YADA Manual — Computational Details

Anders Warne

March 5, 2025

Abstract: YADA (Yet Another Dsge Application) is a Matlab program for Bayesian estimation and evaluation of Dynamic Stochastic General Equilibrium and vector autoregressive models. This paper provides the mathematical details for the various functions used by the software. First, some rather famous examples of DSGE models are presented and all these models are included as examples in the YADA distribution. YADA supports a number of different algorithms for solving log-linearized DSGE models. The fastest algorithm is the so called Anderson-Moore algorithm (AiM), but the approaches of Klein and Sims are also covered and have the benefit of being numerically more robust in certain situations. The AiM parser is used to translate the DSGE model equations into a structural form that the solution algorithms can make use of. Alternatively, parsing of dynare model files is supported and where dynare performs the log-linearization using the AiM option. The output is thereafter handled by YADA to produce files that can be used for solving the DSGE model with the routines supported by YADA. The solution of the DSGE model is expressed as a VAR(1) system that represents the state equations of the state-space representation. Thereafter, the different prior distributions that are supported, the state-space representation and the Kalman filter used to evaluate the log-likelihood are presented. Furthermore, it discusses how the posterior mode is computed, including how the original model parameters can be transformed internally to facilitate the posterior mode estimation. Next, the paper provides some details on the algorithms used for sampling from the posterior distribution: single block and multiple fixed or random block random walk Metropolis and slice sampling algorithms, as well as sequential Monte Carlo. In order to conduct inference based on the draws from the posterior sampler, tools for evaluating convergence are considered next. We are here concerned both with simple graphical tools, as well as formal tools for single and parallel chains. Different methods for estimating the marginal likelihood are considered thereafter. Such estimates may be used to evaluate posterior probabilities for different DSGE models. Various tools for evaluating an estimated DSGE model are provided, including impulse response functions, forecast error variance decompositions, historical forecast error and observed variable decompositions. Forecasting issues, such as the unconditional and conditional predictive distributions, are examined in the following section. The paper thereafter considers frequency domain analysis, such as a decomposition of the population spectrum into shares explained by the underlying structural shocks. Estimation of a VAR model with a prior on the steady state parameters is also discussed. The main concerns are: prior hyperparameters, posterior mode estimation, posterior sampling via the Gibbs sampler, and marginal likelihood calculation (when the full prior is proper), before the topic of forecasting with Bayesian VARs is considered. Next, the paper turns to the important topic of misspecification and goodness-of-fit analysis, where the DSGE-VAR framework is considered in some detail. The expectations formation is thereafter examined and the alternative of adaptive learning is considered as a replacement for rational expectation. Finally, the paper provides information about the various types of input that YADA requires and how these inputs should be prepared.

Contents

In physical science the first essential step in the direction of learning any subject is to find principles of numerical reckoning and practicable methods for measuring some quality connected with it. I often say that when you can measure what you are speaking about, and express it in numbers, you know something about it; but when you cannot measure it, when you cannot express it in numbers, your knowledge is of a meagre and unsatisfactory kind; it may be the beginning of knowledge, but you have scarcely in your thoughts advanced to the state of *Science*, whatever the matter may be.

William Thomson, 1st Baron Kelvin, May 3, 1883 (Thomson, 1891, p. 73).

1. Introduction

The use of Bayesian methods is among some econometricians and statisticians highly controversial. A dogmatic frequentist may argue that using *subjective* information through a prior "pollutes" the information content of the data by deliberately introducing small sample biases. Provided that the data can be regarded as objective information, the choice of model or set of models to use in an empirical study is however subjective and, hence, the use of subjective information is difficult to avoid; for entertaining discussions about the pros and cons of the Bayesian and frequentist approaches to statistical analysis, see Efron (1986), Poirier (1988) with discussions, and Little (2006).

Several arguments for using a Bayesian approach are listed in the introduction of Fernández-Villaverde and Rubio-Ramírez (2004). The first and perhaps most important argument listed there concerns misspecification. Namely, that Bayesian inference relies on the insight that (all) models are false. The subjective information that comes from the prior may therefore to a certain extent "correct" for the effects that misspecification of a model has on the information content in the data.[1] For some recent ideas about model validation in a Bayesian setting, see Geweke (2007).

YADA is a Matlab program for *Bayesian* estimation of and inference in Dynamic Stochastic General Equilibrium (DSGE) and Vector Autoregressive (VAR) models. DSGE models are micro-founded optimization-based models that have become very popular in macroeconomics over the past 25 years. The most recent generation of DSGE models is not just attractive from a theoretical perspective, but is also showing great promise in areas such as forecasting and quantitative policy analysis; see, e.g., Adolfson, Laséen, Lindé, and Villani (2007b), Christiano, Eichenbaum, and Evans (2005), Smets and Wouters (2003, 2005, 2007), and An and Schorfheide (2007). For a historical overview, the reader may, e.g., consult Galí and Gertler (2007) and Mankiw (2006).

The software is developed in connection with the New Area-Wide Model (NAWM) project at the ECB; cf. Christoffel, Coenen, and Warne (2008). Detailed descriptions about how the software has been coded and how functionality can be added to it are given in the document *Extending YADA* which is included in the YADA distribution; cf. Warne (2025).

YADA takes advantage of code made available to the NAWM project by colleagues at both central bank institutions and the academic world. In particular, it relies to some extent on the code written by the group of researchers at Sveriges Riksbank that have developed the Riksbank DSGE model (Ramses). This group includes Malin Adolfson, Stefan Las?en, Jesper Lind?, and Mattias Villani.

A Matlab version of the Anderson-Moore algorithm for solving linear rational expectations models (AiM) and writing them in state-space form is used by YADA; see, e.g., Anderson and Moore (1985), Anderson (1999, 2008, 2010), or Zagaglia (2005). Since only linearized DSGE models can be parsed with and solved through AiM, models based on higher order approximations are not supported; see Fernández-Villaverde and Rubio-Ramírez (2005). However, Dynare can be used to log-linearize a DSGE model from within YADA, while the model solvers supported

---

[1] See also Fernández-Villaverde (2010) for further discussions on the advantages and disadvantages of Bayesian inference.

by YADA can thereafter be used. In addition to the AiM algorithm, the QZ-decomposition (generalized Schur form) based algorithms of Klein (2000) and Sims (2002) are also supported, with the gensys algorithm of Sims being the default model solver since version 3.40 of YADA. Although the Klein algorithm based model solver in YADA does not directly use his *solab* code, I have most definitely taken a peek into it.[2]

Moreover, YADA includes *csminwel* and *gensys*, developed by Christopher Sims, for numerical optimization and solving linear rational expectations models, respectively, as well as code from *Stixbox* by Anders Holtsberg, from the *Lightspeed Toolbox* by Tom Minka, from *Dynare* by Michel Juillard and Stephane Adjemian, from the *Kernel Density Estimation Toolbox* by Christian Beardah, and *kde2d* for bivariate kernel density estimation by Zdravko Botev.[3]

In contrast with other software that can estimate DSGE models, YADA has a Graphical User Inference (GUI) from which all actions and settings are controlled. The current document does not give much information about the GUI. Instead it primarily focuses on the mathematical details of the functions needed to calculate, for instance, the log-likelihood function. The instances when this documents refers to the GUI are always linked to functions that need certain data from the GUI. The help file in the YADA distribution covers the GUI functionality.

This document is structured as follows. In the next section, we present a range of log-linearized DSGE models from the literature, including small-scale and medium-scale models. When we have established the general form of these models, Section 3 provides an overview of the matrix representation of linear rational expectations models and how these can be analysed with the Anderson-Moore algorithm. In addition, the Klein (2000) and Sims (2002) algorithms for solving a DSGE model are discussed, focusing on how the AiM form can be rewritten into a form compatible with the these algorithms. It should be mentioned at this stage that it is not necessary to log-linearize a model analytically before using it with YADA. Instead, (log-)linearization with dynare is supported, where the model is written into a dynare model file and this file is thereafter parsed with YADA's dynare parser function.

Once we know how to solve the models, the issue of estimation can be addressed. The starting point for Bayesian estimation is Bayes theorem, giving the relationship between and the notation for the prior density, the conditional and the marginal density of the data, as well as the posterior density. We then present the density functions that can be used in YADA for the prior distribution of the DSGE model parameters. Parametric definitions of the densities are provided and some of their properties are stated. This leads us into the actual calculation of the likelihood function via the Kalman filter in Section 5. In addition, this Section is concerned with smooth estimation of unobserved variables, such as the structural shocks, the computation of weights on the observed variables for estimates of unobservables, simulation smoothing, square root filtering for handling numerical issues, how to deal with missing observations, univariate filtering for speed concerns, and diffuse initialization.

Since some of the parameters can have bounded support, e.g., that a standard deviation of a shock only takes positive values, the optimization problem for posterior mode estimation typically involves inequality restrictions. Having to take such restrictions into account may slow down the optimization time considerably. A natural way to avoid this issue is to transform the original parameters for estimation such that the domain of the transformed parameters is

---

[2] Paul Klein's homepage can be found through the link: `http://www.paulklein.se/`. A copy of *solab* for Matlab, Gauss, and Fortran can be obtained from there.

[3] The Stixbox toolbox is can be downloaded from `https://forge.scilab.org/index.php/p/stixbox/`; the matlab code for csminwel and gensys can be downloaded from `http://sims.princeton.edu/yftp/optimize/` and `http://sims.princeton.edu/yftp/gensys/` respectively; the Lightspeed Toolbox can be retrieved from Tom Minka's github website at `https://github.com/tminka/lightspeed`; the Dynare code and installer/package are located at `https://www.dynare.org/`; the kernel density estimation toolbox for matlab can be downdloaded from `https://intarch.ac.uk/journal/issue1/beardah/kdeia8.html`; and finally kde2d can be downloaded from `https://www.mathworks.com/matlabcentral/fileexchange/17204-kernel-density-estimation`. In addition, YADA supports an older version of AiM (the binary AiM parser is dated April 8, 1999) for matlab systems that do not support the java-based parser in the more recent versions of AiM. These are available for download from the website of the Federal Reserve at `https://www.federalreserve.gov/econres/ama-index.htm`.

the real line. In this way we shift from a constrained optimization problem to an unconstrained one. The specific transformations that YADA can apply are discussed in Section 6, and how these transformations affect the estimation of the posterior mode is thereafter covered in Section 7.

Once the posterior mode and the inverse Hessian at the mode have been calculated, we can construct draws from the posterior distribution using a Markov Chain Monte Carlo (MCMC) sampler, such as the random walk Metropolis algorithm. In fact, it may be argued that the parameter transformations discussed in Section 6 are more important for posterior sampling than for posterior mode estimation. Specifically, when a Gaussian proposal density is used for the random walk Metropolis algorithm, the proposal draws have the real line as support and, hence, all draws for the transformed parameters are valid candidates. In contrast, when drawing from a Gaussian proposal density for the original parameters it is highly likely that many draws need to be discarded directly since the bounds may often be violated. Moreover, we often have reason to expect the Gaussian proposal density to be a better approximation of the posterior density for parameters that are unbounded than for parameters whose domain is bounded. As a consequence, YADA always performs posterior sampling for the transformed parameters, while the user can select between the original and the transformed parameters for posterior mode estimation. Posterior sampling with the random walk Metropolis algorithm using either a normal or a Student-t proposal density as well as the slice sampler is discussed in Section 8. Furthermore, blocking RWM algorithms based on a normal or a Student-t proposal density are also covered there. The importance sampler, used by for instance DeJong, Ingram, and Whiteman (2000), is not directly supported by YADA as an algorithm by itself. Instead, YADA supports sequential Monte Carlo (SMC), which is closely related to importance sampling, but where the user does not select an importance density and instead employs a so called tempering schedule to generate such a density. One of the SMC algorithms in YADA is based on likelihood tempering and due to Herbst and Schorfheide (2014, 2016), while the other focuses on data tempering and is also based on Durham and Geweke (2014).

Given a sample of draws from the posterior distribution it is important to address the question if the posterior sampler has converged or not. In Section 9 we deal with simple but effective graphical tools as well as formal statistical tools for assessing convergence in a single Markov chain and in parallel chains. When we are satisfied that our posterior sampler has converged, we may turn to other issues regarding Bayesian inference. In Section 10 we examine the problem of computing the marginal likelihood of the DSGE model. This object can be used for cross DSGE model comparisons as long as the same data is covered, but also for comparisons with alternative models, such as Bayesian VARs and DSGE-VARs.

In Section 11 we turn to various tools for analysing the properties of a DSGE model. These tools include impulse response functions, forecast error variance decompositions, conditional correlations, correlation decompositions, observed variable decompositions, and ways of addressing identification concerns. Thereafter, out-of-sample forecasting issues are discussed in Section 12. Both unconditional and conditional forecasting are considered as well as a means for checking if the conditional forecasts are subject to the famous Lucas (1976) critique or not. The following topic concerns frequency domain properties of the DSGE model and of the VAR model and this is covered in Section 13. For example, the population spectrum of the DSGE model can be decomposed at each frequency into the shares explained by the underlying economic shocks. Furthermore, Fisher's information matrix can be computed via the frequency domain using only values for the parameters of the model as input. Provided that we regard identification as a concern which is directly related to the rank of this matrix, identification may therefore be studied in some detail at a very early stage of the analysis, i.e., without the necessity of having access to parameter estimates based on data.

The next topic is Bayesian VARs. In particular, YADA supports VAR models for forecasting purposes. The types of prior that may be used, computation of posterior mode, posterior sampling with the Gibbs sampler, the computation of the marginal likelihood, and forecasting with such models are all given some attention in Section 14. One specific feature of the Bayesian VAR models that YADA support is that the steady state parameters are modelled explicitly.

Next, an important aspect of Bayesian analysis is that it does not rely on the assumption that the model is correctly specified. The so called DSGE-VAR approach, advocated in a series of articles by Del Negro and Schorfheide (2004, 2006, 2009) and Del Negro, Schorfheide, Smets, and Wouters (2007), has been suggested as a tool for measuring the degree of misspecification of a DSGE model by approximating it by a VAR; see also An and Schorfheide (2007). This approach may be also be used as a measure of fit of the DSGE model, and can be used to compute the posterior distribution of the DSGE model parameters when viewed through the lens of the VAR. The setup of DSGE-VARs and their estimation are discussed in some detail in Section 15, while Section 16 turns to the prior and posterior analyses that can be performed through a DSGE-VAR.

Finally, the issue of setting up the DSGE model, VAR, and DSGE-VAR input for YADA is discussed in Section 18. This involves writing the model equations in an AiM model file, specifying a prior for the parameters to estimate, having appropriate parameter functions for parameters that are either calibrated or which are functions of the estimated parameters, the construction of the file for the measurement equations that link the model variables to the observed and the exogenous variables, and finally the file that reads observed data into YADA. Since this paper concerns the implementation of various mathematical issues in a computer program, most sections end with a part that discusses some details of the main functions that are made use of by YADA.

2. DSGE Models

Before we turn our attention to issues related to the estimation of DSGE models it is only natural to first have a look at a few examples and how DSGE models can be solved. In this Section I will therefore provide a bird's eye view of some well know DSGE models, which happen to share the unfortunate fate of being directly supported by YADA. That is, the YADA distribution contains all the necessary files for estimating them. The problem of solving DSGE models is discussed thereafter in Section 3.

A natural point of entry is the benchmark monetary policy analysis model studied in An and Schorfheide (2007). This model is only briefly presented as a close to the simplest possible example of a DSGE model. In addition, a non-linear version of their model is also presented and which is included as an example model in YADA for using dynare to (log-)linearize the non-linear equations. An open economy version of this model from Lubik and Schorfheide (2007a) is thereafter considered. We thereafter turn to the model by Andrés, López-Salido, and Vallés (2006) which includes money and where the data is taken from the Classical Gold Standard era and which has been estimated by Fagan, Lothian, and McNelis (2013). Next, the well-known Smets and Wouters (2007) model is examined in more detail. We first consider the American Economic Review (AER) version of the model, and thereafter the version suggested by, e.g., Del Negro et al. (2007), where detrending is applied with a stochastic rather than a deterministic trend. In addition, we include a small-scale version of the latter model, described by Del Negro and Schorfheide (2013). Moreover, we consider an extension of the stochastic trend version of the Smets and Wouters model which allows for financial frictions, as in Del Negro and Schorfheide (2013), and an extension of the former which allows for unemployment, as in Galí, Smets, and Wouters (2012) for the US and in Smets, Warne, and Wouters (2014) for the euro area.

All the models mentioned above include a monetary policy reaction function, correctly suggesting that the main policy focus within the DSGE model literature is on monetary policy. Fiscal policy is, however, also an issue that has been addressed using log-linearized DSGE model and we therefore consider an interesting example of a model with fiscal polciy rules. The model presented by Leeper, Plante, and Traum (2010) is has government spending, transfer as well as three tax rules and thus covers a wide range of fiscsal policies. This model is also discussed by Herbst and Schorfheide (2016) in their book about Bayesian estimation of DSGE models. The problem of solving a DSGE model will thereafter be addressed in Section 3.

2.1. *The An and Schorfheide Model*

The model economy consists of a final goods producing firm, a continuum of intermediate goods producing firms, a representative household, and a monetary and a fiscal authority. As pointed out by An and Schorfheide (2007), this model has become a benchmark specification for monetary policy analysis; for a detailed derivation see, e.g., King (2000) or Woodford (2003).

Let $\hat{x}_t = \ln(x_t/x)$ denote the natural logarithm of some variable x_t relative to its steady state value x. The log-linearized version of the An and Schorfheide (2007) model we shall consider has 6 equations describing the behavior of (detrended) output (y_t), (detrended) consumption (c_t), (detrended) government spending (g_t), (detrended) technology (z_t), inflation π_t, and a short term nominal interest rate R_t. The equations are given by

$$
\begin{aligned}
\hat{c}_t &= E_t \hat{c}_{t+1} - \frac{1}{\tau}\big(\hat{R}_t - E_t \hat{\pi}_{t+1} - E_t \hat{z}_{t+1}\big), \\
\hat{\pi}_t &= \beta E_t \hat{\pi}_{t+1} + \kappa\big(\hat{y}_t - \hat{g}_t\big), \\
\hat{y}_t &= \hat{c}_t + \hat{g}_t, \\
\hat{R}_t &= \rho_R \hat{R}_{t-1} + (1-\rho_R)\psi_1 \hat{\pi}_t + (1-\rho_R)\psi_2\big(\hat{y}_t - \hat{g}_t\big) + \sigma_R \eta_{R,t}, \\
\hat{g}_t &= \rho_G \hat{g}_{t-1} + \sigma_G \eta_{G,t}, \\
\hat{z}_t &= \rho_Z \hat{z}_{t-1} + \sigma_Z \eta_{Z,t}.
\end{aligned}
\tag{2.1}
$$

The shocks $\eta_{i,t} \sim \text{iid} N(0,1)$ for $i = R, G, Z$, and are called the monetary policy or interest rate shock, the government spending shock, and the technology shock.

The first equation is the log-linearized consumption Euler equation, where τ is the inverse of the intertemporal elasticity of substitution. The second equation is the log-linearized price Phillips curve, with β being the discount factors and κ a function of the inverse of the elasticity of demand (ν), steady-state inflation (π), and the degree of price stickiness (ϕ) according to

$$\kappa = \frac{\tau(1-\nu)}{\nu\pi^2\phi}.$$

The third equation is the log-linearized aggregate resource constraint, while the fourth is the monetary policy rule with $\hat{y}_t - \hat{g}_t$ being equal to the output gap. This output gap measure is a flexible price measure, i.e., based on $\phi = 0$, with potential output thus being equal to government spending. Accordingly, the output gap is equal to consumption in this version of the model. The final two equations in (2.1) are AR(1) processes for the exogenous government spending and technology variables.

The steady state for this model is given by $r = \gamma/\beta$, $R = r\pi^*$, $\pi = \pi^*$, $y = g(1-\nu)^{1/\tau}$, and $c = (1-\nu)^{1/\tau}$. The parameter π^* is the steady-state inflation target of the central bank, while γ is the steady-state growth rate.

The measurement equation linking the data on quarter-to-quarter per capita GDP growth (Δy_t), annualized quarter-to-quarter inflation rates (π_t), and annualized nominal interest rates (R_t) to the model variables are given by:

$$\begin{aligned}
\Delta y_t &= \gamma^{(Q)} + 100\big(\hat{y}_t - \hat{y}_{t-1} + \hat{z}_t\big), \\
\pi_t &= \pi^{(A)} + 400\hat{\pi}_t, \\
R_t &= \pi^{(A)} + r^{(A)} + 4\gamma^{(Q)} + 400\hat{R}_t.
\end{aligned} \tag{2.2}$$

Additional parameter definitions are:

$$\beta = \frac{1}{1 + \dfrac{r^{(A)}}{400}}, \quad \gamma = 1 + \frac{\gamma^{(Q)}}{100}, \quad \pi = 1 + \frac{\pi^{(A)}}{400}, \tag{2.3}$$

where only the β parameter is of real interest since it appears in (2.1). In view of the expression for κ, it follows that the ν and ϕ parameters cannot be identified in the log-linearized version of the model. The parameters to estimate for this model are therefore given by

$$\theta = [\tau\ \kappa\ \psi_1\ \psi_2\ \rho_R\ \rho_G\ \rho_Z\ r^{(A)}\ \pi^{(A)}\ \gamma^{(Q)}\ \sigma_R\ \sigma_G\ \sigma_Z]'. \tag{2.4}$$

When simulating data with the model, the value of θ was given by:

$$\theta = \big[2.00\ 0.15\ 1.50\ 1.00\ 0.60\ 0.95\ 0.65\ 0.40\ 4.00\ 0.50\ 0.002\ 0.008\ 0.0045\big].$$

These values are identical to those reported by An and Schorfheide (2007, Table 2) for their data generating process.

2.1.1. *A Non-Linear Variant of the An and Schorfheide Model*

The first three rows of the log-linearized model in equation (2.1) can be obtained from the following equations:

$$\begin{aligned}
1 &= \beta E_t\big[\exp^{-\tau\hat{c}_{t+1}+\tau\hat{c}_t+\hat{r}_t-\hat{z}_{t+1}-\hat{\pi}_{t+1}}\big], \\
\frac{\kappa}{\tau}\big(\exp^{\tau\hat{c}_t}-1\big) &= \big(\exp^{\hat{\pi}_t}-1\big)\left(\left(1-\frac{1}{2\nu}\right)\exp^{\hat{\pi}_t}+\frac{1}{2\nu}\right) - \\
&\quad \beta E_t\big[\big(\exp^{\hat{\pi}_{t+1}}-1\big)\exp^{-\tau\hat{c}_{t+1}+\tau\hat{c}_t+\hat{y}_{t+1}-\hat{y}_t+\hat{\pi}_{t+1}}\big], \\
\exp^{\hat{c}_t-\hat{y}_t} &= \exp^{-\hat{g}_t} - \frac{\tau(1-\nu)g}{2\nu\kappa}\big(\exp^{\hat{\pi}_t}-1\big)^2.
\end{aligned} \tag{2.5}$$

These equations contain two additional parameters relative to the log-linearized version, ν and g. When (log-)linearizing the equations in (2.5) with dynare these parameters need to take

on some values in the dynare model file, although they have no influence on the parameters of the log-linearized form. In the YADA example files for this particular variant of the model, $\nu = 0.1$ and $g = 1/0.85$ following the values selected for the prior mean in An and Schorfheide (2007, Section 6.3). Since equations (2.5) already contain the log-linearized variables, there is no need to tell dynare to log-linearize them, but simply to linearize.

2.2. *A Small Open Economy DSGE Model: The Lubik and Schorfheide Example*

As an extension of the closed economy An and Schorfheide model example, YADA also comes with a small open economy DSGE model which has been investigated by, e.g., Lubik and Schorfheide (2007a) and Lees, Matheson, and Smith (2011) using actual data. The YADA example is based on data simulated with the DSGE model.

The model is a simplification of Galí and Monacelli (2005) and, like its closed economy counterpart, consists of a forward-looking IS-equation and a Phillips curve. Monetary policy is also given by a Taylor-type interest rate rule, where the exchange rate is introduced via the definition of consumer prices and under the assumption of PPP. In log-linearized form the model can be expressed as

$$
\begin{aligned}
\hat{y}_t &= E_t\hat{y}_{t+1} - \varphi\left(\hat{R}_t - E_t\hat{\pi}_{t+1}\right) - E_t\hat{z}_{t+1} - \alpha\varphi E_t\Delta\hat{q}_{t+1} + \alpha(1-\alpha)\frac{1-\tau}{\tau}E_t\Delta\hat{y}_{t+1}^*, \\
\hat{\pi}_t &= \beta E_t\hat{\pi}_{t+1} + \alpha\beta E_t\Delta\hat{q}_{t+1} - \alpha\Delta\hat{q}_t + \frac{\kappa}{\varphi}\left(\hat{y}_t - \hat{x}_t\right), \\
\hat{R}_t &= \rho_R\hat{R}_{t-1} + (1-\rho_R)\left[\psi_\pi\hat{\pi}_t + \psi_y\hat{y}_t + \psi_{\Delta e}\Delta\hat{e}_t\right] + \sigma_R\eta_{R,t}, \\
\Delta\hat{e}_t &= \hat{\pi}_t - (1-\alpha)\Delta\hat{q}_t - \hat{\pi}_t^*,
\end{aligned}
\tag{2.6}
$$

where $\beta = 1/(1+(r^{(A)}/400))$, $\varphi = \tau + \alpha(2-\alpha)(1-\tau)$, τ is the intertemporal elasticity of substitution, and $0 < \alpha < 1$ is the import share. The closed economy version of the model is obtained when $\alpha = 0$.

Variables denoted with an asterisk superscript are foreign, the nominal exchange rate is given by $\hat{e}$, terms-of-trade (defined as the relative price of exports in terms of imports) by $\hat{q}$, while potential output in the absence of nominal rigidities, $\hat{x}_t$, is determined by the equation:

$$
\hat{x}_t = -\alpha(2-\alpha)\frac{1-\tau}{\tau}\hat{y}_t^*,
$$

To close the model, the remaining 4 variables are assumed to be exogenous and determined by:

$$
\begin{aligned}
\Delta\hat{q}_t &= \rho_Q\Delta\hat{q}_{t-1} + \sigma_Q\eta_{Q,t}, \\
\hat{y}_t^* &= \rho_{Y^*}\hat{y}_{t-1}^* + \sigma_{Y^*}\eta_{Y^*,t}, \\
\hat{\pi}_t^* &= \rho_{\pi^*}\hat{\pi}_{t-1}^* + \sigma_{\pi^*}\eta_{\pi^*,t}, \\
\hat{z}_t &= \rho_Z\hat{z}_{t-1} + \sigma_Z\eta_{Z,t},
\end{aligned}
\tag{2.7}
$$

where $\eta_{i,t} \sim N(0,1)$ for $i = R, Q, Y^*, \pi^*, Z$. When the model is taken literally, the terms-of-trade variable is endogenously determined by the equation:

$$
\varphi\Delta\hat{q}_t = \Delta\hat{y}_t^* - \Delta\hat{y}_t.
$$

However, Lubik and Schorfheide note that such a specification leads to numerical problems when estimating the posterior mode and to implausible parameters estimates and low likelihood values when a mode is located.

The measurement equations for this model are:

$$
\begin{aligned}
\Delta y_t &= \gamma^{(Q)} + \Delta\hat{y}_t + \hat{z}_t, \\
\pi_t &= \pi^{(A)} + \hat{\pi}_t, \\
R_t &= \pi^{(A)} + r^{(A)} + 4\gamma^{(Q)} + \hat{R}_t, \\
\Delta e_t &= \Delta\hat{e}_t, \\
\Delta q_t &= \Delta\hat{q}_t.
\end{aligned}
\tag{2.8}
$$

The model therefore has a total of 19 unknown parameters, collected into

$$\theta = \begin{bmatrix} \psi_\pi & \psi_y & \psi_{\Delta e} & \rho_R & \alpha & r^{(A)} & \kappa & \tau & \rho_Q & \rho_Z \\ \rho_{Y^*} & \rho_{\pi^*} & \pi^{(A)} & \gamma^{(Q)} & \sigma_Q & \sigma_Z & \sigma_R & \sigma_{Y^*} & \sigma_{\pi^*} \end{bmatrix}.$$

When simulating data using this model, the value for θ was given by:

$$\theta = \begin{bmatrix} 1.30 & 0.23 & 0.14 & 0.69 & 0.11 & 0.51 & 0.32 & 0.31 & 0.31 & 0.42 \\ 0.97 & 0.46 & 1.95 & 0.55 & 1.25 & 0.84 & 0.36 & 1.29 & 2.00 \end{bmatrix}.$$

With the exception of the parameters that reflect the steady-state values of the observables, the values are equal to the benchmark estimates for Canada in Lubik and Schorfheide (2007a, Table 3).

2.3. *A Model with Money Demand and Money Supply: Fagan, Lothian and McNelis*

The DSGE model considered by Fagan et al. (2013) is a variant of the model suggested by Andrés et al. (2006), which in turn has predecessors such as Rotemberg and Woodford (1997), McCallum and Nelson (1999), and Ireland (2004), and this version assumes linear separability between money and consumption in preferences. The incorporation of money makes it possible to analyse monetary regimes where interest rates are determined through the interplay between supply and demand, instead of from a policy rule where the Central Bank sets the short-term nominal interest rate. The study by Fagan et al. (2013) focuses on the Gold Standard era and assumes that base money supply is exogenous, but as in Andrés et al. (2006) this assumption may be replaced with having a standard Taylor type of policy rule for determining the interest rate.

Based on the notation from Fagan, Lothian, and McNelis (2013), the log-linearized Euler equation for real GDP is given by

$$\hat{y}_t = \frac{\phi_1}{\phi_1 + \phi_2}\hat{y}_{t-1} + \frac{\beta\phi_1 + \phi_2}{\phi_1 + \phi_2}E_t\hat{y}_{t+1} - \frac{1}{\phi_1 + \phi_2}\left(\hat{r}_t - E_t\hat{\pi}_{t+1}\right)$$
$$- \frac{\beta\phi_1}{\phi_1 + \phi_2}E_t\hat{y}_{t+2} + \frac{\left(1 - \beta h\rho_a\right)\left(1 - \rho_a\right)}{\left(1 - \beta h\right)\left(\phi_1 + \phi_2\right)}\hat{a}_t, \tag{2.9}$$

where $\hat{a}_t$ is an exogenous aggregate demand shock. Real money demand is determined by

$$\hat{m}_t - \hat{p}_t = -\frac{\phi_1}{\delta}\hat{y}_{t-1} + \frac{\phi_2}{\delta}\hat{y}_t - \frac{\beta\phi_1}{\delta}E_t\hat{y}_{t+1} - \frac{1}{\delta(r-1)}\hat{r}_t$$
$$+ \frac{1 - \beta h\rho_a}{\left(1 - \beta h\right)\delta}\hat{a}_t + \frac{\delta - 1}{\delta}\hat{e}_t, \tag{2.10}$$

where $\hat{e}_t$ is an exogenous liquidity preference (money demand) shock.

Real marginal costs are related to output and exogenous shocks in the log-linearized form

$$\widehat{mc}_t = -\phi_1\hat{y}_{t-1} + \left(\chi + \phi_2\right)\hat{y}_t - \beta\phi_1 E_t\hat{y}_{t+1} - \left(1 + \chi\right)\hat{z}_t$$
$$- \frac{\beta h\left(1 - \rho_a\right)}{\left(1 - \beta h\right)}\hat{a}_t, \tag{2.11}$$

where $\hat{z}_t$ is an exogenous technology (productivity) shock. Next, the log-linearized Phillips curve is

$$\hat{\pi}_t = \gamma_f E_t\hat{\pi}_{t+1} + \gamma_b\hat{\pi}_{t-1} + \lambda\widehat{mc}_t, \tag{2.12}$$

and therefore contains both a forward looking and a backward looking part, while λ is the sensitivity of inflation to marginal costs. The real marginal costs function and the log-linearized Phillips curve characterize the supply side of the economy.

Andrés et al. (2006) show that many of the parameters in the log-linearized structural form of the model are functions of the underlying deep structural parameters. Specifically,

$$\lambda = \left(1 - \theta_p\right)\left(1 - \beta\theta_p\right)\left(1 - \omega\right)\xi,$$

$$\xi = \frac{1 - \alpha}{\left[1 + \alpha(\varepsilon - 1)\right]\left[\theta_p + \omega\left(1 - \theta_p[1 - \beta]\right)\right]},$$

$$\chi = \frac{\varphi + \alpha}{1 - \alpha},$$

where $1 - \alpha$ is the elasticity of labor with respect to output, and φ is the labor supply elasticity. Each firm resets its price with probability $1 - \theta_p$ each period, while a fraction θ_p keep their price unchanged; Calvo (1983) mechanism. The ratio $\varepsilon/(1 - \varepsilon)$ is equal to the steady-state price markup. The parameter ω measures the fraction of firms that set prices in a backward-looking way using a simple rule of thumb, while $1 - \omega$ is the fraction of firms that set prices in a forward-looking manner; see Galí and Gertler (1999). The parameter h represents the importance of habit persistence in the utility function, while β is the discount factor and r is the steady-state interest rate. The parameter δ reflects the exponential weight on real money balances in the temporal utility function and serves a similar purpose as σ has for consumption. Specifically, if $\delta = 1$ ($\sigma = 1$) then the log of real money balances (log of current consumption over lagged, habit adjusted consumption) appears in the utility function. The σ parameter is interpreted by Fagan et al. (2013) as the constant relative risk aversion. Additionally,

$$\phi_1 = \frac{(\sigma - 1)h}{1 - \beta h},$$

$$\phi_2 = \frac{\sigma + (\sigma - 1)\beta h^2 - \beta h}{1 - \beta h},$$

$$\gamma_f = \frac{\beta\theta_p}{\theta_p + \omega\left(1 - \theta_p[1 - \beta]\right)},$$

$$\gamma_b = \frac{\omega}{\theta_p + \omega\left(1 - \theta_p[1 - \beta]\right)}.$$

Notice that $\sigma = 1$ implies that $\phi_1 = 0$, while $\phi_2 = 1$. Furthermore, γ_f is increasing in θ_p, while γ_b is increasing in ω. It may also be noted that if $\omega = 0$ such that all firms set prices based on profit maximization, then inflation becomes a purely foreward-looking variable.

The monetary regime over the Gold standard is modelled as an exogenous process for nominal base money growth by Fagan, Lothian, and McNelis (2013), as suggested by, e.g., Cagan (1965).[4] Specifically,

$$\Delta\hat{m}_t = \rho_m\Delta\hat{m}_{t-1} + \sigma_m\eta_{m,t}, \tag{2.13}$$

where $\eta_{m,t}$ is an iid $N(0,1)$ money supply shock. Real money balances obeys the identity

$$\hat{m}_t - \hat{p}_t = \hat{m}_{t-1} - \hat{p}_{t-1} + \Delta\hat{m}_t - \hat{\pi}_t. \tag{2.14}$$

Finally, the aggregate demand, liquidity preference, and technology shocks also follow AR(1) processes with iid $N(0,1)$ innovations, according to

$$\begin{aligned}
\hat{a}_t &= \rho_a\hat{a}_{t-1} + \sigma_a\eta_{a,t}, \\
\hat{e}_t &= \rho_e\hat{e}_{t-1} + \sigma_e\eta_{e,t}, \\
\hat{z}_t &= \rho_z\hat{z}_{t-1} + \sigma_z\eta_{z,t}.
\end{aligned} \tag{2.15}$$

---

[4] Alternatively, Andrés, López-Salido, and Vallés (2006) consider a Taylor type of monetary policy rule, but with the twist that it includes money growth. Specifically,

$$\hat{r}_t = \rho_r\hat{r}_{t-1} + \left(1 - \rho_r\right)\left(\rho_\pi\hat{\pi}_t + \rho_y\hat{y}_t + \rho_m\Delta\hat{m}_t\right) + \sigma_r\eta_{r,t},$$

where $\eta_{r,t}$ is an iid $N(0,1)$ interest rate shock, and ρ_m is the weight on nominal money growth.

The measurement equations for this model are:

$$y_t = \hat{y}_t + \varepsilon_{y,t},$$
$$r_t = \hat{r}_t,$$
$$\pi_t = \hat{\pi}_t,$$
$$\Delta m_t = \Delta \hat{m}_t,$$

(2.16)

where $\varepsilon_{y,t}$ is an iid $N(0, \sigma_y^2)$ measurement error. The observed variables are given by HP-filtered real GDP, the demeaned short-term interest rate (commercial paper rate), the demeaned first difference of the log of the GDP deflator, and the demeaned first difference of the log of base money.

Two of the parameters of the model are calibrated and given by $\beta = 0.988$, and $r = 1/\beta$. The remaining 16 parameters are estimated and given the vector:

$$\theta = \begin{bmatrix} h & \sigma & \chi & \lambda & \theta_p & \omega & \delta & \rho_a & \rho_e & \rho_z & \rho_m & \sigma_a & \sigma_e & \sigma_z & \sigma_m & \sigma_y \end{bmatrix}.$$

The model still has three unidentified deep structural parameters, namely: α, ε, and φ. They are not explicitly needed to solve the log-linearized version of the model, but if we are willing to fix one of them, say $\alpha = 1/3$, then the other two can be solved from the equations for ξ and χ.

2.4. *A Medium-Sized Closed Economy DSGE Model: Smets and Wouters*

A well known example of a medium-sized DSGE model is Smets and Wouters (2007), where the authors study shocks and frictions in US business cycles. Like the two examples discussed above, the Smets and Wouters model is also provided as an example with the YADA distribution. The equations of the model are presented below, while a detailed discussion of the model is found in Smets and Wouters (2007); see also Smets and Wouters (2003, 2005). It should be emphasized that since the model uses a flexible-price based output gap measure in the monetary policy rule, the discussion will first consider the sticky price and wage system, followed by the flexible price and wage system. The equations for the 7 exogenous variables are introduced thereafter, while the steady-state of the system closes the theoretical part of the empirical model. Finally, the model variables are linked to the observed variables via the measurement equations.

2.4.1. *The Sticky Price and Wage Equations*

The log-linearized aggregate resource constraint of this closed economy model is given by

$$\hat{y}_t = c_y \hat{c}_t + i_y \hat{i}_t + z_y \hat{z}_t + \varepsilon_t^g,$$

(2.17)

where $\hat{y}_t$ is (detrended) real GDP. It is absorbed by real private consumption ($\hat{c}_t$), real private investments ($\hat{i}_t$), the capital utilization rate ($\hat{z}_t$), and exogenous spending (ε_t^g). The parameter c_y is the steady-state consumption-output ratio and i_y is the steady-state investment-output ratio, where

$$c_y = 1 - i_y - g_y,$$

and g_y is the steady-state exogenous spending-output ratio. The steady-state investment-output ratio is determined by

$$i_y = (\gamma + \delta - 1) k_y,$$

where k_y is the steady-state capital-output ratio, γ is the steady-state growth rate, and δ is the depreciation rate of capital. Finally,

$$z_y = r^k k_y,$$

where r^k is the steady-state rental rate of capital. The steady-state parameters are shown in Section 2.4.4, but it is noteworthy already at this stage that $z_y = \alpha$, the share of capital in production.

The dynamics of consumption follows from the consumption Euler equation and is equal to

$$\hat{c}_t = c_1 \hat{c}_{t-1} + (1 - c_1) E_t \hat{c}_{t+1} + c_2 \left(\hat{l}_t - E_t \hat{l}_{t+1} \right) - c_3 \left(\hat{r}_t - E_t \hat{\pi}_{t+1} \right) + \varepsilon_t^b,$$

(2.18)

where $\hat{l}_t$ is hours worked, $\hat{r}_t$ is the policy controlled nominal interest rate, and ε_t^b is proportional to the exogenous risk premium, i.e., a wedge between the interest rate controlled by the central bank and the return on assets held by households. It should be noted that in contrast to Smets and Wouters (2007), but identical to Smets and Wouters (2005) and Lindé, Smets, and Wouters (2016), we have moved the risk premium variable outside the expression for the ex ante real interest rate. This means that $\varepsilon_t^b = -c_3 \varepsilon_t^b$, where ε_t^b is the risk premium variable in Smets and Wouters (2007). Building on the work by Krishnamurthy and Vissing-Jorgensen (2012), Fisher (2015) shows that this shock can be given a structural interpretation, namely, as a shock to the demand for safe and liquid assets or, alternatively, as a liquidity preference shock. I have chosen to consider the expression in (2.18) since it is also used in the dynare code that can be downloaded from the American Economic Review web site in connection with the 2007 article.[5]

The parameters of the consumption Euler equation are:

$$c_1 = \frac{\lambda/\gamma}{1+(\lambda/\gamma)}, \quad c_2 = \frac{(\sigma_c - 1)(w^h l/c)}{\sigma_c\left(1+(\lambda/\gamma)\right)}, \quad c_3 = \frac{1-(\lambda/\gamma)}{\sigma_c\left(1+(\lambda/\gamma)\right)},$$

where λ measures external habit formation, σ_c is the inverse of the elasticity of intertemporal substitution for constant labor, while $w^h l/c$ is the steady-state hourly real wage bill to consumption ratio. If $\sigma_c = 1$ (log-utility) and $\lambda = 0$ (no external habit) then the above equation reduces to the familiar purely forward looking consumption Euler equation.

The log-linearized investment Euler equation is given by

$$\hat{i}_t = i_1 \hat{i}_{t-1} + (1-i_1) E_t \hat{i}_{t+1} + i_2 \hat{q}_t + \varepsilon_t^i, \tag{2.19}$$

where $\hat{q}_t$ is the real value of the existing capital stock, while ε_t^i is an exogenous investment-specific technology variable. The parameters of (2.19) are given by

$$i_1 = \frac{1}{1+\beta\gamma^{1-\sigma_c}}, \quad i_2 = \frac{1}{\left(1+\beta\gamma^{1-\sigma_c}\right)\gamma^2\varphi},$$

where β is the discount factor used by households, and φ is the steady-state elasticity of the capital adjustment cost function.

The dynamic equation for the value of the capital stock is

$$\hat{q}_t = q_1 E_t \hat{q}_{t+1} + \left(1-q_1\right) E_t \hat{r}_{t+1}^k - \left(\hat{r}_t - E_t \hat{\pi}_{t+1}\right) + c_3^{-1}\varepsilon_t^b, \tag{2.20}$$

where $\hat{r}_t^k$ is the rental rate of capital. The parameter q_1 is here given by

$$q_1 = \beta\gamma^{-\sigma_c}(1-\delta) = \frac{1-\delta}{r^k + 1 - \delta}.$$

Turning to the supply-side of the economy, the log-linearized aggregate production function can be expressed as

$$\hat{y}_t = \phi_p \left[\alpha \hat{k}_t^s + (1-\alpha)\hat{l}_t + \varepsilon_t^a\right], \tag{2.21}$$

where $\hat{k}_t^s$ is capital services used in production, and ε_t^a an exogenous total factor productivity variable. As mentioned above, the parameter α reflects the share of capital in production, while ϕ_p is equal to one plus the steady-state share of fixed costs in production.

The capital services variable is used to reflect that newly installed capital only becomes effective with a one period lag. This means that

$$\hat{k}_t^s = \hat{k}_{t-1} + \hat{z}_t, \tag{2.22}$$

where $\hat{k}_t$ is the installed capital. The degree of capital utilization is determined from cost minimization of the households that provide capital services and is therefore a positive function of the rental rate of capital. Specifically,

$$\hat{z}_t = z_1 \hat{r}_t^k, \tag{2.23}$$

---

[5] The links to the code and the data as well as the Appendix of Smets and Wouters (2007) can be found next to the electronic version of the paper.

where

$$z_1 = \frac{1 - \psi}{\psi},$$

and ψ is a positive function of the elasticity of the capital adjustment cost function and normalized to be between 0 and 1. The larger ψ is the costlier it is to change the utilization of capital.

The log-linearized equation that specifies the development of installed capital is

$$\hat{k}_t = k_1 \hat{k}_{t-1} + (1 - k_1) \hat{\imath}_t + k_2 \varepsilon_t^i. \tag{2.24}$$

The two parameters are given by

$$k_1 = \frac{1 - \delta}{\gamma}, \quad k_2 = \left(\gamma + \delta - 1\right)\left(1 + \beta\gamma^{1-\sigma_c}\right)\gamma\varphi.$$

From the monopolistically competitive goods market, the price markup ($\hat{\mu}_t^p$) is equal to minus the real marginal cost ($\hat{\mu}_t^c$) under cost minimization by firms. That is,

$$\hat{\mu}_t^p = \alpha\left(\hat{k}_t^s - \hat{l}_t\right) - \hat{w}_t + \varepsilon_t^a, \tag{2.25}$$

where the real wage is given by $\hat{w}_t$. Similarly, the real marginal cost is

$$\hat{\mu}_t^c = \alpha\hat{r}_t^k + (1 - \alpha)\hat{w}_t - \varepsilon_t^a, \tag{2.26}$$

where (2.26) is obtained by substituting for the optimally determined capital-labor ratio in equation (2.28).

Due to price stickiness, as in Calvo (1983), and partial indexation to lagged inflation of those prices that cannot be reoptimized, prices adjust only sluggishly to their desired markups. Profit maximization by price-setting firms yields the log-linearized price Phillips curve

$$\begin{aligned}
\hat{\pi}_t &= \pi_1 \hat{\pi}_{t-1} + \pi_2 E_t \hat{\pi}_{t+1} - \pi_3 \hat{\mu}_t^p + \varepsilon_t^p \\
&= \pi_1 \hat{\pi}_{t-1} + \pi_2 E_t \hat{\pi}_{t+1} + \pi_3 \hat{\mu}_t^c + \varepsilon_t^p,
\end{aligned} \tag{2.27}$$

where ε_t^p is an exogenous price markup process. The parameters of the Phillips curve are given by

$$\pi_1 = \frac{\iota_p}{1 + \beta\gamma^{1-\sigma_c}\iota_p}, \quad \pi_2 = \frac{\beta\gamma^{1-\sigma_c}}{1 + \beta\gamma^{1-\sigma_c}\iota_p}, \quad \pi_3 = \frac{\left(1 - \xi_p\right)\left(1 - \beta\gamma^{1-\sigma_c}\xi_p\right)}{\left(1 + \beta\gamma^{1-\sigma_c}\iota_p\right)\xi_p\left((\phi_p - 1)\varepsilon_p + 1\right)}.$$

The degree of indexation to past inflation is determined by the parameter ι_p, ξ_p measures the degree of price stickiness such that $1 - \xi_p$ is the probability that a firm can reoptimize its price, and ε_p is the curvature of the Kimball (1995) goods market aggregator.

Cost minimization of firms also implies that the rental rate of capital is related to the capital-labor ratio and the real wage according to.

$$\hat{r}_t^k = -\left(\hat{k}_t^s - \hat{l}_t\right) + \hat{w}_t. \tag{2.28}$$

In the monopolistically competitive labor market the wage markup is equal to the difference between the real wage and the marginal rate of substitution between labor and consumption

$$\hat{\mu}_t^w = \hat{w}_t - \left(\sigma_l \hat{l}_t + \frac{1}{1 - (\lambda/\gamma)}\left[\hat{c}_t - \frac{\lambda}{\gamma}\hat{c}_{t-1}\right]\right), \tag{2.29}$$

where σ_l is the elasticity of labor supply with respect to the real wage.

Due to wage stickiness and partial wage indexation, real wages respond gradually to the desired wage markup

$$\hat{w}_t = w_1 \hat{w}_{t-1} + (1 - w_1)\left[E_t \hat{w}_{t+1} + E_t \hat{\pi}_{t+1}\right] - w_2 \hat{\pi}_t + w_3 \hat{\pi}_{t-1} - w_4 \hat{\mu}_t^w + \varepsilon_t^w, \tag{2.30}$$

where ε_t^w is an exogenous wage markup process. The parameters of the wage equation are

$$w_1 = \frac{1}{1 + \beta\gamma^{1-\sigma_c}}, \quad w_2 = \frac{1 + \beta\gamma^{1-\sigma_c}\iota_w}{1 + \beta\gamma^{1-\sigma_c}},$$

$$w_3 = \frac{\iota_w}{1 + \beta\gamma^{1-\sigma_c}}, \quad w_4 = \frac{\left(1 - \xi_w\right)\left(1 - \beta\gamma^{1-\sigma_c}\xi_w\right)}{\left(1 + \beta\gamma^{1-\sigma_c}\right)\xi_w\left(\left(\phi_w - 1\right)\varepsilon_w + 1\right)}.$$

The degree of wage indexation to past inflation is given by the parameter ι_w, while ξ_w is the degree of wage stickiness. The steady-state labor market markup is equal to $\phi_w - 1$ and ε_w is the curvature of the Kimball labor market aggregator.

The sticky price and wage part of the model is closed by adding the monetary policy reaction function

$$\hat{r}_t = \rho\hat{r}_{t-1} + \left(1 - \rho\right)\left[r_\pi\hat{\pi}_t + r_y\left(\hat{y}_t - \hat{y}_t^f\right)\right] + r_{\Delta y}\left[\Delta\hat{y}_t - \Delta\hat{y}_t^f\right] + \varepsilon_t^r, \qquad (2.31)$$

where $\hat{y}_t^f$ is potential output measured as the level of output that would prevail under flexible prices and wages in the absence of the two exogenous markup processes, whereas ε_t^r is an exogenous monetary policy shock process.

2.4.2. *The Flexible Price and Wage Equations*

The flexible price equations are obtained by assuming that the two exogenous markup processes are zero, while $\xi_w = \xi_p = 0$, and $\iota_w = \iota_p = 0$. As a consequence, inflation is always equal to the steady-state inflation rate while real wages are equal to the marginal rate of substitution between labor and consumption as well as to the marginal product of labor. All other aspects of the economy are unaffected. Letting the superscript f denote the flexible price and wage economy versions of the variables we find that

$$\hat{y}_t^f = c_y\hat{c}_t^f + i_y\hat{i}_t^f + z_y\hat{z}_t^f + \varepsilon_t^g,$$

$$\hat{c}_t^f = c_1\hat{c}_{t-1}^f + (1 - c_1)E_t\hat{c}_{t+1}^f + c_2\left(\hat{l}_t^f - E_t\hat{l}_{t+1}^f\right) - c_3\hat{r}_t^f + \varepsilon_t^b,$$

$$\hat{i}_t^f = i_1\hat{i}_{t-1}^f + (1 - i_1)E_t\hat{i}_{t+1}^f + i_2\hat{q}_t^f + \varepsilon_t^i,$$

$$\hat{q}_t^f = q_1E_t\hat{q}_{t+1}^f + \left(1 - q_1\right)E_t\hat{r}_{t+1}^{k,f} - \hat{r}_t^f + c_3^{-1}\varepsilon_t^b,$$

$$\hat{y}_t^f = \phi_p\left[\alpha\hat{k}_t^{s,f} + (1 - \alpha)\hat{l}_t^f + \varepsilon_t^a\right],$$

$$\hat{k}_t^{s,f} = \hat{k}_{t-1}^f + \hat{z}_t^f, \qquad (2.32)$$

$$\hat{z}_t^f = z_1\hat{r}_t^{k,f},$$

$$\hat{k}_t^f = k_1\hat{k}_{t-1}^f + (1 - k_1)\hat{i}_t^f + k_2\varepsilon_t^i,$$

$$\varepsilon_t^a = \alpha\hat{r}_t^{k,f} + (1 - \alpha)\hat{w}_t^f,$$

$$\hat{r}_t^{k,f} = -\left(\hat{k}_t^{s,f} - \hat{l}_t^f\right) + \hat{w}_t^f,$$

$$\hat{w}_t^f = \sigma_l\hat{l}_t^f + \frac{1}{1 - (\lambda/\gamma)}\left[\hat{c}_t^f - \frac{\lambda}{\gamma}\hat{c}_{t-1}^f\right],$$

where $\hat{r}_t^f$ is the real interest rate of the flexible price and wage system.

2.4.3. *The Exogenous Variables*

There are seven exogenous processes in the Smets and Wouters (2007) model. These are generally modelled as AR(1) process with the exception of the exogenous spending process (where the process depends on both the exogenous spending shock η_t^g and the total factor productivity shock η_t^a) and the exogenous price and wage markup processes, which are treated as ARMA(1,1)

processes. This means that

$$\varepsilon_t^g = \rho_g \varepsilon_{t-1}^g + \sigma_g \eta_t^g + \rho_{ga} \sigma_a \eta_t^a,$$
$$\varepsilon_t^b = \rho_b \varepsilon_{t-1}^b + \sigma_b \eta_t^b,$$
$$\varepsilon_t^i = \rho_i \varepsilon_{t-1}^i + \sigma_i \eta_t^i,$$
$$\varepsilon_t^a = \rho_a \varepsilon_{t-1}^a + \sigma_a \eta_t^a, \tag{2.33}$$
$$\varepsilon_t^p = \rho_p \varepsilon_{t-1}^p + \sigma_p \eta_t^p - \mu_p \sigma_p \eta_{t-1}^p,$$
$$\varepsilon_t^w = \rho_w \varepsilon_{t-1}^w + \sigma_w \eta_t^w - \mu_w \sigma_w \eta_{t-1}^w,$$
$$\varepsilon_t^r = \rho_r \varepsilon_{t-1}^r + \sigma_r \eta_t^r.$$

The shocks η_t^j, $j = \{a, b, g, i, p, r, w\}$, are $N(0, 1)$, where η_t^b is a preference shock (proportional to a risk premium shock), η_t^i is an investment-specific technology shock, η_t^p is a price markup shock, η_t^r is a monetary policy or interest rate shock, and η_t^w is a wage markup shock.

2.4.4. *The Steady-State Equations*

It remains to provide expressions for the steady-state values of the capital-output ratio, the rental rate of capital, and the hourly real wage bill to consumption ratio which relate them to the parameters of the model. The steady-state exogenous spending to output ratio g_y is a calibrated parameter and set to 0.18 by Smets and Wouters (2007). From the expressions for the q_1 parameter it follows that

$$r^k = \frac{1}{\beta \gamma^{-\sigma_c}} + \delta - 1.$$

The capital-output ratio is derived in a stepwise manner in the model appendix to Smets and Wouters (2007). The steady-state relations there are

$$w = \left[\frac{\alpha^\alpha (1-\alpha)^{(1-\alpha)}}{\phi_p (r^k)^\alpha} \right]^{1/(1-\alpha)}, \quad \frac{l}{k} = \frac{(1-\alpha) r^k}{\alpha w}, \quad \frac{k}{y} = \phi_p \left[\frac{l}{k} \right]^{\alpha-1},$$

where $k_y = k/y$. From these relationships it is straightforward, albeit tedious, to show that $z_y = r^k k_y = \alpha$.

The steady-state relation between real wages and hourly real wages is

$$w = \phi_w w^h,$$

so that the steady-state hourly real wage bill to consumption ratio is given by

$$\frac{w^h l}{c} = \frac{(1-\alpha) r^k k_y}{\phi_w \alpha c_y} = \frac{1-\alpha}{\phi_w c_y},$$

where the last equality follows from the relationship of $z_y = r^k k_y = \alpha$.

2.4.5. *The Measurement Equations*

The Smets and Wouters (2007) model is consistent with a balanced steady-state growth path driven by deterministic labor augmenting technological progress. The observed variables are given by quarterly data of the log of real GDP per capita (y_t), the log of real consumption per capita (c_t), the log of real investment per capita (i_t), the log of hours per capita (l_t), the log of quarterly GDP deflator inflation (π_t), the log of real wages (w_t), and the federal funds rate (r_t). With all observed variables except hours, inflation, and the federal funds rate being measured

in first differences, the measurement equations are given by

$$
\begin{bmatrix}
\Delta y_t \\
\Delta c_t \\
\Delta i_t \\
\Delta w_t \\
l_t \\
\pi_t \\
r_t
\end{bmatrix}
=
\begin{bmatrix}
\bar{\gamma} \\
\bar{\gamma} \\
\bar{\gamma} \\
\bar{\gamma} \\
\bar{l} \\
\bar{\pi} \\
4\bar{r}
\end{bmatrix}
+
\begin{bmatrix}
\hat{y}_t - \hat{y}_{t-1} \\
\hat{c}_t - \hat{c}_{t-1} \\
\hat{i}_t - \hat{i}_{t-1} \\
\hat{w}_t - \hat{w}_{t-1} \\
\hat{l}_t \\
\hat{\pi}_t \\
4\hat{r}_t
\end{bmatrix}.
\tag{2.34}
$$

Since all observed variables except the federal funds rate (which is already reported in percent) are multiplied by 100, it follows that the steady-state values on the right hand side are given by

$$
\bar{\gamma} = 100\,(\gamma - 1), \quad \bar{\pi} = 100\,(\pi - 1), \quad \bar{r} = 100\left(\frac{\pi}{\beta\gamma^{-\sigma_c}} - 1\right),
$$

where π is steady-state inflation. The federal funds rate is measured in quarterly terms in Smets and Wouters (2007) through division by 4, and is therefore multiplied by 4 in (2.34) to restore it to annual terms.[6] At the same time, the model variable $\hat{r}_t$ is measured in quarterly terms.

Apart from the steady-state exogenous spending-output ratio only four additional parameters are calibrated. These are $\delta = 0.025$, $\phi_w = 1.5$, and $\varepsilon_p = \varepsilon_w = 10$. The remaining 19 structural and 17 shock process parameters are estimated. The prior distributions of the parameters are given in Smets and Wouters (2007, Table 1) and are also provided in the YADA example of their model.

2.5. *Smets and Wouters Model with Stochastic Detrending*

Smets and Wouters (2007) assume that the underlying trend for output, private consumption, etc. is given by γ^t and is therefore deterministic. Del Negro and Schorfheide (2013) consider a detrending approach which allows for a stochastic trend which may even have a unit root. Specifically and as in Smets and Wouters (2007), the total factor productivity (TFP) process around a liner deterministic trend is given by

$$
\varepsilon_t^a = \rho_a \varepsilon_{t-1}^a + \sigma_a \eta_t^a,
$$

where η_t^a, as in Section 2.4.3, is an iid standard normal innovation. Define the trend variable as follows:

$$
\Phi_t = \exp\!\left(\tilde{\gamma} t + (1 - \alpha)^{-1} \varepsilon_t^a\right).
$$

Notice that the deterministic part of the trend is now given by $\exp(\tilde{\gamma} t)$ rather than by γ^t, i.e. we have that $\gamma = \exp(\tilde{\gamma})$. If $\rho_a = 1$, then the log of the trend follows a random walk with drift $\tilde{\gamma}$, while $|\rho_a| < 1$ means that the TFP process is stationary so that the natural logarithm of Φ_t is staionary around a linear deterministic trend.

Del Negro and Schorfheide (2013) suggest detrending the non-stationary variables with Φ_t and as a consequence a number of the equations in Section 2.4 need to take this into account. To this end, they define the variable

$$
\begin{aligned}
\hat{\tau}_t &= \ln\!\left(\Phi_t / \Phi_{t-1}\right) - \tilde{\gamma} \\
&= \frac{1}{1 - \alpha} \Delta \varepsilon_t^a \\
&= \frac{1}{1 - \alpha}\left[(\rho_a - 1)\varepsilon_{t-1}^a + \sigma_a \eta_t^a\right].
\end{aligned}
\tag{2.35}
$$

The $\hat{\tau}_t$ variable is serially correlated when the TFP process is stationary, and white noise when the TFP process has a unit root.

---

[6] For the YADA example of the Smets and Wouters model, the data on the federal funds rate has been redefined into annual terms.

The log-linearized aggregate resource constraint of this closed economy model is now given by

$$\hat{y}_t = c_y \hat{c}_t + i_y \hat{i}_t + z_y \hat{z}_t - \mathbb{I}(\rho_a < 1)\frac{1}{1-\alpha}\varepsilon_t^a + \varepsilon_t^g, \qquad (2.36)$$

where $\mathbb{I}(\rho_a < 1)$ is an indicator function which is unity if $\rho_a < 1$ and zero otherwise; cf. equation (2.17). The steady-state ratios in (2.36) are the same as those in the original Smets and Wouters model.

The consumption Euler equation is now

$$\hat{c}_t = c_1\left(\hat{c}_{t-1} - \hat{\tau}_t\right) + (1 - c_1) E_t\left[\hat{c}_{t+1} + \hat{\tau}_{t+1}\right] + c_2\left(\hat{l}_t - E_t\hat{l}_{t+1}\right) - c_3\left(\hat{r}_t - E_t\hat{\pi}_{t+1}\right) + \varepsilon_t^b, \quad (2.37)$$

where the parameters c_i, $i = 1, 2, 3$, are given below equation (2.18).

Similarly, the investment Euler equation is

$$\hat{i}_t = i_1\left(\hat{i}_{t-1} - \hat{\tau}_t\right) + (1 - i_1) E_t\left[\hat{i}_{t+1} + \hat{\tau}_{t+1}\right] + i_2\hat{q}_t + \varepsilon_t^i, \qquad (2.38)$$

where i_1 and i_2 are specified below equation (2.19). The value of the capital stock evolves according to the equation (2.20), with the parameter q_1 unchanged.

The aggregate production function is now expressed as

$$\hat{y}_t = \phi_p\left[\alpha\hat{k}_t^s + (1 - \alpha)\hat{l}_t\right] + \mathbb{I}(\rho_a < 1)(\phi_p - 1)\frac{1}{1-\alpha}\varepsilon_t^a, \qquad (2.39)$$

while capital services is given by

$$\hat{k}_t^s = \hat{k}_{t-1} + \hat{z}_t - \hat{\tau}_t, \qquad (2.40)$$

and the degree of capital utilization is given by equation (2.23).

The equation determining the evolution of installed capital is given by

$$\hat{k}_t = k_1\left(\hat{k}_{t-1} - \hat{\tau}_t\right) + (1 - k_1)\hat{i}_t + k_2\varepsilon_t^i, \qquad (2.41)$$

while k_1 and k_2 are defined below equation (2.24).

The price markup is like in Section 2.4 equal to minus the real marginal cost under cost minimization. For the alternate detrending method we have that the price markup is

$$\hat{\mu}_t^p = \alpha\left(\hat{k}_t^s - \hat{l}_t\right) - \hat{w}_t, \qquad (2.42)$$

while the real marginal cost is

$$\hat{\mu}_t^c = \alpha\hat{r}_t^k + (1 - \alpha)\hat{w}_t. \qquad (2.43)$$

Notice that the TFP shock is missing from both these equations. Setting (2.42) equal to minus (2.43) and solving for the rental rate on capital we find that equation (2.28) holds.

The price Phillips curve is unaffected by the change in detrending method and is given by equation (2.27), with the parameters π_i, $i = 1, 2, 3$, also being unchanged. The real wage markup is now given by

$$\hat{\mu}_t^w = \hat{w}_t - \left(\sigma_l\hat{l}_t + \frac{1}{1 - (\lambda/\gamma)}\left[\hat{c}_t - \frac{\lambda}{\gamma}\left(\hat{c}_{t-1} - \hat{\tau}_t\right)\right]\right), \qquad (2.44)$$

i.e., the real wage minus the households' marginal rate of substitution between consumption and labor, while real wages are now determined by

$$\hat{w}_t = w_1\left(\hat{w}_{t-1} - \hat{\tau}_t\right) + (1 - w_1) E_t\left[\hat{w}_{t+1} + \hat{\tau}_{t+1} + \hat{\pi}_{t+1}\right] - w_2\hat{\pi}_t + w_3\hat{\pi}_{t-1} - w_4\hat{\mu}_t^w + \varepsilon_t^w. \quad (2.45)$$

The w_i ($i = 1, 2, 3, 4$) parameters are given by the expressions below equation (2.30). Finally, the monetary policy reaction function is given by (2.31).

The equations of the flexible price and wage economy

$$\hat{y}_t^f = c_y \hat{c}_t^f + i_y \hat{i}_t^f + z_y \hat{z}_t^f - \mathbb{I}(\rho_a < 1)(1-\alpha)^{-1} \varepsilon_t^a + \varepsilon_t^g,$$

$$\hat{c}_t^f = c_1 (\hat{c}_{t-1}^f - \hat{\tau}_t) + (1-c_1) E_t [\hat{c}_{t+1}^f + \hat{\tau}_{t+1}] + c_2 (\hat{l}_t^f - E_t \hat{l}_{t+1}^f) - c_3 \hat{r}_t^f + \varepsilon_t^b,$$

$$\hat{i}_t^f = i_1 (\hat{i}_{t-1}^f - \hat{\tau}_t) + (1-i_1) E_t [\hat{i}_{t+1}^f + \hat{\tau}_{t+1}] + i_2 \hat{q}_t^f + \varepsilon_t^i,$$

$$\hat{q}_t^f = q_1 E_t \hat{q}_{t+1}^f + (1-q_1) E_t \hat{r}_{t+1}^{k,f} - \hat{r}_t^f + c_3^{-1} \varepsilon_t^b,$$

$$\hat{y}_t^f = \phi_p \left[\alpha \hat{k}_t^{s,f} + (1-\alpha) \hat{l}_t^f \right] + \mathbb{I}(\rho_a < 1)(\phi_p - 1)(1-\alpha)^{-1} \varepsilon_t^a,$$

$$\hat{k}_t^{s,f} = \hat{k}_{t-1}^f + \hat{z}_t^f - \hat{\tau}_t, \qquad\qquad (2.46)$$

$$\hat{z}_t^f = z_1 \hat{r}_t^{k,f},$$

$$\hat{k}_t^f = k_1 (\hat{k}_{t-1}^f - \hat{\tau}_t) + (1-k_1) \hat{i}_t^f + k_2 \varepsilon_t^i,$$

$$0 = \alpha \hat{r}_t^{k,f} + (1-\alpha) \hat{w}_t^f,$$

$$\hat{r}_t^{k,f} = - \left(\hat{k}_t^{s,f} - \hat{l}_t^f \right) + \hat{w}_t^f,$$

$$\hat{w}_t^f = \sigma_l \hat{l}_t^f + \frac{1}{1-(\lambda/\gamma)} \left[\hat{c}_t^f - \frac{\lambda}{\gamma} (\hat{c}_{t-1}^f - \hat{\tau}_t) \right],$$

where $\hat{r}_t^f$ is the real interest rate of the flexible price and wage economy.

The steady-state values of the model as shown in Section 2.4.4 remain valid. The measurement equations change slightly due to the different detrending method. Specifically, we now obtain

$$\begin{bmatrix} \Delta y_t \\ \Delta c_t \\ \Delta i_t \\ \Delta w_t \\ l_t \\ \pi_t \\ r_t \end{bmatrix} = \begin{bmatrix} \bar{\gamma} \\ \bar{\gamma} \\ \bar{\gamma} \\ \bar{\gamma} \\ \bar{l} \\ \bar{\pi} \\ 4\bar{r} \end{bmatrix} + \begin{bmatrix} \hat{y}_t - \hat{y}_{t-1} + \hat{\tau}_t \\ \hat{c}_t - \hat{c}_{t-1} + \hat{\tau}_t \\ \hat{i}_t - \hat{i}_{t-1} + \hat{\tau}_t \\ \hat{w}_t - \hat{w}_{t-1} + \hat{\tau}_t \\ \hat{l}_t \\ \hat{\pi}_t \\ 4\hat{r}_t \end{bmatrix}, \qquad\qquad (2.47)$$

where $\bar{\gamma}$, $\bar{\pi}$, and $\bar{r}$ are shown below equation (2.34).

2.5.1. *A Small-Scale Version of the Smets and Wouters Model*

Del Negro and Schorfheide (2013) presents a smal-scale version of the Smets and Wouters model in Section 2.5 by removing a number of features, such as capital accumulation (with $\alpha = 0$). After setting $\lambda = 0$ and eliminating the ε_t^b shock, the consumption Euler equation simplifies to

$$\hat{c}_t = E_t [\hat{c}_{t+1} + \hat{\tau}_{t+1}] + \frac{\sigma_c - 1}{\sigma_c} \left(\hat{l}_t - E_t \hat{l}_{t+1} \right) - \frac{1}{\sigma_c} (\hat{r}_t - E_t \hat{\pi}_{t+1}), \qquad (2.48)$$

Notice that hours appears in this formulation when $\sigma_c \neq 1$ and this is based on a unit steady-state wage.[7]

It is also assumed that there is no wage stickiness in the small-scale model. Consequently, the marginal cost is equal to the real wage, and the latter is equal to the marginal rate of substitution between consumption and leisure. In the absence of fixed costs ($\phi_p = 1$) the aggregate production function implies that output equals hours, $\hat{y}_t = \hat{l}_t$, with the results that

$$\hat{\mu}_t^c = \hat{c}_t + \sigma_l \hat{y}_t. \qquad\qquad (2.49)$$

---

[7] This term is missing in equation (22) of Del Negro and Schorfheide (2013). The unit steady-state wage follows from $\phi_w = 1$ and $c_y = 1$ when $\alpha = 0$.

With zero price markup shocks, the price Phillips curve is otherwise given by equation (2.27), noting that the denominator of π_3 simplifies since $\phi_p = 1$. That is,

$$
\hat{\pi}_t = \frac{\iota_p}{1 + \beta\gamma^{1-\sigma_c}\iota_p}\hat{\pi}_{t-1} + \frac{\beta\gamma^{1-\sigma_c}}{1 + \beta\gamma^{1-\sigma_c}\iota_p}E_t\hat{\pi}_{t+1}
$$
$$
+ \frac{\left(1 - \xi_p\right)\left(1 - \beta\gamma^{1-\sigma_c}\xi_p\right)}{\left(1 + \beta\gamma^{1-\sigma_c}\iota_p\right)\xi_p}\hat{\mu}_t^c. \tag{2.50}
$$

Compared with equation (24) in Del Negro and Schorfheide (2013) we here use the term $\beta\gamma^{1-\sigma_c}$ rather than just β, which would only appear if $\sigma_c = 1$.

The monetary policy rule is slightly altered by Del Negro and Schorfheide (2013), where the central bank only reacts to inflation and output growth, while the monetary policy shock is iid. It is also assumed below that the central bank only reacts to inflation and output growth, but the original assumption that monetary policy shocks follow an AR(1) process is retainted. That is,

$$
\hat{r}_t = \rho\hat{r}_{t-1} + \left(1 - \rho\right)\left[r_\pi\hat{\pi}_t + r_y\left(\Delta\hat{y}_t + \hat{\tau}_t\right)\right] + \varepsilon_t^r. \tag{2.51}
$$

The aggregate resource constraint now simplifies to

$$
\hat{y}_t = \hat{c}_t + \varepsilon_t^g, \tag{2.52}
$$

which involves a slight redefinition of the government spending shock since the TFP shock term is missing; cf. equation (2.36). All shocks in this model follow AR(1) processes, including ε_t^a which determines $\hat{\tau}_t$, and the government spending shock ε_t^g.

Finally, the measurement equations are given by

$$
\begin{bmatrix} \Delta y_t \\ \pi_t \\ r_t \end{bmatrix} = \begin{bmatrix} \bar{\gamma} \\ \bar{\pi} \\ 4\bar{r} \end{bmatrix} + \begin{bmatrix} \hat{y}_t - \hat{y}_{t-1} + \hat{\tau}_t \\ \hat{\pi}_t \\ 4\hat{r}_t \end{bmatrix}. \tag{2.53}
$$

The steady-state parameters are equal to those shown below equation (2.34).

2.6. *The Smets and Wouters Model with Financial Frictions*

Del Negro and Schorfheide (2013) and Del Negro, Giannoni, and Schorfheide (2015) introduce financial frictions into their variant of the Smets and Wouters models based on the financial accelerator approach of Bernanke, Gertler, and Gilchrist (1999); see also Christiano, Motto, and Rostagno (2003, 2010) and De Graeve (2008). This amounts to replacing the value of the capital stock equation in (2.20) with

$$
E_t\hat{r}_{t+1}^e - \hat{r}_t = \zeta_{sp,b}\left(\hat{q}_t + \hat{k}_t - \hat{n}_t\right) - c_3^{-1}\varepsilon_t^b + \varepsilon_t^e, \tag{2.54}
$$

and

$$
\hat{r}_t^e - \hat{\pi}_t = \left(1 - q_1\right)\hat{r}_t^k + q_1\hat{q}_t - \hat{q}_{t-1}, \tag{2.55}
$$

where $\hat{r}_t^e$ is the gross return on capital for entrepreneurs, $\hat{n}_t$ is entrepreneurial equity (net worth), and ε_t^e captures mean-preserving changes in the cross-sectional dispersion of ability across entrepreners, a spread shock. The parametes q_1 is here given by

$$
q_1 = \frac{1 - \delta}{r^k + 1 - \delta},
$$

where r^k is generally *not* equal to $\beta^{-1}\gamma^{\sigma_c} + \delta - 1$ since the spread between gross returns on capital for entrepreneurs and the nominal interest rate need not zero. The spread shock is assumed to follows the AR(1) process

$$
\varepsilon_t^e = \rho_e\varepsilon_{t-1}^e + \sigma_e\eta_t^e. \tag{2.56}
$$

The parameters $\zeta_{sp,b}$ is the steady-state elasticity of the spread with respect to leverage. It may be noted that if $\zeta_{sp,b} = \sigma_e = 0$, then the financial frictions are completely shut down and equations (2.54) and (2.55) yield the original value of the capital stock equation (2.20).

The log-linearized net worth of entrepreneurs equation is given by

$$\hat{n}_t + \zeta_{n,\tau}\hat{\tau}_t = \zeta_{n,e}\left(\hat{r}_t^e - \hat{\pi}_t\right) - \zeta_{n,r}\left(\hat{r}_{t-1} - \hat{\pi}_t\right) + \zeta_{n,q}\left(\hat{q}_{t-1} + \hat{k}_{t-1}\right) + \zeta_{n,n}\hat{n}_{t-1} - \frac{\zeta_{n,\sigma_\omega}}{\zeta_{sp,\sigma_\omega}}\varepsilon_{t-1}^e, \quad (2.57)$$

where $\zeta_{n,e}$, $\zeta_{n,r}$, $\zeta_{n,q}$, $\zeta_{n,n}$, and ζ_{n,σ_ω} are the steady-state elasticities of net worth with respect to the return on capital for entrepreneurs, the nominal interest rate, the cost of capital, net worth, and the volatility of the spread shock. Furthermore, $\zeta_{n,\tau}$ is the steady-state elasticity with respect to the TFP shock based trend variable $\hat{\tau}$ and ζ_{sp,σ_ω} is the steady-state elasticity of the spread with respect to the spread shock.

It is typically assumed for the flexible price and wage economy that there are no financial frictions; see, e.g., De Graeve (2008). Accordingly, net worth is zero, the gross real return on capital of entrepreneurs is equal to the real rate $\hat{r}_t^f$, while the value of the capital stock evolves according to

$$\hat{q}_t^f = q_1 E_t\hat{q}_{t+1}^f + \left(1 - q_1\right)E_t\hat{r}_{t+1}^{k,f} - \hat{r}_t^f + c_3^{-1}\varepsilon_t^b.$$

Since the steady-state real rental rate on capital in the flexible price and wage economy without financial frictions is equal to the real interest rate, an option regarding the q_1 parameter in this equation is to set it equal to $q_{1,f} = \beta\gamma^{-\sigma_c}(1 - \delta)$. Formally, this appears to be the correct choice since the steady-state of the flexible price and wage and frictionless economy differs from the steady-state of the economic with sticky prices and wages and financial frictions. However, if the steady-state spread of the former economy, r^e/r, is close to unity, then q_1 and $q_{1,f}$ will be approximately equal.

2.6.1. *Augmented Measurement Equations*

The set of measurement equations is augmented in Del Negro and Schorfheide (2013) by

$$s_t = 4\bar{s} + 4E_t\left[\hat{r}_{t+1}^e - \hat{r}_t\right], \quad (2.58)$$

where $\bar{s}$ is equal to the natural logarithm of the steady-state spread measured in quarterly terms and in percent, $\bar{s} = 100\ln(r^e/r)$, while the spread variable, s_t, is measured as the annualized Moody's seasoned Baa corporate bond yield spread over the 10-year treasury note yield at constant maturity by Del Negro and Schorfheide (2013) and Del Negro et al. (2015). The parameter s is linked to the steady-state values of the model variable variables according to

$$\frac{r^e}{r} = (1 + s/100)^{1/4}, \quad \frac{r}{\pi} = \beta^{-1}\gamma^{\sigma_c}, \quad \frac{r^e}{\pi} = r^k + 1 - \delta.$$

If there are no financial frictions, then $r^e = r$, with the consequence that the steady-state real interest rate is equal to the steady-state real rental rate on capital plus one minus the depreciation rate of capital, i.e., the Smets and Wouters steady-state value; see Section 2.4.4.

Del Negro and Schorfheide (2013) estimate s, $\zeta_{sp,b}$, ρ_e, and σ_e while the parameters $\bar{F}$ and κ^e are calibrated. The latter two parameters will appear in the next Section on the steady-state, but it is useful to know that they represent the steady-state default probability and survival rate of entrepreneurs, respectively, with $\bar{F}$ determined such that in annual terms the default probability is 0.03 (0.0075 in quarterly terms) and $\kappa^e = 0.99$. These values are also used by Del Negro et al. (2015) as well as in the DSGE model of the Federal Reserve Bank of New York; see Del Negro, Eusepi, Giannoni, Sbordone, Tambalotti, Cocci, Hasegawa, and Linder (2013).

2.6.2. *The Steady-State in the Smets and Wouters Model with Financial Frictions*

2.6.2.1. *Preliminaries: The Log-Normal Distribution*

To determine the relationship between the elasticities in equations (2.54) and (2.57) and the steady-state of the model with financial frictions it is important to first sort out some details concerning an ideosyncratic shock which affects entrepreneurs' capital. The distribution of the shock in time t is known before it is realized and this distribution is the same for all entrepreneurs, while the shock is iid across entrepreneurs and over time. In steady state, the natural logarithm of the shock, denoted by $\ln\omega$, is normal with mean m_ω and variance σ_ω^2, where it is assumed that the expected value of ω is unity. Since its distribution is log-normal

(cf. Section 4.2.6) with $\mu_\omega = \exp(m_\omega + (1/2)\sigma_\omega^2) = 1$, it follows that $m_\omega = -(1/2)\sigma_\omega^2$. The probability density function (pdf) of ω is consequently

$$p(\omega) = \frac{1}{\sqrt{2\pi\sigma_\omega^2}}\omega^{-1}\exp\left(-\frac{1}{2\sigma_\omega^2}\left(\ln\omega + (1/2)\sigma_\omega^2\right)^2\right),$$

while the cumulative distribution function (cdf) is

$$F(\omega) = \Phi\left(\frac{\ln\omega + (1/2)\sigma_\omega^2}{\sigma_\omega}\right),$$

where $\Phi(z)$ is the cdf of the normal distribution. For values of ω below a threshold, $\bar\omega$, the enrepreneur defaults on its debt.[8]

The expected value of ω conditional on it being greater than $\bar\omega$ is given by

$$E[\omega|\omega > \bar\omega] = \int_{\bar\omega}^\infty \omega p(\omega)d\omega = \Phi\left(\frac{(1/2)\sigma_\omega^2 - \ln\bar\omega}{\sigma_\omega}\right)$$

$$= 1 - \Phi\left(\frac{\ln\bar\omega - (1/2)\sigma_\omega^2}{\sigma_\omega}\right),$$

where the last equality follows from $\Phi(z) = 1 - \Phi(-z)$, i.e. that the normal distribution is symmetric. We then find that

$$G(\bar\omega) = \int_0^{\bar\omega}\omega p(\omega)d\omega = \int_0^\infty \omega p(\omega)d\omega - \int_{\bar\omega}^\infty \omega p(\omega)d\omega$$

$$= \Phi\left(\frac{\ln\bar\omega - (1/2)\sigma_\omega^2}{\sigma_\omega}\right).$$

Furthermore, let

$$\Gamma(\bar\omega) = \int_0^{\bar\omega}\omega p(\omega)d\omega + \bar\omega\int_{\bar\omega}^\infty p(\omega)d\omega$$

$$= \bar\omega\left[1 - \Phi\left(\frac{\ln\bar\omega + (1/2)\sigma_\omega^2}{\sigma_\omega}\right)\right] + G(\bar\omega).$$

Defining

$$z^\omega = \frac{\ln\bar\omega + (1/2)\sigma_\omega^2}{\sigma_\omega},$$

we obtain

$$G(\bar\omega) = \Phi(z^\omega - \sigma_\omega), \quad \Gamma(\bar\omega) = \bar\omega\left[1 - \Phi(z^\omega)\right] + \Phi(z^\omega - \sigma_\omega). \tag{2.59}$$

Let $\phi(z)$ be the standard normal pdf such that $\partial\Phi(z)/\partial z = \phi(z)$; cf. Section 4.2.6. From the expression for this density it follows that

$$\phi(z^\omega - \sigma_\omega) = \bar\omega\phi(z^\omega), \quad \phi'(z) = \frac{\partial\phi(z)}{\partial z} = -z\phi(z), \quad \frac{\partial z^\omega}{\partial\bar\omega} = -\frac{1}{\bar\omega\sigma_\omega}.$$

Making use of these two results it can be shown that

$$G'(\bar\omega) = -\frac{1}{\sigma_\omega}\phi(z^\omega), \quad G''(\bar\omega) = -\frac{z^\omega}{\bar\omega\sigma_\omega^2}\phi(z^\omega), \tag{2.60}$$

where $G'(\bar\omega) = \partial G(\bar\omega)/\partial\bar\omega$ and $G''(\bar\omega) = \partial^2 G(\omega)/\partial\bar\omega^2$. In addition,

$$\Gamma'(\bar\omega) = \frac{\Gamma(\bar\omega) - G(\bar\omega)}{\bar\omega} = 1 - \Phi(z^\omega), \quad \Gamma''(\bar\omega) = -\frac{1}{\bar\omega\sigma_\omega}\phi(z^\omega), \tag{2.61}$$

---

[8] The entrepreneurs are assumed to be gender free in this economy and supposedly multiply by splitting into two after having too much curry.

where $\Gamma'(\bar{\omega}) = \partial\Gamma(\bar{\omega})/\partial\bar{\omega}$ and $\Gamma''(\bar{\omega}) = \partial^2\Gamma(\omega)/\partial\bar{\omega}^2$. We will also have use for the following derivatives

$$G_{\sigma_\omega}(\bar{\omega}) = -\frac{z^\omega}{\sigma_\omega}\phi(z^\omega - \sigma_\omega), \quad \Gamma_{\sigma_\omega}(\bar{\omega}) = -\phi(z^\omega - \sigma_\omega), \tag{2.62}$$

where $G_{\sigma_\omega}(\bar{\omega}) = \partial G(\bar{\omega})/\partial\sigma_\omega$ and $\Gamma_{\sigma_\omega}(\bar{\omega}) = \partial\Gamma(\bar{\omega})/\partial\sigma_\omega$ and we have made use of

$$\frac{\partial z^\omega}{\partial\sigma_\omega} = 1 - \frac{z^\omega}{\sigma_\omega}.$$

Finally, it can be shown that

$$G'_{\sigma_\omega}(\bar{\omega}) = -\frac{\phi(z^\omega)}{\sigma_\omega^2}\left[1 - z^\omega(z^\omega - \sigma_\omega)\right], \quad \Gamma'_{\sigma_\omega}(\bar{\omega}) = \left(\frac{z^\omega}{\sigma_\omega} - 1\right)\phi(z^\omega), \tag{2.63}$$

where $G'_{\sigma_\omega}(\bar{\omega}) = \partial^2 G(\bar{\omega})/\partial\bar{\omega}\partial\sigma_\omega$ and $\Gamma'_{\sigma_\omega}(\bar{\omega}) = \partial^2\Gamma(\bar{\omega})/\partial\bar{\omega}\partial\sigma_\omega$.

2.6.2.2. Steady-State Elasticities

The steady-state default probability, $\bar{F}$, is assumed to be calibrated at 0.03 (0.0075) in annual (quarterly) terms by Del Negro and Schorfheide (2013). This means that

$$F(\bar{\omega}) = \bar{F}, \tag{2.64}$$

or

$$z^\omega = \frac{\ln\bar{\omega} + (1/2)\sigma_\omega^2}{\sigma_\omega} = \Phi^{-1}(\bar{F}), \tag{2.65}$$

where $\Phi^{-1}(\cdot)$ is the inverted normal. With $\bar{F}$ being known it follows that z^ω is known and that $\bar{\omega}$ may be treated as a function of σ_ω. That is,

$$\bar{\omega}(\sigma_\omega) = \exp(z^\omega\sigma_\omega - (1/2)\sigma_\omega^2). \tag{2.66}$$

Del Negro and Schorfheide (2013) suggest that we may solve for the steady-state value of σ_ω from

$$\zeta_{sp,b} = -\frac{\zeta_{b,\bar{\omega}}/\zeta_{z,\bar{\omega}}}{1 - (\zeta_{b,\bar{\omega}}/\zeta_{z,\bar{\omega}})}\frac{n/k}{1 - (n/k)}, \tag{2.67}$$

where $\zeta_{b,\bar{\omega}}$ and $\zeta_{z,\bar{\omega}}$ are elasticities to be defined below, the parameter $\zeta_{sp,b}$ is estimated, while the steady-state ratio

$$\frac{n}{k}(\bar{\omega}) = 1 - \left[\Gamma(\bar{\omega}) - \mu^e G(\bar{\omega})\right]\frac{r^e}{r}, \tag{2.68}$$

and

$$\mu^e(\bar{\omega}) = \frac{1 - (r/r^e)}{\left(G'(\bar{\omega})/\Gamma'(\bar{\omega})\right)\left[1 - \Gamma(\bar{\omega})\right] + G(\bar{\omega})}. \tag{2.69}$$

It may be noted that $(n/k)/[1 - (n/k)]$ is equal to the inverse of steady-state leverage (where leverage is denoted by ϱ in the Technical Appendix to Del Negro and Schorfheide, 2013), while μ^e is related to the steady-state bankruptcy costs.[9] Furthermore, the ratio r^e/r is determined from the estimate of s; see below equation (2.58).

In the Technical Appendix to Del Negro and Schorfheide (2013) it is shown that

$$\zeta_{b,\bar{\omega}} = \frac{\bar{\omega}\mu^e(n/k)\left[\Gamma''(\bar{\omega})G'(\bar{\omega}) - G''(\bar{\omega})\Gamma'(\bar{\omega})\right]}{\left[\Gamma'(\bar{\omega}) - \mu^e G'(\bar{\omega})\right]^2\left[1 - \Gamma(\bar{\omega}) + \Gamma'(\bar{\omega})\dfrac{\Gamma(\bar{\omega}) - \mu^e G(\bar{\omega})}{\Gamma'(\bar{\omega}) - \mu^e G'(\bar{\omega})}\right](r^e/r)}, \tag{2.70}$$

while

$$\zeta_{z,\bar{\omega}} = \frac{\bar{\omega}\left[\Gamma'(\bar{\omega}) - \mu^e G'(\bar{\omega})\right]}{\Gamma(\bar{\omega}) - \mu^e G(\bar{\omega})}, \tag{2.71}$$

---

[9] The bank monitors the entrepreneurs and extracts a fraction $(1 - \mu^e)$ of its revenues given that the bank assumes the draw of ω is below the threshold $\bar{\omega}$. The remaining fraction μ^e may therefore be regarded as the bankruptcy costs.

while n/k and μ^e are given by (2.68) and (2.69), respectively. Plugging the expression for $\zeta_{b,\bar{\omega}}$, $\zeta_{z,\bar{\omega}}$, μ^e, and n/k into equation (2.67) and making use of (2.66) it is possible to solve for σ_ω numerically using nonlinear methods, such as with the `fzero` function in Matlab.

The elasticities of the net worth equation can now be computed from the parameters of the model. The spread elasticity with respect to the spread shock is

$$\zeta_{sp,\sigma_\omega} = \frac{(\zeta_{b,\bar{\omega}}/\zeta_{z,\bar{\omega}})\zeta_{z,\sigma_\omega} - \zeta_{b,\sigma_\omega}}{1 - (\zeta_{b,\bar{\omega}}/\zeta_{z,\bar{\omega}})}, \tag{2.72}$$

where

$$\zeta_{b,\sigma_\omega} = \frac{\sigma_\omega\left[A(\bar{\omega}) + B(\bar{\omega})\right]}{C(\bar{\omega})}, \tag{2.73}$$

with

$$A(\bar{\omega}) = \left[\frac{1 - \mu^e\big(G_{\sigma_\omega}(\bar{\omega})/\Gamma_{\sigma_\omega}(\bar{\omega})\big)}{1 - \mu^e\big(G'(\bar{\omega})/\Gamma'(\bar{\omega})\big)} - 1\right]\Gamma_{\sigma_\omega}(\bar{\omega})(r^e/r),$$

$$B(\bar{\omega}) = \frac{\mu^e(n/k)\left[G'(\bar{\omega})\Gamma'_{\sigma_\omega}(\bar{\omega}) - \Gamma'(\bar{\omega})G'_{\sigma_\omega}(\bar{\omega})\right]}{\left[\Gamma'(\bar{\omega}) - \mu^e G'(\bar{\omega})\right]^2},$$

$$C(\bar{\omega}) = \left[1 - \Gamma(\bar{\omega})\right](r^e/e) + \frac{\Gamma'(\bar{\omega})}{\Gamma'(\bar{\omega}) - \mu^e G'(\bar{\omega})}\left[1 - (n/k)\right],$$

while

$$\zeta_{z,\sigma_\omega} = \frac{\sigma_\omega\left[\Gamma_{\sigma_\omega}(\bar{\omega}) - \mu^e G_{\sigma_\omega}(\bar{\omega})\right]}{\Gamma(\bar{\omega}) - \mu^e G(\bar{\omega})}. \tag{2.74}$$

Turning to the remaining elasticities of the net worth equation, it is shown in the Technical Appendix of Del Negro and Schorfheide (2013) that

$$\zeta_{n,e} = \kappa^e\frac{r^e}{\pi\gamma}(n/k)^{-1}\left(1 - \mu^e G(\bar{\omega})\left[1 - (\zeta_{G,\bar{\omega}}/\zeta_{z,\bar{\omega}})\right]\right), \tag{2.75}$$

where

$$\zeta_{G,\bar{\omega}} = \frac{\bar{\omega}G'(\bar{\omega})}{G(\bar{\omega})}.$$

Next,

$$\zeta_{n,r} = (\kappa^e/\beta)(n/k)^{-1}\left(1 - (n/k) + \mu^e G(\bar{\omega})(r^e/r)\left[1 - (\zeta_{G,\bar{\omega}}/\zeta_{z,\bar{\omega}})\right]\right), \tag{2.76}$$

$$\zeta_{n,q} = \kappa^e\frac{r^e}{\pi\gamma}(n/k)^{-1}\left[1 - \mu^e G(\bar{\omega})\left(1 + \frac{\zeta_{G,\bar{\omega}}(n/k)}{\zeta_{z,\bar{\omega}}\left[1 - (n/k)\right]}\right)\right] - (\kappa^e/\beta)(n/k)^{-1}, \tag{2.77}$$

$$\zeta_{n,n} = (\kappa^e/\beta) + \kappa^e\frac{r^e}{\pi\gamma}\mu^e G(\bar{\omega})\frac{\zeta_{G,\bar{\omega}}}{\zeta_{z,\bar{\omega}}\left[1 - (n/k)\right]}. \tag{2.78}$$

$$\tag{2.79}$$

Furthermore,

$$\zeta_{n,\tau} = \kappa^e\frac{v}{n}, \tag{2.80}$$

where v is steady-state entrepreneurs' equity. From the Technical Appendix of Del Negro and Schorfheide (2013) we find that

$$v = \frac{n - w^e}{\kappa^e}, \tag{2.81}$$

$$\frac{w^e}{k} = \left(1 - \frac{\kappa^e}{\beta}\right)\frac{n}{k} - \frac{\kappa^e}{\beta}\left[\frac{r^e}{r}\big(1 - \mu^e G(\bar{\omega})\big) - 1\right], \tag{2.82}$$

where w^e is the steady-state net worth transfer received by new entrepreneurs. Combining equations (2.80)–(2.82) it can be shown that

$$\zeta_{n,\tau} = \left(\kappa^e/\beta\right) + \left(\kappa^e/\beta\right)(n/k)^{-1}\left[\frac{r^e}{r}\left(1 - \mu^e G(\bar{\omega})\right) - 1\right]. \tag{2.83}$$

Finally,

$$\zeta_{n,\sigma_\omega} = \kappa^e \mu^e G(\bar{\omega})\frac{r^e}{\pi\gamma}(n/k)^{-1}\left[\zeta_{G,\sigma_\omega} - \left(\zeta_{G,\bar{\omega}}/\zeta_{z,\bar{\omega}}\right)\zeta_{z,\sigma_\omega}\right], \tag{2.84}$$

where

$$\zeta_{G,\sigma_\omega} = \frac{\sigma_\omega G_{\sigma_\omega}(\bar{\omega})}{G(\bar{\omega})}.$$

In case we instead add the BGG-type of financial frictions[10] to the variant of the Smets and Wouters model in Section 2.4, the variable $\hat{\tau}_t$ simply drops out of the net worth equation (2.57). The equation for the spread between expected gross return on capital for entrepreneurs and the short-term nominal interest rate (2.54) and the equation for the real gross return on capital for entrepreneurs (2.55) remain unchanged.

2.7. *The Smets and Wouters Model with Unemployment*

The Galí, Smets, and Wouters (2012, GSW) model is an extension of the standard Smets and Wouters model which explixitly provides a mechanism for explaining unemployment. This is accomplished by modelling the labor supply decisions on the extensive margin (whether to work or not) rather than on the intensive margin (how many hours to work). As a consequence, the unemployment rate is added as an observable variable, while labor supply shocks are allowed for. This allows the authors to overcome the lack of identification of wage markup and labor supply shocks raised by Chari, Kehoe, and McGrattan (2009) in their critique of new Keynesian models. From a technical perspective the GSW model is also based on the assumption of log-utility, i.e. the parameter σ_c is assumed to be unity.

Smets, Warne, and Wouters (2014) present a variant of the GSW model aimed for the euro area and this version is presented below. The log-linearized aggregate resource constraint is given by equation (2.17). The consumption Euler equation may be expressed as

$$\hat{\lambda}_t = E_t\hat{\lambda}_{t+1} + \left(\hat{r}_t - E_t\hat{\pi}_{t+1} - \epsilon_t^b\right), \tag{2.85}$$

where $\hat{\lambda}_t$ is the log-linearized marginal utility of consumption, given by

$$\hat{\lambda}_t = -\frac{1}{1 - (\lambda/\gamma)}\left[\hat{c}_t - \frac{\lambda}{\gamma}\hat{c}_{t-1}\right], \tag{2.86}$$

while ϵ_t^b is a risk premium shock. Making use of (2.85) and (2.86), the consumption Euler equation can be written in the more familiar form

$$\hat{c}_t = c_1\hat{c}_{t-1} + \left(1 - c_1\right)E_t\hat{c}_{t+1} - c_3\left(\hat{r}_t - E_t\hat{\pi}_{t+1} - \epsilon_t^b\right), \tag{2.87}$$

where c_1 and c_3 are given by the expressions below equation (2.18), while $c_2 = 0$ since $\sigma_c = 1$. Notice that the risk premium shock in equation (2.87) satisfies $\epsilon_t^b = c_3^{-1}\varepsilon_t^b$ in equation (2.18).[11]

The investment Euler equation is given by equation (2.19), where $i_2 = 1/(1 + \beta)$ and $i_2 = 1/((1 + \beta)\gamma^2\varphi)$ since $\sigma_c = 1$. The value of the capital stock equation can be expressed in terms of the risk premium shock rather than the intertemporal substitution preference shock and is therefore written as

$$\hat{q}_t = q_1 E_t\hat{q}_{t+1} + \left(1 - q_1\right)E_t\hat{r}_{t+1}^k - \left(\hat{r}_t - E_t\hat{\pi}_{t+1} - \epsilon_t^b\right). \tag{2.88}$$

---

[10] BGG refers to Bernanke, Gertler, and Gilchrist (1999).

[11] Notice that the sign of ϵ_t^b in (2.87) differs from the sign of the risk premium shock in the consumption equation (2) of Smets and Wouters (2007).

The aggregate production function is given by equation (2.21), while the capital service variable is determined by (2.22), capital utilization by (2.23), and installed capital by (2.24), where $k_2 = (\gamma + \delta - 1)(1 + \beta)\gamma\varphi$.

Similarly, the average price markup and the real marginal cost variables are given by equation (2.25) and (2.26), respectively, while the rental rate of capital is determined by equation (2.28).

The log-linearized price Phillips curve is expressed as

$$\hat{\pi}_t = \pi_1 \hat{\pi}_{t-1} + \pi_2 E_t \hat{\pi}_{t+1} - \pi_3 \big(\hat{\mu}_t^p - \hat{\mu}_t^{n,p} \big), \tag{2.89}$$

where the expressions for π_i below equation (2.27) hold, taking into account that $\sigma_c = 1$, and where the natural price markup shock $\hat{\mu}_t^{n,p} = 100\varepsilon_t^p$.

In the GSW model, the wage Phillips curve in equation (2.30) is replaced with the following expression for nominal wage inflation

$$\hat{w}_t - \hat{w}_{t-1} + \hat{\pi}_t = \beta \big(E_t \hat{w}_{t+1} - \hat{w}_t + E_t \hat{\pi}_{t+1} \big) - \beta \iota_w \hat{\pi}_t + \iota_w \hat{\pi}_{t-1}$$
$$- \frac{\big(1 - \xi_w\big)\big(1 - \beta\xi_w\big)}{\xi_w \big(1 + \varepsilon_w \sigma_l\big)} \big[\hat{\mu}_t^w - \hat{\mu}_t^{n,w} \big], \tag{2.90}$$

where $\hat{\mu}_t^w$ is the average wage markup and $\hat{\mu}_t^{n,w}$ is the natural wage markup. Notice that this equation can be rewritten as

$$\hat{w}_t = w_1 \hat{w}_{t-1} + (1 - w_1) \big[E_t \hat{w}_{t+1} + E_t \hat{\pi}_{t+1} \big] - w_2 \hat{\pi}_t + w_3 \hat{\pi}_{t-1} - w_4 \big(\hat{\mu}_t^w - \hat{\mu}_t^{n,w} \big), \tag{2.91}$$

where w_i are given by the expressions below equation (2.30) for $i = 1, 2, 3$ with $\sigma_c = 1$, while

$$w_4 = \frac{\big(1 - \xi_w\big)\big(1 - \beta\xi_w\big)}{\big(1 + \beta\big)\xi_w \big(1 + \varepsilon_w \sigma_l\big)}.$$

In addition, GSW and Smets et al. (2014) let the curvature of the Kimball labor market aggregator be given by

$$\varepsilon_w = \frac{\phi_w}{\phi_w - 1}.$$

The average wage markup is defined as the difference between the real wage and the marginal rate of substitution, which is a function of the smoothed trend in consumption, $\hat{\kappa}_t$, total employment, $\hat{e}_t$, and the labor supply shock. This expression is equal to the elasticity of labor supply times the unemployment rate, i.e.

$$\hat{\mu}_t^w = \hat{w}_t - \big(\hat{\kappa}_t + \varepsilon_t^s + \sigma_l \hat{e}_t \big)$$
$$= \sigma_l \hat{u}_t, \tag{2.92}$$

where unemployment is defined as labor supply minus total employment:

$$\hat{u}_t = \hat{l}_t^s - \hat{e}_t. \tag{2.93}$$

The natural wage markup shock is treated symmetrically with the natural price markup shocks and is expressed as $100\varepsilon_t^w$ and is, in addition, equal to the elasticity of labor supply times the natural rate of unemployment. Accordingly,

$$\hat{\mu}_t^{n,w} = 100\varepsilon_t^w = \sigma_l \hat{u}_t^n \tag{2.94}$$

The natural rate of unemployment, $\hat{u}_t^n$, is defined as the unemployment rate that would prevail in the absence of nominal wage rigidities, and is here proportional to the natural wage markup.

The smoothed trend in consumption is given by

$$\hat{\kappa}_t = \big(1 - \upsilon\big)\hat{\kappa}_{t-1} + \frac{\upsilon}{1 - (\lambda/\gamma)} \left[\hat{c}_t - \frac{\lambda}{\gamma} \hat{c}_{t-1} \right], \tag{2.95}$$

where the parameter υ measures the weight on the marginal utility of consumption under the assumption that $\sigma_c = 1$; cf. equation (2.86). Notice that if $\upsilon = 1$, $\varepsilon_t^s = 0$, and $\hat{u}_t = 0$, the average wage markup in (2.92) is equal to the wage markup in equation (2.29) of the Smets and Wouters model.

Since productivity is written in terms of hours worked, Smets, Warne, and Wouters (2014) follow Smets and Wouters (2003) and introduce an auxiliary equation with Calvo-type rigidity to link observed total employment to unobserved hours worked ($\hat{l}_t$):

$$\hat{e}_t - \hat{e}_{t-1} = \beta\left(E_t\hat{e}_{t+1} - \hat{e}_t\right) + e_1\left(\hat{l}_t - \hat{e}_t\right). \tag{2.96}$$

The parameter e_1 is determined by

$$e_1 = \frac{(1-\xi_e)(1-\beta\xi_e)}{\xi_e},$$

where $1 - \xi_e$ is the fraction of firms that are able to adjust employment to its desired total labor input; see also Adolfson, Laséen, Lindé, and Villani (2005); Adolfson et al. (2007b).

The sticky price and wage part of the GSW model is "closed" by adding the monetary policy reaction function in equation (2.31).

The flexible price and wage equations are obtained by assuming that the natural price and wage markup processes are zero and by setting $\xi_w = \xi_p = 0$ and $\iota_m = \iota_p = 0$, as in Section 2.4.2. Inflation is equal to the steady-state inflation rate and real wages are, as mentioned before, set such that they are equal to the marginal rate of substitution between labor and consumption as well as to the marginal product of labor, with the effect that unemployment is zero. The equations describing the evolution of the flexible price and wage economy are now given by

$$
\begin{aligned}
\hat{y}_t^f &= c_y\hat{c}_t^f + i_y\hat{i}_t^f + z_y\hat{z}_t^f + \varepsilon_t^g, \\
\hat{c}_t^f &= c_1\hat{c}_{t-1}^f + (1-c_1)E_t\hat{c}_{t+1}^f - c_3\left(\hat{r}_t^f - \varepsilon_t^b\right), \\
\hat{i}_t^f &= i_1\hat{i}_{t-1}^f + (1-i_1)E_t\hat{i}_{t+1}^f + i_2\hat{q}_t^f + \varepsilon_t^i, \\
\hat{q}_t^f &= q_1 E_t\hat{q}_{t+1}^f + \left(1-q_1\right)E_t\hat{r}_{t+1}^{k,f} - \left(\hat{r}_t^f - \varepsilon_t^b\right), \\
\hat{y}_t^f &= \phi_p\left[\alpha\hat{k}_t^{s,f} + (1-\alpha)\hat{l}_t^f + \varepsilon_t^a\right], \\
\hat{k}_t^{s,f} &= \hat{k}_{t-1}^f + \hat{z}_t^f, \\
\hat{z}_t^f &= z_1\hat{r}_t^{k,f}, \\
\hat{k}_t^f &= k_1\hat{k}_{t-1}^f + (1-k_1)\hat{i}_t^f + k_2\varepsilon_t^i, \\
\varepsilon_t^a &= \alpha\hat{r}_t^{k,f} + (1-\alpha)\hat{w}_t^f, \\
\hat{r}_t^{k,f} &= -\left(\hat{k}_t^{s,f} - \hat{l}_t^f\right) + \hat{w}_t^f, \\
\hat{w}_t^f &= \sigma_l\hat{e}_t^f + \hat{\kappa}_t^f + \varepsilon_t^s, \\
\hat{\kappa}_t^f &= \left(1-\upsilon\right)\hat{\kappa}_{t-1}^f + \frac{\upsilon}{1-(\lambda/\gamma)}\left[\hat{c}_t^f - \frac{\lambda}{\gamma}\hat{c}_{t-1}^f\right], \\
\hat{e}_t^f &= \hat{e}_{t-1}^f + \beta\left(E_t\hat{e}_{t+1}^f - \hat{e}_t^f\right) + e_1\left(\hat{l}_t^f - \hat{e}_t^f\right).
\end{aligned}
\tag{2.97}
$$

There are eight exogenous processes in the GSW model. These are generally modelled as AR(1) process with the exception of the exogenous spending process (where the process depends on both the exogenous spending shock η_t^g and the total factor productivity shock η_t^a) and the exogenous price and wage markup processes, which are treated as ARMA(1,1) processes.

This means that

$$
\begin{aligned}
\varepsilon_t^g &= \rho_g \varepsilon_{t-1}^g + \sigma_g \eta_t^g + \rho_{ga} \sigma_a \eta_t^a, \\
\varepsilon_t^b &= \rho_b \varepsilon_{t-1}^b + \sigma_b \eta_t^b, \\
\varepsilon_t^i &= \rho_i \varepsilon_{t-1}^i + \sigma_i \eta_t^i, \\
\varepsilon_t^a &= \rho_a \varepsilon_{t-1}^a + \sigma_a \eta_t^a, \\
\varepsilon_t^p &= \rho_p \varepsilon_{t-1}^p + \sigma_p \eta_t^p - \mu_p \sigma_p \eta_{t-1}^p, \\
\varepsilon_t^w &= \rho_w \varepsilon_{t-1}^w + \sigma_w \eta_t^w - \mu_w \sigma_w \eta_{t-1}^w, \\
\varepsilon_t^s &= \rho_s \varepsilon_{t-1}^s + \sigma_s \eta_t^s, \\
\varepsilon_t^r &= \rho_r \varepsilon_{t-1}^r + \sigma_r \eta_t^r.
\end{aligned}
\tag{2.98}
$$

The shocks η_t^j, $j = \{a, b, g, i, p, r, s, w\}$, are iid $N(0,1)$, where η_t^b is the risk premium shock, η_t^i is an investment-specific technology shock, η_t^p is a price markup shock, η_t^r is a monetary policy or interest rate shock, η_t^s is a labor supply, and η_t^w is a wage markup shock.

The steady-state values of the capital-output ratio, etc., are determined as in Section 2.4.4 for the Smets and Wouters model. It should again be kept in mind that $\sigma_c = 1$, such that, e.g., the steady-state rental rate of capital is given by

$$
r^k = \frac{\gamma}{\beta} + \delta - 1.
$$

Since the parameter $c_2 = 0$ it follows that it is not necessary to determine the steady-state hourly wage bill relative to consumption.

The model is consistent with a balanced steady-state growth path, driven by deterministic labor augmenting trend growth. The observed variables for the euro area are given by quarterly data on the log of real GDP (y_t), the log of real private consumption (c_t), the log of real total investment (i_t), the log of the GDP deflator ($p_{y,t}$), the log of real wages (w_t), the log of total employment (e_t), the unemployment rate (u_t), and the short-term nominal interest rate (r_t), represented by the 3-month EURIBOR rate. With all variables except for the unemployment rate and the short-term nominal interest rate being measured in first differences, the measurement equations are given by

$$
\begin{bmatrix}
\Delta y_t \\
\Delta c_t \\
\Delta i_t \\
\Delta w_t \\
\Delta e_t \\
\pi_t \\
u_t \\
r_t
\end{bmatrix}
=
\begin{bmatrix}
\bar{\gamma} + \bar{e} \\
\bar{\gamma} + \bar{e} \\
\bar{\gamma} + \bar{e} \\
\bar{\gamma} \\
\bar{e} \\
\bar{\pi} \\
\bar{u} \\
4\bar{r}
\end{bmatrix}
+
\begin{bmatrix}
\hat{y}_t - \hat{y}_{t-1} \\
\hat{c}_t - \hat{c}_{t-1} \\
\hat{i}_t - \hat{i}_{t-1} \\
\hat{w}_t - \hat{w}_{t-1} \\
\hat{e}_t - \hat{e}_{t-1} \\
\hat{\pi}_t \\
\hat{u}_t \\
4\hat{r}_t
\end{bmatrix}.
\tag{2.99}
$$

The steady-state parameters are determined as

$$
\bar{\gamma} = 100(\gamma - 1), \qquad \bar{\pi} = 100(\pi - 1), \qquad \bar{r} = 100\left(\frac{\pi\gamma}{\beta} - 1\right), \qquad \bar{u} = 100\left(\frac{\phi_w - 1}{\sigma_l}\right),
$$

where ($\phi_w - 1$) is the steady-state labor market markup, π is the steady-state inflation rate, while $\bar{e}$ reflects steady-state labor force growth and is added the real variables that are not measured in per capita terms. Apart from the parameter σ_c, three additional structural parameters are calibrated. These are $g_y = 0.18$, $\delta = 0.025$, and $\epsilon_p = 10$. Furthermore, the AR(1) parameter for the labor supply shock ε_t^s is given by $\rho_s = 0.999$.

2.8. *The Leeper, Plante and Traum Model for Fiscal Policy Analysis*

A common feature of all the models presented above is that macroeconomic (stabilization) policy is confined to interest rate setting with a Taylor-type monetary policy rule, while fiscal policy is, when included, entirely exogenous. The model suggested by Leeper, Plante, and Traum (2010) focuses instead on fiscal policy and includes endogenous policy rules for government spending, lump-sum transfers, and (distortionary) taxation on labor, capital, and consumption expenditures. This LPT model is discussed in some detail below.

2.8.1. *The Log-Linearized Dynamic Equations*

The consumption Euler equation of the log-linearized version of the LPT model is given by

$$\hat{u}_t^b - \frac{\gamma(1+h)}{1-h}\hat{c}_t + \frac{\gamma h}{1-h}\hat{c}_{t-1} - \frac{\tau^c}{1+\tau^c}\hat{\tau}_t^c = \hat{r}_t - \frac{\tau^c}{1+\tau^c}E_t\hat{\tau}_{t+1}^c + E_t\hat{u}_{t+1}^b - \frac{\gamma}{1-h}E_t\hat{c}_{t+1}, \quad (2.100)$$

where $\hat{u}_t^b$ is a general preference shock, $\hat{c}_t$ is real consumption, $\hat{\tau}_t^c$ is the consumption tax rate, and $\hat{r}_t$ is the interest rate on one-period debt outstanding at time t. Furthermore, γ is the risk aversion parameter for a utility maximizing household, h is the habit formation parameter, while τ^c is the steady-state consumption tax rate.

The labor supply Euler equation is

$$\hat{u}_t^l + \left(1+\kappa\right)\hat{l}_t + \frac{\tau^c}{1+\tau^c}\hat{\tau}_t^c = \hat{y}_t - \frac{\tau^l}{1-\tau^l}\hat{\tau}_t^l - \frac{\gamma}{1-h}\hat{c}_t + \frac{\gamma h}{1-h}\hat{c}_{t-1}, \quad (2.101)$$

where $\hat{u}_t^l$ is a labor-supply-specific preference shock, $\hat{l}_t$ is hours worked, $\hat{\tau}_t^l$ is the labor income tax rate, while $\hat{y}_t$ is output. The parameter κ is the inverse Frisch elasticity, whereas τ^l is the steady-state labor income tax rate.

The value of the capital stock, $\hat{q}_t$, is related to consumer behavior according to the following Euler equation

$$\hat{q}_t = E_t\hat{u}_{t+1}^b - \frac{\gamma}{1-h}E_t\hat{c}_{t+1} + \frac{\gamma(1+h)}{1-h}\hat{c}_t - \frac{\tau^c}{1+\tau^c}\left(E_t\hat{\tau}_{t+1}^c - \hat{\tau}_t^c\right) - \hat{u}_t^b - \frac{\gamma h}{1-h}\hat{c}_{t-1}$$
$$+ \beta\left(1-\tau^k\right)R^k\left(E_t\hat{y}_{t+1} - \hat{k}_t\right) - \beta\tau^k R^k E_t\hat{\tau}_{t+1}^k - \beta\delta_1 E_t\hat{v}_{t+1} + \beta\left(1-\delta_0\right)E_t\hat{q}_{t+1}. \quad (2.102)$$

Installed capital is given by $\hat{k}_t$, $\hat{\tau}_t^k$ is the capital income tax rate, and $\hat{v}_t$ is capacity utilization. In (2.102) we have already taken into account that the steady-state rental rate on capital, R^k, is equal to the capital share, α, times the steady-state output-capital ratio, Y/K, i.e.

$$R^k = \frac{\alpha Y}{K}.$$

The discount factor is given by β, τ^k is steady-state capital income tax, while δ_0 and δ_1 are parameters of the quadratic capital utilization cost function.[12] Furthermore, output is linked to capital, its value, and capital utilization through

$$\hat{y}_t - \frac{\tau^k}{1-\tau^k}\hat{\tau}_t^k - \hat{k}_{t-1} = \hat{q}_t + \left(1 + \frac{\delta_2}{\delta_1}\right)\hat{v}_t, \quad (2.103)$$

where δ_2 is yet another capital utilization cost parameter.

The investent Euler equation of the LPT model is

$$\frac{1}{\phi}\hat{q}_t - \left(1+\beta\right)\hat{i}_t + \hat{i}_{t-1} + \beta E_t\hat{i}_{t+1} - \hat{u}_t^i + \beta E_t\hat{u}_{t+1}^i = 0. \quad (2.104)$$

Real investment is here denoted by $\hat{i}_t$, $\hat{u}_t^i$ is an investment-specific technology shock, and ϕ is an investment adjustment cost parameter.

---

[12] Following Schmitt-Grohé and Uribe (2012), LPT assume that owners of physical capital can control the intensity of capital utilization and adopt a quadratic form of this function, with a constant δ_0, a parameter δ_1 on the deviation of capital utilization from unity, and δ_2 on the square of this deviation.

The aggregate resource constraint in log-linearzed form is given by

$$Y\hat{y}_t = C\hat{c}_t + I\hat{i}_t + G\hat{g}_t. \tag{2.105}$$

The steady-state of the model determines the parameters C, I, and G; see Section 2.8.2 for more details.

The capital accumulation equation is

$$\hat{k}_t = (1 - \delta_0)\hat{k}_{t-1} + \delta_1\hat{v}_t + \delta_0\hat{i}_t. \tag{2.106}$$

The log-linearized government budget constraint is

$$B\hat{b}_t + \tau^k\alpha Y\left(\hat{\tau}_t^k + \hat{y}_t\right) + \tau^l(1-\alpha)Y\left(\hat{\tau}_t^l + \hat{y}_t\right) + \tau^c C\left(\hat{\tau}_t^c + \hat{c}_t\right) = \frac{B}{\beta}\left(\hat{r}_{t-1} + \hat{b}_{t-1}\right) \\ + G\hat{g}_t + Z\hat{z}_t. \tag{2.107}$$

The government debt is denoted by $\hat{b}_t$, lump-sum government transfers by $\hat{z}_t$, while B is the steady-state government debt and Z the steady-state government tranfers.

The aggregate production function of the economy is

$$\hat{y}_t = \hat{u}_t^a + \alpha\left(\hat{v}_t + \hat{k}_{t-1}\right) + (1-\alpha)\hat{l}_t, \tag{2.108}$$

where $\hat{u}_t^a$ is a total factor productivity (TFP) process.

The model also includes five fiscal policy rules. Government spending is assumed to be determined by

$$\hat{g}_t = -\varphi_g\hat{y}_t - \gamma_g\hat{b}_{t-1} + \hat{u}_t^g, \tag{2.109}$$

where $\hat{u}_t^g$ is a government spending shock. The φ_g parameter reflects the response to output and γ_g the response to government debt.

The lump-sum government transfers are assumed given by the rule

$$\hat{z}_t = -\varphi_z\hat{y}_t - \gamma_z\hat{b}_{t-1} + \hat{u}_t^z, \tag{2.110}$$

where $\hat{u}_t^z$ is a government transfer shock. The parameters φ_z and γ_z reflect the responses to output and debt in the transfer rule.

The capital income tax rate is determined by the rule

$$\hat{\tau}_t^k = \varphi_k\hat{y}_t + \gamma_k\hat{b}_{t-1} + \phi_{kl}\hat{u}_t^{tl} + \phi_{kc}\hat{u}_t^{tc} + \hat{u}_t^{tk}, \tag{2.111}$$

where $\hat{u}_t^{tl}$ is the labor income tax shock, $\hat{u}_t^{tc}$ is the consumption tax shock, and $\hat{u}_t^{tk}$ the capital income tax shock. The parameter ϕ_{kl} (ϕ_{kc}) measures comovements of the capital income tax shock and the labor income (consumption) tax shocks, while φ_k and γ_k are the response parameters to output and government debt.

The labor income tax rate is similarly determined by the rule

$$\hat{\tau}_t^l = \varphi_l\hat{y}_t + \gamma_l\hat{b}_{t-1} + \phi_{kl}\hat{u}_t^{tk} + \phi_{lc}\hat{u}_t^c + \hat{u}_t^l. \tag{2.112}$$

The parameters φ_l and γ_l give the responses of the labor income tax rate to output and debt, respectively, while ϕ_{lc} is a comovement parameter for labor income tax shocks and consumption tax shocks.

The consumption tax rate rule is given by

$$\hat{\tau}_t^c = \phi_{kc}\hat{u}_t^{tk} + \phi_{lc}\hat{u}_t^{tl} + \hat{u}_t^{tc}. \tag{2.113}$$

Notice that the comovement parameters are the same as those given in the previous tax rate rules, and that this rule does not have any response parameters with respect to output and government debt.

The model has nine exogenous shock process and they are all assumed to follow AR(1) processes. That is,

$$\hat{u}_t^a = \rho_a \hat{u}_{t-1}^a + \sigma_a \eta_t^a,$$
$$\hat{u}_t^b = \rho_b \hat{u}_{t-1}^b + \sigma_b \eta_t^b,$$
$$\hat{u}_t^i = \rho_i \hat{u}_{t-1}^i + \sigma_i \eta_t^i,$$
$$\hat{u}_t^l = \rho_l \hat{u}_{t-1}^l + \sigma_l \eta_t^l,$$
$$\hat{u}_t^g = \rho_g \hat{u}_{t-1}^g + \sigma_g \eta_t^g, \tag{2.114}$$
$$\hat{u}_t^z = \rho_z \hat{u}_{t-1}^z + \sigma_z \eta_t^z,$$
$$\hat{u}_t^{tk} = \rho_{tk} \hat{u}_{t-1}^{tk} + \sigma_{tk} \eta_t^{tk},$$
$$\hat{u}_t^{tl} = \rho_{tl} \hat{u}_{t-1}^{tl} + \sigma_{tl} \eta_t^{tl},$$
$$\hat{u}_t^{tc} = \rho_{tc} \hat{u}_{t-1}^{tc} + \sigma_{tc} \eta_t^{tc}.$$

For $j = a, b, i, l, g, z, tk, tl, tc$, the η_t^j shocks are assumed to be standard normally distributed and to be independent for j and t.

2.8.2. *The Steady-State Equations*

As shown in the Leeper et al. (2010, Appendix A), the steady-state value of the capital stock and capital utilization are both unity, while the interest rate $r = 1/\beta$. The steady-state rental rate on capital satisfies

$$R^k = \frac{r + \delta_0 - 1}{1 - \tau^k},$$

while

$$\delta_1 = R^k (1 - \tau^k).$$

Steady-state investment is linked to the steady-state capital stock through

$$I = \delta_0 K.$$

The steady-state values of Y, K, C, and L can be solved from the following four equations

$$Y(1 - s_g) = C + \delta_0 K,$$
$$Y = K^\alpha L^{1-\alpha},$$
$$R^k = \frac{\alpha Y}{K},$$
$$(1 + \tau^c)L^{1+\kappa} = (C - hC)^{-\gamma}(1 - \tau^l)(1 - \alpha)Y,$$

where $s_g = G/Y \in (0, 1)$. Given a value for s_g and with R^k being determined by, e.g., β, δ_0, and τ^k, these four equations can be solved analytically yielding

$$L = \left[\frac{(1 - \tau^l)(1 - \alpha)\left(\frac{\alpha}{R^k}\right)^{(1-\gamma)\alpha/(1-\alpha)}}{(1 + \tau^c)\left[(1 - h)\left(1 - s_g - \frac{\delta_0 \alpha}{R^k}\right)\right]^\gamma} \right]^{1/(\kappa + \gamma)}$$

It can furthermore be shown that

$$K = \left(\frac{\alpha}{R^k}\right)^{1/(1-\alpha)} L,$$

$$Y = \left(\frac{\alpha}{R^k}\right)^{\alpha/(1-\alpha)} L,$$

$$C = \left(1 - s_g - \frac{\delta_0 \alpha}{R^k}\right) Y.$$

Steady-state lump-sum transfer are thereafter given by

$$Z = \tau^k \alpha Y + \tau^l (1 - \alpha) Y + \tau^c C - \left(s_g + \frac{1 - \beta}{\beta} s_b \right) Y,$$

where $s_b = B / Y \in (0, 1)$.

LPT calibrate a number of the parameters of the model. Specifically, $\beta = 0.99$, $\delta_0 = 0.025$, $\alpha = 0.3$, $s_g = 0.0922$, $s_b = 0.3396$, $\tau^k = 0.184$, $\tau^l = 0.223$, and $\tau^c = 0.028$. Using US data over the sample 1960Q1–2008Q1, they estimate the remaining parameters of the model. From the equations of the model based on optimizing behavior, the parameters to estimate are given by γ, κ, h, ϕ, and δ_2. Furthermore, the 11 parameters of the five fiscal rules, and the 18 parameters of the nine exogenous shock processes are also estimated.

2.8.3. *Measurement Equations*

The measured data are given by natural logarithms of real consumption per capita ($\ln C$), real investment per capital ($\ln I$), hours worked per capital ($\ln H$), government debt per capital ($\ln B$), government spending per capita ($\ln G$), government lump-sum transfers per capita ($\ln Z$), capital income tax revenues per capita ($\ln T^k$), labor income tax revenues per capita ($\ln T^l$), and consumption tax revenues per capita ($\ln T^c$), where a constant and a linear trend has been removed from each of these variables. The observed variables are thereafter linked to the model variables as follows:

$$\begin{bmatrix} \ln C_t \\ \ln I_t \\ \ln H_t \\ \ln B_t \\ \ln G_t \\ \ln Z_t \\ \ln T_t^k \\ \ln T_t^l \\ \ln T_t^c \end{bmatrix} = \begin{bmatrix} \hat{c}_t \\ \hat{i}_t \\ \hat{l}_t \\ \hat{b}_t \\ \hat{g}_t \\ \hat{z}_t \\ \hat{\tau}_t^k + \hat{y}_t \\ \hat{\tau}_t^l + \hat{y}_t \\ \hat{\tau}_t^c + \hat{c}_t \end{bmatrix}. \tag{2.115}$$

Notice the way the observed variables on capital income tax revenues, labor income tax revenues, and consumption tax revenues are linked to the model variables.[13] These three tax revenue variables also appear on the left hand side of the log-linearized government budget constraint in equation (2.107). Additional information about the defintions of the observed variables is available in Leeper et al. (2010, Appendix B) and in Herbst and Schorfheide (2016, Appendix B).

---

[13] This is not documented in the paper, but is shown in the matlab code that can be downloaded from, e.g., Eric Leeper's homepage.

3. Solving a DSGE Model

Sargent (1989) was among the first to recognise that a log-linearized DSGE model with rational expectations can be cast in state-space form, where the observed variables are linked to the model variables through the measurement equation. The state equation provides the reduced form of the DSGE model, mapping current variables to their lags and the iid shocks. The reduced form is obtained by solving for the expectation terms in the structural form of the model using a suitable method; see, e.g., Blanchard and Kahn (1980), Anderson and Moore (1985), Anderson (2008, 2010), Christiano (2002), King and Watson (1998), Klein (2000), Sims (2002), and Meyer-Gohde (2010). If a unique convergent solution is available, the Kalman filter can be applied to compute the value of the log-likelihood function.

Below I shall present the basics underlying three of the above approaches to solving a log-linearized DSGE model with rational expectations. The first is the fastest method supported by YADA, namely, the Anderson-Moore (AiM) approach which relies on a shuffle and eigenvalue method described in various papers over the years. The other two methods make use of the generalized Schur decomposition (QZ decomposition) based on alternative representations of the structural form of the DSGE model.

3.1. *The DSGE Model Specification and Solution*

To make use of the Kalman filter for evaluating the log-likelihood function we need to have a mapping from the structural parameters of the DSGE model to the "reduced form" parameters in the state-space representation. This objective can be achieved by attempting to solve the DSGE model for a given set of parameter values. Given that there exists a unique convergent solution, we can express the solution as a reduced form VAR(1) representation of the form given by the state equation in (5.2).

In this section I will present the general setup for solving linearized rational expectations models with the Anderson-Moore algorithm. This algorithm is implemented in YADA via AiM. Similar to Zagaglia (2005) the DSGE model is expressed as:

$$\sum_{i=1}^{\tau_L} H_{-i} z_{t-i} + H_0 z_t + \sum_{i=1}^{\tau_U} H_i E_t[z_{t+i}] = D\eta_t, \tag{3.1}$$

where $\tau_L > 0$ is the number of lags and $\tau_U > 0$ the number of leads. The z_t ($p \times 1$) vector are here the endogenous variables, while η_t ($q \times 1$) are pure innovations, with zero mean and unit variance conditional on the time $t - 1$ information. The H_i matrices are of dimension $p \times p$ while D is $p \times q$. When $p > q$ the covariance matrix of $\epsilon_t = D\eta_t$, DD', has reduced rank since the number of shocks is less than the number of endogenous variables.[14]

Adolfson, Laséen, Lindé, and Svensson (2008a) shows in some detail how the AiM algorithm can be used to solve the model in equation (3.1) when $\tau_U = \tau_L = 1$.[15] As pointed out in that paper, all linear systems can be reduced to this case by replacing a variable with a long lead or a long lag with a new variable. Consider therefore the system of stochastic difference equations:

$$H_{-1} z_{t-1} + H_0 z_t + H_1 E_t[z_{t+1}] = D\eta_t. \tag{3.2}$$

The AiM algorithm takes the H_i matrices as input and returns B_1, called the convergent autoregressive matrix, and S_0, S_1, such that the solution to (3.2) can be expressed as an autoregressive process

$$z_t = B_1 z_{t-1} + B_0 \eta_t, \tag{3.3}$$

---

[14] The specification of iid shocks and the matrix D is only used here for expositional purposes. AiM does not make any distinction between endogenous variables and shocks. In fact, z_t would include η_t and, thus, H_0 would include D.

[15] Their paper on optimal monetary policy in an operational medium-sized DSGE model is published in Adolfson, Laséen, Lindé, and Svensson (2011).

where

$$B_0 = S_0^{-1}D, \tag{3.4}$$

$$S_0 = H_0 + H_1 B_1, \tag{3.5}$$

$$S_1 = S_0 B_1 = -H_{-1}. \tag{3.6}$$

The S_j matrices ($j = 0, 1$) represent the structural form autoregression, which can be expressed as

$$S_0 z_t = S_1 z_{t-1} + D\eta_t. \tag{3.7}$$

The matrix B_1 satisfies the identity

$$H_{-1} + H_0 B_1 + H_1 B_1^2 = 0. \tag{3.8}$$

This can be seen by leading the system in (3.2) one period and taking the expectation with respect to time t information. Evaluating the expectation through (3.3) yields the identity.[16] From equations (3.5) and (3.8) it can be seen that B_1 and S_j only depend on the H_i matrices, but not on D. This is consistent with the certainty equivalence of the system.

More generally, the conditions for the existence of a unique convergent solution (Anderson and Moore, 1983, 1985, and Anderson, 2008, 2010) can be summarized as follows:

- *Rank condition*:

$$\text{rank}\left(\sum_{i=-\tau_L}^{\tau_U} H_i\right) = \dim(z).$$

- *Boundedness condition*: For all $\{z_i\}_{i=-\tau_L}^{-1}$ there exists $\{z_t\}_{t=0}^{\infty}$ that solves (3.1) such that

$$\lim_{T\to\infty} E_{t+j}\left[z_{t+j+T}\right] = 0, \quad \forall j \geq 0.$$

The rank condition is equivalent to require that the model has a unique non-stochastic steady state, while the boundedness condition requires that the endogenous variables eventually converge to their steady state values; see also Blanchard and Kahn (1980) for discussions on existence and uniqueness.[17]

Given that a unique convergent solution exists, AiM provides an autoregressive solution path

$$z_t = \sum_{i=1}^{\tau_L} B_i z_{t-i} + B_0 \eta_t. \tag{3.9}$$

The VAR(1) companion form of (3.9) is given by

$$\begin{bmatrix} z_t \\ z_{t-1} \\ \vdots \\ z_{t-\tau_L+1} \end{bmatrix} = \begin{bmatrix} B_1 & B_2 & \cdots & B_{\tau_L} \\ I & 0 & \cdots & 0 \\ \vdots & \ddots & & \vdots \\ 0 & \cdots & I & 0 \end{bmatrix} \begin{bmatrix} z_{t-1} \\ z_{t-2} \\ \vdots \\ z_{t-\tau_L} \end{bmatrix} + \begin{bmatrix} B_0\eta_t \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{3.10}$$

With $\xi_t = [z_t \cdots z_{t-\tau_L+1}]'$ the F matrix of the state-space form is immediately retrieved from (3.10), while the state shocks, v_t, are given by the second term on the right hand side; see Section 5. The Q matrix is equal to the zero matrix, except for the upper left corner which is given by $B_0 B_0'$. If $\tau_L = 1$, then $Q = B_0 B_0'$, while $F = B_1$.

Anderson and Moore (1985) presents a 14-steps algorithm for solving the system of equations in (3.1). The AiM matlab code is setup in a different way, where the first 8 steps are performed without relying on a singular value decomposition. Before the coded version of the AiM algorithm is presented, let

$$H = \begin{bmatrix} H_{-\tau_L} & \cdots & H_{-1} & H_0 & H_1 & \cdots & H_{\tau_U} \end{bmatrix},$$

---

[16] Alternatively, (3.8) can be deduced by combining equations (3.5) and (3.6).

[17] For generalizations and discussions of the boundedness condition, see Sims (2002) and Meyer-Gohde (2010); see also Burmeister (1980) for further discussions on convergence and uniqueness.

be a $p \times p(\tau_L + \tau_U + 1)$ matrix, while Q is a zero matrix with dimension $p\tau_U \times p(\tau_L + \tau_U)$. The auxiliary initial conditions in Q are first setup from a series of shift-rights. These are based on locating rows of zero of the last (right-most) p columns of H (initially therefore of H_{τ_U}). Once there are no such rows of zeros, the algorithm is complemented with an eigenvalue computation for determining Q.

Assuming that the matrix H_{τ_U} has $n_1 > 0$ rows of zeros with indices i_1, the shift-right procedure begins with setting the first n_1 rows of Q equal to rows with indices i_1 and the first $p(\tau_U + \tau_L)$ columns of H. The rows in H with indices i_1 are prepended with p columns of zeros while the last p columns are deleted, i.e., the first $p(\tau_U + \tau_L)$ elements in the n_1 rows with indices i_1 are shifted to the right by p columns. The procedure next tries to locate rows of zeros in the last p columns of this reshuffled H matrix. Let n_2 be the number of such rows of zeros with indices i_2. If $n_2 > 0$ and $n_1 + n_2 \leq p\tau_U$, rows $n_1 + 1$ until $n_1 + n_2$ of Q are set equal to the rows with indices i_2 and the first $p(\tau_U + \tau_L)$ columns of H. Furthermore, the rows in H with indices i_2 are prepended with p columns of zeros while the last p columns are deleted. The procedure thereafter checks for rows of zeros in the last p columns of H and repeats the procedure if necessary until no more rows of zero can be located in the last p columns of H. Notice that the procedure also breaks off if $n_1 + n_2 + \ldots + n_k > p\tau_U$, where k is the number of searches for rows of zeros. In that case, AiM reports that there have been too many shift-rights and that it cannot locate a unique convergent solution to the system of equations (too many auxiliary conditions).

Once the shift-right loop has finished, the AiM procedure computes the $p \times p(\tau_U + \tau_L)$ matrix Γ from the reshuffled H matrix according to

$$\Gamma = -H_{\tau_U}^{-1} \begin{bmatrix} H_{-\tau_L} & \cdots & H_{-1} & H_0 & \cdots & H_{\tau_U - 1} \end{bmatrix},$$

where H_{τ_U} is now the $p \times p$ matrix in the last p columns of the reshuffled H, while the $p(\tau_U + \tau_L) \times p(\tau_U + \tau_L)$ companion matrix A is obtained as

$$A = \begin{bmatrix} 0 & I_{p(\tau_U + \tau_L - 1)} \\ \Gamma \end{bmatrix}.$$

These computations correspond to steps 9 and 10 in Anderson and Moore (1985).

The AiM code next checks if there are any columns with only zeros in A. Let m_1 be the number of such columns and j_1 be a vector with indices for these columns. If $m_1 > 0$, rows and columns j_1 are removed from A while keeping track of the columns in the original A matrix that are nonzero and therefore "essential". The code again checks for columns with only zeros, removes the corresponding columns and rows while keeping track of the indices for the original columns of A that have *not* been deleted. This procedure is repeated until there are no more zero columns in the A matrix. While this dimension shrinking step for A is not necessary, it helps speed up the eigenvalue calculation in the following step since inessential lags have been removed from the matrix.

Based on this lower dimensional A matrix, AiM computes a matrix W which contains the remaining stability conditions for verifying the saddlepoint property. Specifically, let $A'V = VD$, where D is a diagonal matrix with the eigenvalues of A', sorted from the largest to the smallest, and V a unitary matrix with the corresponding eigenvectors. The matrix W is now given by the sum of the real and the imaginary part of V, i.e., $W = \Re(V) + \Im(V)$. The large roots of the DSGE model can directly be checked by calculating the number of diagonal elements of D that in absolute terms are greater than a suitable lower bound, such as $1 + g$ with g, the numerical tolerance for the DSGE model solver in YADA, being a small positive scalar.

To test if the DSGE model has a unique convergent solution, AiM adds the total number of shift-rights ($n_1 + \ldots + n_{k^*}$), where k^* is the last time a row of zeros was located in the last p columns of H) and the total number of large roots in A. If this number is less (greater) than $p\tau_U$ then there are too few (many) large roots. With too few large roots there are many convergent solutions, while too many large roots means that the boundedness condition is violated.

The matrix W is next included in the Q matrix as follows: (i) for the Q matrix select rows $n_1 + \ldots + n_{k^*} + 1$ until $p\tau_U$ and the columns determined by the indices with columns of the original A that are used when computing W; (ii) for the W matrix select columns 1 until the number of rows selected in Q; and (iii) set the selected rows and columns of Q in (i) equal to the transpose of the matrix obtained from W using the columns under (ii).

The resulting Q matrix may now be partitioned as

$$Q = \begin{bmatrix} Q_L & Q_R \end{bmatrix},$$

where Q_L is $p\tau_U \times p\tau_L$ and Q_R is $p\tau_U \times p\tau_U$. If Q_R is singular, then the boundedness condition is violated and there does not exist a unique convergent solution. On the other hand, when Q_R is nonsingular a unique convergent solution exists. In that case, the reduced form matrices B_i, $i = 1, \ldots, \tau_L$ are obtained from the first p rows of $-Q_R^{-1} Q_L$, where the reduced form matrices are ordered from left to right as $B_{\tau_L}, B_{\tau_L-1}, \ldots, B_1$.

AiM provides a number of `mcode` values which reflect the findings from running the above procedure. The value 1 means that there exists a unique and convergent solution. The value 3 means that that there are too many large roots and that the boundedness condition is violated, while 4 means that there are too few large roots and that there are multiple convergent solutions. When AiM gives the value 45 there are not only too few large roots, but Q_R is also singular so that the boundedness condition is violated, while 35 implies that there are too many large roots with Q_R is being singular. The case when the total number of shift-rights plus the number of large roots is equal to $p\tau_U$ but when Q_R is singular gives the `mcode` value 5. The values 61 and 62 mean that there are too many exact and numeric shift-rights, respectively, where the exact and numeric shift-rights are two version for searching for rows of zeros in the last p columns of H. YADA also supports two additional `mcode` values. Namely, 7 when the A matrix has infinite or `NaN` entries (the eigenvalues cannot be calculated), and 8 when the H matrix has complex entries. These two cases reflect numerical problems which may be overcome by rewriting the original DSGE model equations.

YADA *requires* that $\tau_L = 1$. This is not a restriction since new variables can be created when lags greater than 1 are required. In fact, this ensures that the dimension r is not needlessly large. To my knowledge, there is not a single DSGE model where *all* z_t variables appear with two or more lags. Hence, by letting $\tau_L \geq 2$ in a DSGE model, the dimension of ξ_t is greater than needed. This will slow down the computations and for medium-sized DSGE models, such as RAMSES (Adolfson et al., 2007b) and the NAWM (Christoffel et al., 2008), the inefficiency losses are likely to be very costly.

3.2. *The Klein Approach*

Klein (2000) shows how the generalized Schur form (QZ decomposition) can be utilized to solve a system of linear rational expectations models; see Golub and van Loan (1983) for details on the QZ decomposition. One issue with the Klein approach that we need to keep in mind is that it divides the variables into predetermined (or backward-looking) and non-predetermined. It is then assumed that the predetermined variables are ordered first and non-predetermined variables thereafter.

Like Blanchard and Kahn (1980), the structural form expression of a linear rational expectations model that Klein considers may be written as

$$A E_t[X_{t+1}] = B X_t + C\eta_t. \tag{3.11}$$

Following, e.g., Meyer-Gohde (2010), we choose to stack the variables in the system (3.2) such that $t-1$ appear above t. That way we know that the first p variables are predetermined. This turns out to simplify the handling of output from the QZ decomposition. In particular, we rewrite the AiM system matrices such that

$$\begin{bmatrix} 0 & -H_1 \\ I & 0 \end{bmatrix} \begin{bmatrix} z_t \\ E_t[z_{t+1}] \end{bmatrix} = \begin{bmatrix} H_{-1} & H_0 \\ 0 & I \end{bmatrix} \begin{bmatrix} z_{t-1} \\ z_t \end{bmatrix} + \begin{bmatrix} -D \\ 0 \end{bmatrix} \eta_t. \tag{3.12}$$

The matrix B_1 in (3.3) can now be computed from a QZ decomposition of the matrices (A, B) conditional on a given ordering of the generalized eigenvalues of the matrix pencil $(Az - B)$, where z is a complex variable. The QZ decomposition of (A, B) is given by $QAZ = S$ and $QBZ = T$, where S and T are upper triangular and possibly complex, Q and Z are unitary, i.e., $Q^*Q = Z^*Z = I$ and Q^* is the complex conjugate of Q.[18] The generalized eigenvalues can be computed from the diagonal elements of S and T, with $\lambda_i(A, B) = t_{ii} / s_{ii}$ for $s_{ii} \neq 0$. We assume that Q, Z, S and T take the ordering of λ_i from the smallest to the largest into account.

Let s be the number of stable generalized eigenvalues, i.e., $\lambda_s(A, B) < 1 + g$, where g is a small positive scalar, while $\lambda_{s+1}(A, B) \geq 1 + g$. Let Z_{11} be the $s \times s$ matrix in the upper left corner of Z corresponding to the s stable eigenvalues. If Z_{11} does not have full rank, then there is no stable solution to the DSGE model. If $\text{rank}(Z_{11}) = s \geq p$ there is a nonempty set of stable solutions, where $s = p$ implies that the solution is unique and convergent, while $s > p$ means that the system is subject to indeterminacy, i.e., there are multiple solutions (too few large roots).

Provided that a unique convergent solution exists, it is given by

$$B_1 = \Re\left(Z_{11} S_{11}^{-1} T_{11} Z_{11}^{-1} \right), \tag{3.13}$$

where $\Re$ denotes a real number, and S_{11} and T_{11} are the $p \times p$ matrices in the upper left corner of S and T, respectively. The B_1 matrix may also be determined directly from Z. Specifically, with Z_{21} being the $p \times p$ matrix in the last p rows and first p columns of Z, it follows that B_1 is also equal to $\Re(Z_{21} Z_{11}^{-1})$. Moreover, let Z_{12} and Z_{22} be the $p \times p$ matrices collected from the first and last p rows, respectively, and the last p columns of Z, while T_{22} is given by the matrix in the last p rows and columns of T and Q_2 is the $p \times 2p$ matrix located in the last p rows of Q. It can now be shown that

$$B_0 = \Re\left(\left(Z_{22} - Z_{21} Z_{11}^{-1} Z_{12} \right) T_{22}^{-1} Q_2 \begin{bmatrix} D \\ 0 \end{bmatrix} \right); \tag{3.14}$$

see, e.g., Meyer-Gohde (2010).

3.3. The Sims Approach

The solution algorithm suggested by Sims (2002) is similar to Klein's in that it uses the QZ decomposition of certain system matrices. However, it does not rely on ordering variables according to those that are predetermined and non-predetermined, respectively. Instead, it introduces auxiliary variables to capture expectational errors which are endogenously determined.

The system matrices defined by Sims can be expressed as

$$\Gamma_0 Y_t = \Gamma_1 Y_{t-1} + \Psi \eta_t + \Pi \varepsilon_t, \tag{3.15}$$

where ϵ_t is an expectational error satisfying $E_t[\varepsilon_{t+1}] = 0$ for all t. To rewrite the structural form in (3.2) into the Sims form (3.15), we introduce the expectational variable $e_t = E_t[z_{t+1}]$. This means that

$$z_t = e_{t-1} + \varepsilon_t, \tag{3.16}$$

where $\varepsilon_t = z_t - E_{t-1}[z_t]$. It now follows that (3.2) can be expressed as

$$\begin{bmatrix} H_0 & H_1 \\ I & 0 \end{bmatrix} \begin{bmatrix} z_t \\ e_t \end{bmatrix} = \begin{bmatrix} -H_{-1} & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} z_{t-1} \\ e_{t-1} \end{bmatrix} + \begin{bmatrix} D \\ 0 \end{bmatrix} \eta_t + \begin{bmatrix} 0 \\ I \end{bmatrix} \varepsilon_t. \tag{3.17}$$

It may be noted that the dimension of Y_t in (3.17) is typically greater than it need be. The reason is that some of the variables in z_t will in most situations not be included in the $H_1 E_t[z_{t+1}]$ term, i.e., the H_1 matrix is typically singular. In fact, it would be sufficient to define the vector e_t such that only those elements of $E_t[z_{t+1}]$ that enter the model would be used. YADA here takes the easy way out and includes all such variables in the definition of e_t. This avoids the hazzle of letting the code try to figure out which variables are (always) included and which are (always)

---

[18] The complex conjugate or conjugate transpose of a complex matrix $A = B + iC$ is equal to $A^* = B' - iC'$. Hence, the complex conjugate of a real matrix is simply the transpose of the matrix.

excluded, a potentially error prone operation. One cost of this decision is that the solution time for the Sims approach is longer than necessary. At the same time, this is not expected to be very important since relative to computing the value of the log-likelihood the time for solving the model is nevertheless short.

Given the form (3.15), the solution method by Sims is also based on the QZ decomposition with the generalized eigenvalues sorted the same way as in Klein; see Sims (2002, equations 44–45) for details on the computation of B_1 and B_0. This also means that the existence of a unique convergent solution can be checked with the same tools.

3.4. *Solving a DSGE Model Subject to a Zero Lower Bound Constraint*

The DSGE models in Section 2 all include the short-term nominal interest rate with a Taylor-type monetary policy rule. Although a policy rate can, in principle, be negative when banks are required to hold a certain fraction of their deposits in reserves at the central bank, it is nevertheless the case that nominal interest rates are subject to some type of (zero) lower bound constraint.

The log-linearized DSGE models do not take such a nonlinearity into account when the short-term nominal interest rate is measured in percent. To deal with this, one option is to let the nominal interest rate be measured as the natural logarithm of the rate, as in Fuhrer and Madigan (1997) who include the first difference of the log of the nominal interest rate as an observable, but this option will not be discussed below as no further changes to the setup would be necessary. Instead, we will consider the approach suggested by Reifschneider and Williams (2000) of adding anticipated shocks to the monetary policy rule, with the important modifications suggested by Hebden, Lindé, and Svensson (2011, HLS); see Erceg and Lindé (2013) for an application of the HLS procedure. The HLS approach takes into account that the anticipated shocks have to be positive when the lower bound is strictly binding. Furthermore, indeterminacy issues are also taken into account by HLS.

3.4.1. *The Zero Lower Bound*

Let $\hat{r}_t$ be the nominal interest rate which is determined by the monetary policy rule[19] and which is therefore not restricted by any lower bound. Let r_t be the observable (actual) nominal interest rate which is now subject to the following nonlinearity

$$r_t = \max\{\bar{r} + \hat{r}_t, r_{ZLB,t}\}, \tag{3.18}$$

where $\bar{r}$ is the steady-state nominal interest rate, and $r_{ZLB,t}$ is the lower bound on the actual nominal interest rate.[20] We allow for the possibility that the lower bound need not be zero, that it can vary over time, but we will also require that the current and T consecutive future values of this lower bound are known at time t, i.e., that $r_{ZLB,t+\tau}$ is known for $\tau = 0, 1, \ldots, T$. This means that we treat the lower bound as a deterministic variable over the period when it may be binding.

The nonlinearity in (3.18) can be rewritten as

$$\tilde{r}_t + \bar{r} - r_{ZLB,t} = \max\{\hat{r}_t + \bar{r} - r_{ZLB,t}, 0\}, \tag{3.19}$$

where $\tilde{r}_t = r_t - \bar{r}$ is the restricted interest rate in the model which satisfies the zero lower bound (ZLB). Notice that the restricted interest rate is equal to the unrestricted rate whenever the zero lower bound is not binding, while it is greater than the unrestricted rate when the ZLB is binding.

---

[19] Hebden et al. (2011) also consider optimal monetary policy, a topic which will not be discussed below.

[20] Notice that the actual interest rate, the notional interest rate ($\hat{r}_t$), the steady-state interest rate and the lower bound are all measured in the same time units here, e.g., quarterly rates. If the actual interest rate is measured in annual terms, then the right hand side of (3.18) is multiplied by 4; see, e.g., the measurement equations (2.34) of the Smets and Wouters model.

3.4.2. *Anticipated Shocks*

Following Laséen and Svensson (2011) the ZLB is emplemented with anticipated shocks. This means that the projected restricted and unrestricted policy rate in each period t satisfy

$$\tilde{r}_{t+\tau,t} = \hat{r}_{t+\tau,t} + \alpha_{t+\tau,t}, \quad \tau \geq 0, \tag{3.20}$$

where $(t+\tau, t)$ denotes the projection at t of period $t+\tau$. Equation (3.20) and $\tilde{r}_{t+\tau,t} \geq \hat{r}_{t+\tau,t}$ implies that all current and future anticipated shocks $\alpha_{t+\tau,t} \geq 0$ and that $\alpha_{t,t} = \alpha_t > 0$ when the ZLB is strictly binding in period t, i.e., when $\tilde{r}_t > \hat{r}_t$. Laséen and Svensson (2011) call the stochastic variable α_t the deviation and assume that it satisfies

$$\alpha_t = \varphi_{t,t} + \sum_{s=1}^{T} \varphi_{t,t-s}, \tag{3.21}$$

for some $T \geq 0$. It is here assumed that the ZLB may be binding in the current or the next finite T periods, but not beyond $t + T$. The vector

$$\Phi_t = \begin{bmatrix} \varphi_{t,t} & \varphi_{t+1,t} & \cdots & \varphi_{t+T,t} \end{bmatrix}',$$

is $(T+1)$-dimensional with zero mean i.i.d. anticipated shocks being realized in period t. From equation (3.21) it follows that

$$\alpha_t = \alpha_{t,t-1} + \varphi_{t,t},$$

and, more generally, that

$$A_t = MA_{t-1} + \Phi_t, \tag{3.22}$$

where

$$A_t = \begin{bmatrix} \alpha_{t,t} & \alpha_{t+1,t} & \cdots & \alpha_{t+T,t} \end{bmatrix}', \qquad M = \begin{bmatrix} 0_{T\times 1} & I_T \\ 0 & 0_{1\times T} \end{bmatrix}.$$

Notice that the vector A_t is $(T+1)$-dimensional while M is $(T+1) \times (T+1)$.

3.4.3. *Solving the DSGE Model with the Klein Solver*

Before we combine the DSGE model in (3.2) with the restricted policy rate and the anticipated shocks, we need to link the monetary policy rate to z_t and the monetary policy rule to the model equations. From this knowledge we can separate the monetary policy variables from the other model variables, as well as the monetary policy rule from the other equations of the model.

To these ends, let $e_{p,r}$ be an p-dimensional vector with unity in the position r of z_t and zeros elsewhere. Position r is identical to the position of $\tilde{r}_t$ in z_t, i.e.

$$\tilde{r}_t = e'_{p,r} z_t.$$

The remaining elements in z_t can be extracted by premultiplying it with $K'_{p,r}$, where the $p \times (p-1)$ matrix $K_{p,r}$ contains all columns of I_p except from column r. Similarly, suppose that the m:th equation of the model system (3.1) is the monetary policy rule. This equation is therefore extracted by premultiplying this system by $e'_{p,m}$, while the remaining equations are similarly selected by premultiplying the system with $K'_{p,m}$.

Under the assumption of one lead and lag, the unrestricted monetary policy rule is now given by

$$\begin{aligned}
e'_{p,m} H_{-1} K_{p,r} K'_{p,r} z_{t-1} + e'_{p,m} H_{-1} e_{p,r} \hat{r}_{t-1} + e'_{p,m} H_0 K_{p,r} K'_{p,r} z_t + e'_{p,m} H_0 e_{p,r} \hat{r}_t \\
+ e'_{p,m} H_1 K_{p,r} K'_{p,r} E_t z_{t+1} + e'_{p,m} H_1 e_{p,r} E_t \hat{r}_{t+1} = e'_{p,m} D \eta_t.
\end{aligned} \tag{3.23}$$

Notice that the restricted policy rate does not enter into this equation. Rather, it is linked to the unrestricted policy rule and the antipicated shocks from equation (3.20), which can be rewritten such that

$$e'_{p,r} z_{t+\tau,t} = \hat{r}_{t+\tau,t} + e_{T+1,\tau+1} A_t, \quad \tau = 0, 1, \ldots, T, \tag{3.24}$$

where $e_{T+1,\tau+1}$ is a $(T+1)$-dimensional vector with unity in its $(\tau+1)$:st element and zeros elsewhere.

The remaining model equations can now be expressed as

$$K'_{p,m}H_{-1}z_{t-1} + K'_{p,m}H_0z_t + K'_{p,m}H_1E_tz_{t+1} = K'_{p,m}D\eta_t. \tag{3.25}$$

Notice that the restricted monetary policy rate enters these equations, while the unrestricted policy rate does not.

Letting $H_i^{(m)} = K'_{p,m}H_i$, $h_i^{(m)} = e'_{p,m}H_iK_{p,r}K'_{p,r}$, $h_i^{(r)} = e'_{p,m}H_ie_{p,r}$ for $i = -1, 0, 1$, and $D^{(m)} = K'_{p,m}D$ and $d^{(m)} = e'_{p,m}D$, we can express the DSGE model subject to possibly nonzero anticipated shocks in stacked form as

$$
\begin{bmatrix}
I_q & 0 & 0 & 0 & 0 & 0 \\
0 & I_p & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & I_{T+1} & 0 & 0 \\
0 & 0 & 0 & 0 & H_1^{(m)} & 0 \\
0 & 0 & 0 & 0 & h_1^{(m)} & h_1^{(r)} \\
0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
\eta_{t+1} \\ z_t \\ \hat{r}_t \\ A_{t+1} \\ E_tz_{t+1} \\ E_t\hat{r}_{t+1}
\end{bmatrix} =
$$

$$
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & I_p & 0 \\
0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & M & 0 & 0 \\
D^{(m)} & -H_{-1}^{(m)} & 0 & 0 & -H_0^{(m)} & 0 \\
d^{(m)} & -h_{-1}^{(m)} & -h_{-1}^{(r)} & 0 & -h_0^{(m)} & -h_0^{(r)} \\
0 & 0 & 0 & -e'_{T+1,1} & e'_{p,r} & -1
\end{bmatrix}
\begin{bmatrix}
\eta_t \\ z_{t-1} \\ \hat{r}_{t-1} \\ A_t \\ z_t \\ \hat{r}_t
\end{bmatrix}
+
\begin{bmatrix}
I_q & 0 \\
0 & 0 \\
0 & 0 \\
0 & I_{T+1} \\
0 & 0 \\
0 & 0 \\
0 & 0
\end{bmatrix}
\begin{bmatrix}
\eta_{t+1} \\ \Phi_{t+1}
\end{bmatrix}. \tag{3.26}
$$

With $V_t = [\eta'_t\ z'_{t-1}\ \hat{r}_{t-1}\ A'_t]'$ and $Y_t = [z'_t\ \hat{r}_t]'$ being $(T + q + p + 2)$ and $(p + 1)$-dimensional vectors, respectively, equation (3.26) can be written more compactly as

$$
\begin{bmatrix} I_{T+q+p+2} & 0 \\ 0 & A_{22} \end{bmatrix}
\begin{bmatrix} V_{t+1} \\ E_tY_{t+1} \end{bmatrix} =
\begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}
\begin{bmatrix} V_t \\ Y_t \end{bmatrix} +
\begin{bmatrix} C_1 \\ 0 \end{bmatrix}
\begin{bmatrix} \eta_{t+1} \\ \Phi_{t+1} \end{bmatrix}, \tag{3.27}
$$

where A_{22} and B_{22} are $(p + 1) \times (p + 1)$ matrices, B_{11} is $(T + p + q + 2) \times (T + p + q + 2)$, while B_{12} and B'_{21} are $(T + q + k + 2) \times (p + 1)$.

An alternative policy rule under the ZLB is suggested by Lindé et al. (2016) where instead of the lagged unrestricted policy rate, the rule has the restricted interest rate. The effect on B_{21} in (3.27) from using this alternative rule is that $h_{-1}^{(r)}$ on $\hat{r}_{t-1}$ is replaced by a zero, while $h_{-1}^{(m)}$ on z_{t-1} is replaced by $h_{-1}^{(m)} + h_{-1}^{(r)}e'_{p,r}$.

Notice that the vector V_t contains the predetermined variables of the system, while Y_t are the forward-looking variables. With A and B representing the matrices in (3.27) with partitions A_{ij} and B_{ij}, respectively, let the QZ decomposition of (A, B) be given by $QAZ = S$ and $QBZ = T$, where S and T are upper triangular and possibly complex, while Q and Z are unitary matrices. Assuming a unique convergent solution exists (cf. Section 3.2) we find that:

$$V_t = GV_{t-1} + C_1 \begin{bmatrix} \eta_t \\ \Phi_t \end{bmatrix}, \tag{3.28}$$

where

$$G = \Re\left(Z_{11}S_{11}^{-1}T_{11}Z_{11}^{-1}\right),$$

with $\Re$ denoting the real part of a complex matrix, and where Z_{11}, S_{11}, and T_{11} are obtained from the upper left $(T + q + p + 2) \times (T + q + p + 2)$ corners of Z, S, and T, respectively. The

forward-looking variables are related to the predetermined variables according to

$$Y_t = PV_t, \tag{3.29}$$

where

$$P = \Re\left(Z_{21} Z_{11}^{-1}\right),$$

and Z_{21} is the bottom left $(p+1) \times (T+q+k+2)$ corner of Z.[21]

Finally, the matrix G has the shape

$$G = \begin{bmatrix} 0 & 0 & 0 & 0 \\ G_{z\eta} & G_{zz} & G_{zr} & G_{zA} \\ G_{r\eta} & G_{rz} & G_{rr} & G_{rA} \\ 0 & 0 & 0 & M \end{bmatrix}.$$

The matrices $G_{z\eta}$, G_{zz}, G_{zr}, and G_{zA} are $p \times q$, $p \times p$, $p \times 1$, and $p \times (T+1)$ respectively, while $G_{r\eta}$, G_{rz}, and G_{rA} are $1 \times q$, $1 \times p$, $1 \times (T+1)$ vectors, respectively, and G_{rr} is a scalar.

Before we proceed, the above is based on solving the model with the Klein solver. Naturally, we may instead use the AiM solver, the Sims solver, or some other valid approach. However, the Klein (2000) is particularly convenient here where in particular the split into predetermined variables (including the anticipated shocks) and forward-looking variables (including the the restricted and unrestriced monetary policy rate) turns out to be useful.

3.4.4. *Policy Rate Projections and the Complementary Slackness Condition*

The projection of the restricted policy rate is

$$\tilde{r}_{t+\tau,t} = e'_{p+1,r} P G^\tau \begin{bmatrix} \eta_t \\ z_{t-1} \\ \hat{r}_{t-1} \\ A_t \end{bmatrix}, \quad \tau = 0, 1, \dots, T, \tag{3.30}$$

while the projection of the unrestricted policy rate is

$$\hat{r}_{t+\tau,t} = e'_{p+1,p+1} P G^\tau \begin{bmatrix} \eta_t \\ z_{t-1} \\ \hat{r}_{t-1} \\ A_t \end{bmatrix}, \quad \tau = 0, 1, \dots, T. \tag{3.31}$$

The HLS approach is to determine the deviation vector A_t such that the policy rate projection (3.30) satisfies the ZLB restriction

$$\tilde{r}_{t+\tau,t} + \bar{r} - r_{ZLB,t+\tau} \geq 0, \quad \tau = 0, 1, \dots, T, \tag{3.32}$$

and the policy rule in (3.19).

When the ZLB restriction is disregarded or not binding, the policy rate projection in period t is given by (3.30) with $A_t = 0$. If the ZLB is disregarded or not binding for any $\tau = 0, 1, \dots, T$, the projections of the restricted and unrestricted policy rates are the same. The policy rate projection in (3.30) with $A_t = 0$ violates the ZLB for one or more periods when

$$\tilde{r}_{t+\tau,t} + \bar{r} - r_{ZLB,t+\tau} < 0, \quad \text{for some } \tau \in \{0, 1, \dots, T\}. \tag{3.33}$$

---

[21] In fact, it can be established that the solution matrices (G, P) of (3.27) satisfy

$$G = B_{11} + B_{12}P,$$
$$A_{22}PG = B_{21} + B_{22}P.$$

while the fact that C_1 remains the matrix that is multiplied by the innovations in (3.28) follows when making use of the former relationship in the sub-system for V_{t+1} in (3.27).

To satisfy the ZLB, we therefore need to find a value of the deviation A_t such that the policy rate projection satisfies (3.32) and

$$\tilde{r}_{t+\tau,t} + \bar{r} - r_{ZLB,t+\tau} = \max\{\hat{r}_{t+\tau,t} + \bar{r} - r_{ZLB,t+\tau}, 0\}. \tag{3.34}$$

This requires that the deviation satisfies

$$\alpha_{t+\tau,t} \geq 0, \quad \tau = 0, 1, \ldots, T, \tag{3.35}$$

and that the policy rate projection and the deviation satisfies the complementary slackness condition

$$\left(\tilde{r}_{t+\tau,t} + \bar{r} - r_{ZLB,t+\tau}\right)\alpha_{t+\tau,t} = 0, \quad \tau = 0, 1, \ldots, T. \tag{3.36}$$

This condition implies that the projected deviation is zero when the left hand side of (3.32) is positive.

We now proceed under the presumption that there exists a unique projection of the deviation A_t that satisfies (3.30) and (3.34)–(3.36). HLS refer to this projection of the deviation and the policy rate projection as the equilibrium projection, where the projection of A_t either has all elements equal to zero with the effect that the ZLB is not binding, or has some elements positive and some equal to zero. Let the set $\mathcal{T}_t$ be defined from

$$\mathcal{T}_t = \left\{ 0 \leq \tau \leq T : \alpha_{t+\tau,t} > 0 \right\}. \tag{3.37}$$

The equilibrium projection satisfies

$$\tilde{r}_{t+\tau,t} = e'_{p,r} P G^\tau \begin{bmatrix} \eta_t \\ z_{t-1} \\ \hat{r}_{t-1} \\ A_t \end{bmatrix} = r_{ZLB,t+\tau} - \bar{r}, \quad \forall \tau \in \mathcal{T}_t. \tag{3.38}$$

Letting n_t denote the number of elements of $\mathcal{T}_t$, the system in (3.38) has n_t equations and n_t elements of A_t that are positive. The solution for A_t and the set $\mathcal{T}_t$ depends on the structural shocks η_t, and HLS emphasize that the set of periods τ in (3.33) for which the policy rate projection in (3.30) with $A_t = 0$ violates the ZLB is not necessarily the same as the set of periods $\mathcal{T}_t$ for which the ZLB is strictly binding in equilibrium. The reason for this is that the projections of the predetermined and forward-looking variables $V_{t+\tau,t}$ and $Y_{t+\tau,t}$ that determine the unrestricted policy rate differ depending on whether A_t is zero or not, i.e., the whole policy-rate path is affected when the ZLB is imposed.

3.4.5. *The Forward-Back Shooting Algorithm*

Given $X_t = [\eta'_t \ z'_{t-1} \ \hat{r}_{t-1}]'$, the problem of imposing the ZLB is now to find the set $\mathcal{T}_t$ for which the ZLB is strictly binding in equilibrium and a contribution of Hebden, Lindé, and Svensson (2011) is the shooting algorithm they suggest for finding this set. The algorithm is initialized by letting $A_t = 0$ and the set $\mathcal{T}_t$ being empty. It proceeds as follows:

Step 1: Given A_t, compute the projected restricted policy rate $\tilde{r}_{t+\tau,t}$ for $\tau = 0, 1, \ldots, T$ from equation (3.30).
  (i) Record the first period $\tau_1 \geq 0$ for which $\tilde{r}_{t+\tau_1,t} + \bar{r} - r_{ZLB,t+\tau_1} < 0$ and which is *not* already an element of $\mathcal{T}_t$. Add this τ_1 to $\mathcal{T}_t$.
  (ii) If $n_t \geq 1$, use (3.38) to solve for the nonzero elements of A_t for all $\tau \in \mathcal{T}_t$, where $\alpha_{t+\tau_1,t} > 0$.

Step 2: Check and eliminate negative $\alpha_{t+\tau,t}$.
  (i) Record the first period $\tau_0 \in \mathcal{T}_t$ for which $\alpha_{t+\tau_0,t} < 0$. Delete τ_0 from $\mathcal{T}_t$ and set $\alpha_{t+\tau_0,t} = 0$.
  (ii) If $n_t \geq 1$, use (3.38) to solve for the nonzero elements of A_t, where $\alpha_{t+\tau_0,t} = 0$.
  (iii) Redo Step 2 until $\alpha_{t+\tau,t} \geq 0$ for all $\tau \in 0, 1, \ldots, T$.

Step 3: Redo Steps 1 and 2. Stop when
  (i) $\alpha_{t+\tau,t} \geq 0$ and $\tilde{r}_{t+\tau_1,t} + \bar{r} - r_{ZLB,t+\tau_1} \geq 0$ for $\tau = 0, 1, \ldots, T$ and the complementary slackness condition (3.36) holds; or

(ii) Eliminating $\alpha_{t+\tau_1,t}$ in Step 2 implies that $\tilde{r}_{t+\tau,t} + \bar{r} - r_{ZLB,t+\tau} < 0$ for some $\tau = 0, 1, \ldots, T$, i.e. a solution cannot be found.

Note that the requirement that τ_1 is not already an element of $\mathcal{T}_t$ in Step 1, part (i), is a slight modification of HLS version of the algorithm. It mainly serves to handle possible numerical issues. The algorithm should normally shoot forward one period at a time to find the positive elements in A_t that are needed to impose the ZLB. At times, the algorithm needs to expand the number of positive elements in A_t backward in time, since adding more and more positive elements in A_t moving forward will depress inflation and output and can thereby cause the unrestricted policy rate path to violate the ZLB for earlies time periods.

It is pointed out by HLS that a solution to equations (3.34)–(3.36) does not always exist. Under standard Taylor-type monetary policy rules, which are locally stable, it is possible that sufficiently adverse shocks along with an unfavorable initial state of the economy (X_t) may require A_t to have negative elements to impose the ZLB for the restricted policy rate. This violates the nonnegativity condition (3.35) for the anticipated shocks.

HLS also emphasize that the solution to (3.34)–(3.36) need not be unique. The forward-back shooting algorithm gives the minimum number of positive elements in A_t in the solution. However, it may be possible to find solutions where the ZLB binds for an extended period and where all elements of A_t are nonnegative.

3.4.6. *Stochastic Simulations*

When conducting stochastic simulations, the forward-back shooting algorithm needs to be run for each period t in the simulation sample, while the solution of the DSGE model with anticipated shocks in (3.28) does not change over time. Let s denote the current simulation where $s = 1, \ldots, \bar{S}$, where the latter integer is the total number of simulations. Also, let $\bar{T}$ be the maximum simulation horizon. Simulated valued of the variables in the stochastic simulations below are all given the superscript (s) to indicate the simulation number.

The following algorithm can be utilized to conduct stochastic simulations with a linear DSGE model subject to the nonlinear ZLB restriction:

Step 0: Set the current simulation to $s = 1$.

Step 1: Initialize with $A_t^{(s)} = MA_{t-1} = 0$. This means that it is either assumed that the ZLB is not binding in period $t - 1$, or that $\hat{r}_{t-1}$ satisfies the ZLB exactly.

Step 2: Draw structural shocks $\eta_t^{(s)}$ from an appropriate distribution, and determine $X_t^{(s)}$ from (3.28).

Step 3: Compute the equilibrium deviation $A_t^{(s)}$ using the forward-back shooting algorithm, and thereafter increase the time index t by one.

Step 4: Update $A_t^{(s)} = MA_{t-1}^{(s)}$, let $\mathcal{T}_t^{(s)}$ be determined from equation (3.37), and redo Steps 2–4 until $t > \bar{T}$.

Step 5: Reset the time index t to its initial value, increase s by one, and redo Steps 1–4 until $s > \bar{S}$.

When prior or posterior draws of the model parameters are investigated (see Sections 4 and 8, respectively), the DSGE model would need to be solved for each of these parameter draws. The above algorithm can easily be adapted to handle such cases.

In Step 2 it is conceivable that one may wish to apply some modification to the typically assumed $N(0, I_q)$ for the structural shocks. While the unconditional forecasts will be based on this assumption (see Section 12.1), the conditional forecasts require that both the mean and the covariance matrix are adapted and the details for the adaption depends on the selected conditioning method; see Section 12.2. Similarly, if the predictive distributions are re-centered on actual values, then the mean of the shock distribution will need to be adapted.

An important issue concerns what to do in the event that there is no solution for $A_t^{(s)}$ in Step 3. One option would be to redo Step 2, but the implication of this is that the distribution of the shocks is affected. That is, discarding certain $\eta_t^{(s)}$ draws means that we are truncating the shock distribution. Provided that the number of times this happens is small compared with the

total number of simulations, the effect on the distribution of the structural shocks is likely to be negligible. Still, when using this option it is advisable that the number of such "bad draws" is kept track of, including the time periods when they occur, and that the sample mean vector and the sample covariance matrix of these discarded draws is also recorded for each time period. The alternative to this would be to either use another method for solving the DSGE model subject to the ZLB, or to revise the model.

3.4.7. A Structural Form Representation of the Zero Lower Bound Solution

Like the structural form autoregressive representation of the solution in equation (3.7) of the DSGE model, the solution in (3.29) for the zero lower bound case with the Klein solver can also be expressed in such a form. Notice first that the P matrix can be decomposed as

$$P = \begin{bmatrix} P_\eta & P_Y & P_A \end{bmatrix},$$

where P_η has dimension $(p+1) \times q$, P_Y is $(p+1) \times (p+1)$, while P_A is $(p+1) \times (T+1)$. From equation (3.29) it follows that

$$Y_t = P_Y Y_{t-1} + P_\eta \eta_t + P_A A_t, \tag{3.39}$$

an autoregressive form of the solution for the forward-looking variables.

From the lower block of the system in (3.27) we have that

$$A_{22} E_t Y_{t+1} = B_{21} V_t + B_{22} Y_t. \tag{3.40}$$

In order to substitute for the expectation term, we make use of (3.29), noting that

$$E_t Y_{t+1} = P E_t V_{t+1}$$

$$= \begin{bmatrix} P_\eta & P_Y & P_A \end{bmatrix} \begin{bmatrix} 0 \\ Y_t \\ M A_t \end{bmatrix}$$

$$= P_Y Y_t + P_A M A_t.$$

Substituting for the expectation term on the left hand side of (3.40), taking the definition of V_t into account, and rearranging terms we obtain

$$\left(A_{22} P_Y - B_{22} \right) Y_t = B_{21,Y} Y_{t-1} + B_{21,\eta} \eta_t + \left(B_{21,A} - A_{22} P_A M \right) A_t$$

$$= \left(B_{21} - \begin{bmatrix} 0 & 0 & A_{22} P_A M \end{bmatrix} \right) V_t. \tag{3.41}$$

The matrix $B_{21,Y}$ has dimension $(p+1) \times (p+1)$, $B_{21,\eta}$ is $(p+1) \times q$, while P_A is $(p+1) \times (T+1)$. From this equation we may deduce that the P matrix satisfies

$$P = \left(A_{22} P_Y - B_{22} \right)^{-1} \left(B_{21} - \begin{bmatrix} 0 & 0 & A_{22} P_A M \end{bmatrix} \right).$$

Moreover, the first line of equation (3.41) can be written as

$$S_0 Y_t = S_1 Y_{t-1} + S_\eta \eta_t + S_A A_t.$$

In other words, we may treat (3.41) as a structural form autoregressive representation of the zero lower bound solution in (3.39).

3.5. YADA Code

YADA uses only Matlab *functions* for running the AiM procedures, whereas most Matlab implementations include *script* files. The main reason for this change is that the construction of various outputs can more easily be traced to a particular Matlab file when functions are used, while script files tend to hide a lot of variables, most of which are only needed locally. Moreover, inputs are also easier to keep track of, since they can be given local names in a function.

The main AiM functions in YADA for solving the DSGE model and setting up the output as required by the Kalman filter are: `AiMInitialize`, `AiMSolver`, and `AiMtoStateSpace`. A number

of other functions are also included for utilizing AiM, but these are not discussed here.[22] It should be noted that the computationally slowest function, `AiMInitialize`, needs only be run once for a given model specification. The other two main functions need to be run for each set of parameter values to be analysed by the code.

If the original model is given by a dynare model file, the dynare parser can be utilized by YADA. Like `AiMInitialize`, the function `DynareParser` needs only be run one as it produces the same AiM related files as the AiM parser. One advantage of the dynare approach is that models can be expressed in non-linear form, while the dynare parser linearizes or log-linearizes the model. For this case, it is important to keep in mind that non-linear models tend to imply that steady-state values of the model variables are included as parameters in the AiM output file for solving a DSGE model and that these steady-state values must be either calibrated or derived in the two parameter functions that can be used for each DSGE model in YADA. The names of these steady-state parameter will be set equal to the names of the state variables by YADA and they should not be specified as such in the dynare model file.

YADA also supports the Klein (2000) and Sims (2002) approaches to solving a DSGE model. The AiM parser, run through the `AiMInitialize` function, is still required for these approaches since we need to write the DSGE model on the structural form in (3.2). The Klein approach is handled with the function `KleinSolver`, while Sims' `gensys` solver is run via the function `SimsSolver`. Finaly, in the event that the model solver takes the zero lower bound into account, then YADA uses the function `ZLBKleinSolver`, while the forward-back shooting algorithm for computing the anticipated shocks is run in the function `ForwardBackShootingAlgorithm`. Finally, YADA also supports external model solvers. The only requirements that YADA imposes on such alternatives is that they are: (i) matlab functions; (ii) support the same input and output variables as the Klein and Sims solvers; (iii) as well as three additional input variables `StateVariableNames`, `StateEquationNames` and `StateShockNames`. The last three variables provoides an external solver with direct access to the names of the state variables, the state equations and the structural shocks.

3.5.1. `AiMInitialize`

The function `AiMInitialize` runs the AiM parser on `ModelFile`, a text file that sets up the DSGE model in a syntax that the AiM parser can interpret. YADA refers to this file as the AiM model file. If parsing is successful (the syntax of the `ModelFile` is valid and the model is otherwise properly specified), the AiM parser writes two Matlab files to disk. The first is the function `compute_aim_data.m` and the second the script file `compute_aim_matrices.m`. The latter file is then internally parsed by `AiMInitialize`, rewriting it as a function that accepts a structure `ModelParameters` as input, where the fields of the structure are simply the parameter names as they have been baptized in the AiM model file, and provides the necessary output. For example, if the model file has a parameter called `omega`, then the structure `ModelParameters` has a field with the same name, i.e., `ModelParameters.omega`.

The functions `compute_aim_data.m` and `compute_aim_matrices.m` are stored on disk in a sub-directory to the directory where the AiM model file is located. By default, the name of this directory depends only on the name of the model specification (which can be different from the AiM model file, since the latter can be shared by many model specifications). `AiMInitialize` therefore also takes the input arguments `NameOfModel` and (optionally) `OutputDirectory`.

`AiMInitialize` also runs the function `compute_aim_data.m` and stores the relevant output from this function in a mat-file located in the same directory as the `compute_aim_data.m` file. Finally, `AiMInitialize` provides as output the status of the AiM parsing, and the output given by the `compute_aim_data.m` function. The status variable is 0 when everything went OK; it is 1 if the parsing did not provide the required output; 2 if the number of data variables did

---

[22] Most, if not all, of these Matlab functions originate from the AiM implementation at the Federal Reserve System; see, e.g., Zagaglia (2005).

not match the number of stochastic equations; 3 if illegal parameter names were used;[23] and 4 if the number of lags (τ_L) is greater than 1. All output variables from `AiMInitialize` are required, while the required input variables are given by `ModelFile`, being a string vector containing the full path plus name and extension of the model file, and `NameOfModel`, a string vector containing the name of the model specification. The final input variable is optional and is locally called `OutputDirectory`, the directory where the AiM output is stored. The `NameOfModel` variable determines the name of the `mat`-file that is created when running the function `compute_aim_data.m`.

3.5.2. `AiMSolver`

The function `AiMSolver` attempts to solve the DSGE model. To this end it requires as inputs the `ModelParameters` structure (containing values for all model parameters), the number of AiM equations (`NumEq`, often being at least $p + q + 1$), the number of lags (`NumLag` being τ_L), the number of leads (`NumLead` being τ_U), and the numerical tolerance for AiM and the other DSGE model solvers (`AIMTolerance`).

As output the function provides a scalar `mcode` with information about the solvability properties of the DSGE model for the parameter values found in the `ModelParameters` structure. When a unique convergent solution exists the `mcode` variable returns 1, while other values reflect various problems with the selected parameters (see the `AiMSolver` file for details).

Given that a unique convergent solution exists, the solution matrices as well as the maximum absolute error (`MaxAbsError`) when computing the solution are calculated. The solution matrices are given by all the B_i's, provided in `BMatrix` ($[B_{\tau_L} \cdots B_1]$), and all the S_j's, returned as the matrix `SMatrix` ($[S_{\tau_L} \cdots S_1\ S_0]$). These matrices have dimensions `NumEq` $\times\ \tau_L$`NumEq` and `NumEq` $\times\ (\tau_L + 1)$`NumEq`, respectively. Since YADA only accepts DSGE models that have been specified such that $\tau_L = 1$, the dimensions of these matrices are not unnecessarily made larger than they need be.

Finally, the function yields the output variable `ModelEigenvalues`, a structure that contains information about the eigenvalues of the reduced form, i.e., the solution of the model.

3.5.3. `AiMtoStateSpace`

The function `AiMtoStateSpace` creates the F matrix for the state equation (5.2) based on the input matrix `BMatrix` and B_0 from `SMatrix`. Since the output from `AiMSolver` treats all equations in a similar fashion, the vectors z_t and η_t are both often included as separate equations. Hence, `NumEq` $\geq p + q$. The additional input variables `StateVariablePositions`, `StateEquationPositions`, and `StateShockPositions` are therefore needed to locate which rows and columns of `BMatrix` and `SMatrix` that contain the coefficients on the z and η variables. These input vectors are created with the YADA GUI.

3.5.4. `DynareParser`

The function `DynareParser` creates same named AiM output files as `AiMInitialize`, having exactly the same purpose. In addition, this function creates a dummy AiM file for the model which has not information about the model equations or the state/model variables, but which serves as a place holder. Hence, it should not be parsed by AiM once it exists.

3.5.5. `KleinSolver`

The function `KleinSolver` requires 7 input variables to perform its task, i.e., to solve the DSGE model with the Klein (2000) approach. The variables are: `ModelParameters`, `NumLead`, `StateVariablePositions`, `StateEquationPositions`, `StateShockPositions`, `AIMTolerance`,

---

[23] YADA has only reserved 4 names as illegal. First of all, in order to allow for parameters called `g` and `h` (matrix names in the function `compute_aim_matrices.m`) YADA temporarily renames them `YADAg` and `YADAh`, respectively, when it rewrites the file from a Matlab script file to a Matlab function. For this reason, parameters cannot be named `YADAg` and `YADAh`. Furthermore, the name `UserVariables` is reserved for passing user determined data to the parameter functions that YADA supports, while the name `YADA` is reserved for internally disseminating the state equation matrices to the measurement equation function; see Section 18.4.

and `OrderQZ`. The first two and the fifth input variable is identical to the same variables in `AiMSolver`, while the third and the fourth variable are used by `AiMtoStateSpace`. The last input variable is a boolean that is unity if the function `ordqz` is a built-in Matlab function, and zero otherwise. All Matlab version greater than or equal to version 7 have this function.

The function provides 3 required and 2 optional output variables. The required outputs are `F`, `B0`, and `mcode`. The first two are matrices on lagged state variables and current state shocks in the state-pace representation, i.e., the solution to the DSGE model and are therefore the same as the output variables from `AiMtoStateSpace`. The `mcode` variable is shared with `AiMSolver`, but supports slightly different values. The optional variables are `MaxAbsError` and `ModelEigenvalues` which are also provided by `AiMSolver`. The latter structure is now based on the generalized eigenvalues of the structural form of the model, and has fewer fields than the variable provided by `AiMSolver`.

3.5.6. `SimsSolver`

The function `SimsSolver` supports the same input and output variables as `KleinSolver`. It rewrites the structural form of the DSGE model into the Sims (2002) form in equation (3.17) and sends the matrices to `gensys`, the Sims solver. The function used by YADA for this is called `YADAgensys` and is a slight rewrite of Sims' original function. In particular, it makes it possible to run the Matlab function `ordqz` rather than `gensys`' own `qzdiv`. The built-in Matlab function is considerably faster than `qzdiv`, but is not included in older versions of Matlab.

3.5.7. `ZLBKleinSolver`

The function `ZLBKleinSolver` requires 10 input variables to solve a DSGE model with anticipated shocks in the monetary policy rule. The first 5 and last 2 variables are identical to the input variables for `KleinSolver`. The 6th through 8th input variables are given by the integers `T_ZLB`, `RtildePosition`, and `REqPosition`. The `T_ZLB` variable gives the length of the sample over which the zero lower bound may be binding, while `RtildePosition` gives the position of the policy rate in z_t, and `REqPosition` is the position of the monetary policy rule among the model equations.

The function gives 4 output variables: the matrices `G`, `C1`, and `P` from equations (3.28) and (3.29), and the integer `mcode`. Note that the number of rows of `P` is generally equal to $\tau_U(p+1)$, where τ_U is equal to `NumLead`. The `mcode` output variable is 1 if a unique convergent solution is located, -1 if there is no stable solution, -2 if there are too many large eigenvalues, and -3 if there are too few such eigenvalues.

3.5.8. `ForwardBackShootingAlgorithm`

The function requires 6 input variables: `Xt`, `At`, `ePG`, `Rzlb`, `Rbar`, and `AIMTolerance`. The vector `Xt` gives the initial values of the current shocks, lagged state variables, and the unrestricted policy rate of the vector V_t, while `At` is the vector of initial values if the anticipated shocks, i.e., the bottom part of V_t. The matrix `ePG` has dimension $(T+1) \times (T+q+p+2)$ and when multiplying this matrix with V_t, which has dimension $(T+q+p+2)$, it generates the $(T+1)$ projections of the restricted polic rate in equation (3.30). The vector `Rzlb` has dimension $(T+1)$ and it gives the path of the zero lower bound, while the vector `Rbar` has the same dimension and gives the path of the steady-state of the policy rate. Finally, the input variable `AIMTolerance` gives the numerical tolerance of the lower bound solution.

The function provides 3 output variables: `At`, `status`, and `AlgorithmInformation`. The first is the vector of anticipated shocks, while the second output is a boolean that takes the value 1 if a solution is found, and 0 otherwise. The last output is a 5-dimensional vector with information collected when running the algorithm. The first element is unity if the solution is bad (invertibility problem when trying to solve for the anticipated shocks) and 0 otherwise; the second element is unity of there is no solution based on Step 3, part (ii) of the algorithm, and 0 otherwise; the third element is unity if the complementary slackness condition is not satisfies and 0 otherwise; the fourth element gives the number of times the algorithm has been run for

a given time period t; and the last gives you the maximum number of times the algorithm could have been executed for the same time period.

4. Prior and Posterior Distributions

It has been pointed out by, e.g., Fernández-Villaverde and Rubio-Ramírez (2004) that a DSGE model is always *false* since such an economy is an artificial construction. This simple understanding generates two important challenges for econometric analysis. First, how to select values for the so called *deep* parameters of the model, i.e., parameters that describe preferences, technology, policy rules, etc. Second, how to compare two or more possibly non-nested and misspecified models. A Bayesian approach to dealing with these difficult tasks is in principle easy. Parameters are given values through their posterior distribution, which is linked to prior information and the observed data through Bayes theorem. Model comparisons are performed through the use of the posterior odds ratio, i.e., the ratio of marginal density of the data for the models times the prior odds ratio.

4.1. *Bayes Theorem*

Let y_t denote the observed variables, a vector of dimension n. Furthermore, the sample is given by $t = 1, \ldots, T$ and we collect the data into the $n \times T$ matrix Y. For simplicity we here neglect any exogenous or predetermined variables as they do not matter for the exposition.

The density function for a random matrix Y conditional on θ is given by $p(Y|\theta)$, where θ is a vector of parameters. The joint prior distribution of θ is denoted by $p(\theta)$. From Bayes theorem we then know that the posterior distribution of θ, denoted by $p(\theta|Y)$, is related to these functions through

$$p(\theta|Y) = \frac{p(Y|\theta)p(\theta)}{p(Y)}, \tag{4.1}$$

where $p(Y)$ is the marginal density of the data, defined from

$$p(Y) = \int_{\theta \in \Theta} p(Y|\theta)p(\theta)d\theta, \tag{4.2}$$

with Θ being the support of θ. Since $p(Y)$ is a constant when Y has been realized we know that the posterior density of θ is proportional to the product $p(Y|\theta)p(\theta)$. Hence, if we can characterize the distribution of this product we would know the posterior distribution of θ. For complex models like those belonging to the DSGE family this characterization is usually not possible. Methods based on Markov Chain Monte Carlo (MCMC) theory can instead be applied to generate draws from the posterior.

Still, without having to resort to such often time consuming calculations it should be noted that the mode of the posterior density can be found by maximizing the product $p(Y|\theta)p(\theta)$. Since this product is usually highly complex, analytical approaches to maximization are ruled out from the start. Instead the posterior mode, denoted by $\tilde{\theta}$, can be estimated using numerical methods. In Section 4.2 we provide details on the individual prior distributions for the elements of θ that YADA supports. Through the independence assumption, the joint prior $p(\theta)$ is simply the product of these individual (and marginal) prior densities.[24] The computation of the likelihood function for any given value of θ is thereafter discussed in Section 5.

4.2. *Prior Distributions*

In the Bayesian DSGE framework it is usually assumed that the parameters to be estimated, denoted here by θ, are a priori independent. For parameters that have support $\mathbb{R}$, the prior distribution is typically Gaussian. Parameters that instead have support $\mathbb{R}_+$ tend to have either gamma or inverted gamma prior distributions, while parameters with support (c, d), where $d > c$ and both are finite, are usually assumed to have beta prior distributions; see, e.g., An and Schorfheide (2007). In some cases, e.g., Adolfson et al. (2007b), the distribution may be left truncated normal for a certain parameter. The density functions of these distributions as

---

[24] If one wishes to make use of parameters that are a priori dependent, one may formulate parameter functions and treat elements of θ as auxiliary parameters for the ones of interest. YADA supports such parameter functions and, hence, an assumption of a priori independent parameters is *not* restrictive.

well as of the uniform, the Student-t (and Cauchy), the logistic, the Gumbel, and the Pareto distributions are given below. Some of these have, to my knowledge, not been used in the empirical DSGE modelling literature, but it seems reasonable to, e.g., consider using a Student-t or a logistic as an alternative to the normal prior.

YADA can also support a number of additional distributions through parameter transformation functions. These include but are not limited to the Weibull and the Snedecor (better known as the F or Fisher) distributions.[25] The densities of such additional distribution are derived through a useful result which directly relates the density of a monotonic transformation of a continuous random variable to the density of that variable. Next, the gamma and beta functions are presented since they often appear in the integration constants of certain important distributions. Thereafter, we examine the prior distributions which are directly supported by YADA, focusing on the specific parameterizations used and relate these parameters to moments of the distributions. Furthermore, we discuss some distributions which can be derived from the directly supported ones, and which are therefore indirectly supported. In addition, we also reflect on some interesting special cases of the directly supported priors. The section continues with a dicussion about random number generators. Before the YADA based code is discussed, the section also considers so called *system priors* which allow the user to include a prior on some system or model feature which is otherwhise solely determined from the solution of the model. One such example is the population standard deviation of an observed variable conditional on the parameters.

4.2.1. *Monotonic Functions of Continuous Random Variables*

Suppose that a continuous random variable x has density function $p_X(x)$. The general principle for determining the density of a random variable $z = f(x)$, where $f(\cdot)$ is *monotonic* (order preserving), is the following:

$$p_Z(z) = \left| \frac{1}{f'\big(f^{-1}(z)\big)} \right| p_X\big(f^{-1}(z)\big) \tag{4.3}$$

The derivative of f is given by $dz/dx = f'(\cdot)$, while $f^{-1}(\cdot)$ is the inverse function, i.e., $x = f^{-1}(z)$; see, e.g., Bernardo and Smith (2000, p. 111). This powerful result makes it straightforward to determine the density of any monotonic transformation of a random variable.

An intuition for the result in equation (4.3) can be obtained by recalling that the integral of the density of x over its domain is equal to unity. To calculate this integral we multiply the density of x by dx and then perform the integration. When integrating over the domain of z we instead multiply the density of x by dz. To ensure that the integral is still equal to unity, we must therefore multiply this expression by $|dx/dz| = |1/(dz/dx)|$, where the absolute value guarantees that the sign of the integral does not change.

Since YADA supports functions of parameters, the relationship in equation (4.3) means that YADA indirectly supports all prior distributions where the corresponding random variable can be expressed as a monotonic function of one of the basic priors directly supported by YADA. Furthermore, the relationship between the joint density and the conditional and the marginal densities (i.e., the foundation for Bayes Theorem) makes it possible to further enhance the set of density functions which YADA can indirectly support to include also mixtures and multivariate priors.

---

[25] YADA can indirectly support multivariate extensions of the distributions. For example, one may wish to have a Dirichlet (multivariate beta), a multivariate normal prior, or an inverted Wishart prior for a vector of parameters. For these cases, parameter transformation functions can be used to allow for a multivariate prior. In the case of a multivariate normal prior, we would define the prior as univariate normal priors for auxiliary parameters, e.g., for one "conditional" and for one "marginal" parameter, while the transformation would be applied to the conditional parameter. Similarly, the Dirichlet distribution is supported through univariate beta priors for auxiliary parameters; see, e.g., Connor and Mosimann (1969). In other words, YADA can support multivariate priors through its use of a parameter transformation function; see Section 18.3.

4.2.2. *The Gamma and Beta Functions*

The *gamma* function is defined by the following integral identity:

$$\Gamma(a) = \int_0^\infty x^{a-1} \exp(-x) dx, \quad a > 0. \tag{4.4}$$

In many cases integer or half integer values of a are used. Here it is useful to know that $\Gamma(1) = 1$, while $\Gamma(1/2) = \sqrt{\pi}$. Integration by parts of (4.4) gives for $a > 1$ that $\Gamma(a) = (a-1)\Gamma(a-1)$.

The *beta* function $\beta(a, b)$ for $a, b > 0$ is defined as:

$$\beta(a, b) = \int_0^1 x^{a-1}(1-x)^{b-1} dx. \tag{4.5}$$

It is related to the gamma function through the following relationship:

$$\beta(a, b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a + b)}. \tag{4.6}$$

Matlab contains two useful functions for dealing with the gamma function. One is `gamma` which works well for relatively small values of a. The other is `gammaln` which calculates the natural logarithm of the gamma function and works well also with large values of a. Similarly, for the beta function Matlab provides the functions `beta` and `betaln`.

4.2.3. *Gamma, χ^2, Exponential, Erlang and Weibull Distributions*

A random variable $z > 0$ has a *gamma distribution* with shape parameter $a > 0$ and scale parameter $b > 0$, denoted by $z \sim G(a, b)$ if and only if its pdf is given by

$$p_G(z|a, b) = \frac{1}{\Gamma(a)b^a} z^{a-1} \exp\left(\frac{-z}{b}\right). \tag{4.7}$$

It is worthwhile to keep in mind that if $z \sim G(a, b)$ and $y = z/b$, then $y \sim G(a, 1)$. Furthermore, $E[z] = ab$, while $E[(z - ab)^2] = ab^2$; see, e.g., Bauwens, Lubrano, and Richard (1999) or Zellner (1971). If $a > 1$, then the pdf has a unique mode at $\tilde{\mu}_\Gamma = b(a - 1)$. The difference between the mean and the mode is b, implying that the mean is greater than the mode. Furtermore, skewness is equal to $2/\sqrt{a}$, while excess kurtosis is $6/a$.

Letting μ_Γ and σ_Γ^2 denote the mean and the variance, respectively, we can directly see that

$$a = \frac{\mu_\Gamma^2}{\sigma_\Gamma^2}, \quad b = \frac{\sigma_\Gamma^2}{\mu_\Gamma}. \tag{4.8}$$

In practise, most economists (and econometricians) are probably more comfortable formulating a prior in terms of the mean and the standard deviation, than in terms of a and b.

The mode can, when it exists, also be expressed in terms of the mean and the variance parameters. Equation (4.8) and the expression for the mode give us

$$\tilde{\mu}_\Gamma = \mu_\Gamma - \frac{\sigma_\Gamma^2}{\mu_\Gamma}.$$

The mode therefore exists when $\mu_\Gamma^2 > \sigma_\Gamma^2$, i.e., when the mean is greater than the standard deviation.

A few examples of the gamma distribution have been plotted in the upper left panel of Figure 1 (with the lower bound being equal to zero). The mean has been set to 0.2 in three cases while the standard deviation takes the values $(0.05, 0.1, 0.2)$. For the two cases when the mean is greater than the standard deviation the mode exists, while $\mu_\Gamma = \sigma_\Gamma$ results in $a = 1$ so that the mode does not exist. For the cases when the mode exists, the height of the density is negatively related the standard deviation. Furthermore, for a given mean, the mode lies closer to the mean as the standard deviation decreases since the ratio $\sigma_\Gamma^2/\mu_\Gamma$ becomes smaller. Moreover, since a lower standard deviation for fixed mean implies that a increases we also know that skewness decreases. Hence, the gamma distribution with mean 0.2 and standard deviation 0.1 (blue solid

FIGURE 1: Examples of gamma, inverted gamma, beta, normal and left truncated normal distributions.

Gamma Distribution

Inverted Gamma Distribution

Beta Distribution

Normal and Left Truncated Normal Distributions
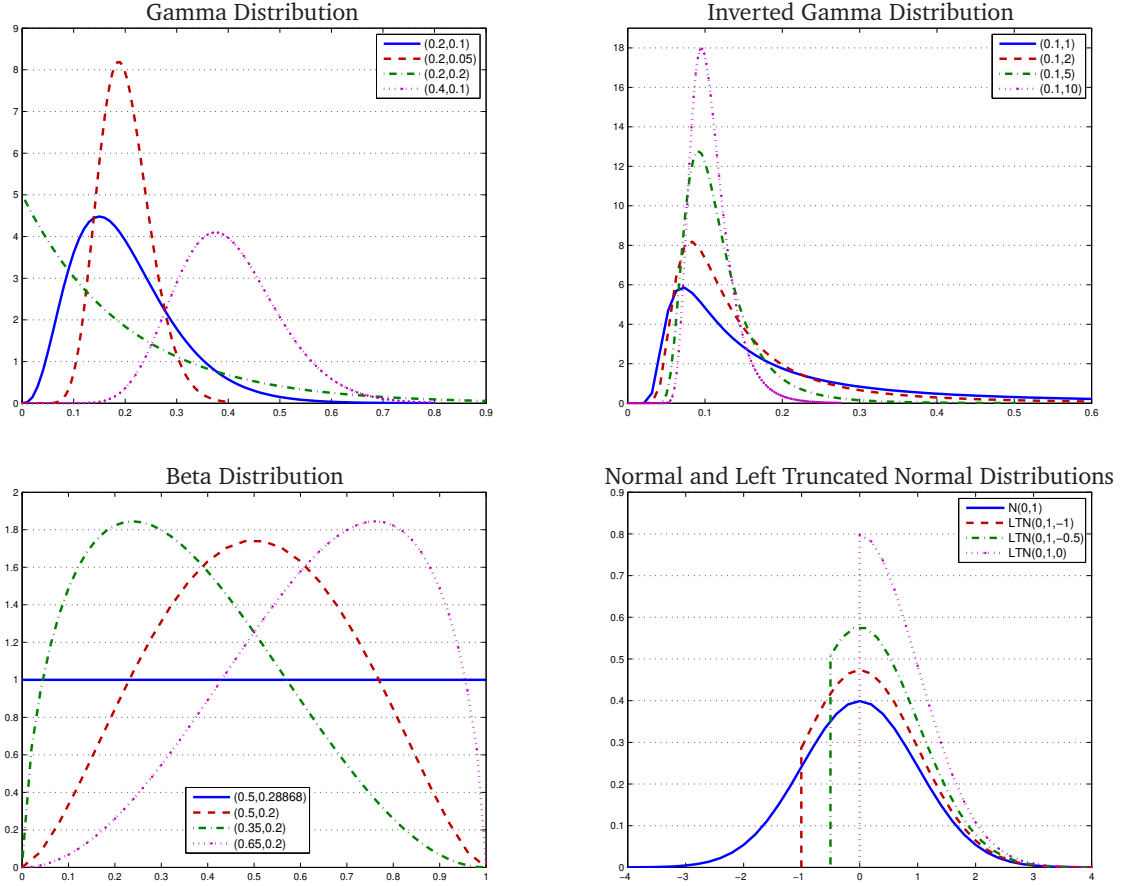
line in Figure 1) is more skewed than the gamma with mean 0.2 and standard deviation 0.05 (red dashed line).

The last example covers the case when the mean increases while the standard deviation is fixed, i.e., the blue solid line relative to the magenta colored dotted line. The distance between the mode and the mean now also decreases since the ratio $\sigma_\Gamma^2/\mu_\Gamma$ becomes smaller. In terms of the shape and scale parameters a and b, we know from (4.8) that a increases with μ_Γ while b decreases. Moreover, since a increases it follows from the skewness expression that it decreases.

One special case of the gamma distribution is the $\chi^2(q)$. Specifically, if $z \sim \chi^2(q)$ then this is equivalent to stating that $z \sim G(q/2, 2)$, with mean q and variance $2q$. The mode of this distribution exists and is unique when $q \geq 3$ and is equal to $q - 2$.

Another special case of the gamma distribution is the *exponential distribution*. This is obtained by letting $a = 1$ in (4.7). With $z \sim \mathcal{E}(b) \equiv G(1, b)$, we find that the mean is equal to $\mu_{\mathcal{E}} = b$ and the variance is $\sigma_{\mathcal{E}}^2 = b^2$.

Similarly, the *Erlang distribution* is the special case of the gamma when a is an integer. For this case we have that $\Gamma(a) = (a - 1)!$, where ! is the factorial function. The parameterization of the Erlang density is usually written in terms of $\lambda = 1/b$, a rate parameter.

YADA can also support the *Weibull distribution* through the gamma prior and the so called *file with parameters to update*; see Section 18.3. Specifically, if $x \sim G(1, 1) \equiv \mathcal{E}(1)$ and $x = (z/a)^b$ with $a, b > 0$, then z has a Weibull distribution with scale parameter a and shape parameter b, i.e. $z \sim W(a, b)$. In YADA one would specify a prior for the random variable x and compute

$z = ax^{1/b}$ in the file with parameters to update. The density function is now

$$p_W(z|a, b) = \frac{b}{a^b} z^{b-1} \exp\left(-\left[\frac{z}{a}\right]^b\right), \qquad z > 0, \tag{4.9}$$

since dx/dz is positive and given by the term $(b/a^b)z^{b-1}$.[26]

The mean of the Weibull distribution is $\mu_W = a\Gamma(1 + (1/b))$, the variance is $\sigma_W^2 = a^2[\Gamma(1 + (2/b)) - (\Gamma(1 + (1/b)))^2]$, whereas the mode exists and is given by $\tilde{\mu}_W = a((b-1)/b)^{(1/b)}$ when $b > 1$.

4.2.4. *Inverted Gamma and Inverted Wishart Distributions*

A random variable $z > 0$ has an *inverted gamma distribution* with shape parameter $a > 0$ and scale parameter $b > 0$, denoted by $z \sim IG(a, b)$, if and only if its pdf is given by

$$p_{IG}(z|a, b) = \frac{2}{\Gamma(a)b^a} z^{-(2a+1)} \exp\left(\frac{-1}{bz^2}\right). \tag{4.10}$$

This pdf has a unique mode at $\tilde{\mu}_{IG} = (2/(b(2a+1)))^{1/2}$; cf. Zellner (1971).[27] Moreover, the statement $z \sim IG(a, b)$ is equivalent to $z = 1/\sqrt{x}$ where $x \sim G(a, b)$.

The inverted gamma distribution is an often used prior for a standard deviation parameter. Letting $\sigma = z$, $a = q/2$, and $b = 2/qs^2$, we get

$$p_{IG}(\sigma|s, q) = \frac{2}{\Gamma(q/2)} \left(\frac{qs^2}{2}\right)^{q/2} \sigma^{-(q+1)} \exp\left(\frac{-qs^2}{2\sigma^2}\right), \tag{4.11}$$

where $s, q > 0$. The parameter q is typically an integer (degrees of freedom), but is only restricted such that $q > 0$, while $s > 0$ is a location parameter. This pdf has a unique mode at $\tilde{\mu}_{IG} = s(q/(q+1))^{1/2}$. Hence, the mode is below s for finite q and converges to s when $q \to \infty$.

The moments of this distribution exists when q is sufficiently large. For example, if $q \geq 2$, then the mean is

$$\mu_{IG} = \frac{\Gamma((q-1)/2)}{\Gamma(q/2))} \left(\frac{q}{2}\right)^{1/2} s,$$

while if $q \geq 3$ then the variance is given by

$$\sigma_{IG}^2 = \frac{q}{q-2} s^2 - \mu_{IG}^2.$$

Hence, both the mean and the variance are decreasing functions of q; see Zellner (1971) for details.

Moreover, if $q \geq 4$ then the third moment also exists. The exact expression can be found in Zellner (1971, eq. (A.48)), but since that expression is very messy an alternative skewness measure may be of interest. One such simpler alternative is the *Pearson measure of skewness*, defined as the mean minus the mode and divided by the standard deviation. For the inverted gamma we here find that

$$S_{P,IG} = \frac{\mu_{IG} - \tilde{\mu}_{IG}}{\sigma_{IG}} = \frac{\dfrac{\Gamma\left((q-1)/2\right)}{\Gamma(q/2)}\left(\dfrac{q}{2}\right)^{1/2} - \left(\dfrac{q}{q+1}\right)^{1/2}}{\left[\dfrac{q}{q-2} - \left(\dfrac{\Gamma\left((q-1)/2\right)}{\Gamma(q/2)}\right)^2 \dfrac{q}{2}\right]^{1/2}}, \qquad q \geq 3.$$

---

[26] With $p_X(x) = \exp(-x)$ and $z = ax^{1/b}$ we find from equation (4.3) that $f^{-1}(z) = (z/a)^b$. Moreover, $f'(x) = (a/b)x^{(1-b)/b}$ so that $f'(f^{-1}(z)) = (a^b/b)z^{1-b}$. By multiplying terms we obtain the density function in (4.9). Notice that $dx/dz = 1/f'(f^{-1}(z))$, the Jacobian of the transformation z into x.

[27] Bauwens, Lubrano, and Richard (1999) refer to the inverted gamma distribution as the inverted gamma-1 distribution. The inverted gamma-2 distribution is then defined for a variable $x = z^2$, where z follows an inverted gamma-1 distribution.

This expression is positive for finite q and the inverted gamma distribution is therefore right-skewed. As q gets large, the skewness measure $S_{P,IG} \to 0$. Both the numerator and the denominator are decreasing in q and for $q > 5$ the ratio is decreasing.[28]

A few examples of the inverted gamma distribution have been plotted in the upper right panel of Figure 1. The location parameter is for simplicity kept fixed at 0.1, while the number of degrees of freedom are given by $q = (1, 2, 5, 10)$. It can be seen that the height of the density increases as q becomes larger.[29] Moreover, the variance is smaller while skewness appears to be lower for $q = 10$ than for $q = 5$. The latter is consistent with the results for the Pearson measure of skewness, $S_{P,IG}$.

Another parameterization of the inverted gamma distribution is used in the software developed by Adolfson et al. (2007b). Letting $a = d/2$ and $b = 2/c$, the pdf in (4.10) can be written as:

$$p_{IG}(z|c, d) = \frac{2}{\Gamma(d/2)} \left(\frac{c}{2}\right)^{d/2} z^{-(d+1)} \exp\left(\frac{-c}{2z^2}\right).$$

The mode of this parameterization is found by setting $\tilde{\mu}_{IG} = (c/(d+1))^{1/2}$. With $c = qs^2$ and $d = q$ this parameterization is equal to that in equation (4.11) with $z = \sigma$.

A multivariate extension of the inverted gamma distribution is given by the *inverted Wishart distribution*. Specifically, when a $p \times p$ positive definite matrix Ω is inverted Wishart, denoted by $\Omega \sim IW_p(A, v)$, its density is given by

$$p(\Omega) = \frac{|A|^{v/2}}{2^{vp/2} \pi^{p(p-1)/4} \Gamma_p(v)} |\Omega|^{-(v+p+1)/2} \exp\left(-\frac{1}{2} \text{tr}\left[\Omega^{-1} A\right]\right), \tag{4.12}$$

where $\Gamma_b(a) = \prod_{i=1}^{b} \Gamma([a-i+1]/2)$ for positive integers a and b, with $a \geq b$, and $\Gamma(\cdot)$ being the gamma function in (4.4). The parameters of this distribution are given by the positive definite location matrix A and the degrees of freedom parameter $v \geq p$. The mode of the inverted Wishart is given by $(1/(p + v + 1))A$, while the mean exists if $v \geq p + 2$ and is then given by $E[\Omega] = (1/(v - p - 1))A$; see, e.g., Zellner (1971, Appendix B.4) and Bauwens et al. (1999, Appendix A) for details.

Suppose for simplicity that $p = 2$ and let us partition Ω and A conformably

$$\Omega = \begin{bmatrix} \Omega_{11} & \Omega_{12} \\ \Omega_{12} & \Omega_{22} \end{bmatrix}, \qquad A = \begin{bmatrix} A_{11} & A_{12} \\ A_{12} & A_{22} \end{bmatrix}.$$

It now follows from, e.g., Bauwens et al. (1999, Theorem A.17) that:

(1) Ω_{11} is independent of Ω_{12}/Ω_{11} and of $\Omega_{22\cdot1} = \Omega_{22} - \Omega_{12}^2/\Omega_{11}$;
(2) $\Omega_{11} \sim IW_1(A_{11}, v-1)$;
(3) $\Omega_{12}/\Omega_{11}|\Omega_{22\cdot1} \sim N(A_{12}/A_{11}, \Omega_{22\cdot1}/A_{11})$, where $N(\mu, \sigma^2)$ denotes the univariate normal distribution with mean μ and variance σ^2 (see, e.g., Section 4.2.6 for details); and
(4) $\Omega_{22\cdot1} \sim IW_1(A_{22\cdot1}, v)$, where $A_{22\cdot1} = A_{22} - A_{12}^2/A_{11}$.

From these results it is straightforward to deduce that the multivariate random matrix Ω may be represented by three independent univariate random variables. Specifically, let

$$\sigma_1 \sim IG(s_1, v-1), \quad \sigma_2 \sim IG(s_2, v), \quad \text{and} \quad \rho \sim N(0, 1). \tag{4.13}$$

With $\Omega_{11} = \sigma_1^2$ and $A_{11} = (v-1)s_1^2$, it can be shown that $\Omega_{11} \sim IW_1(A_{11}, v-1)$ by evaluating the inverted gamma density at Ω_{11}, A_{11}, and multiplying this density with the inverse of the derivative of Ω_{11} with respect to σ_1, i.e., by $(1/2)\Omega_{11}^{-1/2}$; recall equation (4.3) in Section 4.2.1. Furthermore, letting $\Omega_{22\cdot1} = \sigma_2^2$ and $A_{22\cdot1} = vs_2^2$ we likewise find that $\Omega_{22\cdot1} \sim IW_1(A_{22\cdot1}, v)$. Trivially, we also know that $\Omega_{12}/\Omega_{11} = A_{12}/A_{11} + \sqrt{\Omega_{22\cdot1}/A_{11}}\rho$ implies that $\Omega_{12}/\Omega_{11}|\Omega_{22\cdot1} \sim N(A_{12}/A_{11}, \Omega_{22\cdot1}/A_{11})$.

---

[28] Skewness is defined as the third standardized central moment, i.e., the third central moment divided by the standard deviation to the power of 3. There is no guarantee that the sign of this measure always corresponds to the sign of the Pearson measure.

[29] This can also be seen if we let $\sigma = \tilde{\mu}_{IG}$ in equation (4.11).

Together, these results therefore ensure that $\Omega \sim IW_2(A, \nu)$, where

$$\Omega_{11} = \sigma_1^2,$$

$$\Omega_{12} = \Omega_{11} \left[\frac{A_{12}}{A_{11}} + \sqrt{\frac{\sigma_2^2}{A_{11}}} \rho \right],$$

$$\Omega_{22} = \sigma_2^2 + \frac{\Omega_{12}^2}{\Omega_{11}}.$$

It may also be noted that one can derive an inverted Wishart distribution for the general $p \times p$ case based on p univariate inverted gamma random variables and $p(p-1)/2$ univariate standard normal random variables, and where all univariate variables are independent. The precise transformations needed to obtain Ω from these univariate variables can be determined by using Theorem A.17 from Bauwens et al. (1999) in a sequential manner.

4.2.5. *Beta, Snedecor (F), and Dirichlet Distributions*

A random variable $c < x < d$ has a *beta distribution* with parameters $a > 0$, $b > 0$, $c \in \mathbb{R}$ and $d > c$, denoted by $x \sim B(a, b, c, d)$ if and only if its pdf is given by

$$p_B(x|a, b, c, d) = \frac{1}{(d-c)\beta(a,b)} \left(\frac{x-c}{d-c} \right)^{a-1} \left(\frac{d-x}{d-c} \right)^{b-1}. \tag{4.14}$$

The standardized beta distribution can directly be determined from (4.14) by defining the random variable $z = (x-c)/(d-c)$. Hence, $0 < z < 1$ has a beta distribution with parameters $a > 0$ and $b > 0$, denoted by $z \sim B(a, b)$ if and only if its pdf is given by

$$p_{SB}(z|a, b) = \frac{1}{\beta(a,b)} z^{a-1} (1-z)^{b-1}. \tag{4.15}$$

For $a, b > 1$, the mode of (4.15) is given by $\tilde{\mu}_{SB} = (a-1)/(a+b-2)$. Zellner (1971) provides general expressions for the moments of the beta pdf in (4.15). For example, the mean of the standardized beta is $\mu_{SB} = a/(a+b)$, while the variance is $\sigma_{SB}^2 = ab/((a+b)^2(a+b+1))$.

The a and b parameters of the beta distribution can be expressed as functions of the mean and the variance. Some algebra later we find that

$$a = \frac{\mu_{SB}}{\sigma_{SB}^2} \left[\mu_{SB}(1 - \mu_{SB}) - \sigma_{SB}^2 \right],$$

$$b = \frac{(1 - \mu_{SB})}{\mu_{SB}} a. \tag{4.16}$$

From these expressions we see that a and b are defined from μ_{SB} and σ_{SB}^2 when $\mu_{SB}(1 - \mu_{SB}) > \sigma_{SB}^2 > 0$ with $0 < \mu_{SB} < 1$.

Letting μ_B and σ_B^2 be the mean and the variance of $x \sim B(a, b, c, d)$, it is straightforward to show that:

$$\mu_B = c + (d - c)\mu_{SB},$$

$$\sigma_B^2 = (d - c)^2 \sigma_{SB}^2. \tag{4.17}$$

This means that we can express a and b as functions of μ_B, σ_B, c, and d:

$$a = \frac{(\mu_B - c)}{(d-c)\sigma_B^2} \left[(\mu_B - c)(d - \mu_B) - \sigma_B^2 \right],$$

$$b = \frac{(d - \mu_B)}{(\mu_B - c)} a. \tag{4.18}$$

The conditions that $a > 0$ and $b > 0$ means that $c < \mu_B < d$, while $(\mu_B - c)(d - \mu_B) > \sigma_B^2$. The mode still exists when $a, b > 1$ and is in that case given by $\tilde{\mu}_B = c + (d - c)\tilde{\mu}_{SB} = c + (d - c)(a - 1)/(a + b - 2)$.

Moreover, skewness is given by:

$$S_B = \frac{2ab(b-a)}{(a+b)^3(a+b+1)(a+b+2)\left[\frac{ab}{(a+b)^2(a+b+1)}\right]^{3/2}}.$$

Hence, if $a = b$, then the beta distribution is symmetric, while $b > a$ ($a > b$) implies that it is right-skewed (left-skewed). Since $b > a$ implies that $\mu_B < (d+c)/2$, it follows that the mean lies below the mid-point of the range $[c, d]$.

The beta distribution is related to the gamma distribution in a particular way. Suppose $x \sim G(a, 1)$ while $y \sim G(b, 1)$. As shown by, e.g., Bauwens, Lubrano, and Richard (1999, Theorem A.3), the random variable $z = x/(x + y) \sim B(a, b)$.

The beta distribution is plotted in the lower left panel of Figure 1 for a few examples. In all cases the lower bound $c = 0$ and the upper bound $b = 1$. For the baseline case the mean is 0.5 while the standard deviation is $1/\sqrt{12} \approx 0.28868$ and this is displayed as the horizontal blue solid line in the figure. This means that the beta distribution is identical to the uniform distribution. When the standard deviation drops, the distribution becomes bell shaped (red dashed line) and since the mean is exactly at the center between the lower and the upper bound, the distribution becomes symmetric; cf. equal (4.17) where $a = b$. As noted above, when the mean of the beta distribution is smaller (greater) than the mid-point in the support, the the distribution is right-skewed (left-skewed) since $b > a$ (a>b).

The beta distribution is also related to the *Snedecor* or F (Fisher) *distribution*. For example, suppose that $x \sim B(a/2, b/2)$ with a, b being positive integers. Then $z = bx/(a(1-x))$ can be shown to have an $F(a, b)$ distribution; cf. Bernardo and Smith (2000, Chapter 3). That is,

$$p_F(z|a, b) = \frac{a^{(a/2)}b^{(b/2)}}{\beta(a/2, b/2)} z^{(a/2)-1}\left(b + az\right)^{(a+b)/2}, \qquad z > 0.$$

The mean of this distribution exists if $b > 2$ and is then $\mu_F = b/(b-2)$. The mode exists and is unique with $\tilde{\mu}_F = (a-2)b/(a(b+2))$ when $a > 2$. Finally, if $b > 4$ then the variance exists and is given by $\sigma_F^2 = 2b^2(a+b-2)/(a(b-4)(b-2)^2)$.

Although YADA does not directly support the F distribution, the combination of the beta prior and the *file with parameters to update* (see Section 18.3) makes it possible to indirectly support this as a prior.

The multivariate extension of the beta distribution is the so called *Dirichlet distribution*. The marginal distribution for one element of a Dirichlet distributed random vector is the beta distribution; cf. Gelman, Carlin, Stern, and Rubin (2004, Appendix A). YADA does not directly support prior distributions that include dependence between parameters. However, by using the *file with parameters to update* (see Section 18.3) the user can circumvent this "restriction".

Specifically, suppose $x_i \sim B(a_i, b_i)$ are mutually independent for $i = 1, \ldots, k-1$ with $k \geq 3$. Defining $z_i = x_i \prod_{j=1}^{i-1}(1 - x_j)$ for $i = 2, \ldots, k-1$, $z_1 = x_1$, and assuming that $b_{i-1} = a_i + b_i$ for $i = 2, \ldots, k-1$, it is shown in Connor and Mosimann (1969) that the density for $(z_1, \ldots, z_{k-1})$ is given by

$$p_D(z_1, \ldots, z_{k-1}|\alpha_1, \ldots, \alpha_k) = \frac{\Gamma(\sum_{i=1}^{k} \alpha_i)}{\prod_{i=1}^{k} \Gamma(\alpha_i)} \prod_{i=1}^{k} z_i^{\alpha_i-1},$$

where $z_k = 1 - \sum_{i=1}^{k-1} z_i$, $\alpha_i = a_i$ for $i = 1, \ldots, k-1$ and $\alpha_k = b_{k-1}$. This is the density function of the Dirichlet distributed vector $z = (z_1, \ldots, z_{k-1}) \sim D(\alpha_1, \ldots, \alpha_k)$.

The first two moments of the standardized Dirichlet distribution exist and are, for example, given in Gelman et al. (2004, Appendix A). Specifically, let $\alpha_0 = \sum_{j=1}^{k} \alpha_j$. The mean of z_i is $\mu_{D,i} = \alpha_i/\alpha_0$, while the mode is equal to $\tilde{\mu}_{D,i} = (\alpha_i - 1)/(\alpha_0 - k)$ when it exists. The variance

and the covariance are

$$\sigma_{D,i}^2 = \frac{\alpha_i(\alpha_0 - \alpha_i)}{\alpha_0^2(\alpha_0 + 1)},$$

$$\sigma_{D,ij} = \frac{-\alpha_i\alpha_j}{\alpha_0^2(\alpha_0 + 1)}.$$

From the expressions for the mean and the variance of the Dirichlet, the relation to the mean and the variance of the (univariate) beta distribution can be seen.

To use the Dirichlet prior in YADA, the user should setup a prior for the auxiliary parameters x_i and compute z_i in the file with parameters to update. Cases when the Dirichlet may be of interest include models that have parameter pairs than are restricted to, e.g., be positive and to sum to something less than unity.[30]

4.2.6. *Normal and Log-Normal Distributions*

For completeness, the Gaussian density function is also provided. Specifically, a random variable z is said to have a *normal distribution* with location parameter $\mu \in \mathbb{R}$ and scale parameter $\sigma > 0$, denoted by $z \sim N(\mu, \sigma^2)$, if and only if its pdf is given by

$$p_N(z|\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(z-\mu)^2}{2\sigma^2}\right). \tag{4.19}$$

The mode of this density is $z = \mu$, while the mean is also equal to μ and the variance is σ^2.

YADA does not directly support the log-normal as a prior distribution. Nevertheless, log-normal priors can be used by combining the normal distribution with the *file with parameters to update*; cf. Section 18.3. That is, the normal prior is specified for the random variable x, while $z = \exp(x)$ is given in the file with parameters to update.

The density function of the *log-normal distribution* is

$$p_{LN}(z|\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} z^{-1} \exp\left(-\frac{1}{2\sigma^2}\left(\ln z - \mu\right)^2\right), \qquad z > 0.$$

The mean of the log-normal distribution is $\mu_{LN} = \exp(\mu + (\sigma^2/2))$, the variance is $\sigma_{LN}^2 = \exp(2\mu + \sigma^2)(\exp(\sigma^2) - 1)$, while the mode is $\tilde{\mu}_{LN} = \exp(\mu - \sigma^2)$; see Gelman et al. (2004).

It is also possible to compute the μ and σ parameters from μ_{LN} and σ_{LN}. In this case we have that:

$$\sigma^2 = \ln\left(\frac{\sigma_{LN}^2}{\mu_{LN}^2} + 1\right), \quad \mu = \ln\left(\mu_{LN}\right) - \frac{\sigma^2}{2}.$$

4.2.7. *Left Truncated Normal Distribution*

The *left truncated normal distribution* can be defined from (4.19) by introducing a lower bound c. This means that a random variable $z \geq c$ is left truncated normal with location parameter $\mu \in \mathbb{R}$, scale parameter $\sigma > 0$, and finite c, denoted by $z \sim LTN(\mu, \sigma^2, c)$, if and only if its pdf is

$$p_{LTN}(z|\mu,\sigma,c) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(z-\mu)^2}{2\sigma^2}\right) \left(1 - \Phi((c-\mu)/\sigma)\right)^{-1}, \tag{4.20}$$

---

[30] One such example is when the model contains an AR(2) process and where the AR parameters should both be positive and add up to something less than unity. This is sufficient but not necessary for stability. By transforming z_i even further one could also consider the general conditions for stability of an AR(2) process. For instance, let

$$y_1 = 2z_2 + 4(1 - z_2)z_1 - 2,$$
$$y_2 = 2z_2 - 1.$$

It can now be shown that $y_1 + y_2 < 1$, $y_2 - y_1 < 1$, and $-1 < y_2 < 1$ for $(z_1, z_2) \sim D(\alpha_1, \alpha_2, \alpha_3)$. The last condition follows from $z_2 = (y_2 + 1)/2 \in (0, 1)$. The first two conditions are satisfied if we notice that $y_2 - 1 < y_1 < 1 - y_2$. We may therefore let $z_1 = (y_1 - y_2 + 1)/(2 - 2y_2) \in (0, 1)$. Based on the means and covariance of z_i we can directly determine the means of y_i. Notice that the stability conditions are also satisfied if we let $z_i \sim U(0, 1)$, with z_1 and z_2 independent.

where

$$\Phi(a) = \begin{cases} (1 + \kappa(a/\sqrt{2}))/2 & \text{if } a > 0 \\ (1 - \kappa(-a/\sqrt{2}))/2 & \text{otherwise,} \end{cases} \tag{4.21}$$

$$\kappa(b) = \frac{2}{\sqrt{\pi}} \int_0^b \exp(-x^2) dx.$$

Hence, the left truncated normal density is given by the normal density divided by 1 minus the cumulative normal distribution up to the point of left truncation, i.e., $(c - \mu)/\sigma$. The function $\kappa(b)$ is often called the error function. In Matlab, its name is `erf`.

As long as $\mu \geq c$, the mode is given by $\tilde{\mu}_{LTN} = \mu$, while $\mu < c$ means that the mode is $\tilde{\mu}_{LTN} = c$.

Three examples of the left truncated normal distribution along with the normal distribution are plotted in the lower right panel of Figure 1. As c increases the height of the density relative to it highest point based on the normal for the same support increases.

4.2.8. *Uniform Distribution*

A random variable z is said to have a *uniform distribution* with parameters a and b with $b > a$, denoted by $z \sim U(a, b)$ if and only if its pdf is given by

$$p_U(z|a, b) = \frac{1}{b - a}. \tag{4.22}$$

The mean and the variance of this distribution are:

$$\mu_U = \frac{a + b}{2},$$

$$\sigma_U^2 = \frac{(b - a)^2}{12}.$$

The beta distribution is equivalent to a uniform distribution with lower bound c and upper bound d when $\mu_B = (c + d)/2$ and $\sigma_B^2 = (d - c)^2/12$; see, also, Bauwens, Lubrano, and Richard (1999) for additional properties of the uniform distribution.

4.2.9. *Student-t and Cauchy Distribution*

A random variable z is said to have a *Student-t distribution* with location parameter $\mu \in \mathbb{R}$, scale parameter $\sigma > 0$, and degrees of freedom parameter d (a positive integer), denoted by $z \sim t(\mu, \sigma, d)$, if and only if its pdf is given by

$$p_S(z|\mu, \sigma, d) = \frac{\Gamma((d + 1)/2)}{\Gamma(d/2)\sigma\sqrt{d\pi}} \left(1 + \frac{1}{d}\left(\frac{z - \mu}{\sigma}\right)^2\right)^{-(d+1)/2}. \tag{4.23}$$

The Student-t distribution is symmetric around the mode μ, while the mean exists if $d > 1$, and the variance exists if $d > 2$. The first two central moments are then given by

$$\mu_S = \mu,$$

$$\sigma_S^2 = \frac{d}{d - 2}\sigma^2.$$

The distribution has heavier tails (higher kurtosis) for finite d than the normal distribution.[31]

When $d \to \infty$, the density in (4.23) converges to the density of the normal distribution. At the other extreme, i.e., $d = 1$, the distribution is also known as the *Cauchy*, denoted by $z \sim C(\mu, \sigma)$. The density function now simplifies to

$$p_C(z|\mu, \sigma) = \frac{\sigma}{\pi(\sigma^2 + (z - \mu)^2)}. \tag{4.24}$$

---

[31] Specifically, if $d > 4$, then excess kurtosis is $6/(d - 4)$

FIGURE 2: Examples of normal, Student-*t*, Cauchy, logistic, Gumbel and Pareto distributions.
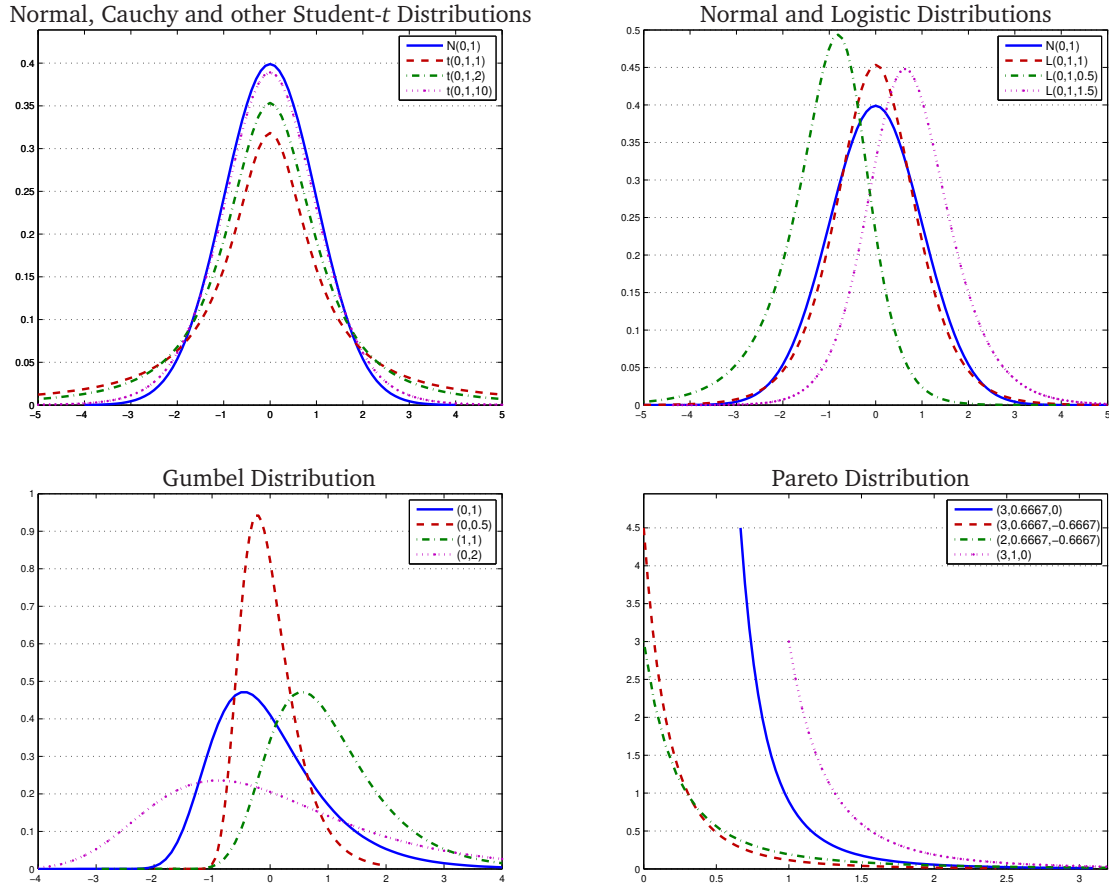
The mean and the variance for the Cauchy distribution do not exist; see, also, Bauwens, Lubrano, and Richard (1999) for additional properties of the Student-*t* distribution.

The normal, the Cauchy, and some other Student-*t* distributions are shown in the upper left panel of Figure 2. As *q* increases, the height of the *t*-density approaches the height of the normal density from below. For the standard distributions with location parameter zero and unit scale parameter, the tails of the Student-*t* become thicker than the normal once the distance is at leats 2 standard deviations away from the mean of the normal distribution.

A random vector z of dimension n is said to have a multivariate Student-*t* distribution with location vector μ, positive definite scale matrix Σ, and degrees of freedom parameter d, denoted by $z \sim T_n(\mu, \Sigma, d)$, if its pdf is given by

$$
\begin{aligned}
p_S(z|\mu, \Sigma, d) &= \frac{\Gamma((d+n)/2)}{\Gamma(d/2)\pi^{n/2}|\Sigma|^{1/2}d^{-d/2}} \left(d + (z-\mu)'\Sigma^{-1}(z-\mu)\right)^{-(d+n)/2} \\
&= \frac{\Gamma((d+n)/2)}{\Gamma(d/2)|\Sigma|^{1/2}(d\pi)^{n/2}} \left(1 + \frac{1}{d}(z-\mu)'\Sigma^{-1}(z-\mu)\right)^{-(d+n)/2}.
\end{aligned}
\tag{4.25}
$$

The mode of this distribution is given by μ and when $d > 1$, the mean exists and is equal to μ. Provided that $d > 2$, the covariance also exists and is given by

$$
E\left[(z-\mu)(z-\mu)'\right] = \frac{d}{d-2}\Sigma.
$$

4.2.10. *Logistic Distributions*

A random variable z is said to have a *logistic distribution* with location parameter $\mu \in \mathbb{R}$ and scale parameter $\sigma > 0$, denoted by $z \sim L(\mu, \sigma)$, if and only if its density function is

$$p_L(z|\mu, \sigma) = \frac{\exp\left(-(z - \mu)/\sigma\right)}{\sigma\left[1 + \exp\left(-(z - \mu)/\sigma\right)\right]^2}. \tag{4.26}$$

Because the pdf can be expressed in terms of the square of the hyperbolic secant function[32] it is sometimes referred to as the *sech-squared distribution*.

The logistic distribution receives its name from its cdf, which is an instance of the family of logistic functions. Specifically, the cdf is given by

$$F_L(z|\mu, \sigma) = \frac{1}{1 + \exp\left(-(z - \mu)/\sigma\right)}.$$

The logistic distribution is symmetric and resembles the normal distribution in shape, but has heavier tails (higher kurtosis). The mean and variance of the distribution are:

$$
\begin{aligned}
\mu_L &= \mu, \\
\sigma_L^2 &= \frac{\pi^2}{3}\sigma^2,
\end{aligned}
\tag{4.27}
$$

while excess kurtosis is $6/5$.

One extension of the logistic distribution that may be of interest is the so called Type I generalized (or reversed) logistic distribution; cf. Balakrishnan and Leung (1988). A random variable z is said to have a *Type I generalized logistic distribution* with location parameter $\mu \in \mathbb{R}$, scale parameter $\sigma > 0$, and shape parameter $c > 0$ if and only if its density function is

$$p_{GL}(z|\mu, \sigma, c) = \frac{c\exp\left(-(z - \mu)/\sigma\right)}{\sigma\left[1 + \exp\left(-(z - \mu)/\sigma\right)\right]^{1+c}}. \tag{4.28}$$

In this case the cdf is given by

$$F_{GL}(z|\mu, \sigma, c) = \frac{1}{\left[1 + \exp\left(-(z - \mu)/\sigma\right)\right]^c}. \tag{4.29}$$

The distribution is left-skewed for $c < 1$, right-skewed for $c > 1$, and for $c = 1$ it is symmetric and is identical to the logistic. The mode of the distribution is given by $\tilde{\mu}_{GL} = \mu + \sigma \ln(c)$.[33]

The mean and the variance of the Type I generalized logistic distribution exist and are given by

$$
\begin{aligned}
\mu_{GL} &= \mu + \left(\gamma + \psi(c)\right)\sigma, \\
\sigma_{GL}^2 &= \left[\frac{\pi^2}{6} + \psi'(c)\right]\sigma^2.
\end{aligned}
\tag{4.30}
$$

---

[32] The hyperbolic secant function is given by $\text{sech}(x) = 2/(\exp(x) + \exp(-x))$ for $x \in \mathbb{R}$.

[33] In fact, skewness for the Type I generalized logistic is:

$$S_{GL} = \frac{\psi''(c) + 2\zeta(3)}{\frac{\pi^2}{6} + \psi'(c)},$$

where $\psi''(c)$ is the second derivative of the ψ function, i.e., the third derivative of the log of the gamma function. The Riemann zeta function is given by $\zeta(s) = \sum_{k=1}^{\infty} 1/k^s$, where $\zeta(3) \approx 1.20205690$ is also known as Apéry's constant after Roger Apéry who proved that it is an irrational number. For $c = 1$ the numerator is zero, and for $c < 1$ ($c>1$) it is negative (positive).

The term γ is *Euler's constant*, while $\psi(c)$ is the *digamma function* and $\psi'(c)$ its first derivative, the so called trigamma function.[34] In Matlab, $\psi(c)$ and its derivatives can be computed through the function `psi`. But since this function was not present until version 6.5 of Matlab, the functions `DiGamma` and `TriGamma` are included in YADA, using the algorithms in Bernardo (1976) and Schneider (1978), respectively.[35]

The logistic distribution is compared with the normal distribution in the upper right panel of Figure 2. Focusing on a mean of zero, we find that the height of the symmetric logistic ($c = 1$) is greater than the height of the normal close to the mean. Between around one and 2.5 standard deviations away from the mean of the normal the height of the normal is somewhat greater than that of the logistic, whereas further out in the tails the height of the logistic is greater than the height of the normal. As $c < 1$ the distribution becomes left-skewed (green dash-dotted line), and for $c > 1$ it is right-skewed (magenta dotted line). The skewness effect in Figure 2 seems to be larger in absolute terms when $c < 1$ than when $c > 1$. This is also confirmed when applying the skewness formula. For $c = 0.5$ we find that skewness is approximately -2.19, whereas for $c = 1.5$ ($c = 100$) it is roughly 0.61 (1.45). As c becomes very large, skewness appears to converge to around 1.46. Hence, a higher degree of left-skewness than right-skewness can be achieved through the Type I generalized logistic distribution.

4.2.11. *Gumbel Distribution*

A random variable z is said to have a *Gumbel distribution* with location parameter $\mu \in \mathbb{R}$ and scale parameter $\sigma > 0$, denoted by $z \sim Gu(\mu, \sigma)$, if and only if its density function is

$$p_{Gu}(z|\mu, \sigma) = \frac{1}{\sigma} \exp\left(-\frac{z-\mu}{\sigma}\right) \exp\left(-\exp\left(-\frac{z-\mu}{\sigma}\right)\right). \tag{4.31}$$

The Gumbel distribution is the most common of the three types of Fisher-Tippett extreme value distribution. It is therefore sometimes called the *Type I extreme value distribution*. The cdf of the Gumbel is given by

$$F_{Gu}(z|\mu, \sigma) = \exp\left(-\exp\left(-\frac{z-\mu}{\sigma}\right)\right). \tag{4.32}$$

The Gumbel distribution is right-skewed for z and therefore left-skewed for $x = -z$. The mean and the variance exist and are given by:

$$\mu_{Gu} = \mu + \gamma\sigma,$$
$$\sigma_{Gu}^2 = \frac{\pi^2}{6}\sigma^2. \tag{4.33}$$

Skewness is roughly equal to 1.1395, while excess kurtosis is exactly $12/5$.[36] The mode is given by the location parameter $\tilde{\mu}_{Gu} = \mu$, while the median is $\mu - \sigma \ln(\ln(2))$; see Johnson, Kotz, and Balakrishnan (1995).

A few examples of the Gumbel distribution have been plotted in the lower left panel of Figure 2. It is noteworthy that a shift in the mean for fixed standard deviation implies an equal size shift in the location parameter μ. This is illustrated by comparing the solid blue line for $Gu(0, 1)$ to the dash-dotted green line for $Gu(1, 1)$. This is a direct consequence of equation (4.33), where $d\mu/d\mu_{Gu} = 1$. Moreover, and as also implied by that equation and the fact that

---

[34] Euler's constant, also known as the Euler-Mascheroni constant, is defined as $\gamma = \lim_{n\to\infty}[H_n - \ln n]$, where $H_n = \sum_{k=1}^{n} 1/k$ is a harmonic number. For Euler's constant we know that $\gamma = -\psi(1) \approx 0.57721566$. This number was calculated to 16 integers by Euler in 1781, Mascheroni calculated it to 32 by 1790, although only the first 19 were later shown to be correct; see Havil (2003, p. 89-90). It has now been calculated to at least 2 billion digits. Still, it is not known if this constant is irrational. The digamma function is, as the name suggests, the logarithmic derivative of the gamma function. It may also be noted that $\psi'(1) = \pi^2/6$.

[35] The algorithm also makes use of modifications suggested by Tom Minka in the functions `digamma.m` and `trigamma.m` from the *Lightspeed Toolbox*. See https://github.com/tminka/lightspeed for details.

[36] To be precise, skewness is given by $S_{Gu} = 12\sqrt{6}\zeta(3)/\pi^3$, where $\zeta(3)$ is equal to Apéry's constant; cf. footnote 33.

μ is the mode, a higher (lower) standard deviation for fixed mean results in a lower (higher) mode, i.e., $d\tilde{\mu}_{Gu}/d\sigma_{Gu} = -\gamma\sqrt{6}/\pi \approx -0.45$.

4.2.12. *Pareto Distribution*

A random variable $z \geq b$ is said to have a *Pareto distribution* with shape parameter $a > 0$ and location parameter $b > 0$, denoted by $z \sim P(a, b)$, if and only if its pdf is equal to

$$p_P(z|a, b) = ab^a z^{-(a+1)}. \tag{4.34}$$

This distribution was originally used by Vilfredo Pareto to describe the allocation of wealth among individuals. The idea was to represent the "80-20 rule", which states that 20 percent of the population control 80 percent of the wealth. The power law probability distribution has the simple cdf

$$F_P(z|a, b) = 1 - \left(\frac{b}{z}\right)^a. \tag{4.35}$$

The mode of the distribution is b and the density declines exponentially toward zero as z becomes larger. The mean exists if $a > 1$, while the variance exists if $a > 2$. The central moments are then given by

$$\mu_P = \frac{ab}{a-1},$$

$$\sigma_P^2 = \frac{ab^2}{(a-1)^2(a-2)}.$$

Moreover, if $a > 3$ then skewness exists and is given by:

$$S_P = \frac{2(a+1)\sqrt{a-2}}{(a-3)\sqrt{a}}.$$

Since $S_P > 0$ it follows that the distribution is right-skewed. Moreover, S_P is a decreasing function of a and $S_P \rightarrow 2$ as $a \rightarrow \infty$.

The Pareto distribution is plotted for a 3 parameter case in the lower right panel of Figure 2. The third parameter is called the origin parameter, c, and is applied such that $z = c + x$, where $x \sim P(a, b)$. In other words, the density function for z is given by

$$p_P(z|a, b, c) = ab^a(z-c)^{-(a+1)}.$$

Comparing the baseline case where $z \sim P(3, 2/3, 0)$ (blue solid line) to the case when $c = -b$ (red dashed line) it is clear that the origin parameter merely affects the lower bound of the distribution. On the other hand, a drop of the shape parameter a from 3 to 2 (green dash-dotted line) lowers the height of the distribution around the mode and increases the mass of the distribution in the right tail.[37] Finally, an increase in b has no effect on skewness and, hence, it increases the height of the distribution over its support.

4.2.13. *Discussion*

YADA needs input from the user regarding the type of prior to use for each parameter it should estimate. In the case of the beta, normal, logistic, and Gumbel distributions the parameters needed as input are assumed to be the mean and the standard deviation. If you wish to have a general beta prior you need to provide the upper and lower bounds as well or YADA will set these to 1 and 0, respectively. If you wish to have a Type I generalized logistic distribution you need to provide the shape parameter c in (4.28).

---

[37] Excess kurtosis is also a function of a only. Specifically, provided that $a > 4$ it is given by:

$$K_P = \frac{6\left(a^3 + a^2 - 6a - 2\right)}{a\left(a^2 - 7a + 12\right)}.$$

It can be shown that excess kurtosis is a decreasing function of a and that $K_P \rightarrow 6$ from above as $a \rightarrow \infty$.

For the gamma and the left truncated normal distribution, the parameters to assign values for are given by μ, σ, and a lower bound c. For the gamma distribution this the first two parameters are the mean and the standard deviation, while for the left truncated normal they are defined in equation (4.20) and are the location and scale parameters, respectively.

Similarly, for the inverted gamma distribution the parameters to select values for are s, q, and a lower bound c. The s parameter is, as mentioned above, a location parameter and q is a degrees of freedom parameter that takes on integer values. The location parameter s can, e.g., be selected such that the prior has a desired mode. Relative to equation (4.11) we are now dealing with the location parameter $(s - c)$ and the random variable $(\sigma - c)$ since the density is expressed for a random variable that is positive. The mode for this parameterization is $\tilde{\mu}_{IG} = s(q/(q+1))^{1/2} + c(1 - (q/(q+1))^{1/2})$.

As can be expected, the uniform distribution requires the lower and upper bound, i.e., a and b in (4.22).

For the the Student-t, the required parameters are μ, σ, and d, while the Cauchy only takes the first two parameters. Finally, the Pareto distribution takes the shape and location parameters (a, b). In addition, YADA also accepts an origin parameter c which shifts z by a constant, i.e., $y = z + c$, where $z \sim P(a, b)$. This means that $y \geq b + c$, i.e., $b + c$ is both the mode and the lower bound of y.

For all distributions but the beta, gamma, logistic, and Gumbel there is no need for internally transforming the distribution parameters. For these 4 distributions, however, transformations into the a and b parameters (μ and σ for the logistic and Gumbel) are needed, using the mean and the standard deviation as input (as well as the shape parameter c for the logistic). YADA has 4 functions that deal with these transformations, `MomentToParamGammaPDF` (for the gamma), `MomentToParamStdbetaPDF` (for the standardized beta), `MomentToParamLogisticPDF` (for the logistic), and `MomentToParamGumbelPDF` (for the Gumbel distribution). These functions take vectors as input as provide vectors as output. The formulas used are found in equations (4.8) and (4.16) above for the gamma and the beta distributions, the inverse of (4.27) for the logistic distribution, i.e., $\mu = \mu_L$ and $\sigma = (\sqrt{3}/\pi)\sigma_L$ when $c = 1$, the inverse of (4.30) when $c \neq 1$, and the inverse of (4.33) for the Gumbel distribution. Since YADA supports a lower bound that can be different from zero for the gamma prior, the mean minus the lower bound is used as input for the transformation function `MomentToParamGammaPDF`. Similarly, the mean and the standard deviation of the standardized beta distribution are computed from the mean and the standard deviation as well as the upper and lower bounds of the general beta distribution. Recall that these relations are $\mu_{SB} = (\mu_B - c)/(d - c)$ and $\sigma_{SB} = \sigma_B/(d - c)$.

4.3. *Random Number Generators*

For each prior distribution that YADA has direct support for, it can also provide random draws. These draws are, for instance, used to compute impulse responses based on the prior distribution. In this subsection, the specific random number generators that YADA makes use of will be discussed.

The basic random number generators are given by the `rand` and `randn` Matlab function. The first provides draws from a standardized uniform distribution and the second from a standardized normal distribution. Supposing that $p \sim U(0, 1)$ it follows by the relationship $p = (z - a)/(b - a)$, where $b > a$ are the upper and lower bound for z that

$$z = a + (b - a)p.$$

Hence, random draws for z are obtained by drawing p from $U(0, 1)$ via the function `rand` and computing z from the above relationship.

Similarly, with $x \sim N(0, 1)$ and $x = (z - \mu)/\sigma$ it follows that random draws for $z \sim N(0, \sigma^2)$ can be obtained by drawing x from $N(0, 1)$ via the function `randn` and using the relationship $z = \mu + \sigma x$.

To obtained draws of z from $LTN(\mu, \sigma, c)$ YADA uses a very simple approach. First of all, x is drawn from $N(\mu, \sigma^2)$. All draws such that $x \geq c$ are given to z, while the draws $x < c$ are discarded.

If $z \sim G(a, b)$, then YADA checks if the *Statistics Toolbox* is available on the computer. If its existence is confirmed, then the function `gamrnd` is used. On the other hand, if this toolbox is missing, then YADA uses the function `YADARndGammaUnitScale` function to obtained $x \sim G(a, 1)$, while $z = bx \sim G(a, b)$ follows from this relationship. The function that draws from a gamma distribution with unit scale ($b = 1$) is based on the function `rgamma` from the *Stixbox Toolbox* by Anders Holtsberg.[38] Since gamma distributed random variables can have a lower bound (c) different from zero, this parameter is added to the gamma draws.

Similarly, to obtain draws from a beta distribution, YADA first checks if the Statistics Toolbox is available. If the test provides a positive response the function `betarnd` is used to generate $z \sim B(a, b)$. With a negative response to this test, YADA uses the random number generator for the gamma distribution for $y_1 \sim G(a, 1)$ and $y_2 \sim G(b, 1)$, and determines z from $z = y_1 / (y_1 + y_2)$. For both cases, YADA makes use of the relationship $x = c + (d - c)z$, with $d > c$. It now follows that $x \sim B(a, b, c, d)$.

Furthermore, to draw $z \sim IG(s, q)$ YADA again makes use of the gamma distribution. First of all, the pair $(a, b) = (q/2, 2/qs^2)$ is computed such that $z \sim IG(a, b) \equiv IG(s, q)$. Using the fact that z being $IG(a, b)$ is equivalent to $x = z^{-2} \sim G(a, b)$. Hence, x is drawn from the gamma distribution, while $z = 1/\sqrt{x}$.

For the Cauchy and, more generally, the Student-t we make use of the result that the standardized Student-t density is given by the ratio between the standardized normal density and the density of $\sqrt{z/q}$ where $z \sim \chi_q^2$; see, e.g., Bauwens et al. (1999, p. 318). That is, we let $y_1 \sim N(0, 1)$ and $y_2 \sim G(d/2, 2)$ and let $x = y_1 \sqrt{d/y_2}$. This means that $x \sim t(0, 1, d)$. Finally, the get the random draws $z \sim t(\mu, \sigma, d)$ we employ the relationship $z = \mu + \sigma x$. Again, random draws from the Cauchy distribution $C(\mu, \sigma)$ are given by setting $d = 1$ for the draws from the Student-t distribution; see, e.g, Gelman et al. (2004, Appendix A, p. 581).[39]

To obtain draws from the multivariare Student-t distribution, YADA makes use of the second algorithm in Bauwens et al. (1999, Section B.4.2, p. 320), with $s = \nu = d$ and $\Sigma = M^{-1}$. That is, it takes the lower triangular Choleski decomposition of Σ and calls it C, i.e. $CC' = \Sigma$. We now let $y_1 \sim N(0, I_n)$ and $y_2 \sim G(d/2, 2)$ and let $x = y_1 \sqrt{d/y_2}$. This means that $x \sim T_n(0, I_n, d)$. The variable $z \sim T_n(\mu, \Sigma, d)$ is now obtained from the transformation $z = \mu + Cx$. For low degrees of freedom, one usually needs a lot of draws to simulate the multivariate Student-t distribution well with this algorithm.

To obtain random draws from the Type I generalized logistic distribution, YADA makes use of the cdf in equation (4.29). That is, we replace $F_{GL}(z|\mu, \sigma, c)$ with $p \sim U(0, 1)$ and compute z by inverting the cdf. This provides us with

$$z = \mu - \sigma \ln \left(\frac{1}{p^{1/c}} - 1 \right).$$

The same approach is used for the Gumbel and the Pareto distributions. By inverting (4.32) we obtain

$$z = \mu - \sigma \ln \big(\ln(p) \big),$$

where $p \sim U(0, 1)$. It now follows that $z \sim Gu(\mu, \sigma)$.

Similarly, by inverting equation (4.35), and taking the origin parameter into account, it is straightforward to show that

$$z = c + b \frac{1}{(1 - p)^{1/a}}.$$

---

[38] The Stixbox Toolbox can be download from https://forge.scilab.org/index.php/p/stixbox/.

[39] The Statistics Toolbox comes with the function `trnd` which returns draws from the standardized Student-t distribution $t(0, 1, d)$. This function is not used by YADA and the main reason is that it does not seem to improve upon the routines provided by YADA itself. In fact, for small d many draws from the Student-t appear to be extremely large or small. To avoid such extreme draws, YADA excludes all draws that are outside the range $[\mu - 6\sigma, \mu + 6\sigma]$. While the choice of 6 times the scale factor is arbitrary, this seems to work well in practise. In particular, when estimating the density via a kernel density estimator using, e.g, and Epanechnikov or normal kernel, the resulting density seems to match a grid-based estimator of the Student-t density very well when based on occular inspection.

With $p \sim U(0, 1)$ we find that $z \sim P(a, b, c)$.

4.4. *System Priors*

The priors considered so far are straightforward to use in settings where the parameters are independent. Such a property is standard in DSGE modelling and is used because of its convenience. At the same time, independent prior for the structural parameters is often at odds with what we a prior expect. For example, the parameters σ_c and λ in the log-linearized consumption equation (2.18) of the Smets and Wouters model are likely to be a priori correlated since they both measure aspects of the consumer's preferences. At the same time, it is not obvious how to model such dependence between parameters as economic theory gives little or no guidance.

Although correlations between structural parameters can be introduced in many ways, one natural approach is to consider a particular model or system feature which a researcher would like to condition the empirical analyses on. Several papers have introduced ways that such information can be accounted for in DSGE models, giving it different names. Del Negro and Schorfheide (2008) suggest an approach for introducing beliefs about steady-state relationships and second moments of the endogenous variables. Christiano, Trabandt, and Walentin (2011) consider what they refer to as *endogenous priors* and focus on the population standard deviations of the observed variables, while Andrle and Beneš (2013) more broadly discuss what they call *system priors* and which include priors about, e.g., the sacrifice ratio, conditional and unconditional population moments of the model, policy scenarios, impulse respones function, and frequency response functions and spectral characteristics.

YADA uses this latter terminology and refers to a system prior as the researcher's prior beliefs about some system characteristic which can be modelled with a suitable density function conditional on the DSGE model parameters. For example, a researcher may wish to condition the DSGE model on the prior that the standard deviation of real GDP growth has mean 0.5. This can, for example, be achieved by assuming that the population standard deviation has an inverted gamma distribution with mean 0.5 and a variance that suitably captures the prior uncertainty about this mean. For example, letting $s = 0.46$ and $q = 10$ for the inverted gamma parameterization in equation (4.11) gives a mean $\mu_{IG} \approx 0.4985$ and $\sigma_{IG} \approx 0.1264$, which may serve the intended purpose well.

Let ω denote the system properties we would like to control with the system prior, where $\omega = h(\theta)$ for a known function θ. As in Andrle and Beneš (2013) we assume that ω is endowed with a complete probabilistic model with hyperparameters Φ_ω. The likelihood function for ω is given by $p(\Phi_\omega|\theta, h)$, while $p(\theta)$ is the original prior density for the DSGE model parameters. This function is typically a product of marginal prior distributions of the individual DSGE model parameters. The full system prior of ω and θ is now

$$p(\theta|\Phi_\omega, h) \propto p(\Phi_\omega|\theta, h)p(\theta), \tag{4.36}$$

and where $p(\theta)$ is the marginal distribution of θ. In the inverted gamma example above, $\Phi_\omega = (\mu_{IG}, \sigma_{IG})$ or, equivantently, $\Phi_\omega = (s, q)$, while h is the function that gives the standard deviation of real GDP growth based on the DSGE model parameters.

When estimating the θ parameters, we are interested in the prior density of θ conditional on Φ_ω. Using Bayes theorem, as in Section 4.1, we find that

$$p(\theta|\Phi_\omega, h) = \frac{p(\Phi_\omega|\theta, h)p(\theta)}{p(\Phi_\omega|h)}. \tag{4.37}$$

This means that the marginal prior likelihood of the hyperparameters Φ_ω needs to be determined.

Taking a step back first, notice that direct sampling of the DSGE model parameters θ from the density in equation (4.37) is not possible since its analytical form is generally unknown. This sampling problem is analogous to the problem of sampling from the posterior distribution, discussed below in Section 8. In fact, we may use the Markov Chain Monte Carlo (MCMC) methods from posterior simulation for system prior sampling, where $p(\Phi_\omega|\theta, h)$ replaces the likelihood function for the posterior simulator. Furthermore, the system prior mode can be

estimated using the joint log prior kernel of the right hand side in (4.36) using, for example, numerical optimization. The likelihood $p(\Phi_\omega|\theta, h)$ is typically rapidly calculated once $h(\theta)$ has been determined. For example, if h is a vector of population standard deviations of the observable variables conditional on θ, then the main computational part of the system prior is to solve the DSGE model.

Once prior draws have been obtained via a suitable sampler, the marginal prior likelihood $p(\Phi_\omega|h)$ can be estimated with, e.g., the algorithms discussed below in Section 10. This computation is actually only relevant when the researcher is interested in the marginal likelihood of the observed data, since the constant $p(\Phi_\omega|h)$ needs to be accounted for this estimate. In contrast, posterior mode estimation or posterior simulation are not affected by this constant.

4.5. *YADA Code*

The density functions presented above are all written in natural logarithm form in YADA. The main reason for this is to keep the scale manageable. For example, the exponential function in Matlab, like any other computer software available today, cannot deal with large or small real numbers. If one attempts to calculate e^{700} one obtains `exp(700) = 1.0142e+304`, while `exp(720) = Inf`. Furthermore, and as discussed in Section 4.2.2, the gamma and beta functions return infinite values for large (and finite) input values, while the natural logarithm of these functions return finite values for the same large input values.

4.5.1. `logGammaPDF`

The function `logGammaPDF` calculates $\ln p_G(z|a, b)$ in (4.7). Required inputs are `z`, `a`, `b`, while the output is `lnG`. Notice that all inputs can be vectors, but that the function does not check that the dimensions match. This is instead handled internally in the files calling or setting up the input vectors for `logGammaPDF`.

4.5.2. `logInverseGammaPDF`

The function `logInverseGammaPDF` calculates $\ln p_{IG}(\sigma|s, q)$ in (4.11). Required inputs are `sigma`, `s`, `q`, while the output is `lnIG`. Notice that all inputs can be vectors, but that the function does not check that the dimensions match. This is instead handled internally in the files calling or setting up the input vectors for `logInverseGammaPDF`.

4.5.3. `logBetaPDF`

The function `logBetaPDF` calculates $\ln p_B(z|a, b, c, d)$ in (4.14). Required inputs are `z`, `a`, `b`, `c`, and `d`, while the output is `lnB`. Notice that all inputs can be vectors, but that the function does not check that the dimensions match. This is instead handled internally in the files calling or setting up the input vectors for `logBetaPDF`.

4.5.4. `logNormalPDF`

The function `logNormalPDF` calculates $\ln p_N(z|\mu, \sigma)$ in (4.19). Required inputs are `z`, `mu`, `sigma`, while the output is `lnN`. Notice that all inputs can be vectors, but that the function does not check that the dimensions match. This is instead handled internally in the files calling or setting up the input vectors for `logNormalPDF`.

4.5.5. `logLTNormalPDF`

The function `logLTNormalPDF` calculates $\ln p_{LTN}(z|\mu, \sigma, c)$ in (4.20). Required inputs are `z`, `mu`, `sigma`, `c`, while the output is `lnLTN`. This function calls the `PhiFunction` described below. Notice that the function does not check if $z \geq c$ holds. This is instead handled by the functions that call `logLTNormalPDF`.

4.5.6. `logUniformPDF`

The function `logUniformPDF` calculates $\ln P_U(z|a, b)$ in (4.22), i.e., the log height of the uniform density, i.e., $-\ln(b - a)$. The required inputs are `a` and `b`, where the former is the lower bound and the latter the upper bound of the uniformly distributed random vector z.

4.5.7. logStudentTAltPDF

The function logStudentTAltPDF calculates $\ln p_S(z|\mu,\sigma,d)$ in (4.23), i.e., the log height of the Student-t density. The required input variables are z, mu, sigma, and df, while the output is lnS. Notice that all inputs can be vectors, but that the function does not check that the dimensions match. This is instead handled internally in the files calling or setting up the input vectors for logStudentTAltPDF.

4.5.8. logCauchyPDF

The function logCauchyPDF calculates $\ln p_C(z|\mu,\sigma)$ in (4.24), i.e., the log height of the Cauchy density. The required input variables are z, mu, and sigma, while the output is lnC. Notice that all inputs can be vectors, but that the function does not check that the dimensions match. This is instead handled internally in the files calling or setting up the input vectors for logCauchyPDF.

4.5.9. logLogisticPDF

The function logLogisticPDF calculates $\ln p_{GL}(z|\mu,\sigma,c)$ in (4.28), i.e., the log height of the (Type I generalized) logistic density. The required input variables are z, mu, sigma and c, while the output is lnL. Notice that all inputs can be vectors, but that the function does not check that the dimensions match. This is instead handled internally in the files calling or setting up the input vectors for logLogisticPDF.

4.5.10. logGumbelPDF

The function logGumbelPDF calculates $\ln p_{Gu}(z|\mu,\sigma)$ in (4.31), i.e., the log height of the Gumbel density. The required input variables are z, mu, and sigma, while the output is lnGu. Notice that all inpute variables can be vectors, but that the function does not check that the dimensions match. This is instead handled internally in the files calling or setting up the input vectors for logGumbelPDF.

4.5.11. logParetoPDF

The function logParetoPDF calculates $\ln p_P(z|a,b)$ in (4.34), i.e., the log height of the Pareto density. The required input variables are z, a, and b, while the output is lnP. Notice that all inputs can be vectors, but that the function does not check that the dimensions match. This is instead handled internally in the files calling or setting up the input vectors for logParetoPDF.

4.5.12. PhiFunction

The function PhiFunction evaluates the expression for $\Phi(a)$ in (4.21). The required input is the vector a, while the output is PhiValue, a vector with real numbers between 0 and 1.

4.5.13. GammaRndFcn

The function GammaRndFcn computes random draws from a gamma distribution. The function takes two required input variables, a and b. These contain the shape and the scale parameters from the gamma distribution with lower bound 0 and both are treated as vectors; see equation (4.7). Notice that the function does not check that the dimensions of a and b match. An optional input variable for total number of draws, NumDraws, is also accepted. The default value for this integer is 1. The algorithms used by the function are discussed above in Section 4.3.

The function provides one output variable, z, a matrix with row dimension equal to the length of a and column dimension equal to NumDraws.

4.5.14. InvGammaRndFcn

The function InvGammaRndFcn computes random draws from an inverted gamma distribution. The function takes two required input variables, s and q. These contain the location and degrees of freedom parameters from the inverted gamma distribution with lower bound 0 and both are treated as vectors; see equation (4.11). Notice that the function does not check that the dimensions of s and q match. An optional input variable for total number of draws, NumDraws,

is also accepted. The default value for this integer is 1. The algorithms used by the function are discussed above in Section 4.3.

The function provides one output variable, `sigma`, a matrix with row dimension equal to the length of `s` and column dimension equal to `NumDraws`.

4.5.15. BetaRndFcn

The function `BetaRndFcn` computes random draws from a beta distribution. The function takes 4 required input variables, `a`, `b`, `c` and `d`. These contain the shape parameters as well as the lower and upper bound from the beta and are all treated as vector; see equation (4.14). The function does not check if the dimensions of these variables match; this is instead handled by the function that calls it. An optional input variable for total number of draws, `NumDraws`, is also accepted. The default value for this integer is 1. The algorithms used by the function are discussed above in Section 4.3.

The function provides one output variable, `x`, a matrix with row dimension equal to the length of `a` and column dimension equal to `NumDraws`.

4.5.16. NormalRndFcn

The function `NormalRndFcn` computes random draws from a normal distribution. The function takes two required input variables, `mu` and `sigma`. These contain the mean (location) and standard deviation (scale) parameters; see equation (4.19). The function does not check if the dimensions of these variables match; this is instead handled by the function that calls it. An optional input variable for total number of draws, `NumDraws`, is also accepted. The default value for this integer is 1. The algorithm used by the function is discussed above in Section 4.3.

The function provides one output variable, `z`, a matrix with row dimension equal to the length of `mu` and column dimension equal to `NumDraws`.

4.5.17. LTNormalRndFcn

The function `LTNormalRndFcn` computes random draws from a left truncated normal distribution. The function takes 3 required input variables, `mu`, `sigma` and `c`. These contain the location, scale and lower bound (left truncation) parameters; see equation (4.20). The function does not check if the dimensions of these variables match; this is instead handled by the function that calls it. An optional input variable for total number of draws, `NumDraws`, is also accepted. The default value for this integer is 1. The algorithm used by the function is discussed above in Section 4.3.

The function provides one output variable, `z`, a matrix with row dimension equal to the length of `mu` and column dimension equal to `NumDraws`.

4.5.18. UniformRndFcn

The function `UniformRndFcn` computes random draws from a uniform distribution. The function takes two required input variables, `a` and `b`. These contain the lower and upper bound parameters; see equation (4.22). The function does not check if the dimensions of these variables match; this is instead handled by the function that calls it. An optional input variable for total number of draws, `NumDraws`, is also accepted. The default value for this integer is 1. The algorithm used by the function is discussed above in Section 4.3.

The function provides one output variable, `z`, a matrix with row dimension equal to the length of `a` and column dimension equal to `NumDraws`.

4.5.19. StudentTAltRndFcn

The function `StudentTAltRndFcn` computes random draws from a Student-*t* distribution. The function takes 3 required input variables, `mu`, `sigma` and `d`. These contain the location, scale and degrees of freedom parameters; see equation (4.23). The function does not check if the dimensions of these variables match; this is instead handled by the function that calls it. An optional input variable for total number of draws, `NumDraws`, is also accepted. The default value for this integer is 1. The algorithm used by the function is discussed above in Section 4.3.

The function provides one output variable, z, a matrix with row dimension equal to the length of mu and column dimension equal to NumDraws.

4.5.20. CauchyRndFcn

The function CauchyRndFcn computes random draws from a Cauchy distribution. The function takes two required input variables, mu and sigma. These contain the location and scale parameters; see equation (4.24). The function does not check if the dimensions of these variables match; this is instead handled by the function that calls it. An optional input variable for total number of draws, NumDraws, is also accepted. The default value for this integer is 1. The algorithm used by the function is discussed above in Section 4.3.

The function provides one output variable, z, a matrix with row dimension equal to the length of mu and column dimension equal to NumDraws.

4.5.21. MultiStudentTRndFcn

The function MultiStudentTRndFcn generates a desired number of draws from the multivariate Student-t distribution. As input the function needs mu, CholSigma, and d. These contain the location vector μ, the lower triangular Choleski decomposition of the positive definite scale matrix Σ, and degrees of freedom parameter d, respectively; see equation (4.25). The last input, NumDraws, is optional and defaults to 1.

As output the function gives z, a matrix with as the same number of rows as the dimension of the mean and number of columns gives by NumDraws.

4.5.22. LogisticRndFcn

The function LogisticRndFcn computes random draws from a Type I generalized logistic distribution. The function takes 3 required input variables, mu, sigma and c. These contain the location, scale and shape parameters; see equation (4.28). The function does not check if the dimensions of these variables match; this is instead handled by the function that calls it. An optional input variable for total number of draws, NumDraws, is also accepted. The default value for this integer is 1. The algorithm used by the function is discussed above in Section 4.3.

The function provides one output variable, z, a matrix with row dimension equal to the length of mu and column dimension equal to NumDraws.

4.5.23. GumbelRndFcn

The function GumbelRndFcn computes random draws from a Gumbel distribution. The function takes two required input variables, mu and sigma. These contain the location and scale parameters; see equation (4.31). The function does not check if the dimensions of these variables match; this is instead handled by the function that calls it. An optional input variable for total number of draws, NumDraws, is also accepted. The default value for this integer is 1. The algorithm used by the function is discussed above in Section 4.3.

The function provides one output variable, z, a matrix with row dimension equal to the length of mu and column dimension equal to NumDraws.

4.5.24. ParetoRndFcn

The function ParetoRndFcn computes random draws from a Pareto distribution. The function takes two required input variables, a and b. These contain the shape and location parameters; see equation (4.34). The function does not check if the dimensions of these variables match; this is instead handled by the function that calls it. An optional input variable for total number of draws, NumDraws, is also accepted. The default value for this integer is 1. The algorithm used by the function is discussed above in Section 4.3. Since the function is based on a zero value for the origin parameter, it can be added to the draws as needed.

The function provides one output variable, z, a matrix with row dimension equal to the length of a and column dimension equal to NumDraws.

5. The Kalman Filter

The solution to a log-linearized DSGE model can be written as a VAR model, where some of the variables may be unobserved. This suggests that any approach to estimation of its parameters is closely linked to estimation of state-space models. In the case of DSGE models, the solution of the model is represented by the state equation, while the measurement equation provides the link from the state (or model) variables into the observable variables, the steady-state of the DSGE model, and possible measurement errors.

The Kalman filter was originally developed by Kalman (1960) and Kalman and Bucy (1961) to estimate unobserved variables from observables via a state-space structure. A classic example for its use concern tracking a satellite's orbit around the earth. The unknown state would then be the position and speed of the satellite at a point in time with respect to a spherical coordinate system with the origin at the center of the earth. These quantities cannot be measured directly, while tracking stations around the earth may collect measurements of the distance to the satellite and the angles of measurement. The Kalman filter not only provides an optimal estimate (in a mean-squared error sense) of the position and speed of the satellite at the next instant in time, but also an optimal estimate of its current position and speed.

The output from the Kalman filter can also be used to compute the sample log-likelihood function of the data for a given set of parameter values, i.e., $\ln p(Y|\theta)$, once we have made distributional assumptions about the noise terms in the state-space model. Another by-product of the filter is an optimal forecast of the observed variables in the model. The notation used in this section follows the notation in Hamilton (1994, Chapter 13) closely, where details on the derivation of the filter are also found; see also Anderson and Moore (1979), Durbin and Koopman (2012) and Harvey (1989) for details and applications based on the Kalman filter. For a Bayesian interpretation of the filter, see, e.g., Meinhold and Singpurwalla (1983).

Apart from presenting the standard Kalman filter and smoothing recursions and the calculation of the log-likelihood function, this Section covers a number of additional aspects. The estimators of the unobserved variables can be decomposed into shares determined by the individual observables over the full conditioning sample as well as specific periods in time. It is thereby possible to analyse which individual observation matter for the estimates of unobserved variables, such as the output gap. Estimation of the conditional distribution of the entire *sample* of the unobserved variables is handled by the so called simulation smoother. Furthermore, numerical aspects can be improved on by relying on square root filtering and smoothing. The issue of missing observations is easily dealt with in state-space models and is briefly discussed. Moreover, the issue of initialization is dealt with, where the case of exact diffuse initialization gives rise to modifications of the filtering and smoothing recursions during an initial sample. Finally, it is possible to express the multivariate filtering and smoothing problems as univariate problems and thereby avoid certain matrix inversions and other possibly large matrix manipulations. This suggests that the univariate approach may improve computational speed as well as numerical accuracy, especially under diffuse initialization.

5.1. *The State-Space Representation*

Let y_t denote an n-dimensional vector of variables that are observed at date t. The *measurement* (or observation) equation for y is given by:

$$y_t = A'x_t + H'\xi_t + w_t. \tag{5.1}$$

The vector x_t is k-dimensional and contains only deterministic variables. The vector ξ_t is r-dimensional and is known as the state vector and contains possibly unobserved variables. The term w_t is white noise and is called the measurement error.

The *state* (or transition) equation determines the dynamic development of the state variables. It is here given by:

$$\xi_t = F\xi_{t-1} + v_t, \tag{5.2}$$

where F is the state transition matrix. The term v_t is white noise and it is assumed that v_t and w_τ are uncorrelated for all t and τ, with

$$E[v_t v_\tau'] = \begin{cases} Q \text{ for } t = \tau, \\ 0 \text{ otherwise,} \end{cases}$$

while

$$E[w_t w_\tau'] = \begin{cases} R \text{ for } t = \tau, \\ 0 \text{ otherwise.} \end{cases}$$

The parameter matrices are given by A $(k \times n)$, H $(r \times n)$, F $(r \times r)$, Q $(r \times r)$, and R $(n \times n)$. These matrices are known once we provide a value for θ. To initialize the process described by (5.1) and (5.2), it is assumed that ξ_1 is uncorrelated with any realizations of v_t or w_t.

5.2. *The Kalman Filter Recursion*

Let $\mathcal{Y}_t = \{y_t, y_{t-1}, \ldots, y_1, x_t, x_{t-1}, \ldots, x_1\}$ denote the set of observations up to and including period t. The Kalman filter provides a method for computing optimal 1-step ahead forecasts of y_t conditional on its past values and on the vector x_t as well as the associated forecast error covariance matrix.[40] These forecasts and their mean squared errors can then be used to compute the value of the log-likelihood function for y. Given the state-space representation in (5.1) and (5.2), it can directly be seen that the calculation of such forecasts requires forecasts of the state vector ξ_t conditional on the observed variables.

Let $\xi_{t+1|t}$ denote the linear projection of ξ_{t+1} on $\mathcal{Y}_t$. The Kalman filter calculates these forecasts recursively, generating $\xi_{1|0}$, $\xi_{2|1}$, and so on. Associated with each of these forecasts is a mean squared error matrix, represented by

$$P_{t+1|t} = E\left[(\xi_{t+1} - \xi_{t+1|t})(\xi_{t+1} - \xi_{t+1|t})'\right].$$

Similarly, let $y_{t|t-1}$ be the linear projection of y_t on $\mathcal{Y}_{t-1}$ and x_t. From the measurement equation (5.1) and the assumption about w_t we have that:

$$y_{t|t-1} = A'x_t + H'\xi_{t|t-1}. \tag{5.3}$$

The forecast error for the observed variables can therefore be expressed as

$$y_t - y_{t|t-1} = H'\left(\xi_t - \xi_{t|t-1}\right) + w_t. \tag{5.4}$$

It follows that the 1-step ahead forecast error covariance matrix for the observed variables is given by

$$E\left[(y_t - y_{t|t-1})(y_t - y_{t|t-1})'\right] \equiv \Sigma_{y,t|t-1} = H'P_{t|t-1}H + R. \tag{5.5}$$

To compute the forecasts and forecast error covariance matrices for the observed variables (y_t) we therefore need to know the sequence of forecasts and forecast error covariance matrices of the state variables (ξ_t).

Projecting the state variables in period $t+1$ on $\mathcal{Y}_t$ we find from equation (5.2) that

$$\xi_{t+1|t} = F\xi_{t|t}, \tag{5.6}$$

where $\xi_{t|t}$ is called the updated or filtered value of ξ_t. From standard results on linear projections (see, e.g., Hamilton, 1994, Chapter 4.5), the updated value of ξ_t relative to its forecasted value is given by:

$$\xi_{t|t} = \xi_{t|t-1} + P_{t|t-1}H\Sigma_{y,t|t-1}^{-1}\left(y_t - y_{t|t-1}\right). \tag{5.7}$$

Substituting (5.7) into (5.6), gives us:

$$\xi_{t+1|t} = F\xi_{t|t-1} + K_t\left(y_t - y_{t|t-1}\right), \tag{5.8}$$

where

$$K_t = FP_{t|t-1}H\Sigma_{y,t|t-1}^{-1}, \tag{5.9}$$

---

[40] The forecasts are optimal in a mean squared error sense among any functions of $(x_t, \mathcal{Y}_{t-1})$.

is known as the *Kalman gain matrix.*

The product between $P_{t|t-1}$ and H in (5.7) is equal to the covariance matrix for the state forecast errors, $\xi_t - \xi_{t|t-1}$, and the observed variable forecast errors in (5.4). This means that the difference between the update and the forecast of state variables (the time t revision to the time $t-1$ forecast) is proportional to the time t forecast error for the observed variables. The $r \times n$ matrix which is premultiplied to the latter error is simply the covariance matrix between the state and observed variables forecast errors times the inverse of the forecast error covariance matrix of the observed variables.

From (5.7) it can also be seen that that the update error for the state variables, $\xi_t - \xi_{t|t}$, is equal to the forecast error of the state variables minus the impact of the observed variable forecast error on the state variable update, i.e., the second term of the right hand side of this equation. Taking the correlation between the forecast errors for the state variables and the observed variables into accound, the update covariance matrix for the state variables can be shown to be equal to

$$P_{t|t} = P_{t|t-1} - P_{t|t-1}H\Sigma_{y,t|t-1}^{-1}H'P_{t|t-1}. \tag{5.10}$$

We may alternatively define the update innovation

$$r_{t|t} = H\Sigma_{y,t|t-1}^{-1}(y_t - y_{t|t-1}), \tag{5.11}$$

so that the Kalman update equation is

$$\xi_{t|t} = \xi_{t|t-1} + P_{t|t-1}r_{t|t}. \tag{5.12}$$

It may be noted that $r_{t|t}$ has mean zero and covariance matrix $H\Sigma_{y,t|t-1}^{-1}H'$. This innovation will be shown to be useful below when we are interested in computing the state equation innovations conditional on $\mathcal{Y}_t$.

To complete the filter, it remains to calculate $P_{t+1|t}$. It is straightforward to relate this matrix to $P_{t|t}$ through

$$P_{t+1|t} = FP_{t|t}F' + Q. \tag{5.13}$$

There are several ways to compute $P_{t+1|t}$ matrix using $P_{t|t-1}$ and one is based on the Kalman gain matrix. This updating formula states that

$$P_{t+1|t} = (F - K_tH')P_{t|t-1}(F - K_tH')' + K_tRK_t' + Q. \tag{5.14}$$

The Kalman gain-based expression for $P_{t+1|t}$ follows from noticing the that difference between the state variables at $t+1$ and the forecast can be expressed as

$$\xi_{t+1} - \xi_{t+1|t} = (F - K_tH')(\xi_t - \xi_{t|t-1}) - K_tw_t + v_{t+1}.$$

This relationship is obtained by using equation (5.8) for $\xi_{t+1|t}$, the state equation for ξ_{t+1}, and the measurement equation for y_t. The three terms on the right hand side are uncorrelated conditional on the information at t, thus yielding (5.14).

To summarize, the Kalman filter recursions for forecasting the state variables are given by (5.8) and (5.14), while equations (5.7) (or (5.11) and (5.12)) and (5.10) are the recursions for updating the state variables. Finally, equations (5.3) and (5.5) determines the 1-step ahead forecast and associated error covariance matrix for the observed variables.

5.3. *Initializing the Kalman Filter*

To run the Kalman filter recursions we now need to have initial values for the state variable forecast $\xi_{1|0}$ and its covariance matrix $P_{1|0}$. To start up the algorithm we may let:

$$\xi_{1|0} = E[\xi_1] = \mu_\xi.$$

In a DSGE model, the state variables are measured as deviations from steady state. A candidate value for μ_ξ is therefore zero so that all state variables are initially in steady state. This candidate value is, for instance, reasonable when ξ_t is covariance stationary, i.e., if all eigenvalues of F are inside the unit circle. Provided that this is the case, the unconditional covariance matrix

$E[\xi_t \xi_t'] = \Sigma_\xi$ exists. From the state equation (5.2) we find that:

$$\Sigma_\xi = F\Sigma_\xi F' + Q. \tag{5.15}$$

The solution to (5.15) is given by

$$\mathrm{vec}(\Sigma_\xi) = \left[I_{r^2} - (F \otimes F)\right]^{-1}\mathrm{vec}(Q), \tag{5.16}$$

where vec is the column stacking operator, and $\otimes$ the Kronecker product. One candidate for $P_{1|0}$ is therefore Σ_ξ.

However, if r is large then the calculation of Σ_ξ in (5.16) may be too cumbersome, especially if this has to be performed frequently (e.g., during the stage of drawing from the posterior). In such cases, it may be better to make use of the *doubling algorithm*. Equation (5.15) is a Lyapunov equation, i.e., a special case of the Sylvester equation. Letting $\gamma_0 = Q$ and $\alpha_0 = F$ we can express the iterations

$$\gamma_k = \gamma_{k-1} + \alpha_{k-1}\gamma_{k-1}\alpha_{k-1}', \quad k = 1, 2, \ldots \tag{5.17}$$

where

$$\alpha_k = \alpha_{k-1}\alpha_{k-1}.$$

The specification in (5.17) is equivalent to expressing:

$$\gamma_k = \sum_{j=0}^{2^k-1} F^j Q F'^j.$$

From this relation we can see that $\lim_{k \to \infty} \gamma_k = \Sigma_\xi$. Moreover, each iteration doubles the number of terms in the sum and we expect the algorithm to converge quickly since $\|\alpha_k\|$ should be close to zero also for relatively small k when all the eigenvalues of F lie inside the unit circle.

Alternatively, it may be better to let $P_{1|0} = cI_r$ for some constant c. The larger c is the less informative the initialization is for the filter. YADA allows for both alternatives to using equation (5.16) for initializing $P_{1|0}$. In addition, the exact treatment of diffuse initialization when $c \to \infty$ is discussed in Section 5.14.

5.4. *The Likelihood Function*

One important reason for running the Kalman filter for DSGE models is that as a by-product it provides us with the value of the log-likelihood function when the state shocks and the measurement errors have know distributions. The log-likelihood function for $y_T, y_{T-1}, \ldots, y_1$ can be expressed as:

$$\ln L\left(y_T, y_{T-1}, \ldots, y_1 \,|\, x_T, \ldots, x_1 \,; \theta\right) = \sum_{t=1}^{T} \ln p\left(y_t | x_t, \mathcal{Y}_{t-1}; \theta\right), \tag{5.18}$$

by the usual factorization and assumption regarding x_t. To compute the right hand side for a given θ, we need to make some distributional assumptions regarding ξ_1, v_t, and w_t.

In YADA it is assumed that ξ_1, v_t, and w_t are multivariate Gaussian with Q and R being positive semidefinite. This means that:

$$\begin{aligned}
\ln p\left(y_t | x_t, \mathcal{Y}_{t-1}; \theta\right) = &-\frac{n}{2}\ln(2\pi) - \frac{1}{2}\ln\left|\Sigma_{y,t|t-1}\right| + \\
&-\frac{1}{2}\left(y_t - y_{t|t-1}\right)'\Sigma_{y,t|t-1}^{-1}\left(y_t - y_{t|t-1}\right).
\end{aligned} \tag{5.19}$$

The value of the sample log-likelihood can thus be calculated directly from the 1-step ahead forecast errors of y_t and the associated forecast error covariance matrix.

5.5. *Smoothed Projections of the State Variables*

Since many of the elements of the state vector are given structural interpretations in the DSGE framework, it is important to use as much information as possible to project this vector. That is, we are concerned with the smooth projections $\xi_{t|T}$. In equation (5.7) we already have such a projection for $t = T$. It thus remains to determine the backward looking projections for $t < T$.

Hamilton (1994, Chapter 13.6) shows that the smooth projections of the state vector are given by:

$$\xi_{t|T} = \xi_{t|t} + J_t\big(\xi_{t+1|T} - \xi_{t+1|t}\big), \quad t = 1, 2, \ldots, T - 1, \tag{5.20}$$

where the Kalman smoothing matrix J_t is given by

$$J_t = P_{t|t} F' P_{t+1|t}^{-1}.$$

The mean squared error matrix of the smooth projection $\xi_{t|T}$ is now:

$$P_{t|T} = P_{t|t} + J_t\big(P_{t+1|T} - P_{t+1|t}\big)J_t'. \tag{5.21}$$

To calculate $\xi_{t|T}$ and $P_{t|T}$ one therefore starts in period $t = T - 1$ and then iterates backwards until $t = 1$. Derivations of these equations can be found in, e.g., Ansley and Kohn (1982).

The smoothing expression in (5.20) requires that $P_{t+1|t}$ is a full rank matrix. This assumption is violated if, for instance, $R = 0$ (no measurement errors) and rank(Q) $< r$ with $P_{1|0} = \Sigma_\xi$. In this case we may use an eigenvalue decomposition such that $P_{t+1|t} = S\Lambda S'$, where S is $r \times s$, $r > s$, Λ is a diagonal full rank $s \times s$ matrix, and $S'S = I_s$. Replacing J_t with

$$J_t^* = P_{t|t} F' S (S' P_{t+1|t} S)^{-1} S',$$

the smoothing algorithm remains otherwise intact; see, e.g., Kohn and Ansley (1983).[41]

There is an alternative algorithm for computing the smoother which does not require an explicit expression for the inverse of the state forecast covariance matrix $P_{t+1|t}$; see, e.g., De Jong (1988, 1989), Kohn and Ansley (1989), or Koopman and Harvey (2003). The smoothed state vector can be rewritten as

$$\xi_{t|T} = \xi_{t|t-1} + P_{t|t-1} r_{t|T}, \quad t = 1, 2, \ldots, T. \tag{5.22}$$

The vector $r_{t|T}$ is here given by

$$r_{t|T} = H\Sigma_{y,t|t-1}^{-1}\big(y_t - y_{t|t-1}\big) + \big(F - K_t H'\big)' r_{t+1|T}, \tag{5.23}$$

where $r_{T+1|T} = 0$.[42] The only matrix that has to be inverted is $\Sigma_{y,t|t-1}$, the forecast error covariance matrix of the observed variables, i.e., the same matrix that needs to be invertible for the Kalman filter to be valid.[43] Furthermore, notice that $P_{t|t-1}$ times the first term of the

---

[41] To show that $P^- = S\Lambda^{-1}S'$ is a generalized inverse of $P = S\Lambda S'$ we need to establish that $PP^-P = P$; see, e.g., Magnus and Neudecker (1988, p. 38). This follows directly from $S'S = I_s$, while $\Lambda^{-1} = (S'PS)^{-1}$ means that the generalized inverse may also be written as in the expression for J_t^*.

[42] These expressions can be derived by first substituting for $\xi_{t|t}$ in (5.20) using equation (5.7), replacing J_t with its definition, and substituting for $P_{t|t}$ from (5.10). Next, we take the definition of the Kalman gain matrix K_t into account and rearrange terms. This gives us (5.22), with

$$r_{t|T} = H\Sigma_{y,t|t-1}^{-1}\big(y_t - y_{t|t-1}\big) + \big(F - K_t H'\big)' P_{t+1|t}^{-1}\big(\xi_{t+1|T} - \xi_{t+1|t}\big), \quad t = 1, 2, \ldots, T - 1. \tag{5.23'}$$

The difference between the smoothed value and the forecast value of ξ_{t+1} in period $t = T - 1$ is obtained from (5.7), yielding

$$\xi_{T|T} - \xi_{T|T-1} = P_{T|T-1} H\Sigma_{y,T|T-1}^{-1}\big(y_T - y_{T|T-1}\big).$$

Subsituting this into (5.23') for $t = T - 1$ gives us

$$r_{T-1|T} = H\Sigma_{y,T-1|T-2}^{-1}\big(y_{T-1} - y_{T-1|T-2}\big) + \big(F - K_{T-1}H'\big)' H\Sigma_{y,T|T-1}^{-1}\big(y_T - y_{T|T-1}\big)$$

$$= H\Sigma_{y,T-1|T-2}^{-1}\big(y_{T-1} - y_{T-1|T-2}\big) + \big(F - K_{T-1}H'\big)' r_{T|T},$$

where $r_{T|T} = H\Sigma_{y,T|T-1}^{-1}(y_T - y_{T|T-1})$. This implies that (5.23) holds for $t = T - 1$ and, in addition, for $t = T$ provided that $r_{T+1|T} = 0$. Notice also that if we substitute the definition of $r_{t|T}$ into equation (5.22) for $t = T$ it gives us the identical expression to what we have in (5.7).

Returning to (5.22) we now have that $(\xi_{T-1|T} - \xi_{T-1|T-2}) = P_{T-1|T-2} r_{T-1|T}$ so that $P_{T-1|T-2}^{-1}(\xi_{T-1|T} - \xi_{T-1|T-2}) = r_{T-1|T}$. From (5.23') we then find that (5.23) holds $T - 2$ as well. Continuing backwards recursively it can be established that (5.23) holds for $t = 1, 2, \ldots, T - 1$.

[43] When the Kalman filter and smoother is used to estimate unobserved variables of the model, then the inverse of $\Sigma_{y,t|t-1}$ may be replaced with a generalized inverse; see, e.g., Harvey (1989, p. 106). However, a singular covariance matrix is not compatible with the existence of a density for $y_t|\mathcal{Y}_{t-1}, \theta$. Hence, the log-likelihood function cannot be calculated for models where $\Sigma_{y,t|t-1}$ does not have full rank.

expression for $r_{t|T}$ gives the update part for ξ_t relative to the forecast part, while the same matrix times the second term generates the forward looking part. Hence, the alternative expression for the smoother has an intuitive interpretation, where the smoother is directly linked to the information in the data at $t-1$, t, and therafter from $t+1$ until T.

The mean squared error matrix for the smoothed projections can likewise be expressed as

$$P_{t|T} = P_{t|t-1} - P_{t|t-1} N_{t|T} P_{t|t-1}, \quad t = 1, 2, \ldots, T, \tag{5.24}$$

where

$$N_{t|T} = H \Sigma_{y,t|t-1}^{-1} H' + \left(F - K_t H' \right)' N_{t+1|T} \left(F - K_t H' \right), \tag{5.25}$$

and $N_{T+1|T} = 0$. It can again be seen that the only matrix that has to be inverted is the forecast error covariance matrix of the observed variables. Notice also that the first term in the expression for $N_{t|T}$ is related to the update part of the covariance matrix for the state variables, while the second term is linked to the forward looking part. Moreover, these terms clarify why $P_{t|T} \leq P_{t|t}$ in a matrix sense.

5.6. Smoothed and Updated Projections of State Shocks and Measurement Errors

Smoothing allows us to estimate the shocks to the state equations using as much information as possible. In particular, by projecting both sides of equation (5.2) on the data until period T we find that

$$v_{t|T} = \xi_{t|T} - F \xi_{t-1|T}, \quad t = 2, \ldots, T. \tag{5.26}$$

To estimate v_1 using the full sample, it would seem from this equation that we require an estimate of $\xi_{0|T}$. However, there is a simple way around this issue.

For $\xi_{t-1|T}$ we premultiply both sides of (5.20) with F, substitute (5.6) for $F\xi_{t-1|t-1}$, and replace (5.22) for $(\xi_{t|T} - \xi_{t|t-1})$. Next, take the definition of J_{t-1} into account and use (5.13) for $FP_{t-1|t-1}F'$. These manipulations give us

$$F \xi_{t-1|T} = \xi_{t|t-1} + \left(P_{t|t-1} - Q \right) r_{t|T}.$$

At the same time equation (5.22) gives us an expression for $\xi_{t|T}$. Using (5.26) it follows that

$$v_{t|T} = Q r_{t|T}, \quad t = 2, \ldots, T. \tag{5.27}$$

The variable $r_{t|T}$ in (5.22) is therefore seen to have the natural interpretation that when premultiplied by the covariance matrix of the state shock (Q), we obtain the smoothed projection of the state shock. In other words, $r_{t|T}$ in (5.22) is an innovation with mean zero and covariance matrix $N_{t|T}$.

Equation (5.27) provides us with a simple means for estimating $v_{t|T}$ for $t = 1$ since $r_{t|T}$ is defined for that period. In other words, equation (5.27) can be applied to compute the smooth projection of the state shocks not only for periods $t = 2, \ldots, T$ but also for period $t = 1$, i.e.

$$v_{1|T} = Q r_{1|T}.$$

To calculate update projections of the state shocks, denoted by $v_{t|t}$, we need the smooth projection $\xi_{t-1|t}$. Since equation (5.20) holds for all $T > t$ we can replace T with $t + 1$. Using the definition of J_t and equation (5.7) for $(\xi_{t+1|t+1} - \xi_{t+1|t})$ we obtain

$$\xi_{t|t+1} = \xi_{t|t} + P_{t|t} F' H \Sigma_{y,t+1|t}^{-1} \left(y_{t+1} - y_{t+1|t} \right), \quad t = 1, \ldots, T-1. \tag{5.28}$$

Premultiplying $\xi_{t-1|t}$ with F, replacing $FP_{t-1|t-1}F'$ with $P_{t|t-1} - Q$ (see equation 5.13), and taking equation (5.6) into account we find that

$$F \xi_{t-1|t} = \xi_{t|t-1} + \left(P_{t|t-1} - Q \right) H \Sigma_{y,t|t-1}^{-1} \left(y_t - y_{t|t-1} \right).$$

It now follows from (5.7) and (5.11) that

$$v_{t|t} = QH \Sigma_{y,t|t-1}^{-1} \left(y_t - y_{t|t-1} \right) = Q r_{t|t}, \quad t = 2, \ldots, T. \tag{5.29}$$

Hence, the state shock projected on observations up to period t is a linear combination of the forecast error for the observed variables. Notice also that $v_{t|t}$ is equal to Q times the first term

on the right hand side of (5.23) and therefore has a natural connection with the smoothed projection of the state shock.

Moreover, a simple estimator of $v_{1|1}$ is suggested by equation (5.29). Specifically,

$$v_{1|1} = QH \left[H'P_{1|0}H + R \right]^{-1} \left(y_1 - A'x_1 - H'\xi_{1|0} \right) = Qr_{1|1}.$$

Implicitly, this assumes that $F\xi_{0|1}$ is equal to

$$F\xi_{0|1} = \xi_{1|0} + \left(P_{1|0} - Q \right) H \left[H'P_{1|0}H + R \right]^{-1} \left(y_1 - A'x_1 - H'\xi_{1|0} \right).$$

It is also interesting to note that the covariance matrix of the updated projection of the state shock is given by

$$E\left[v_{t|t} v'_{t|t} \right] = QH\Sigma^{-1}_{y,t|t-1}H'Q.$$

This matrix is generally not equal to Q, the covariance matrix of v_t, unless ξ_t is observable at t.[44] In fact, it can be shown that $Q \geq QH\Sigma^{-1}_{y,t|t-1}H'Q$ in a matrix sense since the difference between these matrices is equal to the covariance matrix of $(v_t - v_{t|t})$. That is,

$$E\left[(v_t - v_{t|t})(v_t - v_{t|t})'|\mathcal{Y}_t \right] = Q - QH\Sigma^{-1}_{y,t|t-1}H'Q, \quad t = 1, \ldots, T. \tag{5.30}$$

Moreover, the covariance matrix of the smoothed projection of the state shock is

$$E\left[v_{t|T} v'_{t|T} \right] = QN_{t|T}Q.$$

From (5.25) if also follows that $QN_{t|T}Q \geq QH\Sigma^{-1}_{y,t|t-1}H'Q$ for $t = 1, \ldots, T$. Moreover, we find that

$$E\left[(v_t - v_{t|T})(v_t - v_{t|T})'|\mathcal{Y}_T \right] = Q - QN_{t|T}Q, \quad t = 1, \ldots, T. \tag{5.31}$$

Additional expressions for covariance matrices of the state shocks are given in Koopman (1993); see also Durbin and Koopman (2012, Chapter 4.7).

The measurement errors can likewise be estimated using the sample information once the update or smoothed state variables have been computed. In both cases, we turn to the measurement equation (5.1) and replace the state variables with the update or smooth estimator. For the update estimator of the state variables we find that

$$w_{t|t} = y_t - A'x_t - H'\xi_{t|t} = R\Sigma^{-1}_{y,t|t-1} \left(y_t - y_{t|t-1} \right), \quad t = 1, \ldots, T, \tag{5.32}$$

where we have substituted for update estimate of the state variables using equation (5.7) and rearranging terms. It follows from (5.32) that the update estimate of the measurement error is zero when $R = 0$. Moreover, we find from this equation that the covariance matrix is given by

$$E\left[w_{t|t} w'_{t|t} \right] = R\Sigma^{-1}_{y,t|t-1}R.$$

It follows that $R \geq R\Sigma^{-1}_{y,t|t-1}R$ in a matrix sense since the difference is a positive semidefinite matrix. In fact,

$$E\left[(w_t - w_{t|t})(w_t - w_{t|t})'|\mathcal{Y}_t \right] = R - R\Sigma^{-1}_{y,t|t-1}R, \quad t = 1, \ldots, T. \tag{5.33}$$

It may also be noted that the population cross-covariance matrix for the update estimates of the state shocks and the measurement errors is given by

$$E\left[v_{t|t} w'_{t|t} \right] = QH\Sigma^{-1}_{y,t|t-1}R.$$

Unlike the population cross-covariance matrix for the state shocks and measurement errors, this matrix is generally not zero.

Similarly, the smoothed estimate of the measurement error can be computed from the measurement equation. If we subsitute for the smooth estimate of the state variables and rerrange

---

[44] Observability of ξ_t at t means that $P_{t|t} = 0$. From equation (5.13) we therefore know that $P_{t|t-1} = Q$. The claim follows by noting that $P_{t|t}$ always satisfies equation (5.10) and that $\Sigma_{y,t|t-1} = H'QH + R$.

terms

$$w_{t|T} = y_t - A'x_t - H'\xi_{t|T}$$
$$= R\Sigma_{y,t|t-1}^{-1} \left(y_t - y_{t|t-1} - H'P_{t|t-1}F'r_{t+1|T} \right), \quad t = 1, \ldots, T. \tag{5.34}$$

If the covariance matrix $R = 0$, then $w_{t|T} = 0$ by construction. Notice also that the covariance matrix is equal to:

$$E\left[w_{t|T} w_{t|T}' \right] = R\Sigma_{y,t|t-1}^{-1} \left(\Sigma_{y,t|t-1} + H'P_{t|t-1}F'N_{t+1|T}FP_{t|t-1}H \right) \Sigma_{y,t|t-1}^{-1} R,$$
$$= R \left(\Sigma_{y,t|t-1}^{-1} + K_t'N_{t+1|T}K_t \right) R.$$

From this it follows that $E[w_{t|T}w_{t|T}'] \geq E[w_{t|t}w_{t|t}']$ in a matrix sense. Moreover,

$$E\left[(w_t - w_{t|T})(w_t - w_{t|T})' | \mathcal{Y}_T \right] = R - R \left(\Sigma_{y,t|t-1}^{-1} + K_t'N_{t+1|T}K_t \right) R, \quad t = 1, \ldots, T. \tag{5.35}$$

The measurement equation can also be extended to the case when the measurement matrix H is time-varying. With H replaced by H_t in (5.1), the filtering, updating, and smoothing equations are otherwise unaffected as long as H_t is treated as known. YADA is well equipped to deal with a time-varying measurement matrix; see Sections 5.17 and 18.4 for details.

5.7. Multistep Forecasting

The calculation of h-step ahead forecasts of y is now straightforward. From the state equation (5.2) we know that for any $h \geq 1$

$$\xi_{t+h|t} = F\xi_{t+h-1|t}. \tag{5.36}$$

The h-step ahead forecast of y is therefore:

$$y_{t+h|t} = A'x_{t+h} + H'\xi_{t+h|t}. \tag{5.37}$$

The mean squared error matrix for the $\xi_{t+h|t}$ forecast is simply:

$$P_{t+h|t} = F^h P_{t|t}(F')^h + \sum_{i=0}^{h-1} F^i Q(F')^i$$
$$= FP_{t+h-1|t}F' + Q. \tag{5.38}$$

Finally, the mean squared error matrix for the $y_{t+h|t}$ forecast is:

$$E\left[(y_{t+h} - y_{t+h|t})(y_{t+h} - y_{t+h|t})' \right] = H'P_{t+h|t}H + R. \tag{5.39}$$

The h-step ahead forecasts of y and the mean squared error matrix can thus be built iteratively from the forecasts of the state variables and their mean squared error matrix.

5.8. Covariance Properties of the Observed and the State Variables

It is sometimes of interest to compare the covariances of the observed variables based on the model to the actual data or some other model. It is straightforward to determine these moments provided that the state vector is stationary, i.e., that the eigenvalues of F are inside the unit circle. The unconditional covariance matrix for the state vector is then given by Σ_ξ, which satisfies equation (5.15).

From the state equation (5.2) we can rewrite the first order VAR as follows for any non-negative integer h:

$$\xi_t = F^h \xi_{t-h} + \sum_{i=0}^{h-1} F^i v_{t-i}. \tag{5.40}$$

From this it immediately follows that the autocovariance function of the state variables, given the parameter values, based on the state-space model is:

$$E\left[\xi_t \xi_{t-h}' \right] = F^h \Sigma_\xi. \tag{5.41}$$

The autocovariance function of the observed variables can now be calculated from the measurement equation (5.1). With $E[y_t] = A'x_t$, it follows that for any non-negative integer h

$$E\big[(y_t - A'x_t)(y_{t-h} - A'x_{t-h})'\big] = \begin{cases} H'\Sigma_\xi H + R, & \text{if } h = 0, \\ H'F^h\Sigma_\xi H, & \text{otherwise.} \end{cases} \quad (5.42)$$

From equations (5.41) and (5.42) it can be seen that the autocovariances tend to zero as h increases. For given parameter values we may, e.g., compare these autocovariances to those obtained directly from the data.

5.9. *Computing Weights on Observations for the State Variables*

The Kalman filter and associated algorithms calculates optimal estimates of the unobserved state variables using the parameters, the initial conditions for the state variables, and the data for the observed variables y and x. Since the state-space model is linear, this means that we expect that the forecast, update and smooth projections of the state variables can be expressed as weighted sums of the inputs.

Koopman and Harvey (2003) show how such weights can be determined analytically for a linear state-space model using inputs from the Kalman filter.[45] Although the model in (5.1) and (5.2) is a special case of their setup, the results that they derive are based on the assumption that the initial value of the state vector, $\xi_{1|0}$, is zero. Since YADA allows for a non-zero initial value it makes sense to reconsider their approach to this case. In addition, some useful expressions for how the weights can be computed more efficiently in practise will be considered below.

5.9.1. *Weights for the Forecasted State Variable Projections*

Letting $z_t = y_t - A'x_t$ and $L_t = F - K_t H'$ we can rewrite equation (5.8) such that

$$\xi_{t+1|t} = L_t \xi_{t|t-1} + K_t z_t, \quad t = 1, \ldots, T-1. \quad (5.43)$$

Substituting for $\xi_{t|t-1}, \xi_{t-1|t-2}, \ldots, \xi_{2|1}$ in (5.43) and rearranging terms it can be shown that

$$\xi_{t+1|t} = \sum_{\tau=1}^{t} \alpha_\tau(\xi_{t+1|t}) z_\tau + \beta_0(\xi_{t+1|t})\xi_{1|0}, \quad t = 1, \ldots, T-1, \quad (5.44)$$

where the $r \times n$ matrix

$$\alpha_\tau(\xi_{t+1|t}) = \beta_\tau(\xi_{t+1|t})K_\tau, \quad \tau = 1, \ldots, t, \quad (5.45)$$

while the $r \times r$ matrix

$$\beta_\tau(\xi_{t+1|t}) = \begin{cases} \prod_{i=0}^{t-1-\tau} L_{t-i}, & \text{if } \tau = 0, 1, \ldots, t-1, \\ I_r, & \text{if } \tau = t. \end{cases} \quad (5.46)$$

Notice that $\beta_\tau(\xi_{t|t-1}) = L_{t-1}\cdots L_{\tau+1}$ for $\tau = 0, 1, \ldots, t-1$. We can therefore also express this matrix as

$$\beta_\tau(\xi_{t+1|t}) = L_t\beta_\tau(\xi_{t|t-1}), \quad \tau = 0, 1, \ldots t-1.$$

If the intension is to compute the state variable forecasts for all $(t = 1, 2, \ldots, T-1)$, this latter relationship between β-matrices is very useful in practise. It means that for the forecast of ξ_{t+1} we premultiply all the β-matrices for the forecast of ξ_t by the same matrix L_t and then add $\beta_t(\xi_{t+1|t}) = I_r$ to the set of β-matrices.

---

[45] Their version of the state-space model allows the matrices H, R, F, and Q to be time-varying and known for all $t = 1, \ldots, T$ when $t \geq 1$. See also Gómez (2007) for conditions when the weighting matrices of Koopman and Harvey (2003) converge to those obtained from steady-state recursions, i.e., to the asymptotes.

Moreover, the α_τ-matrices for the forecast of ξ_{t+1} are related to those for the forecast of ξ_t through

$$\alpha_\tau\left(\xi_{t+1|t}\right) = \begin{cases} L_t \alpha_\tau\left(\xi_{t|t-1}\right), & \text{if } \tau = 1, \ldots, t-1, \\ K_t, & \text{if } \tau = t. \end{cases}$$

In fact, the relationship between $\alpha_\tau(\xi_{t+1|t})$ and $\alpha_\tau(\xi_{t|t-1})$ can be inferred directly from (5.43). Furthermore, from (5.44) we see that the only β-matrix we need to keep track of is $\beta_0(\xi_{t+1|t})$ since it is the weight on the initial condition $\xi_{1|0}$.

5.9.2. *Weights for the Updated State Variable Projections*

With $M_t = H\Sigma_{y,t|t-1}^{-1}$ and $G_t = M_t H'$, the updated projection of ξ_t in (5.7) can be rewritten as

$$\xi_{t|t} = \left[I_r - P_{t|t-1}G_t\right]\xi_{t|t-1} + P_{t|t-1}M_t z_t. \tag{5.47}$$

Using equation (5.44) for the forecast of ξ_t it follows that we can express the updated projection of ξ_t as

$$\xi_{t|t} = \sum_{\tau=1}^{t} \alpha_\tau\left(\xi_{t|t}\right)z_\tau + \beta_0\left(\xi_{t|t}\right)\xi_{1|0}, \tag{5.48}$$

where

$$\alpha_\tau\left(\xi_{t|t}\right) = \begin{cases} \left[I_r - P_{t|t-1}G_t\right]\alpha_\tau\left(\xi_{t|t-1}\right), & \text{if } \tau = 1, \ldots, t-1, \\ P_{t|t-1}M_t, & \text{if } \tau = t, \end{cases} \tag{5.49}$$

while $\beta_0(\xi_{t|t}) = [I_r - P_{t|t-1}G_t]\beta_0(\xi_{t|t-1})$.

5.9.3. *Weights for the Smoothed State Variable Projections*

To compute the weights for the smoothed estimator of the state variables we may proceed in two step. First, we determine the weights for the $r_{t|T}$ vector via (5.23) through a backward recursion. Once we have these weights, it is straightforward to determine the weights for $\xi_{t|T}$ via (5.22).

The equation for $r_{t|T}$ in (5.23) can be rewritten as

$$r_{t|T} = M_t z_t - G_t \xi_{t|t-1} + L_t' r_{t+1|T}. \tag{5.50}$$

The general expression for $r_{t|T}$ as a weighted series of the observed data and the initial conditions is given by

$$r_{t|T} = \sum_{\tau=1}^{T} \alpha_\tau\left(r_{t|T}\right)z_\tau + \beta_0\left(r_{t|T}\right)\xi_{1|0}, \quad t = 1, \ldots, T.$$

For period $t = T$, where $r_{T+1|T} = 0$, it then follows from (5.50) that

$$\alpha_\tau\left(r_{T|T}\right) = \begin{cases} -G_T \alpha_\tau\left(\xi_{T|T-1}\right), & \text{if } \tau = 1, \ldots, T-1, \\ M_T, & \text{if } \tau = T, \end{cases} \tag{5.51}$$

and $\beta_0(r_{T|T}) = -G_T\beta_0(\xi_{T|T-1})$. For periods $t = 1, \ldots, T-1$ it likewise follows from (5.50) that

$$\alpha_\tau\left(r_{t|T}\right) = \begin{cases} L_t'\alpha_\tau\left(r_{t+1|T}\right) - G_t\alpha_\tau\left(\xi_{t|t-1}\right), & \text{if } \tau = 1, \ldots, t-1, \\ M_t + L_t'\alpha_\tau\left(r_{t+1|T}\right) & \text{if } \tau = t, \\ L_t'\alpha_\tau\left(r_{t+1|T}\right) & \text{if } \tau = t+1, \ldots, T. \end{cases} \tag{5.52}$$

The weights on the initial condition is given by $\beta_0(r_{t|T}) = L_t'\beta_0(r_{t+1|T}) - G_t\beta_0(\xi_{t|t-1})$.

The smoothed state variable projections are expressed as a weighted sum of the observed variables and the initial conditions as follows:

$$\xi_{t|T} = \sum_{\tau=1}^{T} \alpha_\tau(\xi_{t|T}) z_\tau + \beta_0(\xi_{t|T}) \xi_{1|0}, \quad t = 1, \ldots, T, \tag{5.53}$$

where the weights on the observations z_τ are obtained through (5.22) and are given by:

$$\alpha_\tau(\xi_{t|T}) = \begin{cases} \alpha_\tau(\xi_{t|t-1}) + P_{t|t-1}\alpha_\tau(r_{t|T}), & \text{if } \tau = 1, \ldots, t-1, \\ P_{t|t-1}\alpha_\tau(r_{t|T}), & \text{if } \tau = t, \ldots, T. \end{cases} \tag{5.54}$$

The weights on the initial conditions is $\beta_0(\xi_{t|T}) = \beta_0(\xi_{t|t-1}) + P_{t|t-1}\beta_0(r_{t|T})$.

These expressions allow us to examine a number of interesting questions. For example, column j of the $\alpha_\tau(\xi_{t|T})$ weight matrix tells us how much the smooth estimate of the state variables for period t will change if the j:th observed variable changes by 1 unit in period τ. Letting this column be denoted by $\alpha_{\tau,j}(\xi_{t|T})$ while $z_{j,\tau}$, denotes the j:th element of z_τ, we can decompose the smooth estimates such that

$$\xi_{t|T} = \sum_{j=1}^{n} \sum_{\tau=1}^{T} \alpha_{\tau,j}(\xi_{t|T}) z_{j,\tau} + \beta_0(\xi_{t|T}) \xi_{1|0}.$$

Hence, we obtain an expression for the share of the smoothed projection of the state variables at t which is determined by each observed variable and of the initial condition. Similar expression can be computed for the state forecasts and the updates.

Since $z_t = y_t - A' x_t$ it follows that the weights for the state forecast, update and smoother are equal to the weights on y_t. If we wish to derive the specific weights on x_t we simply postmultiply the α_τ-matrices with $-A'$. For example, in case $x_t = 1$ we have that

$$\xi_{t|T} = \sum_{j=1}^{n} \sum_{\tau=1}^{T} \alpha_{\tau,j}(\xi_{t|T}) y_{j,\tau} + \sum_{\tau=1}^{T} \gamma_\tau(\xi_{t|T}) + \beta_0(\xi_{t|T}) \xi_{1|0},$$

where the r-dimensional vector $\gamma_\tau(\xi_{t|T}) = -\alpha_\tau(\xi_{t|T}) A'$ for $\tau = 1, \ldots, T$.

It is also straightforward to express the smooth projection of the measurement error in (5.34) as a weighted sum of the observed variables and the initial condition. Specifically, from (5.53) we obtain

$$w_{t|T} = \sum_{\tau=1}^{T} \alpha_\tau(w_{t|T}) z_\tau + \beta_0(w_{t|T}) \xi_{1|0}, \quad t = 1, \ldots, T, \tag{5.55}$$

where

$$\alpha_\tau(w_{t|T}) = \begin{cases} -H'\alpha_\tau(\xi_{t|T}), & \text{if } \tau = 1, \ldots, t-1, t+1, \ldots, T, \\ I_n - H'\alpha_\tau(\xi_{t|T}), & \text{if } \tau = t. \end{cases} \tag{5.56}$$

Moreover, the $n \times r$ matrix with weights on the initial condition is $\beta_0(w_{t|T}) = -H'\beta_0(\xi_{t|T})$. Observation weights for the state shocks can be determined from the observation weights for the economic shocks. This topic is covered in Section 11.1.

5.10. *Simulation Smoothing*

The smooth estimates of the state variables, shocks and measurement errors that we discussed in Sections 5.5 and 5.6 above give the expected value from the distributions of these unobserved variables conditional on the data (and the parameters of the state-space model). Suppose instead that we would like to draw samples of these variables from their conditional distributions. Such draws can be obtained through what is generally known as *simulation smoothing*; see, e.g., Durbin and Koopman (2012, Chapter 4.9).

Frühwirth-Schnatter (1994) and Carter and Kohn (1994) independently developed techniques for simulation smoothing of the state variables based on

$$p(\xi_1, \ldots, \xi_T | y_T) = p(\xi_T | y_T) \cdot p(\xi_{T-1} | \xi_T, y_T) \cdots p(\xi_1 | \xi_2, \ldots, \xi_T, y_T). \tag{5.57}$$

The methods suggested in these papers are based on drawing recursively from the densities on the right hand side of (5.57), starting with $p(\xi_T|\mathcal{Y})$, then from $p(\xi_{T-1}|\xi_T, \mathcal{Y}_T)$, and so on. Their techniques were improved on in De Jong and Shephard (1995) who concentrated on sampling the shocks (and measurement errors) and thereafter the state variables. The algorithm that I shall discuss relies on the further significant enhancements suggested by Durbin and Koopman (2002), which have made simulation smoothing both simpler and faster, and which is also discussed by Durbin and Koopman (2012).

Let $\omega = [w_1' \, v_1' \, \cdots \, w_T' \, v_T']'$, a vector of dimension $(n + r)T$ with the measurement errors and state shocks. We know that the conditional distribution of ω is normal, where the smooth estimates in Section 5.6 gives the conditional mean, $E[\omega|\mathcal{Y}_T]$, and that the covariance matrix is independent of $\mathcal{Y}_T$, i.e., $Cov[\omega|\mathcal{Y}_T] = C$. We also know that the marginal distribution of ω is $N(0, \Omega)$, where $\Omega = (I_T \otimes \text{diag}[R, Q])$, while $\xi_1 \sim N(\mu_\xi, \Sigma_\xi)$ and independent of ω.

Drawing vectors $\tilde{\omega}$ from $p(\omega|\mathcal{Y}_T)$ can be achieved by drawing vectors from $N(0, C)$ and adding the draws to $E[\omega|\mathcal{Y}_T]$. Durbin and Koopman suggest that this can be achieved through the following three steps.

(1) Draw $\omega^{(i)}$ from $N(0, \Omega)$ and $\xi_1^{(i)}$ from $N(\mu_\xi, \Sigma_\xi)$. Substitute these values into the state-space model in (5.1)–(5.2) to simulate a sequence for $y_t^{(i)}$, $t = 1, \ldots, T$, denoted by $\mathcal{Y}_T^{(i)}$.

(2) Compute $E[\omega|\mathcal{Y}_T^{(i)}]$ via equations (5.27) and (5.34).

(3) Take $\tilde{\omega}^{(i)} = E[\omega|\mathcal{Y}_T] + \omega^{(i)} - E[\omega|\mathcal{Y}_T^{(i)}]$ as a draw from $p(\omega|\mathcal{Y}_T)$.

If the covariances matrices R and Q have reduced rank, we can lower the dimension of ω such that only the unique number of sources of error are considered. Moreover, if we are only interested in the simulation smoother for, say, the state shocks, then we need only compute the smooth estimates $v_{t|T}$ and $v_{t|T}^{(i)} = E[v_t|\mathcal{Y}_T^{(i)}]$, while $\tilde{v}_t^{(i)} = v_{t|T} + v_t^{(i)} - v_{t|T}^{(i)}$, $t = 1, \ldots, T$, is a draw from $p(v_1, \ldots, v_T|\mathcal{Y}_T)$. Notice that the first step of the algorithm always has to be performed.

To see why the third step of the algorithm gives a draw from $p(\omega|\mathcal{Y}_T)$, note that $\omega^{(i)} - E[\omega|\mathcal{Y}_T^{(i)}]$ is independent of $\mathcal{Y}_T$. This means that, conditional on $\mathcal{Y}_T$, the vector $\tilde{\omega}^{(i)}$ is drawn from a normal distribution with mean $E[\omega|\mathcal{Y}_T]$ and covariance matrix $E[(\omega^{(i)} - E[\omega|\mathcal{Y}_T^{(i)}])(\omega^{(i)} - E[\omega|\mathcal{Y}_T^{(i)}])'] = C$. Furthermore, a draw of ξ_t, t=1, ..., T, from $p(\xi_1, \ldots, \xi_T|\mathcal{Y}_T)$ is directly obtained as a by-product from the above algorithm by letting $\tilde{\xi}_T^{(i)} = \xi_{t|T} + \xi_t^{(i)} - \xi_{t|T}^{(i)}$.

Durbin and Koopman (2002) also suggest that *antithetic variables* can be computed via these results. Recall that an antithetic variable in this context is a function of the random draw which is equiprobable with the draw, and which when used together with the draw increases the efficiency of the estimation; see, e.g., Mikhail (1972) and Geweke (1988). It follows that $\tilde{\omega}^{(-i)} = E[\omega|\mathcal{Y}_T] - \omega^{(i)} + E[\omega|\mathcal{Y}_T^{(i)}]$ is one such antithetic variable for $\tilde{\omega}^{(i)}$. If a simulation smoother of ω is run for $i = 1, \ldots, N$, then the simulation average of $\tilde{\omega}^{(i)}$ is not exactly equal to $E[\omega|\mathcal{Y}_T]$, while the simulation average of $\tilde{\omega}^{(i)}$ plus $\tilde{\omega}^{(-i)}$ is, i.e., the simulation sample is *balanced for location*. Additional antithetic variables for dealing with second moments of the simulation can be constructed as suggested by Durbin and Koopman (1997); see also Durbin and Koopman (2012, Chapter 11.4.3). The calculation of such variables for balancing the scale of the simulation and how further efficiency gains can be made from properly taking the rank of R and Q, respectively, into account are discussed in Section 11.1.

5.11. *Chandrasekhar Recursions*

An equivalent formulation of the state covariance updates in equation (5.14) is given by the matrix Riccati difference equation

$$P_{t+1|t} = F\left[P_{t|t-1} - P_{t|t-1}H\Sigma_{y,t|t-1}^{-1}H'P_{t|t-1}\right]F' + Q. \tag{5.58}$$

When the number of state variables (r) is large relative to the number of observed variables (n), it is possible to use the *Chandrasehkar recursions* of the state covariance matrix, originally developed by Morf, Sidhu, and Kailath (1974) and recently suggested by Herbst (2015), to

improve the computational time; see also Anderson and Moore (1979, Chapter 6.7). For these recursions to be valid, the system matrices need to be constant or, as in Aknouche and Hamdi (2007), at least periodic, while the state variables are stationary.

Defining $\Delta P_{t+1|t} = P_{t+1|t} - P_{t|t-1}$ and utilizing, e.g., Lemma 7.1 in Anderson and Moore (1979) (or equivalently the Lemma in Herbst, 2015) we find that

$$\Delta P_{t+1|t} = \left(F - K_t H'\right)\left(\Delta P_{t|t-1} - \Delta P_{t|t-1} H \Sigma_{y,t-1|t-2}^{-1} H' \Delta P_{t|t-1}\right)\left(F - K_t H'\right)' \tag{5.59}$$

With $\Delta P_{t+1|t} = W_t M_t W_t'$ the following recursive expressions hold

$$\Sigma_{y,t|t-1} = \Sigma_{y,t-1|t-2} + H' W_{t-1} M_{t-1} W_{t-1}' H, \tag{5.60}$$

$$K_t = \left(K_{t-1}\Sigma_{y,t-1|t-2} + F W_{t-1} M_{t-1} W_{t-1}' H\right)\Sigma_{y,t|t-1}^{-1}, \tag{5.61}$$

$$M_t = M_{t-1} + M_{t-1} W_{t-1}' H \Sigma_{y,t-1|t-2}^{-1} H' W_{t-1} M_{t-1}, \tag{5.62}$$

$$W_t = \left(F - K_t H'\right)W_{t-1}. \tag{5.63}$$

Notice that the computation of $P_{t+1|t}$ involves multiplication of $r \times r$ matrices in (5.14) and (5.58). Provided that n is sufficiently smaller than r, as is typically the case with DSGE models, the computational gains from the Chandrasekhar recursions in (5.60)–(5.63) can be substantial. This is particularly the case when we only wish to evaluate the log-likelihood function in (5.19) since the state covariance matrix is not explicitly needed in this expression. Being able to recursive compute $\Sigma_{y,t|t-1}$ without involving multiplications of $r \times r$ matrices can therefore be highly beneficial.

To initialize these recursions we need values of W_1 and M_1, Herbst (2015) notes that for $t = 1$ the matrix Riccati equation in (5.58) can be expresses as

$$P_{2|1} = F P_{1|0} F' + Q - K_1 \Sigma_{y,1|0} K_1'$$

Provided that we have initialized the Kalman filter with $P_{1|0} = \Sigma_\xi$ in equation (5.15), it follows that

$$P_{1|0} = F P_{1|0} F' + Q,$$

so that

$$\Delta P_{2|1} = -K_1 \Sigma_{y,1|0} K_1' = W_1 M_1 W_1'.$$

This suggests that

$$W_1 = K_1, \quad M_1 = -\Sigma_{y,1|0}. \tag{5.64}$$

Alternatively, we can replace recursions of the Kalman gain matrix with recursions of $\tilde{K}_t = F P_{t|t-1} H$. This means that we replace equation (5.61) with

$$\tilde{K}_t = \tilde{K}_{t-1} + F W_{t-1} M_{t-1} W_{t-1}' H.$$

This equation involves fewer multiplications than the recursive equation for the Kalman gain matrix and may therefore be preferred. Furthermore, equation (5.63) can be rewritten as

$$W_t = \left(F - \tilde{K}_t \Sigma_{y,t|t-1}^{-1} H'\right)W_{t-1}.$$

The intializations of W_1 and M_1 are affected by these changes to the Chandrasekhar recursions and are now given by

$$W_1 = \tilde{K}_1, \quad M_1 = -\Sigma_{y,1|0}^{-1}.$$

This also emphasizes that the matrices W_t and M_t are not uniquely determined.

As pointed out by Herbst (2015), the Chandrasekhar recursions are sensitive to the numerical accuracy of the solution to the Lyapunov equation in (5.15). When the doubling algorithm in Section 5.3 is used to compute $P_{1|0}$ it is therefore recommended that the number of recursions k is large enough for the error matrix

$$\Gamma_{\epsilon,k+1} = \gamma_{k+1} - \gamma_k = \sum_{j=2^k}^{2^{k+1}-1} F Q F'^j,$$

to be sufficiently close to zero. The convergence criterion for the doubling algorithm in YADA concerns the norm of $\Gamma_{\epsilon,k+1}$, i.e., the largest singular value of this matrix. This makes it straightforward to have control over the numerical accuracy of the solution to the Lyapunov equation when this fast, numerical approach is used.

It may also be noted that if $\Sigma_\xi = \gamma_k$, then the error of the Lyapunov equation (5.15) is given by

$$F\gamma_k F' + Q - \gamma_k = F^{2^k} Q F'^{2^k}.$$

This is equal to the term in $\Gamma_{\epsilon,k+1}$ with the smallest exponent and since each increment to the error matrix is positive semidefinite the norm of the solution error for γ_k is smaller than (or equal to) the norm of the error matrix. Furthermore, using γ_{k+1} as the solution for the Lyapunov equation yields a solution error which is never bigger than that of γ_k.

5.12. *Square Root Filtering*

It is well known that the standard Kalman filter can sometimes fail to provide an error covariance matrix which is positive semidefinite. For instance, this can happen when some of the observed variables have a very small variance or when pairs of variables are highly correlated. The remedy to such purely numerical problems is to use a square root filter; see Morf and Kailath (1975), Anderson and Moore (1979, Chapter 6.5), or Durbin and Koopman (2012, Chapter 6.3).

To setup a square root filter we first need to define the square roots of some of the relevant matrices. Specifically, let $P_{t|t-1}^{1/2}$ be the $r \times r$ square root of $P_{t|t-1}$, $R^{1/2}$ the $n \times n$ square root of R, while B_0 is an $r \times q$ square root of Q, with $r \geq q$.[46] That is,

$$P_{t|t-1} = P_{t|t-1}^{1/2} P_{t|t-1}^{1/2\prime}, \quad R = R^{1/2} R^{1/2\prime}, \quad Q = B_0 B_0'. \tag{5.65}$$

Let us now define the $(n + r) \times (r + n + q)$ matrix

$$U_t = \begin{bmatrix} H' P_{t|t-1}^{1/2} & R^{1/2} & 0 \\ F P_{t|t-1}^{1/2} & 0 & B_0 \end{bmatrix}. \tag{5.66}$$

Notice that

$$U_t U_t' = \begin{bmatrix} \Sigma_{y,t|t-1} & H' P_{t|t-1} F' \\ F P_{t|t-1} H & F P_{t|t-1} F' + Q \end{bmatrix}.$$

The matrix U_t can be transformed to a lower triangular matrix using the orthogonal matrix G, such that $GG' = I_{n+r+q}$. Postmultiplying U_t by G we obtain

$$U_t G = U_t^*, \tag{5.67}$$

where

$$U_t^* = \begin{bmatrix} U_{1,t}^* & 0 & 0 \\ U_{2,t}^* & U_{3,t}^* & 0 \end{bmatrix}. \tag{5.68}$$

The three matrices inside U_t^* are given by

$$U_{1,t}^* = \Sigma_{y,t|t-1}^{1/2}, \quad U_{2,t}^* = F P_{t|t-1} H \Sigma_{y,t|t-1}^{-1/2\prime}, \quad U_{3,t}^* = P_{t+1|t}^{1/2}.$$

This means that the Kalman filter step in (5.8) can be expressed as

$$\xi_{t+1|t} = F \xi_{t|t-1} + U_{2,t}^* \left(U_{1,t}^* \right)^{-1} \left(y_t - y_{t|t-1} \right). \tag{5.69}$$

Moreover, the state covariance matrix in (5.14) can instead be computed from

$$P_{t+1|t} = U_{3,t}^* U_{3,t}^{*\prime}. \tag{5.70}$$

---

[46] The matrix B_0 is defined in Section 3 and satisfies $v_t = B_0 \eta_t$, where $\eta_t \sim N(0, I_q)$, where q is the number of iid economic shocks. Hence, $Q = B_0 B_0'$ and B_0 may therefore be regarded as a proper square root matrix of Q.

The right hand side of (5.70) is positive semidefinite by construction and the same is true for

$$\Sigma_{y,t|t-1} = U_{1,t}^* U_{1,t}^{*\prime}.$$

The crucial step in the square root filter is the computation of the orthogonal matrix G. One approach to calculating this matrix along with U_t^* is the so called *Q-R factorization*. For an $n \times m$ matrix A this factorization is given by $A = QR$, where the $n \times n$ matrix Q is orthogonal ($QQ' = I_n$), while the $n \times m$ matrix R is upper triangular; see, e.g., Golub and van Loan (1983, p. 147) or the qr function in Matlab. To obtain U_t^* we therefore compute the Q-R factorization of U_t' and let U_t^* be the tranpose of the upper triangular matrix R (which should not be confused with the covariance matrix of the measurement errors), while G is given by the Q matrix.

It is also possible to calculate square root versions of the update and smooth estimates of the states and of the state covariance matrices. Concerning the update estimates we have that $\xi_{t|t}$ is determined as in equation (5.12), where the square root filter means that $P_{t|t-1}$ is based on the lag of equation (5.70), while

$$r_{t|t} = H \left(U_{1,t}^* U_{1,t}^{*\prime} \right)^{-1} \left(y_t - y_{t|t-1} \right). \tag{5.71}$$

Furthermore, the update covariance matrix is determined by equation (5.10) where $P_{t|t-1}$ and $\Sigma_{y,t|t-1}$ are computed from the square root expressions.

The smothed state variables can similarly be estimated using the square root filter by utilizing equation (5.22), where $P_{t|t-1}$ is again determined from the output of the square root filter, while the expression for the smooth innovations in equation (5.23) can now be expressed as

$$r_{t|T} = r_{t|t} + \left(F - U_{2,t}^* \left(U_{1,t}^* \right)^{-1} H' \right)' r_{t+1|T}, \tag{5.72}$$

with $r_{T+1|T} = 0$ and where $r_{t|t}$ is given by (5.71).

The state covariance matrix for the smooth estimates is computed as in equation (5.24), but where a square root formula for $N_{t|T}$ is now required. Following Durbin and Koopman (2012) we introduce the $r \times r$ lower triangular matrix $N_{t+1|T}^*$ which satisfies

$$N_{t+1|T} = N_{t+1|T}^* N_{t+1|T}^{*\prime}.$$

Next, define the $r \times (n + r)$ matrix

$$\tilde{N}_{t|T} = \left[H \left(U_{1,t}^{*\prime} \right)^{-1} \quad \left(F - U_{2,t}^* \left(U_{1,t}^* \right)^{-1} H' \right)' N_{t+1|T}^* \right], \tag{5.73}$$

from which it follows that $N_{t|T} = \tilde{N}_{t|T} \tilde{N}_{t|T}'$; see equation (5.25). The last step is now to compute $N_{t|T}^*$ from $\tilde{N}_{t|T}$ through the $(n + r) \times (n + r)$ matrix G, i.e.,

$$\tilde{N}_{t|T} G = \left[N_{t|T}^* \quad 0 \right], \tag{5.74}$$

where $GG' = I_{n+r}$. A Q-R factorization of $\tilde{N}_{t|T}'$ may therefore be considered, where $N_{t|T}^*$ is obtained as the r first columns of the transpose of the R matrix from the factorization. The square root based algorithm for $N_{t|T}$ is initialized by letting $N_{T+1|T}^* = 0$.

5.13. *Missing Observations*

Dealing with missing observations is relatively straightforward in state-space models. We can simply think of this as adding a layer of measurement on top of the original measurement equation (5.1); see, e.g., Durbin and Koopman (2012, Chapter 4.10), Harvey (1989, Chapter 3.4.7), Harvey and Pierse (1984), or Kohn and Ansley (1983).

Specifically, suppose that $y_t^{(o)}$ is n_t-dimension and related to y_t through the equation

$$y_t^{(o)} = S_t' y_t, \quad t = 1, \ldots, T, \tag{5.75}$$

where the matrix S_t is $n \times n_t$ and $n_t \leq n$ with rank equal to n_t. The columns of this matrix are given by columns of I_n, whose i:th column is denoted by e_i. The i:th element of y_t, denoted

by $y_{i,t}$, is observed at t if and only if e_i is a column of S_t, i.e., $e_i'S_t \neq 0$. In the event that observations on all variables are missing for some t, then S_t is empty for that period. Hence, the vector $y_t^{(o)}$ is given by all observed elements in the vector y_t, while the elements with missing values are skipped.

The measurement equation of the state-space model can now be expressed as

$$y_t^{(o)} = A_t^{(o)\prime} x_t + H_t^{(o)\prime} \xi_t + w_t^{(o)}, \quad t = 1, \dots, T, \tag{5.76}$$

where $A_t^{(o)} = AS_t$, $H_t^{(o)} = HS_t$, and $w_t^{(o)} = S_t'w_t$. This means that the measurement errors are given by $w_t^{(o)} \sim N(0, R_t^{(o)})$, where the covariance matrix is given by $R_t^{(o)} = S_t'RS_t$.

The Kalman filtering and smoothing equations under missing observations have the same general form as before, except that the parameter matrices in (5.76) replace A, H, and R when $n_t \geq 1$. In the event that $n_t = 0$, i.e., there are no observation in period t, then the 1-step ahead forecasts of the state variables become

$$\xi_{t+1|t} = F\xi_{t|t-1}, \tag{5.77}$$

while the covariance matrix of the 1-step ahead forecast errors of the state variables is

$$P_{t+1|t} = FP_{t|t-1}F' + Q. \tag{5.78}$$

That is, the Kalman gain matrix, K_t, is set to zero; cf. equations (5.8) and (5.14). For the update estimates we likewise have that $r_{t|t} = 0$, $\xi_{t|t} = \xi_{t|t-1}$, and $P_{t|t} = P_{t|t-1}$.

Furthermore, when $n_t = 0$ then the smoothing equation in (5.23) is replaced by

$$r_{t|T} = F'r_{t+1|T}, \tag{5.79}$$

while the smoothing equation in (5.25) is instead given by

$$N_{t|T} = F'N_{t+1|T}F. \tag{5.80}$$

Regarding the log-likelihood function in (5.19) we simply set it to 0 for the time periods when $n_t = 0$, while n_t replaces n, $y_t^{(o)}$ is used instead of y_t, and $\Sigma_{y^{(o)},t|t-1}$ instead of $\Sigma_{y,t|t-1}$ for all time periods when $n_t \geq 1$.

5.14. *Diffuse Initialization of the Kalman Filter*

The Kalman filters and smoothers that we have considered thus far rely on the assumption that the initial state ξ_1 has a (Gaussian) distribution with mean μ_ξ and finite covariance matrix Σ_ξ, where the filtering recursions are initialized by $\xi_{1|0} = \mu_\xi$ and $P_{1|0} = \Sigma_\xi$; this case may be referred to as the *standard initialization*.

More generally, the initial state vector ξ_1 can be specified as

$$\xi_1 = \mu_\xi + S\delta + S_\perp \nu_1, \tag{5.81}$$

where S is an $r \times s$ selection matrix with $s \leq r$ columns from I_r, $S_\perp$ is an $r \times (r - s)$ selection matrix with the remaining columns of I_r ($S'S_\perp = 0$), while $\delta \sim N(0, cI_s)$ and $\nu_1 \sim N(0, \Sigma_\nu)$. If we let $s = 0$ and $\Sigma_\nu = \Sigma_\xi$ the initialization in (5.81) is identical to the standard initialization. Furthermore, if $s = r$ and c is finite another common initialization for the Kalman filter is obtained with $\xi_1 \sim N(\mu_\xi, cI_r)$. A large value for c may be regarded as an approximation of diffuse initialization where the Kalman filters and smoothers discussed above are still valid. However, this may be numerically problematic and an exact treatment of the case when $c \to \infty$ is instead likely to be preferred. In what follows, we shall consider how the Kalman filter and smoothing recursions need to be adapted to allow for such an initialization. If $0 < s < r$ this means that the initialization is *partially diffuse* and when $s = r$ it is *fully diffuse*.

An exact treatment of diffuse Kalman filtering and smoothing was first suggested by Ansley and Kohn (1985, 1990) and later simplified by Koopman (1997) and Koopman and Durbin (2003); see also Durbin and Koopman (2012, Chapter 5). The transition to the standard Kalman filter occurs at some point in time $t = d$ and is automatic. An alternative approach based on augmentation was given by De Jong (1991) for filtering and smoothing, where the latter step was simplified by De Jong and Chu-Chun-Lin (2003). A transition to the usual Kalman filter

after some point in time $t = d$ is here optional, but advisable for computational efficiency. The discussion below follows the treatment in Koopman and Durbin (2003).

It should already at this point be stressed that diffuse initialization will *not* converge at a finite point in time $t = d$ if some state variable has a unit root, such as for a random walk. Moreover, near non-stationarity is also a source for concern. In this situation d will be finite, but can still be very large, difficult to determine numerically, and on top of this greater than the sample size. Numerical issues are likely to play an important role under near unit roots with certain covariance matrices possibly being close to singular or with problems determining their rank. For such models it is recommended to stick with a finite initialization of the state covariance matrix.

5.14.1. *Diffuse Kalman Filtering*

From equation (5.81) we define the covariance matrix of the initial value for the state variables with finite c as

$$P = cP_\infty + P_*,$$

where $P_\infty = SS'$ and $P_* = S_\perp \Sigma_v S'_\perp$. The case of full diffuse initialization means that $s = r$ so that $P_\infty = I_r$ while $P_* = 0$, while $0 < s < r$ concerns partial diffuse initialization. The implications for the Kalman filter and smoother are not affected by the choice of s, but the implementation in YADA mainly concerns $s = r$. In fact, if none of the variables have been specifically selected as a unit-root process, then full diffuse initialization is used. Similarly, state variables that have been selected as "unit-root" processes, whether they are or not, will be excluded from the s diffuse variables such that the corresponding diagonal element of P_∞ is zero, while the same diagonal element of P_* is unity. Further details on the selection of unit-root state variables is found in the YADA help file, while such variables can be selected via the *Actions menu* in YADA.

The 1-step ahead forecast error covariance matrix for the state variables can be decomposed in the same way as P so that

$$P_{t|t-1} = cP_{\infty,t|t-1} + P_{*,t|t-1} + O(c^{-1}), \quad t = 1, \ldots, T,$$

where $P_{\infty,t|t-1}$ and $P_{*,t|t-1}$ do not depend on c.[47] It is shown by Ansley and Kohn (1985) and Koopman (1997) that the influence of the term $P_{\infty,t|t-1}$ will disappear at some $t = d$ and that the usual Kalman filtering equations in Section 5.2 can be applied for $t = d + 1, \ldots, T$.

Consider the expansion of the inverse of the matrix

$$\Sigma_{y,t|t-1} = c\Sigma_{\infty,t|t-1} + \Sigma_{*,t|t-1} + O(c^{-1}), \quad t = 1, \ldots, T, \tag{5.82}$$

which appears in the definition of K_t in (5.9). The covariances matrices on the right hand side are given by

$$\begin{aligned} \Sigma_{\infty,t|t-1} &= H'P_{\infty,t|t-1}H, \\ \Sigma_{*,t|t-1} &= H'P_{*,t|t-1}H + R. \end{aligned} \tag{5.83}$$

Koopman and Durbin (2003) provide the Kalman filter equations for the cases when $\Sigma_{\infty,t|t-1}$ is nonsingular and $\Sigma_{\infty,t|t-1} = 0$. According to the authors, the case when $\Sigma_{\infty,t|t-1}$ has reduced rank $0 < n_t < n$ is rare, although an explicit solution is given by Koopman (1997). The authors suggest, however, that for the reduced rank case the univariate approach to multivariate Kalman filtering in Koopman and Durbin (2000) should be used instead. This is also the stance taken by YADA when $\Sigma_{\infty,t|t-1}$ is singular but not zero. The univariate approach for both the standard initialization and diffuse initialization is discussed in Section 5.15.

The exact initial state 1-step ahead forecast equations when $\Sigma_{\infty,t|t-1}$ is nonsingular and $c \to \infty$ are given by

$$\begin{aligned} \xi_{t+1|t} &= F\xi_{t|t-1} + K_{\infty,t}\left(y_t - y_{t|t-1}\right), \\ P_{\infty,t+1|t} &= FP_{\infty,t|t-1}L'_{\infty,t}, \\ P_{*,t+1|t} &= FP_{*,t|t-1}L'_{\infty,t} - K_{\infty,t}\Sigma_{\infty,t|t-1}K'_{*,t} + Q, \end{aligned} \tag{5.84}$$

---

[47] The term $O(c^{-1})$ refers to a function $f(c)$ such that $cf(c)$ is finite as $c \to \infty$.

for $t = 1, \ldots, d$, with

$$K_{\infty,t} = FP_{\infty,t|t-1}H\Sigma^{-1}_{\infty,t|t-1},$$

$$K_{*,t} = \left(FP_{*,t|t-1}H - K_{\infty,t}\Sigma_{*,t|t-1}\right)\Sigma^{-1}_{\infty,t|t-1},$$

$$L_{\infty,t} = F - K_{\infty,t}H',$$

with the initialization $\xi_{1|0} = \mu_\xi$, $P_{*,1|0} = P_*$ and $P_{\infty,1|0} = P_\infty$. The forecast $y_{t|t-1}$ is given by equation (5.3).[48]

From the forecast equations in (5.84) it can deduced that the Kalman update equation can be written as

$$\xi_{t|t} = \xi_{t|t-1} + P_{\infty,t|t-1}H\Sigma^{-1}_{\infty,t|t-1}\left(y_t - y_{t|t-1}\right)$$
$$= \xi_{t|t-1} + P_{\infty,t|t-1}r_{\infty,t|t}. \tag{5.85}$$

while the update state covariance matrices are

$$P_{\infty,t|t} = P_{\infty,t|t-1} - P_{\infty,t|t-1}H\Sigma^{-1}_{\infty,t|t-1}H'P_{\infty,t|t-1},$$

$$P_{*,t|t} = P_{*,t|t-1} - P_{*,t|t-1}H\Sigma^{-1}_{\infty,t|t-1}H'P_{\infty,t|t-1} - P_{\infty,t|t-1}H\Sigma^{-1}_{\infty,t|t-1}H'P_{*,t|t-1} \tag{5.86}$$

$$+ P_{\infty,t|t-1}H\Sigma^{-1}_{\infty,t|t-1}\Sigma_{*,t|t-1}\Sigma^{-1}_{\infty,t|t-1}H'P_{\infty,t|t-1}.$$

If $\Sigma_{\infty,t|t-1} = 0$, the exact initial state 1-step ahead forecast equations are instead

$$\xi_{t+1|t} = F\xi_{t|t-1} + K_{*,t}\left(y_t - y_{t|t-1}\right),$$

$$P_{\infty,t+1|t} = FP_{\infty,t|t-1}F', \tag{5.87}$$

$$P_{*,t+1|t} = FP_{*,t|t-1}L'_{*,t} + Q,$$

where

$$K_{*,t} = FP_{*,t|t-1}H\Sigma^{-1}_{*,t|t-1},$$

$$L_{*,t} = F - K_{*,t}H'.$$

It can here be seen that the forecast equations are identical to the standard Kalman filter equations where $P_{*,t+1|t}$ serves the same function as $P_{t+1|t}$, while an additional equation is provided for $P_{\infty,t+1|t}$. When $P_{\infty,d|d-1} = 0$ it follows that $\Sigma_{\infty,d|d-1} = 0$ and that it is no longer necessary to compute the equation for $P_{\infty,t+1|t}$. Accordingly, the diffuse Kalman filter has automatically shifted to the standard Kalman filter.

From (5.87) it is straightforward to show that the Kalman state variable update equation when $\Sigma_{\infty,t|t-1} = 0$ is

$$\xi_{t|t} = \xi_{t|t-1} + P_{*,t|t-1}H\Sigma^{-1}_{*,t|t-1}\left(y_t - y_{t|t-1}\right)$$
$$= \xi_{t|t-1} + P_{*,t|t-1}r_{*,t|t}, \tag{5.88}$$

while the update covariance matrices are given by

$$P_{\infty,t|t} = P_{\infty,t|t-1},$$

$$P_{*,t|t} = P_{*,t|t-1} - P_{*,t|t-1}H\Sigma^{-1}_{*,t|t-1}H'P_{*,t|t-1}. \tag{5.89}$$

Regarding the automatic collapse to the standard Kalman filter, for stationary state-space models Koopman (1997) notes that

$$\text{rank}\left(P_{\infty,t+1|t}\right) \leq \min\left\{\text{rank}\left(P_{\infty,t|t-1}\right) - \text{rank}\left(\Sigma_{\infty,t|t-1}\right), \text{rank}\left(F\right)\right\},$$

as long as the left hand side is nonnegative; see also Ansley and Kohn (1985). For example, with the intialization $P_* = 0$ and $P_\infty = I_r$, it is straightforward to show that

$$P_{\infty,2|1} = FP_HF', \qquad \Sigma_{\infty,2|1} = H'FP_HF'H,$$

[48] Notice that the second term within parenthesis in the expression for $K_{*,t}$ is subtracted from the first rather than added to it. The sign error in Koopman and Durbin (2003, page 89) can be derived using equations (5.7), (5.10), (5.12), and (5.14) in Durbin and Koopman (2012).

where $P_H = I_r - H(H'H)^{-1}H'$ and $\text{rank}(P_H) = r - n$. This means that the rank of $P_{\infty,2|1}$ is less than or equal to the minimum of $r - n$ and the rank of F. As long as $\Sigma_{\infty,t|t-1}$ is nonsingular it follows that the rank of $P_{\infty,t+1|t}$ drops quickly towards zero.

From the updating equation for $P_{\infty,t+1|t}$ in (5.87) it can be seen that if a row of F is equal to a row from the identity matrix, i.e., the corresponding state variable has a unit root, then the diagonal element in $P_{\infty,t+1|t}$ is equal to the same element in $P_{\infty,t|t-1}$. This means that the rank of $P_{\infty,t|t-1}$ will never converge to zero and whether the diffuse Kalman filter shifts to the standard filter depends on the development of the covariance matrix $\Sigma_{\infty,t|t-1}$. Moreover, in the case of near non-stationary where, say, a state variable follows an AR(1) process with autoregressive coefficient close to unity, the diffuse Kalman filter need not converge to standard filter within the chosen sample period.

When $\Sigma_{\infty,t|t-1}$ is nonsingular, we may deduce from the update equation (5.85) that the update estimator of the state shock is given by

$$v_{t|t} = Qr_{\infty,t|t}, \tag{5.90}$$

while the update measurement error is zero, i.e., $w_{t|t} = 0$. Similarly, when $\Sigma_{\infty,t|t-1} = 0$ it can be shown from equation (5.88) that the update estimator of the state shock is

$$v_{t|t} = Qr_{*,t|t}, \tag{5.91}$$

while the update estimate of the measurement error is equal to

$$w_{t|t} = R\Sigma_{*,t|t-1}^{-1}\left(y_t - y_{t|t-1}\right). \tag{5.92}$$

The log-likelihood function can also be computed for periods $t = 1,\dots,d$ as shown by Koopman (1997). In case $\Sigma_{\infty,t|t-1}$ is nonsingular the time t log-likelihood is equal to

$$\ln p\left(y_t|x_t, \mathcal{Y}_{t-1}; \theta\right) = -\frac{n}{2}\ln(2\pi) - \frac{1}{2}\ln\left|\Sigma_{\infty,t|t-1}\right|, \tag{5.93}$$

and if $\Sigma_{\infty,t|t-1} = 0$ we have that

$$\ln p\left(y_t|x_t, \mathcal{Y}_{t-1}; \theta\right) = -\frac{n}{2}\ln(2\pi) - \frac{1}{2}\ln\left|\Sigma_{*,t|t-1}\right| - \frac{1}{2}\left(y_t - y_{t|t-1}\right)'\Sigma_{*,t|t-1}^{-1}\left(y_t - y_{t|t-1}\right). \tag{5.94}$$

As expected, the log-likelihood in the latter case is exactly the same as for the standard filter; cf. equation (5.19).

5.14.2. *Diffuse Kalman Smoothing*

The diffuse Kalman filtering stage is covered by the sample $t = 1,\dots,d$, while the standard Kalman filter is used for the remaining sample dates $t = d+1,\dots,T$. This means that the standard Kalman smoother can be utilized for all dates *after* period d, while a modified smoother needs to be applied for the sample *until* period d.

The diffuse Kalman smoother for the state variables is given by

$$\xi_{t|T} = \xi_{t|t-1} + P_{*,t|t-1}r_{t|T}^{(0)} + P_{\infty,t|t-1}r_{t|T}^{(1)}, \quad t = d, d-1,\dots,1 \tag{5.95}$$

while the corresponding state covariance matrix is determined through

$$\begin{aligned}
P_{t|T} = &P_{*,t|t-1} - P_{*,t|t-1}N_{t|T}^{(0)}P_{*,t|t-1} - P_{\infty,t|t-1}N_{t|T}^{(1)}P_{*,t|t-1} \\
&- P_{*,t|t-1}N_{t|T}^{(1)'}P_{\infty,t|t-1} - P_{\infty,t|t-1}N_{t|T}^{(2)}P_{\infty,t|t-1}.
\end{aligned} \tag{5.96}$$

The initialization of the smoothing algorithm is:

$$r_{d+1|T}^{(0)} = r_{d+1|T}, \quad r_{d+1|T}^{(1)} = 0, \quad N_{d+1|T}^{(0)} = N_{d+1|T}, \quad N_{d+1|T}^{(1)} = N_{d+1|T}^{(2)} = 0.$$

For the case when $\Sigma_{\infty,t|t-1}$ is nonsingular it is shown by Koopman and Durbin (2003) that

$$\begin{aligned}
r_{t|T}^{(0)} &= L_{\infty,t}'r_{t+1|T}^{(0)}, \\
r_{t|T}^{(1)} &= H\left[\Sigma_{\infty,t|t-1}^{-1}\left(y_t - y_{t|t-1}\right) - K_{*,t}'r_{t+1|T}^{(0)}\right] + L_{\infty,t}'r_{t+1|T}^{(1)}.
\end{aligned} \tag{5.97}$$

The covariance matrices similarly evolve according to

$$
\begin{aligned}
N_{t|T}^{(0)} &= L_{\infty,t}' N_{t+1|T}^{(0)} L_{\infty,t}, \\
N_{t|T}^{(1)} &= H \Sigma_{\infty,t|t-1}^{-1} H' + L_{\infty,t}' N_{t+1|T}^{(1)} L_{\infty,t} - L_{\infty,t}' N_{t+1|T}^{(0)} K_{*,t} H' - H K_{*,t}' N_{t+1|T}^{(0)'} L_{\infty,t}, \\
N_{t|T}^{(2)} &= H \left[K_{*,t}' N_{t+1|T}^{(0)} K_{*,t} - \Sigma_{\infty,t|t-1}^{-1} \Sigma_{*,t|t-1} \Sigma_{\infty,t|t-1}^{-1} \right] H' + L_{\infty,t}' N_{t+1|T}^{(2)} L_{\infty,t} \\
&\quad - L_{\infty,t}' N_{t+1|T}^{(1)} K_{*,t} H' - H K_{*,t}' N_{t+1|T}^{(1)'} L_{\infty,t}.
\end{aligned}
\tag{5.98}
$$

In case $\Sigma_{\infty,t|t-1} = 0$ we instead find that

$$
\begin{aligned}
r_{t|T}^{(0)} &= H \Sigma_{*,t|t-1}^{-1} \left(y_t - y_{t|t-1} \right) + L_{*,t}' r_{t+1|T}^{(0)}, \\
r_{t|T}^{(1)} &= F' r_{t+1|T}^{(1)},
\end{aligned}
\tag{5.99}
$$

while the covariance matrices are given by

$$
\begin{aligned}
N_{t|T}^{(0)} &= H \Sigma_{*,t|t-1}^{-1} H' + L_{*,t}' N_{t+1|T}^{(0)} L_{*,t}, \\
N_{t|T}^{(1)} &= F' N_{t+1|T}^{(1)} L_{*,t}, \\
N_{t|T}^{(2)} &= F' N_{t+1|T}^{(2)} F.
\end{aligned}
\tag{5.100}
$$

The smoothing recursions in (5.97)–(5.98) and (5.99)–(5.100) may also be written more compactly as in Koopman and Durbin (2003, Section 4). Specifically, let $\tilde{H} = [I_2 \otimes H]$, whereas

$$
\tilde{r}_{t|T} = \begin{bmatrix} r_{t|T}^{(0)} \\ r_{t|T}^{(1)} \end{bmatrix}, \quad
\tilde{N}_{t|T} = \begin{bmatrix} N_{t|T}^{(0)} & N_{t|T}^{(1)} \\ N_{t|T}^{(1)} & N_{t|T}^{(2)} \end{bmatrix}, \quad
\tilde{z}_t = \begin{bmatrix} y_t - y_{t|t-1} \\ 0 \end{bmatrix},
$$

and the vector $\tilde{z}_t$ is $2n$-dimensional. If $\Sigma_{\infty,t|t-1}$ is nonsingular we let

$$
\tilde{\Sigma}_{t|t-1} = \begin{bmatrix} 0 & \Sigma_{\infty,t|t-1}^{-1} \\ \Sigma_{\infty,t|t-1}^{-1} & -\Sigma_{\infty,t|t-1}^{-1} \Sigma_{*,t|t-1} \Sigma_{\infty,t|t-1}^{-1} \end{bmatrix}, \quad
\tilde{L}_t = \begin{bmatrix} L_{\infty,t} & -K_{*,t} H' \\ 0 & L_{\infty,t} \end{bmatrix}.
$$

On the other hand, when $\Sigma_{\infty,t|t-1} = 0$ we instead define

$$
\tilde{\Sigma}_{t|t-1} = \begin{bmatrix} \Sigma_{*,t|t-1}^{-1} & 0 \\ 0 & 0 \end{bmatrix}, \quad
\tilde{L}_t = \begin{bmatrix} L_{*,t} & 0 \\ 0 & F \end{bmatrix}.
$$

For both cases, the recursions in (5.97)–(5.98) and (5.99)–(5.100) may be expressed as

$$
\tilde{r}_{t|T} = \tilde{H} \tilde{\Sigma}_{t|t-1} \tilde{z}_t + \tilde{L}_t' \tilde{r}_{t+1|T},
\tag{5.101}
$$

$$
\tilde{N}_{t|T} = \tilde{H} \tilde{\Sigma}_{t|t-1} \tilde{H}' + \tilde{L}_t' \tilde{N}_{t+1|T} \tilde{L}_t.
\tag{5.102}
$$

For the state shocks and the measurement errors, a nonsingular $\Sigma_{\infty,t|t-1}$ matrix yields the following smooth estimates

$$
v_{t|T} = Q r_{t|T}^{(0)}, \quad w_{t|T} = -R K_{\infty,t}' r_{t+1|T}^{(0)}.
\tag{5.103}
$$

The corresponding covariance matrices are

$$
C\left(v_t | \mathcal{Y}_T \right) = Q - Q N_{t|T}^{(0)} Q, \quad C\left(w_t | \mathcal{Y}_T \right) = R - R K_{\infty,t}' N_{t+1|T}^{(0)} K_{\infty,t} R.
$$

Similarly, when $\Sigma_{\infty,t|t-1} = 0$ we find that

$$
v_{t|T} = Q r_{t|T}^{(0)}, \quad w_{t|T} = R \left[\Sigma_{*,t|t-1}^{-1} \left(y_t - y_{t|t-1} \right) - K_{*,t}' r_{t+1|T}^{(0)} \right].
\tag{5.104}
$$

The covariance matrices for these estimates are

$$
C\left(v_t | \mathcal{Y}_T \right) = Q - Q N_{t|T}^{(0)} Q, \quad C\left(w_t | \mathcal{Y}_T \right) = R - R \left[\Sigma_{*,t|t-1}^{-1} + K_{*,t}' N_{t+1|T}^{(0)} K_{*,t} \right] R.
$$

Kalman filtering and smoothing when $\Sigma_{\infty,t|t-1}$ is singular but has at least rank one is covered by the univariate approach discussed in Section 5.15.2. When YADA encounters this case it automatically switches to a univariate routine for this time period and reverts back to the multivariate routine when the $\Sigma_{\infty,t|t-1}$ is either nonsingular or zero. The multivariate approach in Ansley and Kohn (1985) and Koopman (1997) is not be covered since it is mathematically complex and computationally inefficient.

5.15. A Univariate Approach to the Multivariate Kalman Filter

The standard approach to Kalman filtering and smoothing is based on taking the full observation vector into account at each point in time. The basic idea of univariate filtering is to incrementally add the individual elements of the vector of observed variables; see Anderson and Moore (1979, Chapter 6.4), Koopman and Durbin (2000), and Durbin and Koopman (2012, Chapter 6.4). One important reason for considering such an approach is computational efficiency, which can be particularly relevant when diffuse initialization is considered. In this subsection we shall first consider the univariate approach, also known as sequential processing, under standard initialization of the state vector, and thereafter the case of diffuse initialization.

5.15.1. Univariate Filtering and Smoothing with Standard Initialization

Correlated measurement errors are not taken into account by the univariate representation of the state-space model which underlies the univariate filtering and smoothing algorthms; see, e.g., Durbin and Koopman (2012, eq. 6.10–11). In order to deal with this consideration, one solution is to move the measurement errors from the measurement equation to the state equation. Alternatively, the measurement equation can be transformed such the the measurement errors become uncorrelated through the transformation. This latter case may be handled by a Schur decomposition of $R = S\Lambda S'$, where Λ is diagonal and holds the eigenvalues of R, while S is orthogonal, i.e., $S'S = I_n$.[49] By premultiplying the measurement equation by S' we get

$$y_t^* = A^{*'}x_t + H^{*'}\xi_t + w_t^*, \quad t = 1, \ldots, T,$$

where $y_t^* = S'y_t$, $A^* = AS$, $H^* = HS$, $w_t^* = S'w_t$. This means that $E[w_t^* w_t^{*'}] = \Lambda$, a diagonal matrix, while the state equation is unaffected by the transformation. Since r can be a large number for DSGE models, YADA always transforms the measurement equation when R is not diagonal, rather than augmenting the state vector.

Assuming that R is diagonal, the Kalman forecasting and updating equations can be determined from the following univariate filtering equations

$$\xi_{t,i+1} = \xi_{t,i} + K_{t,i}\Sigma_{t,i}^{-1}z_{t,i}, \tag{5.105}$$

$$P_{t,i+1} = P_{t,i} - K_{t,i}\Sigma_{t,i}^{-1}K_{t,i}', \tag{5.106}$$

for $i = 1, \ldots, n_t$, where $n_t \le n$ is equal to the actual number of observed variables at t (thereby allowing for missing observations), and

$$z_{t,i} = y_{t,i} - A_i'x_t - H_i'\xi_{t,i}, \tag{5.107}$$

$$\Sigma_{t,i} = H_i'P_{t,i}H_i + R_i, \tag{5.108}$$

$$K_{t,i} = P_{t,i}H_i. \tag{5.109}$$

Observed element i in y_t is denoted by $y_{t,i}$ and the corresponding columns of A and H are given by A_i and H_i, respectively, while R_i is the measurement error variance for $w_{t,i}$. The univariate transition equations are

$$\xi_{t+1,1} = F\xi_{t,n_t+1}, \tag{5.110}$$

$$P_{t+1,1} = FP_{t,n_t+1}F' + Q. \tag{5.111}$$

---

[49] Since R is a square matrix, the Schur decomposition is identical to the eigenvalue decomposition.

It follows that the forecasting and updating estimates for the state variables and the corresponding covariance matrices are

$$\xi_{t+1|t} = \xi_{t+1,1}, \quad \xi_{t|t} = \xi_{t,n_t+1}, \quad P_{t+1|t} = P_{t+1,1}, \quad P_{t|t} = P_{t,n_t+1}, \tag{5.112}$$

while the standard initialization is handled by $\xi_{1,1} = \mu_\xi$ and $P_{1,1} = \Sigma_\xi$.

Note that although the univariate Kalman filter requires looping over all $i = 1, \ldots, n_t$, it also avoids inverting $\Sigma_{y,t|t-1}$ and two matrix multiplications. For larger models, computational gains can therefore be quite important. Furthermore, the log-likelihood can also be determined without inverting $\Sigma_{y,t|t-1}$. Specifically, it can be shown that

$$\ln p(y_t | x_t, \mathcal{Y}_{t-1}; \theta) = -\frac{1}{2} \sum_{i=1}^{n_t} \left(\ln(2\pi) + \ln\left(\Sigma_{t,i}\right) + z_{t,i}^2 / \Sigma_{t,i} \right). \tag{5.113}$$

Recall that we are assuming a diagonal covariance matrix for the measurement errors. This assumption is not a restriction for the log-likelihood since the orthogonal transformation matrix S has determinant one, i.e., the value of the log-likelihood function is invariant to S.

Smoothed estimates of the unobserved variables can likewise be determined from univariate smoothing recursions. Let $r_{T,n_T} = 0$ and $N_{T,n_T} = 0$ intialize the univariate smoother while

$$r_{t,i-1} = H_i \Sigma_{t,i}^{-1} z_{t,i} + L_{t,i}' r_{t,i}, \tag{5.114}$$

$$N_{t,i-1} = H_i \Sigma_{t,i}^{-1} H_i' + L_{t,i}' N_{t,i} L_{t,i}, \tag{5.115}$$

where $L_{t,i} = I_r - K_{t,i} H_i' \Sigma_{t,i}^{-1}$, $i = n_t, \ldots, 1$, $t = T, \ldots 1$, and with transitions

$$r_{t-1,n_{t-1}} = F' r_{t,0}, \tag{5.116}$$

$$N_{t-1,n_{t-1}} = F' N_{t,0} F. \tag{5.117}$$

The smooth innovations and covariance matrices are given by

$$r_{t|T} = r_{t,0}, \quad N_{t|T} = N_{t,0}. \tag{5.118}$$

Smooth estimates of the state variables and their covariances satisfy equations (5.22) and (5.24), while smooth estimates of the measurement errors and the state shocks can be computed from the smooth innovations as in Section 5.6.

The above algorithm may also be used for computing update estimates of the state shocks (and measurement errors) with the $r_{t|t}$ innovation vector. Let $u_{t,n_t} = 0$ for $t = 1, \ldots, T$ and consider the recursion

$$u_{t,i-1} = H_i \Sigma_{t,i}^{-1} z_{t,i} + L_{t,i}' u_{t,i}, \quad i = n_t, \ldots, 1. \tag{5.119}$$

This procedure gives us

$$r_{t|t} = u_{t,0},$$

so that the update estimator of the state shocks and the measurement errors are

$$v_{t|t} = Q u_{t,0}, \quad w_{t|t} = R \left(H'H \right)^{-1} H' u_{t,0}, \quad t = 1, \ldots, T. \tag{5.120}$$

For the calculation of the measurement error in (5.120) it is important to note that the original H and R matrices are used, instead of those obtained when transforming the measurement equation such that the errors have a diagonal covariance matrix. Naturally, the update estimator of the measurement errors may also be computed directly from the measurement equation using the update estimator of the state variables.

5.15.2. *Univariate Filtering and Smoothing with Diffuse Initialization*

The diffuse initialization of the Kalman filter considered in Section 5.14 implies that the matrix $P_{t,i}$, the vector $K_{t,i}$ and the scalar $\Sigma_{t,i}$ can be decomposed as

$$P_{t,i} = P_{*,t,i} + c P_{\infty,t,i} + O(c^{-1}),$$

$$K_{t,i} = K_{*,t,i} + c K_{\infty,t,i} + O(c^{-1}), \tag{5.121}$$

$$\Sigma_{t,i} = \Sigma_{*,t,i} + c \Sigma_{\infty,t,i} + O(c^{-1}),$$

where we have used the following

$$\Sigma_{\infty,t,i} = H_i' P_{\infty,t,i} H_i, \quad \Sigma_{*,t,i} = H_i' P_{*,t,i} H_i + R_i,$$
$$K_{\infty,t,i} = P_{\infty,t,i} H_i, \qquad K_{*,t,i} = P_{*,t,i} H_i. \tag{5.122}$$

Notice that $\Sigma_{\infty,t,i} = 0$ implies that $K_{\infty,t,i} = 0$ since H_i has rank one.

To obtain the diffuse filtering recursions, the scalar $\Sigma_{t,i}^{-1}$ needs to be expanded as a power series in c^{-j}. This yields

$$\Sigma_{t,i}^{-1} = \begin{cases} c^{-1}\Sigma_{\infty,t,i}^{-1} - c^{-2}\Sigma_{*,t,i}\Sigma_{\infty,t,i}^{-2} + O(c^{-3}), & \text{if } \Sigma_{\infty,t,i} > 0, \\ \Sigma_{*,t,i}^{-1} & \text{otherwise.} \end{cases}$$

This is easily established by computing $(\Sigma_{*,t,i} + c\Sigma_{\infty,t,i})\Sigma_{t,i}^{-1} = 1$. Provided $\Sigma_{\infty,t,i} > 0$, equations (5.105) and (5.106) then gives us

$$\xi_{t,i+1} = \xi_{t,i} + K_{\infty,t,i}\Sigma_{\infty,t,i}^{-1} z_{t,i}, \tag{5.123}$$

$$P_{\infty,t,i+1} = P_{\infty,t,i} - K_{\infty,t,i} K_{\infty,t,i}' \Sigma_{\infty,t,i}^{-1}, \tag{5.124}$$

$$P_{*,t,i+1} = P_{*,t,i} + K_{\infty,t,i} K_{\infty,t,i}' \Sigma_{*,t,i}\Sigma_{\infty,t,i}^{-2} - \left(K_{*,t,i} K_{\infty,t,i}' + K_{\infty,t,i} K_{*,t,i}'\right)\Sigma_{\infty,t,i}^{-1}, \tag{5.125}$$

for $i = 1, \ldots, n_t$. When $\Sigma_{\infty,t,i} = 0$, the univariate filtering equations for the standard initialization apply, i.e.,

$$\xi_{t,i+1} = \xi_{t,i} + K_{*,t,i}\Sigma_{*,t,i}^{-1} z_{t,i}, \tag{5.126}$$

$$P_{\infty,t,i+1} = P_{\infty,t,i}, \tag{5.127}$$

$$P_{*,t,i+1} = P_{*,t,i} - K_{*,t,i} K_{*,t,i}' \Sigma_{*,t,i}^{-1}, \tag{5.128}$$

for $i = 1, \ldots, n_t$. Notice that $P_{*,t,i+1}$ plays the role of $P_{t,i+1}$ under standard initialization and that the filter under diffuse initialization is augmented with equation (5.127). The transition from t to $t + 1$ satisfies the following

$$\xi_{t+1,1} = F\xi_{t,n_t+1}, \tag{5.129}$$

$$P_{\infty,t+1,1} = F P_{\infty,t,n_t+1} F', \tag{5.130}$$

$$P_{*,t+1,1} = F P_{*,t,n_t+1} F' + Q. \tag{5.131}$$

As pointed out by Koopman and Durbin (2000), it is required that

$$\text{rank}\left[P_{\infty,t+1,1}\right] = \text{rank}\left[P_{\infty,t,n_t+1}\right],$$

and that this is not a restriction for a properly defined model, i.e., the rank of F does not influence the rank of $P_{\infty,t+1,1}$. Moreover, when $\Sigma_{\infty,t,i} > 0$ it holds that[50]

$$\text{rank}\left[P_{\infty,t,i+1}\right] = \text{rank}\left[P_{\infty,t,i}\right] - 1.$$

The diffuse recursions are continued until the matrix $P_{\infty,t,i+1}$ becomes zero at $t = d$ and $i = i^*$. From then on, the univariate Kalman filter in Section 5.15.1 is used with $P_{t,i+1} = P_{*,t,i+1}$.

Compared with the exact diffuse multivariate filtering approach in Section 5.14.1, we now have that the forecasting and updating estimates of the state variables and their covariance matrices are

$$\xi_{t+1|t} = \xi_{t+1,1}, \qquad \xi_{t|t} = \xi_{t,n_t+1}, \quad P_{\infty,t+1|t} = P_{\infty,t+1,1},$$
$$P_{\infty,t|t} = P_{\infty,t,n_t+1}, \quad P_{*,t+1|t} = P_{*,t+1,1}, \qquad P_{*,t|t} = P_{*,t,n_t+1}. \tag{5.132}$$

while initialization is handled by $\xi_{1,1} = \mu_\xi$, $P_{\infty,1,1} = P_\infty$, and $P_{*,1,1} = P_*$.

---

[50] See Ansley and Kohn (1985, 1990), Koopman (1997), or Section 5.14.1.

The log-likelihood function can be calculated without the need for computing some inverses of $\Sigma_{\infty,t|t-1}$ and $\Sigma_{*,t|t-1}$. We here have that

$$\ln p(y_t | x_t, \mathcal{Y}_{t-1}; \theta) = -\frac{1}{2} \sum_{i=1}^{n_t} \ln p_{t,i}, \tag{5.133}$$

where

$$\ln p_{t,i} = \begin{cases} \ln 2\pi + \ln \Sigma_{\infty,t,i} & \text{if } \Sigma_{\infty,t,i} > 0, \\ \ln 2\pi + \ln \Sigma_{*,t,i} + z_{t,i}^2 / \Sigma_{*,t,i} & \text{otherwise.} \end{cases}$$

Turning to smoothing, the diffuse univariate recursions apply to the same sample as the diffuse univariate filter, i.e., to the indices

$$(t, i) = (d, i^*), (d, i^* - 1), \ldots, (d, 1), \ldots, (1, 1).$$

Expanding $r_{t,i}$ and $N_{t,i}$ in equations (5.114) and (5.115), respectively, in terms of reciprocals of c in the same way as for $\Sigma_{t,i}^{-1}$ we obtain

$$r_{t,i} = r_{t,i}^{(0)} + c^{-1} r_{t,i}^{(1)} + O(c^{-2}),$$

$$N_{t,i} = N_{t,i}^{(0)} + c^{-1} N_{t,i}^{(1)} + c^{-2} N_{t,i}^{(2)} + O(c^{-3}),$$

with initialization $r_{d,i^*}^{(0)} = r_{d,i^*}$, $r_{d,i^*}^{(1)} = 0$, $N_{d,i^*}^{(0)} = N_{d,i^*}$, and $N_{d,i^*}^{(1)} = N_{d,i^*}^{(2)} = 0$. The initial values r_{d,i^*} and N_{d,i^*} are determined by equations (5.114) and (5.115), respectively. By defining

$$\tilde{r}_{t,i} = \begin{bmatrix} r_{t,i}^{(0)} \\ r_{t,i}^{(1)} \end{bmatrix}, \quad \tilde{N}_{t,i} = \begin{bmatrix} N_{t,i}^{(0)} & N_{t,i}^{(1)} \\ N_{t,i}^{(1)} & N_{t,i}^{(2)} \end{bmatrix},$$

it can be shown using equations (5.114) and (5.115) that for $\Sigma_{\infty,t,i} > 0$

$$\tilde{r}_{t,i-1} = \begin{bmatrix} 0 \\ H_i \Sigma_{\infty,t,i}^{-1} z_{t,i} \end{bmatrix} + \begin{bmatrix} L_{\infty,t,i} & L_{0,t,i} \\ 0 & L_{\infty,t,i} \end{bmatrix}' \tilde{r}_{t,i}, \quad i = n_t, \ldots, 1, \tag{5.134}$$

where

$$L_{\infty,t,i} = I_r - K_{\infty,t,i} H_i' \Sigma_{\infty,t,i}^{-1},$$

$$L_{0,t,i} = \left(K_{\infty,t,i} \Sigma_{*,t,i} \Sigma_{\infty,t,i}^{-1} - K_{*,t,i} \right) H_i' \Sigma_{\infty,t,i}^{-1}.$$

Furthermore,

$$\tilde{N}_{t,i-1} = \begin{bmatrix} 0 & H_i H_i' \Sigma_{\infty,t,i}^{-1} \\ H_i H_i' \Sigma_{\infty,t,i}^{-1} & H_i H_i' \Sigma_{*,t,i} \Sigma_{\infty,t,i}^{-2} \end{bmatrix} + \begin{bmatrix} L_{\infty,t,i} & L_{0,t,i} \\ 0 & L_{\infty,t,i} \end{bmatrix}' \tilde{N}_{t,i} \begin{bmatrix} L_{\infty,t,i} & L_{0,t,i} \\ 0 & L_{\infty,t,i} \end{bmatrix}, \tag{5.135}$$

for $i = n_t, \ldots, 1,$, with transitions

$$\tilde{r}_{t-1,n_{t-1}} = (I_2 \otimes F') \tilde{r}_{t,0}, \quad \tilde{N}_{t-1,n_{t-1}} = (I_2 \otimes F') \tilde{N}_{t,0} (I_2 \otimes F), \tag{5.136}$$

for $t = d, \ldots, 1$.

If $\Sigma_{\infty,t,i} = 0$, it can likewise be shown that

$$\tilde{r}_{t,i-1} = \begin{bmatrix} H_i \Sigma_{*,t,i}^{-1} z_{t,i} \\ 0 \end{bmatrix} + \begin{bmatrix} L_{*,t,i} & 0 \\ 0 & L_{*,t,i} \end{bmatrix}' \tilde{r}_{t,i}, \quad i = n_t, \ldots, 1, \tag{5.137}$$

where $L_{*,t,i} = I_r - K_{*,t,i} H_i' \Sigma_{*,t,i}^{-1}$. Moreover,

$$\tilde{N}_{t,i-1} = \begin{bmatrix} H_i H_i' \Sigma_{*,t,i}^{-1} & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} L_{*,t,i} & 0 \\ 0 & L_{*,t,i} \end{bmatrix}' \tilde{N}_{t,i} \begin{bmatrix} L_{*,t,i} & 0 \\ 0 & L_{*,t,i} \end{bmatrix}, \tag{5.138}$$

for $i = n_t, \ldots, 1$. The transition from t to $t - 1$ is again covered by equation (5.136).

For both these cases, the diffuse state smoothing equations are given by

$$\xi_{t|T} = \xi_{t,1} + \tilde{P}_{t,1}\tilde{r}_{t,0}, \tag{5.139}$$

$$P_{t|T} = P_{*,t,1} - \tilde{P}_{t,1}\tilde{N}_{t,0}\tilde{P}'_{t,1}, \tag{5.140}$$

where

$$\tilde{P}_{t,1} = \begin{bmatrix} P_{*,t,1} & P_{\infty,t,1} \end{bmatrix}.$$

Furthermore, the smooth state shocks are given by

$$v_{t|T} = Qr_{t,0}^{(0)}. \tag{5.141}$$

We can also determine update estimates of the state shocks using the above algorithm. Specifically, let $u_{t,n_t}^{(0)} = u_{t,n_t}^{(1)} = 0$ for all t and define the stacked vector

$$\tilde{u}_{t,i} = \begin{bmatrix} u_{t,i}^{(0)} \\ u_{t,i}^{(1)} \end{bmatrix}.$$

If $\Sigma_{\infty,t,i} > 0$ we let

$$\tilde{u}_{t,i-1} = \begin{bmatrix} 0 \\ H_i\Sigma_{\infty,t,i}^{-1}z_{t,i} \end{bmatrix} + \begin{bmatrix} L_{\infty,t,i} & L_{0,t,i} \\ 0 & L_{\infty,t,i} \end{bmatrix}' \tilde{u}_{t,i}, \tag{5.142}$$

and if $\Sigma_{\infty,t,i} = 0$ we have that

$$\tilde{u}_{t,i-1} = \begin{bmatrix} H_i\Sigma_{*,t,i}^{-1}z_{t,i} \\ 0 \end{bmatrix} + \begin{bmatrix} L_{*,t,i} & 0 \\ 0 & L_{*,t,i} \end{bmatrix}' \tilde{u}_{t,i}, \quad i = n_t, \dots, 1. \tag{5.143}$$

The update estimates of the state shocks are now given by equation (5.120) with $u_{t,0}$ being replaced with $u_{t,0}^{(0)}$. Update estimates of the measurement errors can likewise be determined by replacing $u_{t,0}$ with $u_{t,0}^{(0)}$ in this equation. As in the case of standard initialization for the univariate case, care has to be taken to ensure that the original H and R matrices are used, rather than those implied by the transformation S.

5.16. *Observation Weights for Unobserved Variables under Diffuse Initialization*

Like it was shown in Section 5.9 with an initialization based on a finite covariance matrix of the initial state, it is also possible to compute weights on the observed variables for unobserved variables under diffuse initialization. Below we shall first cover forecasted state variables, thereafter updated state variables, and finally the smooth estimates of the state variables and the measurement errors. Since the estimators of the unobserved variables when $t > d$ are equal under diffuse and finite initialization, the presentation below focuses on the initialization sample, i.e., $t = 1, \dots, d$.

5.16.1. *Weights for the Forecasted State Variables under Diffuse Initialization*

To determine the weights on the observed variables for the forecasted state variables under the assumption of diffuse initialization we need to take three cases into account. Namely, the covariance matrix $\Sigma_{\infty,t|t-1}$ has full rank n_t, it has rank zero, and it has rank less than n_t but greater than zero. The last possibility is, as in Section 5.14, handled through the univariate filter with diffuse initialization.

When $\Sigma_{\infty,t|t-1}$ has full rank n_t equation (5.84) for the state variable forecast can be rewritten as

$$\xi_{t+1|t} = L_{\infty,t}\xi_{t|t-1} + K_{\infty,t}z_t,$$

where $z_t = y_t - A'x_t$. Similarly, when $\Sigma_{\infty,t|t-1}$ has rank zero, equation (5.87) can be rearranged according to

$$\xi_{t+1|t} = L_{*,t}\xi_{t|t-1} + K_{*,t}z_t.$$

These two equations have the same general form as (5.43) in Section 5.9, thus suggesting how we may proceed to determine the observation weights.

The third case to examine is when $\Sigma_{\infty,t|t-1}$ has reduced rank but is nonzero. From the univariate filter in Section 5.15.2 it may be recalled that $\xi_{t+1|t} = \xi_{t+1,1} = F\xi_{t,n_t+1} = F\xi_{t|t}$, while

$$
\xi_{t,i+1} = \begin{cases} \xi_{t,i} + K_{\infty,t,i}\Sigma_{\infty,t,i}^{-1}z_{t,i}, & \text{if } \Sigma_{\infty,t,i} > 0, \\ \xi_{t,i} + K_{*,t,i}\Sigma_{*,t,i}^{-1}z_{t,i}, & \text{otherwise.} \end{cases}
$$

From equation (5.107) we know that $z_{t,i} = y_{t,i} - A_i'x_t - H_i'\xi_{t,i}$ and, hence, we may rewrite the above equation as

$$
\xi_{t,i+1} = L_{t,i}\xi_{t,i} + \kappa_{t,i}\tilde{z}_{t,i}, \tag{5.144}
$$

where $\tilde{z}_{t,i} = y_{t,i} - A_i'x_t$, and

$$
L_{t,i} = \begin{cases} L_{\infty,t,i}, & \text{if } \Sigma_{\infty,t,i} > 0, \\ L_{*,t,i}, & \text{otherwise,} \end{cases} \qquad \kappa_{t,i} = \begin{cases} K_{\infty,t,i}\Sigma_{\infty,t,i}^{-1}, & \text{if } \Sigma_{\infty,t,i} > 0, \\ K_{*,t,i}\Sigma_{*,t,i}^{-1}, & \text{otherwise.} \end{cases}
$$

From recursive substitution for $\xi_{t,i}$ in (5.144) it can be shown that

$$
\xi_{t,n_t+1} = \left(\prod_{i=1}^{n_t} L_{t,n_t+1-i}\right)\xi_{t,1} + \sum_{i=1}^{n_t}\left(\prod_{j=i+1}^{n_t} L_{t,n_t+i+1-j}\right)\kappa_{t,i}\tilde{z}_{t,i}. \tag{5.145}
$$

Notice that the order of multiplication of the $L_{t,i}$ matrices is important with L_{t,n_t} being the first, L_{t,n_t-1} next and so on.

Since the univariate filter is based on uncorrelated measurement errors, the transformation $R = S\Lambda S'$ discussed at the beginning of Section 5.15.1 must to be taken into account. Letting $\tilde{z}_t$ be the n_t-dimensional vector with typical element $\tilde{z}_{t,i}$, we know that $\tilde{z}_t = S'z_t$. This means that equation (5.145) can be rewritten as

$$
\xi_{t+1|t} = F\bar{L}_t\xi_{t|t-1} + F\bar{K}_t z_t, \tag{5.146}
$$

where

$$
\bar{L}_t = \prod_{i=1}^{n_t} L_{t,n_t+1-i}, \qquad \bar{K}_t = \left[\left(\prod_{j=2}^{n_t} L_{t,n_t+2-j}\right)\kappa_{t,1} \quad \cdots \quad L_{t,n_t}\kappa_{t,n_t-1} \quad \kappa_{t,n_t}\right]S'.
$$

Combining the results for the three cases concerning the rank of $\Sigma_{\infty,t|t-1}$ we find that

$$
\xi_{t+1|t} = \hat{L}_t\xi_{t|t-1} + \hat{K}_t z_t, \quad t = 1,\dots,d-1, \tag{5.147}
$$

where

$$
\hat{L}_t = \begin{cases} L_{\infty,t}, & \text{if } \mathrm{rank}\left(\Sigma_{\infty,t|t-1}\right) = n_t, \\ L_{*,t}, & \text{if } \mathrm{rank}\left(\Sigma_{\infty,t|t-1}\right) = 0, \\ F\bar{L}_t, & \text{otherwise,} \end{cases}
$$

while $\hat{K}_t$ is defined in a similar fashion from $K_{\infty,t}$, $K_{*,t}$, and $F\bar{K}_t$, respectively. It now follows that equation (5.44) is also valid under diffuse initialization, but where the weights need to be redefined for the initialization sample $t = 1,\dots,d-1$. Specifically, for this sample we have that the weights are determined according to the forward recursion

$$
\alpha_\tau\left(\xi_{t+1|t}\right) = \begin{cases} \hat{L}_t\alpha_\tau\left(\xi_{t|t-1}\right), & \text{if } \tau = 1,\dots,t-1, \\ \hat{K}_t, & \text{if } \tau = t. \end{cases}
$$

The weight on $\xi_{1|0}$ for the initialization sample is likewise given by

$$
\beta_0\left(\xi_{t+1|t}\right) = \prod_{i=0}^{t-1} \hat{L}_{t-i}.
$$

For $t = d, \ldots, T - 1$, the weights on the observed variables are determined as in Section 5.9 but with the caveat that the new weights for the initialization sample need to be used.

It should be emphasized that if the univariate filtering under diffuse initialization is applied, then the following matrices are redefined as follows

$$\hat{L}_t = F\bar{L}_t, \qquad \hat{K}_t = F\bar{K}_t,$$

for the initialization sample.

5.16.2. *Weights for the Updated State Variable Projections under Diffuse Initialization*

It is now straightforward to determine the weights on the observed variables over the initialization sample for the Kalman updater. Specifically, we now find that

$$\xi_{t|t} = \hat{M}_t \xi_{t|t-1} + \hat{N}_t z_t, \tag{5.148}$$

where

$$\hat{M}_t = \begin{cases} I_r - P_{\infty,t|t-1} H \Sigma_{\infty,t|t-1}^{-1} H', & \text{if rank} \left(\Sigma_{\infty,t|t-1} \right) = n_t, \\ I_r - P_{*,t|t-1} H \Sigma_{*,t|t-1}^{-1} H', & \text{if rank} \left(\Sigma_{\infty,t|t-1} \right) = 0, \\ \bar{L}_t, & \text{otherwise,} \end{cases}$$

and

$$\hat{N}_t = \begin{cases} P_{\infty,t|t-1} H \Sigma_{\infty,t|t-1}^{-1}, & \text{if rank} \left(\Sigma_{\infty,t|t-1} \right) = n_t, \\ P_{*,t|t-1} H \Sigma_{*,t|t-1}^{-1}, & \text{if rank} \left(\Sigma_{\infty,t|t-1} \right) = 0, \\ \bar{K}_t, & \text{otherwise.} \end{cases}$$

This means that equation (5.48) is also valid for the intialization sample $t = 1, \ldots, d$, except that the weights on the observed variables are now determined by

$$\alpha_\tau \left(\xi_{t|t} \right) = \begin{cases} \hat{M}_t \alpha_\tau \left(\xi_{t|t-1} \right), & \text{if } \tau = 1, \ldots, t - 1, \\ \hat{N}_t, & \text{if } \tau = t, \end{cases}$$

while the weights on the initial state for the intialization sample is

$$\beta_0 \left(\xi_{t|t} \right) = \hat{M}_t \beta_0 \left(\xi_{t|t-1} \right).$$

If univariate filtering has been used, then the following matrices are redefined as follows

$$\hat{M}_t = \bar{L}_t, \quad \hat{N}_t = \bar{K}_t,$$

for the initialization sample.

Finally, when initializing the computations of weights for the forecasted variables in period $t = d + 1$ we need to take the initial sample outcome into account. A simple way of achieving this is to base the initialization on $\xi_{d+1|d} = F\xi_{d|d}$. This means that the L_d and K_d matrices in equation (5.43) are replaced with $F\hat{M}_t$ and $F\hat{N}_t$, respectively.

5.16.3. *Weights for the Smoothed State Variable Projections under Diffuse Initialization*

The smooth estimates of the states variables for the initialization sample can be expressed as

$$\xi_{t|T} = \xi_{t|t-1} + \tilde{P}_{t|t-1} \tilde{r}_{t|T}, \quad t = 1, \ldots, d,$$

where $\tilde{P}_{t|t-1} = [P_{*,t|t-1} \ P_{\infty,t|t-1}]$ and $\tilde{r}_{t|T}$ is the extended innovation vector in (5.101) or in (5.139). To determine observation weights for the smoothed state variables it is therefore convenient to first compute the observation weights for the innovation vector $\tilde{r}_{t|T}$.

In the event that $\Sigma_{\infty,t|t-1}$ has rank n_t or zero is can be shown that

$$\tilde{r}_{t|T} = \tilde{H}\tilde{\Sigma}_{t|t-1} J_n z_t - \tilde{H}\tilde{\Sigma}_{t|t-1} J_n H' \xi_{t|t-1} + \tilde{L}'_t \tilde{r}_{t+1|T}, \tag{5.149}$$

where $J_n = [I_n \ 0]'$ is $2n \times n$ and the remaining the matrices are defined in Section 5.14.2.

For the case when the rank of $\Sigma_{\infty,t|t-1}$ is greater than zero but less than n_t we examine the univariate smoother in more detail. From equation (5.134) and (5.137) we have that

$$\tilde{r}_{t,i-1} = A_{t,i}z_{t,i} + B_{t,i}\tilde{r}_{t,i}, \quad i = n_t, \ldots, 1, \tag{5.150}$$

where

$$A_{t,i} = \begin{cases} \begin{bmatrix} 0 \\ H_i\Sigma_{\infty,t,i}^{-1} \end{bmatrix}, & \text{if } \Sigma_{\infty,t,i} > 0 \\[2ex] \begin{bmatrix} H_i\Sigma_{*,t,i}^{-1} \\ 0 \end{bmatrix}, & \text{otherwise,} \end{cases}$$

while

$$B_{t,i} = \begin{cases} \begin{bmatrix} L_{\infty,t,i} & L_{0,t,i} \\ 0 & L_{\infty,t,i} \end{bmatrix}', & \text{if } \Sigma_{\infty,t,i} > 0 \\[3ex] \begin{bmatrix} L_{*,t,i} & 0 \\ 0 & L_{*,t,i} \end{bmatrix}', & \text{otherwise.} \end{cases}$$

By recursive substitution for $\tilde{r}_{t,i}$ in equation (5.150), making use of the transition equation (5.136) for the innovations, and the relation $\tilde{r}_{t,0} = \tilde{r}_{t|T}$ it can be shown that

$$\tilde{r}_{t|T} = \sum_{i=1}^{n_t} \left(\prod_{j=1}^{i-1} B_{t,j} \right) A_{t,i}z_{t,i} + \left(\prod_{i=1}^{n_t} B_{t,i} \right) \left(I_2 \otimes F'\right) \tilde{r}_{t+1|T}. \tag{5.151}$$

To obtain an equation similar to (5.149) we need to find an expression for $z_{t,i}$ in terms of z_t and $\xi_{t|t-1}$. By making use of equation (5.144) along with recursive substitution and recalling that $\xi_{t,1} = \xi_{t|t-1}$ some algebra leads to

$$\xi_{t,i} = \left(\prod_{j=1}^{i-1} L_{t,i-j} \right) \xi_{t|t-1} + \sum_{k=1}^{i-1} \left(\prod_{j=k+1}^{i-1} L_{t,i+k-j} \right) \kappa_{t,k}\tilde{z}_{t,k}, \quad i = 1, \ldots, n_t. \tag{5.152}$$

Next, by substituting the right hand side of (5.152) into equation (5.107) for $\xi_{t,i}$ we obtain

$$z_{t,i} = \tilde{z}_{t,i} - H_i' \sum_{k=1}^{i-1} \left(\prod_{j=k+1}^{i-1} L_{t,i+k-j} \right) \kappa_{t,k}\tilde{z}_{t,k} - H_i' \left(\prod_{j=1}^{i-1} L_{t,i-j} \right) \xi_{t|t-1}, \quad i = 1, \ldots, n_t. \tag{5.153}$$

The first two terms on the right hand side of (5.153) can be further simplified to $C_{t,i}S'z_t$, where $C_{t,i}$ is the $1 \times n_t$ vector given by

$$C_{t,i} = \begin{bmatrix} -H_i' \left(\prod_{j=2}^{i-1} L_{t,i+1-j} \right) \kappa_{t,1} & \cdots & -H_i'L_{t,i-1}\kappa_{t,i-2} & -H_i'\kappa_{t,i-1} & 1 & 0 & \cdots & 0 \end{bmatrix},$$

for $i = 1, \ldots, n_t$. It follows that (5.151) can be rewritten as

$$\tilde{r}_{t|T} = \left[\sum_{i=1}^{n_t} \left(\prod_{j=1}^{i-1} B_{t,j} \right) A_{t,i}C_{t,i} \right] S'z_t - \sum_{i=1}^{n_t} \left(\prod_{j=1}^{i-1} B_{t,j} \right) A_{t,i}H_i' \left(\prod_{j=1}^{i-1} L_{t,i-j} \right) \xi_{t|t-1}$$
$$+ \left(\prod_{i=1}^{n_t} B_{t,i} \right) \left(I_2 \otimes F'\right) \tilde{r}_{t+1|T},$$

or more compactly

$$\tilde{r}_{t|T} = \bar{F}_tz_t - \bar{G}_t\xi_{t|t-1} + \bar{H}_t\tilde{r}_{t+1|T}. \tag{5.154}$$

Combining the results for the rank of $\Sigma_{\infty,t|t-1}$ being n_t, zero, or between these numbers in equation (5.149) and (5.154) we find that

$$\tilde{r}_{t|T} = \hat{F}_tz_t - \hat{G}_t\xi_{t|t-1} + \hat{H}_t\tilde{r}_{t+1|T}, \tag{5.155}$$

where

$$\hat{F}_t = \begin{cases} \tilde{H}\tilde{\Sigma}_{t|t-1}J_n, & \text{if rank}\left(\Sigma_{\infty,t|t-1}\right) \in \{0, n_t\}, \\ \bar{F}_t & \text{otherwise,} \end{cases}$$

$$\hat{G}_t = \begin{cases} \tilde{H}\tilde{\Sigma}_{t|t-1}J_nH', & \text{if rank}\left(\Sigma_{\infty,t|t-1}\right) \in \{0, n_t\}, \\ \bar{G}_t & \text{otherwise,} \end{cases}$$

$$\hat{H}_t = \begin{cases} \check{L}'_t & \text{if rank}\left(\Sigma_{\infty,t|t-1}\right) \in \{0, n_t\}, \\ \bar{H}_t & \text{otherwise.} \end{cases}$$

If univariate filtering and smoothing has been applied, then the following equalities hold: $\hat{F}_t = \bar{F}_t$, $\hat{G}_t = \bar{G}_t$, and $\hat{H}_t = \bar{H}_t$.

The observation weights for the innovations $\tilde{r}_{t|T}$ over the initialization sample ($t = 1, \ldots, d$) can now be established from (5.155). Specifically,

$$\alpha_\tau\left(\tilde{r}_{t|T}\right) = \begin{cases} \hat{H}_t\alpha_\tau\left(\tilde{r}_{t+1|T}\right) - \hat{G}_t\alpha_\tau\left(\xi_{t|t-1}\right), & \text{if } \tau = 1, \ldots, t-1, \\ \hat{F}_t + \hat{H}_t\alpha_\tau\left(\tilde{r}_{t+1|T}\right), & \text{if } \tau = t, \\ \hat{H}_t\alpha_\tau\left(\tilde{r}_{t+1|T}\right), & \text{if } \tau = t+1, \ldots, T. \end{cases}$$

The weights in the $2r \times n_t$ matrix $\alpha_\tau(\tilde{r}_{t|T})$ are initialized at $t = d + 1$ by

$$\alpha_\tau\left(\tilde{r}_{d+1|T}\right) = \begin{bmatrix} \alpha_\tau\left(r_{d+1|T}\right) \\ 0 \end{bmatrix}.$$

Furthermore, the weights on the initial state for $t = 1, \ldots, d$ are

$$\beta_0\left(\tilde{r}_{t|T}\right) = \hat{H}_t\beta_0\left(\tilde{r}_{t+1|T}\right) - \hat{G}_t\beta_0\left(\xi_{t|t-1}\right), \qquad \beta_0\left(\tilde{r}_{d+1|T}\right) = \begin{bmatrix} \beta_0\left(r_{d+1|T}\right) \\ 0 \end{bmatrix}.$$

Given the observation weights for the innovations $\tilde{r}_{t|T}$, $t = 1, \ldots, d$, it is straightforward to determine the observation weights for the smooth estimates of the state variables. Specifically, equation (5.53) remains valid for the full sample, but where the weights for the initialization sample are now given by

$$\alpha_\tau\left(\xi_{t|T}\right) = \begin{cases} \alpha_\tau\left(\xi_{t|t-1}\right) + \tilde{P}_{t|t-1}\alpha_\tau\left(\tilde{r}_{t|T}\right), & \text{if } \tau = 1, \ldots, t-1, \\ \tilde{P}_{t|t-1}\alpha_\tau\left(\tilde{r}_{t|T}\right), & \text{if } \tau = t, \ldots, T. \end{cases}$$

The weights for the initial state is similarly equal to

$$\beta_0\left(\xi_{t|T}\right) = \beta_0\left(\xi_{t|t-1}\right) + \tilde{P}_{t|t-1}\beta_0\left(\tilde{r}_{t|T}\right).$$

Since the measurement errors can be directly estimated using either updated or smoothed estimates of the state variables, the observation weights under diffuse initialization can be determined from the same relations. That is, the relationships in equations (5.55) and (5.56) are still valid for the smooth estimates of the measurement errors. When calculating weights for the update estimates of the measurement errors, the weights for the smooth state variables in (5.56) are replaced with the weights for the update estimates of the state variables, recalling that $\alpha_\tau(\xi_{t|t}) = 0$ for $\tau \geq t + 1$.

5.17. *YADA Code*

5.17.1. `KalmanFilter(Ht)`

The function `KalmanFilter` in YADA computes the value of the log-likelihood function in (5.18) for a given set of parameter values. It requires a $n \times T$ matrix $Y = [y_1 \cdots y_T]$ with the observed

variables, a $k \times T$ matrix $X = [x_1 \cdots x_T]$ with exogenous variables, and parameter matrices A, H, F, Q, and R. The F and Q matrices are constructed based on the output from the solution to the DSGE model, while the A, H, and R matrices are specified in a user-defined function that determines the measurement equation; cf. Section 18.4. Moreover, the vector with initial state values $\xi_{1|0}$ is needed. This input is denoted by KsiInit and is by default the zero vector.

Furthermore, KalmanFilter requires input on the variable initP. If this variable is 1, then the function calculates an initial value for the matrix $P_{1|0}$ as described in equation (5.15). If this variable is set to 2, then the doubling algorithm is used to calculate an approximation of Σ_ξ (see DoublingAlgorithmLyapunov below). Next, the input variables MaxIter and Tolerance are accepted and are used by the doubling algorithm function. The input variable StartPeriod is used to start the sample at period $t_m \geq 1$. The default value of this parameter is 1, i.e., not to skip any observations. Moreover, the boolean variable AllowUnitRoot is needed to determine if undefined unit roots are accepted in the state equation or not. Finally, if initP is 3, then $P_{1|0} = cI_r$, where $c > 0$ needs to be specified; its default value is 100.

The function KalmanFilterHt takes exactly the same input variables as KalmanFilter. While the input variable H is $r \times n$ for the latter function, it is now $r \times n \times T$ for the former function. This means that KalmanFilterHt allows for a time-varying measurement matrix.

As output, KalmanFilter (KalmanFilterHt) provides lnL, the value of the log-likelihood function in (5.18), where the summation is taken from t_m until T. Furthermore, output is optionally provided for $y_{t|t-1}$, $H'P_{t|t-1}H + R$ (or $H'_tP_{t|t-1}H_t + R$ when the measurement matrix is time-varying), $\xi_{t|t-1}$, $P_{t|t-1}$, $\ln p(y_t|x_t, \mathcal{Y}_{t-1}; \theta)$ from t_m until T, etc. The dimensions of the outputs are:

> lnL: scalar containing the value of the log-likelihood function in (5.18).
> status: indicator variable being 0 if all the eigenvalues of F are inside the unit circle, and 1 otherwise. In the latter case, KalmanFilter (KalmanFilterHt) sets initP to 3. In addition, this varaible takes the value -1 in the event that the value of the log-likelihood function is not a real number. The latter can happen if the forecast covariance matrix of the observed variables, $H'P_{t|t-1}H + R$, is not positive definite for some time period t.
> lnLt: $1 \times (T - t_m + 1)$ vector $[\ln p(y_{t_m}|x_{t_m}, \mathcal{Y}_{t_m-1}; \theta) \cdots \ln p(y_T|x_T, \mathcal{Y}_{T-1}; \theta)]$. [Optional]
> Yhat: $n \times (T - t_m + 1)$ matrix $[y_{t_m|t_m-1} \cdots y_{T|T-1}]$. [Optional]
> MSEY: $n \times n \times (T - t_m + 1)$ 3 dimensional matrix where MSEY$(:, :, t - t_m + 1) = H'P_{t|t-1}H + R$. [Optional]
> Ksitt1: $r \times (T - t_m + 1)$ matrix $[\xi_{t_m|t_m-1} \cdots \xi_{T|T-1}]$. [Optional]
> Ptt1: $r \times r \times (T - t_m + 1)$ 3 dimensional matrix where Ptt1$(:, :, t - t_m + 1) = P_{t|t-1}$. [Optional]

The inputs are given by Y, X, A, H, F, Q, R, KsiInit, initP, MaxIter, Tolerance, StartPeriod, and c. All inputs are required by the function. The integer MaxIter is the maximum number of iterations that the doubling algorithm can use when initP is 2. In this case, the parameter Tolerance, i.e., the tolerance value for the algorithm, is also used.

5.17.2. UnitRootKalmanFilter(Ht)

The function UnitRootKalmanFilter (UnitRootKalmanFilterHt) takes all the input variables that KalmanFilter (KalmanFilterHt) accepts. In addition, this unit-root consistent version of the Kalman filter needs to know the location of the stationary state variables. This input vector is given by StationaryPos. Using this information the function sets up an initial value for the rows and columns of $P_{1|0}$ using the algorithm determined through initP. If this integer is 1 or 2, then the rows and columns of F and Q determined by StationaryPos are used. The remaining entries of the $P_{1|0}$ are set to zero if off-diagonal and to c if diagonal.

The output variables from UnitRootKalmanFilter (UnitRootKalmanFilterHt) are identical to those from KalmanFilter (KalmanFilterHt).

5.17.3. ChandrasekharRecursions

The function `ChandrasekharRecursions` takes the same input variables as `KalmanFilter`, except for the last two variables `AllowUnitRoot` and `c` in `KalmanFilter`. These inputs are not needed by the Chandrasekhar recursion since they require that the state covariance matrix is initialized with Σ_ξ, the unconditional state covariance matrix.

The output variables from the function are the same as those from the `KalmanFilter` function.

5.17.4. StateSmoother(Ht)

The function `StateSmoother` (`StateSmootherHt`) computes $\xi_{t|t}$, $\xi_{t|T}$, $P_{t|t}$, $P_{t|T}$, and $\xi_{t-1|t}$ using y_t, $y_{t|t-1}$, $\xi_{t|t-1}$ and $P_{t|t-1}$ as well as the parameter matrices H, R, F, and B_0 as input. The dimensions of the outputs are:

    `Ksitt`: $r \times (T - t_m + 1)$ matrix $[\xi_{t_m|t_m} \cdots \xi_{T|T}]$.
    `Ptt`: $r \times r \times (T - t_m + 1)$ 3 dimensional matrix where `Ptt`$(:,:,t - t_m + 1) = P_{t|t}$.
    `KsitT`: $r \times (T - t_m + 1)$ matrix $[\xi_{t_m|T} \cdots \xi_{T|T}]$.
    `PtT`: $r \times r \times (T - t_m + 1)$ 3 dimensional matrix where `PtT`$(:,:,t - t_m + 1) = P_{t|T}$.
    `Ksit1t`: $r \times (T - t_m + 1)$ matrix with the 1-step smoothed projections $\xi_{t-1|t}$.
    `rtvec`: $r \times 1$ vector with the smoothed innovation vector $r_{t_m|T}$. [Optional]
    `NtMat`: $r \times r \times (T - t_m + 1)$ matrix with the smoothed innovation covariances $N_{t|T}$. [Optional]

The required inputs are given by Y, Yhat, Ksitt1, Ptt1, H, F, R, and B0. For the `StateSmoother` version, the H matrix has dimension $r \times n$, while for `StateSmootherHt` is has dimension $r \times n \times T$.

5.17.5. SquareRootKalmanFilter(Ht)

The functions `SquareRootKalmanFilter` and `SquareRootKalmanFilterHt` compute the value of the log-likelihood function using the square root filter rather than the standard filter; cf. `KalmanFilter` and `KalmanFilterHt`. The functions take the same input variables as the standard filter functions except that Q is replaced with B0.

The required output variables are `lnL` and `status`. The optional variables are the same as for the standard Kalman filter functions. In addition, the functions can compute output variables `Yerror` ($n \times T - t_m + 1$ matrix with the 1-step ahead forecast errors of the observed variables), `SigmaSqRoot` ($n \times n \times T - t_m + 1$ matrix with the square root of the 1-step ahead forecast error covariance matrix of the observed variables), `InvMSEY` ($n \times n \times T - t_m + 1$ matrix with the inverse of $\Sigma_{y,t|t-1}$), and `KalmanGain` ($r \times n \times T - t_m + 1$ matrix with the Kalman gain matrix based on the square root calculations).

5.17.6. UnitRootSquareRootKalmanFilter(Ht)

The unit root consistent function `UnitRootSquareRootKalmanFilter` and calculates the value of the log-likelihood function using the square root filter rather than the standard filter; cf. `UnitRootKalmanFilter`. The function take the same input variables as the standard filter functions except that Q is replaced with B0.

The output variables are exactly the same as the function `SquareRootKalmanFilter` provides.

The functions with `Ht` appended are mirror images exacpt that they allow for a time-varying H matrix on the state variables in the measurement equation.

5.17.7. SquareRootSmoother(Ht)

The square root smoother are computed with the aid of the function `SquareRootSmoother` for the case of a constant H matrix and by `SquareRootSmootherHt` for a time-varying H matrix. The input variables are slightly different than those accepted by the standard smoother functions, `StateSmoother` and `StateSmootherHt`. In particular, the square root smoother functions accept the input variables: `Yerror`, `SigmaSqRoot`, `InvMSEY`, `KalmanGain`, `Ksitt1`, `Ptt1`, `H`, `F`, and `B0`. The first 4 variables are output variables from the square root filter functions discussed above, while the remaining input variables are shared with the standard smoother functions.

The output variables are the same as those given by `StateSmoother` and `StateSmootherHt`.

5.17.8. UnivariateKalmanFilter(Ht)

The functions `UnivariateKalmanFilter` and `UnivariateKalmanFilterHt` compute the value of the log-likelihood function using the univariate filter rather than the standard filter; cf. `KalmanFilter` and `KalmanFilterHt`. The functions take the same input variables as the standard filter functions.

The required output variables are `lnL` and `status`. The optional variables are the same as for the standard Kalman filter functions. In addition, the functions can compute output variables `sigma2i` (cell array of dimension $1 \times T$ where the cells contain the $1 \times n$ vector with scalars $\Sigma_{t,i}$ for $i = 1, \ldots, n$), `Kti` (cell array of dimension $1 \times T$ where the cells contains the $r \times n$ matrix whose columns are given by $K_{t,i}$ for $i = 1, \ldots, n$), `zti` (cell array of dimension $1 \times T$ where the cells contain the $1 \times n$ vector with scalars $z_{t,i}$ for $i = 1, \ldots, n$), and `Hti` (cell array of dimension $1 \times T$ where the cells contain the $r \times n$ matrix H, whose columns are given by H_i for $i = 1, \ldots, n$). The variables are presented in Section 5.15.1.

5.17.9. UnitRootUnivariateKalmanFilter(Ht)

The function `UnitRootUnivariateKalmanFilter` (`UnitRootUnivariateKalmanFilterHt`) is responsible for computing the value of the log-likelihood function using the univariate filter rather than the standard filter; see, for instance, `UnitRootKalmanFilter`. The function takes the same input variables as the standard filter functions.

The required output variables are `lnL` and `status`. The optional variables are the same as for the univariate Kalman filter function `UnivariateKalmanFilter`.

5.17.10. UnivariateStateSmoother

The univariate Kalman smoother is computed with the function `UnivariateStateSmoother`. The input 11 variables are given by `zti`, `sigma2i`, `Hti`, `Kti`, `Ksitt1`, `Ptt1`, `Yerror`, `H`, `F`, `R`, and `B0`. The first 4 input variables are given by the same named output variables of the univariate Kalman filter. The final 7 input variables are all used by the other smoother functions.

The output variables are the same as those given by the standard and square root smoother functions.

5.17.11. KalmanFilterMO(Ht)

The standard Kalman filtering subject to possibly missing observations is handled by the function `KalmanFilterMO` for the case of a constant H matrix and by `KalmanFilterMOHt` for a time-varying matrix. The input and output variables are identical to those for the standard Kalman filter functions.

5.17.12. UnitRootKalmanFilterMO(Ht)

The standard Kalman filtering allowing for unit roots and possibly subject to missing observations is handled by `UnitRootKalmanFilterMO` for the case of a constant H matrix and by `UnitRootKalmanFilterMOHt` for a time-varying matrix. The input and output variables are identical to those for the standard Kalman filter functions that allow for unit roots.

5.17.13. SquareRootKalmanFilterMO(Ht)

The square root Kalman filtering subject to possibly missing observations is handled by the function `SquareRootKalmanFilterMO` for the case of a constant H matrix and for a time-varying H matrix by `SquareRootKalmanFilterMOHt`. The input and output variables are identical to those for the square root Kalman filter functions.

5.17.14. `UnitRootSquareRootKalmanFilterMO(Ht)`

The square root Kalman filtering allowing for unit roots and possibly subject to missing observations is handled by `UnitRootSquareRootKalmanFilterMO` for the case of a constant H matrix and for a time-varying H matrix by `UnitRootSquareRootKalmanFilterMOHt`. The input and output variables are identical to those for the square root Kalman filter functions which allow for unit roots.

5.17.15. `UnivariateKalmanFilterMO(Ht)`

The univariate Kalman filtering subject to possibly missing observations is handled by the function `UnivariateKalmanFilterMO` for the case of a constant H matrix and for a time-varying H matrix by `UnivariateKalmanFilterMOHt`. The input and output variables are identical to those for the univariate Kalman filter functions.

5.17.16. `UnitRootUnivariateKalmanFilterMO(Ht)`

The univariate Kalman filtering allowing for unit roots and possibly subject to missing observations is handled by `UnitRootUnivariateKalmanFilterMO` for the case of a constant H matrix and for a time-varying H matrix by `UnitRootUnivariateKalmanFilterMOHt`. The input and output variables are identical to those for the univariate Kalman filter functions which allow for unit roots.

5.17.17. `StateSmootherMO(Ht)`

The standard Kalman smoothing subject to possibly missing observations is handled by the function `StateSmootherMO` when the H matrix in the measurement equation is constant and by `StateSmootherMOHt` when it is time-varying. The input and output variables are exactly the same as those provided by the standard Kalman smoothing functions.

5.17.18. `SquareRootSmootherMO(Ht)`

The square root Kalman smoothing subject to possibly missing observations is handled by `SquareRootSmootherMO` when the H matrix in the measurement equation is constant and by `SquareRootSmootherMOHt` when it is time-varying. The input variables are exactly the same as those needed by the square root Kalman smoothing functions without missing observations. The output variables are extended with two optional variables. Namely, the $r \times T$ matrices `rtt` and `rtT` with estimates of the update and smooth innovations, $r_{t|t}$ and $r_{t|T}$, respectively.

5.17.19. `UnivariateStateSmootherMO`

The univariate Kalman smoothing calculations subject to possibly missing observations is handled by the function `UnivariateStateSmootherMO` for the cases of a constant or a time-varying H matrix in the measurement equation. The input variables are exactly the same as those needed by the function `UnivariateStateSmoother`. The output variables are extended with two optional variables. Namely, the $r \times T$ matrices `rtt` and `rtT` with estimates of the update and smooth innovations, $r_{t|t}$ and $r_{t|T}$, respectively.

5.17.20. `DiffuseKalmanFilter(MO)(Ht)`

The functions `DiffuseKalmanFilter(MO)(Ht)` computes the standard Kalman filter with diffuse initialization, where functions with the addition `MO` to the name handle possible missing observations, and functions with the addition `Ht` cover a time-varying H-matrix in the measurement equation. The 11 input variables are: `Y`, `X`, `A`, `H`, `F`, `Q`, `R`, `KsiLast`, `StartPeriod`, `AllowUnitRoot`, and `StationaryPos` which are all shared with other functions for the standard Kalman filter.

The functions provides 2 required and 7 optional output variables: `lnL`, `status`, `lnLt`, `Yhat`, `MSEY`, `Ksitt1`, `Ptt1`, `SigmaRank`, and `SmoothData`. Only the last two are unique to functions with diffuse initialization. The variable `SigmaRank` is a matrix with at most 2 rows and d columns. The first row holds the rank of $\Sigma_{\infty,t|t-1}$ over the initialization sample, while the second row (when it exsts) holds the number of observed variables for the same sample. The last

output variable, `SmoothData`, is a structure with fields containing data needed for computing the smooth and update estimates over the initialization sample.

5.17.21. `DiffuseSquareRootKalmanFilter(MO)(Ht)`

The functions `DiffuseSquareRootKalmanFilter(MO)(Ht)` computes the square-root Kalman filter with diffuse initialization. The function shares its 11 input variables with the function for the standard Kalman filter with diffuse initialization, except that `Q` is replaced with `B0`.

The 9 output variables are shared with `DiffuseKalmanFilter(MO)(Ht)`.

5.17.22. `DiffuseUnivariateKalmanFilter(MO)(Ht)`

The functions `DiffuseUnivariateKalmanFilter(MO)(Ht)` computes the univariate Kalman filter with diffuse initialization. The function shares its 11 input variables with the function for the standard Kalman filter with diffuse initialization.

The 9 output variables are shared with `DiffuseKalmanFilter(MO)(Ht)`.

5.17.23. `DiffuseStateSmoother(MO)(Ht)`

The functions `DiffuseStateSmoother(MO)(Ht)` computes the smooth and update estimates of the state variables based on the standard Kalman smoother with diffuse initialization. The function requires 10 input variables: `Y`, `Yhat`, `Ksitt1`, `Ptt1`, `H`, `F`, `R`, `B0`, `SigmaRank`, and `SmoothData`. The first 8 inputs are shared with `StateSmoother(MO)(Ht)`, while the last two are outputs from the Kalman filter routine subject to diffuse initialization.

The functions provides at least 5 and at most 9 output variables: `Ksitt`, `Ptt`, `KsitT`, `PtT`, `Ksit1t`, `rtvec`, `NtMat`, `rtt`, and `rtT`. The first 7 are also obtained from the standard Kalman smoother. The last two optional output variables are only provided by the smoothers that support missing observations. For these functions, the `rtt` variable is a matrix of dimension $r \times T$ with the update estimates of the disturbances $r_{t|t}$, while the `rtT` variable is a matrix of the same dimension with the smooth estimates of the distrurbances $r_{t|T}$.

5.17.24. `DiffuseSquareRootSmoother(MO)(Ht)`

The functions `DiffuseSquareRootSmoother(MO)(Ht)` computes the smooth and update estimates of the state variables based on the square-root Kalman smoother with diffuse initialization. The input and output variables are shared with the standard Kalman smoother functions `DiffuseStateSmoother(MO)(Ht)`.

5.17.25. `DiffuseUnivariateStateSmoother(MO)`

The functions `DiffuseUnivariateStateSmoother(MO)` computes the smooth and update estimates of the state variables based on the univariate Kalman smoother with diffuse initialization. The input and output variables are shared with the standard Kalman smoother functions `DiffuseStateSmoother(MO)(Ht)`.

5.17.26. `DoublingAlgorithmLyapunov`

The function `DoublingAlgorithmLyapunov` computes Σ_ξ, the unconditional covariance matrix of the state vector ξ, using S ($m \times m$), W ($m \times m$ and positive semidefinite) as inputs as well as the positive integer `MaxIter`, reflecting the maximum number of iterations to perform, and the positive real `ConvValue`, measuring the value when the convergence criterion is satisfied. The convergence criterion is simply the largest singular value of $|\gamma_{k+1} - \gamma_k|$; see Matlab's `norm` function. The dimensions of the outputs are:

   `M:` $m \times m$ positive semidefinite matrix.
   `status:` Boolean variable which is 0 if the algorithm converged and 1 otherwise.

When called from `KalmanFilter`, the first two inputs are given by `F` and `Q`, while the maximum number of iterations and the tolerance value for the function can be determined by the user.[51]

5.17.27. `DSGEObservationWeightsTheta` & `DSGEObservationWeightsThetaDiffuse`

The functions `DSGEObservationWeightsTheta` and `DSGEObservationWeightsThetaDiffuse` are used to provide the observation weights and decompositions of the state variables and the structural shocks for fixed parameter values based on the standard and the diffuse initialization of the Kalman filter, respectively. To fulfill their purpose, the functions require six input variables: `theta`, `thetaPositions`, `ModelParameters`, `VarType`, `DSGEModel`, and `CurrINI`. Among these inputs only the integer `VarType` variable needs to be mentioned. It takes the value 1 if state variable decompositions should be prepared, 2 if structural shock decompositions are needed, 3 if state variable weights are sought, and 4 if structural shock weights are requested by YADA.

The only required output variable is the structure `StateDecomp`, with available fields that depend on the input variable `VarType`. The two optional output variables are given by `status` and `kalmanstatus`.

---

[51] The settings tab in YADA contains options for selecting the doubling algorithm rather than the vectorized solution technique, and for selecting the maximum number of iterations and the tolerance level for the algorithm. The default values are 100 and `1.0e-8`.

6. PARAMETER TRANSFORMATIONS

If some of the parameters in θ have a gamma, inverted gamma, left truncated normal, or Pareto prior distribution, then the support for these parameters is bounded from below. Similarly, if some of the θ parameters have a beta or uniform prior distribution, then the support is bounded from below and above. Rather than maximizing the log posterior of θ subject to these bounds on the support, it is common practise to transform the parameters of θ such that the support of the transformed parameters is unbounded. Before turning the attention to posterior mode estimation, the discussion will first consider the parameter transformation that YADA can apply.

6.1. *Transformation Functions for the Original Parameters*

For the p_1 parameters with a gamma, inverted gamma, left truncated normal, or Pareto prior distribution, denoted by θ_1, the transformation function that is typically applied is the natural logarithm

$$\phi_{i,1} = \ln(\theta_{i,1} - c_{i,1}), \quad i = 1, \ldots, p_1, \tag{6.1}$$

where $c_{i,1}$ is the lower bound. Please note that YADA sets the lower bound of the Pareto distribution to $c + b$, where c is the origin parameter and b the location parameter for $y = c + z$ with z having lower bound b; cf. equation (4.34).

Letting θ_2 denote the p_2 parameters of θ with a beta or uniform prior distribution, the transformation function is the generalized logit

$$\phi_{i,2} = \ln\left(\frac{\theta_{i,2} - a_i}{b_i - \theta_{i,2}}\right), \quad i = 1, \ldots, p_2, \tag{6.2}$$

where $b_i > a_i$ gives the upper and the lower bounds.[52] The remaining p_0 parameters are given by $\phi_0 = \theta_0$, while $\phi = [\phi_0' \; \phi_1' \; \phi_2']'$ and $\theta = [\theta_0' \; \theta_1' \; \theta_2']'$. The overall transformation of θ into ϕ may be expressed as $\phi = g(\theta)$, where $g(\theta)$ is a vector of monotonic functions.[53]

We can likewise define a transformation from ϕ back into θ by inverting the above relations. That is,

$$\theta_{i,1} = \exp(\phi_{i,1}) + c_{i,1}, \quad i = 1, \ldots, p_1, \tag{6.3}$$

and

$$\theta_{i,2} = \frac{a_i + b_i \exp(\phi_{i,2})}{1 + \exp(\phi_{i,2})}, \quad i = 1, \ldots, p_2, \tag{6.4}$$

while $\theta_0 = \phi_0$. The full transformation can be expressed as $\theta = g^{-1}(\phi)$.

6.2. *The Jacobian Matrix*

When the ϕ parameters are used for evaluating the posterior distribution, it should be noted that the log-likelihood function is invariant to the transformation, i.e., $p(Y|\theta) = p(Y|g^{-1}(\phi)) = p(Y|\phi)$. Next, the value of the joint prior density $p(\phi)$ can be determined in the usual way by using the fact that $\phi = g(\theta)$; recall Section 4.2.1. That is, we need to take the Jacobian in the transformation into account.

Since the individial parameters are assumed to be independent, the prior density of θ is equal to the product of the marginal prior densities for each individual θ parameter. Moreover, since the matrix with partial derivatives of θ with respect to ϕ is diagonal and equal to the inverse of the matrix with partial derivatives of ϕ with respect to θ, it follows that the individual ϕ parameters are also a priori independent, that the individual Jacobians are given by

$$\left|\frac{1}{g'\left(g^{-1}(\phi_{i,j})\right)}\right| = \left|\frac{d\theta_{i,j}}{d\phi_{i,j}}\right|, \quad i = 1, \ldots, p_j \text{ and } j = 0, 1, 2,$$

and that $p(\phi)$ is equal to the product of the individual ϕ prior densities.

---

[52] We may think of this transformation as a generalized logit since $a_i = 0$ and $b_i = 1$ implies the logit function.

[53] There is no need to order the parameters according to their prior distribution. This is only done here for clarifying reasons. YADA knows from reading the prior distribution input which parameters have a beta, a gamma, etc, prior distribution.

For the θ parameters that have a gamma, inverted gamma, left truncated normal, or Pareto distribution, the log of the Jacobian is simply

$$\ln\left(\frac{d\theta_{i,1}}{d\phi_{i,1}}\right) = \phi_{i,1}, \quad i = 1, \ldots, p_1. \tag{6.5}$$

For the θ parameters with a beta or a uniform prior, the log of the Jacobian is

$$\ln\left(\frac{d\theta_{i,2}}{d\phi_{i,2}}\right) = \ln\left(b_i - a_i\right) + \phi_{i,2} - 2\ln\left(1 + \exp(\phi_{i,2})\right), \quad i = 1, \ldots, p_2. \tag{6.6}$$

Finally, for the θ parameters with a normal, Student-t (and Cauchy), logistic, or Gumbel prior the log of the Jacobian is zero since $d\theta_{i,0}/d\phi_{i,0} = 1$.

Notice that the Jacobians are positive for all parameter transformations and, hence, each Jacobian is equal to its absolute value. The sum of the log Jacobians in equations (6.5) and (6.6) should now be added to the log prior of θ, evaluated at $\theta = g^{-1}(\phi)$, to obtain the value of the log prior of ϕ; cf. equation (4.3) in Section 4.2.1.

6.3. *YADA Code*

YADA has four functions that handle the parameter transformations discussed above. The $\phi = g(\theta)$ mapping is handled by ThetaToPhi, the $\theta = g^{-1}(\phi)$ mapping by PhiToTheta, while logJacobian takes care of calculating the log of the Jacobian. In addition, there is a function (PartialThetaPartialPhi) that computes the matrix with partial derivatives of θ with respect to ϕ.

6.3.1. ThetaToPhi

The function ThetaToPhi calculates the mapping $\phi = g(\theta)$. It requires the inputs theta, the θ vector; thetaIndex, a vector with the same length as θ with unit entries for all parameters that have a gamma, an inverted gamma, a left truncated normal, or a Pareto prior distribution, with zero entries for all parameters with a normal, a Student-t, a Cauchy, a logistic, or a Gumbel prior, with 2 for the beta prior, and 3 for the uniform prior; UniformBounds a matrix with lower and upper bounds of any uniformly and beta distributed parameters (for all other parameters the elements are 0 and 1); and LowerBound, a vector of the same length as θ with the lower bound parameters $c_{i,1}$; see also Section 7.4 regarding the function VerifyPriorData. The output is given by phi.

6.3.2. PhiToTheta

The function PhiToTheta calculates the mapping $\theta = g^{-1}(\phi)$. It requires the inputs phi, the ϕ vector; thetaIndex; UniformBounds; and LowerBound. The output is given by theta.

6.3.3. logJacobian

The function logJacobian calculates the sum of the log of the Jacobian for the mapping $\theta = g^{-1}(\phi)$. It requires the inputs phi, the ϕ vector; thetaIndex; and UniformBounds. The output is given by lnjac.

6.3.4. PartialThetaPartialPhi

The function PartialThetaPartialPhi calculates the partial derivatives of θ with respect to ϕ. The required input is phi, thetaIndex, and UniformBounds. The output is given by the diagonal matrix ThetaPhiPartial. This function is used when approximating the inverse Hessian at the posterior mode of θ with the inverse Hessian at the posterior mode of ϕ.

7. Computing the Posterior Mode

The estimation of the posterior mode is either performed using the transformed parameters ϕ or the original parameters θ. Letting m be their common dimension, the posterior mode of ϕ can be expressed as:

$$\tilde{\phi} = \arg \max_{\phi \in \mathbb{R}^m} \left(\ln L\left(Y; g^{-1}(\phi)\right) + \ln p\left(g^{-1}(\phi)\right) + \ln J(\phi) + \ln p\left(\Phi_\omega | g^{-1}(\phi), h\right) \right). \quad (7.1)$$

The matrix Y represents the observed data, $L(\cdot)$ is the likelihood function, $\theta = g^{-1}(\phi)$, while $J(\phi)$ is the determinant of the (diagonal) Jacobian; cf. Section 4.2.1. The posterior estimate of θ is then given by $\bar{\theta} = g^{-1}(\tilde{\phi})$.

Similarly, the posterior mode of θ is given by

$$\tilde{\theta} = \arg \max_{\theta \in \Theta} \left(\ln L\left(Y; \theta\right) + \ln p\left(\theta\right) + \ln p\left(\Phi_\omega | \theta, h\right) \right), \quad (7.2)$$

where $\Theta \subset \mathbb{R}^m$. In Section 7.1 we shall discuss when the posterior estimate $\bar{\theta}$ is close to the posterior mode $\tilde{\theta}$.

In case the user has provided a system prior file, this file determines $\ln p(\Phi_\omega | \theta, h)$; see Section 4.4. If the prior does not include a system prior part, but simply the marginal prior $p(\theta)$, then $\ln p(\Phi_\omega | \theta, h) = 0$ for all θ and therefore drops out of the optimization problem.

The actual optimization of the log posterior of ϕ or θ is performed numerically in YADA. The user can choose between Christopher Sims' `csminwel` routine, Marco Ratto's `newrat`, Dynare's `gmhmaxlik`, and Matlab's `fminunc` (provided that the Optimization Toolbox is installed and YADA-related diff-files have been taken into account) and whether the transformed parameters (ϕ) or the original parameters (θ) should be targetted. All these optimization routines provide an estimate of the posterior mode of the targetted parameters and of the inverse Hessian at the mode, denoted by $\tilde{\Sigma}$.[54] The inverse Hessian at the mode is one candidate for the covariance matrix of the proposal density that the random walk Metropolis algorithm discussed in Section 8.1 needs for generating candidate draws from the posterior distribution of ϕ and of θ.

Note that the YADA specific version of `fminunc` is not supplied with the public version of YADA. Moreover, the original Matlab version of `fminunc` is not supported by the posterior mode estimation routine in YADA since it uses an edited version of the function (named `YADAfminuncx`, where x should be replaced with 5 or 7). The YADA specific version has some additional output fields and also supports a progress dialog. To make it possible for users that have Matlab's *Optimization Toolbox* installed to use `fminunc` for posterior mode estimation in YADA, diff-files are available for download from the YADA website. In addition, instructions on how to make use of the diff-files are provided in the YADA help file.[55]

7.1. Comparing the Posterior Modes

The posterior estimate $\bar{\theta}$ based on the posterior mode of ϕ in (7.1) is *approximately* equal to the posterior mode of θ in (7.2) provided that either the data is informative about the parameters or the log Jacobian is constant; for the transformations in Section 6 the Jacobian is constant when $\theta = \phi$. While the latter case is obvious, the former has a large sample explanation. That is, once the likelihood dominates in the posterior over the prior for *all* elements of θ (and ϕ), the posterior estimate $\bar{\theta}$ is almost equal to the mode of $p(\theta|Y)$ for finite T, with equality holding asymptotically.

To show that $\bar{\theta}$ need not be the mode of $p(\theta|Y)$, consider for simplicity a one parameter problem where $\phi = \ln(\theta)$, while $\theta \sim G(a, b)$, with $a > 1$. For this case we know that the prior mode of θ is $\tilde{\theta} = b(a - 1)$; cf. Section 4.2.3. This corresponds to the prior estimate $\bar{\phi} = \ln(b(a - 1))$. We may then ask if $\bar{\phi}$ is also the mode of $p(\phi)$.

---

[54] The Matlab function `fminunc` actually produces an estimate of the Hessian at the mode.

[55] The YADA website is located at: https://www.texlips.net/yada/. The YADA help file can also be read there.

The prior density of ϕ can be found using the result in Section 4.2.1. This means that

$$p(\phi) = \frac{1}{\Gamma(a)b^a} \exp(\phi)^a \exp\left(-\frac{\exp(\phi)}{b}\right).$$

This density resembles an extreme value distribution where, for instance, $a = 1$ yields a Gumbel distribution (for $\varphi = -\phi$) with location parameter $\mu = 0$ and scale parameter $\sigma = b$; see Section 4.2.11. It is straightforward to show that the mode of the density is $\tilde{\phi} = \ln(ab)$. This translates to the prior estimate $\bar{\theta} = ab$ and, hence, the mode of $p(\theta)$ is not equal to the value of $\theta = \exp(\phi)$ when ϕ is evaluated at the mode of $p(\phi)$.

Furthermore, we know that the posterior distribution of θ is equal to its prior distribution when the data is not informative about the parameter θ. Similarly, the posterior distribution of ϕ is also equal to its prior in this case and since the distributions have different modes, it follows that the posterior distributions do as well.

One implication of this is that it may be useful to perform the optimum check discussed in Section 7.2 also for the original parameters. Should the posterior estimate $\bar{\theta} = g^{-1}(\tilde{\phi})$ be close to the posterior mode for θ for all parameters, it suggests that the likelihood may be dominating and, hence, that the data may be informative about all the parameters.[56] On the other hand, if the posterior mode of some element of θ appears to be far away from its posterior estimate using $\bar{\theta}$, then the data is unlikely to be informative about this parameter. Hence, not only can the plots of the log posteriors be useful when tuning a proposal density, but it may also be informative about identification issues.

7.2. Checking the Optimum

In order to check if the value $\tilde{\phi}$ is a local optimum, YADA makes use of some tools suggested and originally coded by Mattias Villani. For each element ϕ_i of the vector ϕ a suitable grid with d elements is constructed from the lower and upper bounds $(\tilde{\phi}_i - c\tilde{\Sigma}_{i,i}^{1/2}, \tilde{\phi}_i + c\tilde{\Sigma}_{i,i}^{1/2})$, where $c > 0$ and $\tilde{\Sigma}_{i,i}$ is element (i,i) of $\tilde{\Sigma}$, the inverse Hessian of the log posterior at the mode. Let ϕ_{-i} be a vector with all elements of ϕ except element i. For each ϕ_i in the grid, the log posterior is evaluated at $(\phi_i, \tilde{\phi}_{-i})$. For parameter ϕ_i this provides us with d values of the log posterior of ϕ_i conditional on $\tilde{\phi}_{-i}$.

One proposal density that YADA can use for posterior sampling is $N(\phi, \tilde{\Sigma})$, where the value of ϕ is determined from the previous draw from the posterior. Since the computed values of the log posterior of ϕ_i are conditional on all parameters being equal to their values at the posterior mode, it is natural to compare them to a conditional proposal density for ϕ_i. For the grid values of ϕ_i that were used to calculate the conditional log posterior values of the parameter, one such density is the log of the normal density with mean $\tilde{\phi}_i$ and variance $\tilde{\Omega}_{i|-i} = \tilde{\Sigma}_{i,i} - \tilde{\Sigma}_{i,-i}\tilde{\Sigma}_{-i,-i}^{-1}\tilde{\Sigma}_{-i,i}$. The vector $\tilde{\Sigma}_{i,-i}$ is equal to the i:th row of $\tilde{\Sigma}$ with element i removed. Similarly, the matrix $\tilde{\Sigma}_{-i,-i}$ is obtained by removing row and column i from $\tilde{\Sigma}$.

We can also estimate the variance of a conditional proposal density by running a regression of the log posterior values on a constant, ϕ_i, and ϕ_i^2. The estimated variance is now given by the inverse of the absolute value of two times the coefficient on ϕ_i^2. A modified proposal density can now be evaluated for each ϕ_i by using the log of the normal density with mean $\tilde{\phi}_i$ and variance given by the estimate in question.[57]

---

[56] Note that this does not imply that the data is informative. It is possible that the prior mode of θ is close to the mode of the prior for ϕ. In the one parameter example, large values for b imply that $\bar{\theta}$ is close to $\tilde{\theta}$.

[57] Such an estimated conditional variance can also be transformed into a marginal variance if, e.g., we are willing to use the correlation structure from the inverse Hessian at the posterior mode. Let $C = \Sigma \ominus \varsigma\varsigma'$, where $\ominus$ denotes element-by-element division, and ς is the square root of the diagonal of Σ. Let $\Omega_{i|-i}$ be the conditional variance, while $\Sigma_{i,i}$ is the marginal variance. For a normal distribution we know that $\Omega_{i|-i} = \Sigma_{i,i} - \Sigma_{i,-i}\Sigma_{-i,-i}^{-1}\Sigma_{-i,i}$. This relationship can also be expressed through the correlation structure as $\Omega_{i|-i} = (1 - C_{i,-i}C_{-i,-i}^{-1}C_{-i,i})\Sigma_{i,i}$. Hence, if we have an estimate of the conditional variance $\Omega_{i|-i}$ and the correlation structure C, we can compute the marginal variance $\Sigma_{i,i}$ by inverting this expression.

FIGURE 3: Plot of the conditional log posterior density around the estimated posterior mode along with two conditional proposal densities (left) and the log-likelihood (right) for the parameters τ, ρ_G, and σ_Z.
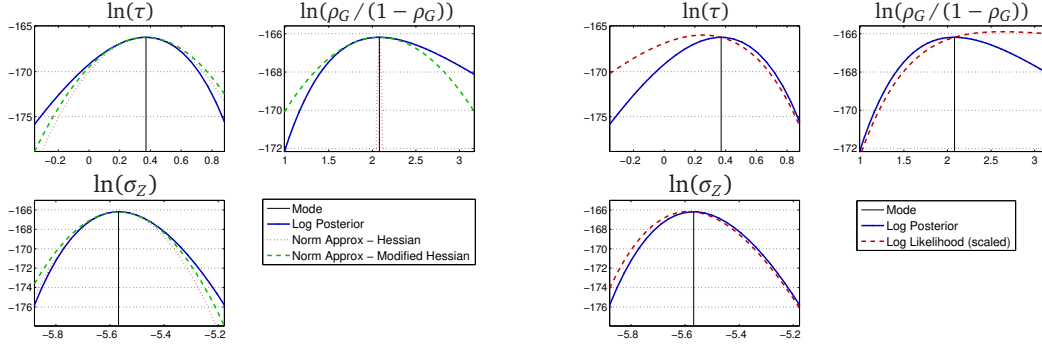
Using these ideas, the 3 plots to the left in Figure 3 provides graphs of the conditional log posterior density (blue solid line) of the parameters τ, ρ_G, and σ_Z from the An and Schorfheide model in Section 18. The transformed (ϕ) space for the parameters is used here, i.e., the log of τ and σ_Z and the logit of ρ_G. Since the support for ϕ is the real line, it is seems a priori more likely that a normal distribution can serve well as a proposal density for ϕ than for θ, where, for instance, ρ_G is restricted to be the 0-1 interval; cf. Section 6. The red dotted line shows the normal approximation of the log posterior using the posterior mode as mean and conditional variance based on the inverse Hessian at the mode. The green dashed line is similarly based on the normal approximation with the same mean, but with the conditional variance estimated as discussed in the previous paragraph.

It is worth noticing from the Figure that the normal approximations based on the inverse Hessian and on the modification are close to the log posterior for all these parameters except for ρ_G. In the case of ρ_G, however, these differences seem to be particularly severe and indicates that if the proposal density has the inverse Hessian, $\tilde{\Sigma}$, as its covariance matrix, then it may take a long time before the support of ρ_G is sufficiently covered. By comparison, the proposal density based on the modification lies closer to the log posterior and is therefore a better approximation.

The posterior mode checking facilities in YADA also produces a second set of plots. These graphs operate over the same grid as those discussed above, but instead of studying proposal densities they plot the log-likelihood against the log posterior over the grid for each parameter. To avoid scaling problems in the graphs, YADA adds the value of the log prior at the mode to the log-likelihood. One important feature of these plots is that potential identification issues can be detected from the slope of the log-likelihood. Moreover, they give an idea of how far away a local maximum for the log-likelihood is relative to the local maximum of the posterior for each parameter.

The 3 plots to the right in Figure 3 provides graphs of the conditional log posterior density (blue solid line) along with the log-likelihood (red dashed line) for the same 3 parameters. As mentioned above, the log-likelihood has been scaled such that the value of the log prior at the mode has been added to each value. For the τ parameter it can be seen that the local maximum of the log-likelihood gives a value for τ is somewhat smaller than the posterior mode value, while the local maximum of the log-likelihood for σ_Z is very close to the posterior mode. It is also noteworthy that the log-likelihood for ρ_G is increasing for values of this parameter that are greater than the posterior mode value, but that the slope is rather flat. This suggests that at least locally this parameter is not well identified. This is also supported by the result that the prior mode of ρ_G ($11/13 \approx 0.8462$) is very close to the posterior estimate (0.8890).

7.3. *A Monte Carlo Based Optimization Routine*

There are situations when it may be difficult to locate a suitable posterior mode through standard numerical optimization routines. The posterior distributions samplers discussed in, e.g., Section 8 do not necessarily need an estimate of the posterior mode and the inverse Hessian at the mode in order to be applied. What is needed is a suitable covariance matrix for the jumping distribution. At the same time, YADA requires that some estimate of the posterior mode exists in order to run these posterior samplers.

Since at least version 4, Dynare includes a Monte Carlo based routine that attempts to locate a parameter point with a high posterior probability through simulation. Based on a simple selection procedure, similar to the random walk Metropolis routine considered in Section 8.1, the Monte Carlo optimization procedure requires only a suitable proposal density. Specifically, let this density be given by

$$\varphi^{(s)} \sim N_m(\phi^{(s-1)}, c^2 \Sigma_\phi), \quad s = 1, 2, \ldots, S, \tag{7.3}$$

where S is the maximum number of draws to use, m is the dimension of the parameter vector, c is a positive scale factor, and Σ_ϕ is a positive definite matrix. The latter matrix may, e.g., be diagonal with the prior variances of ϕ in the diagonal. Should the variances not exists, then the corresponding element may be replaced with a large constant. Furthermore, the initial value $\phi^{(0)}$ may be taken from the prior distribution information in YADA.

The vector $\phi^{(s)}$ can now be updated as in Section 8.1,[58] while the posterior mean and posterior covariance matrix are updated according to

$$\mu^{(s)} = \mu^{(s-1)} + (1/s) \left[\phi^{(s)} - \mu^{(s-1)} \right],$$
$$\Sigma^{(s)} = \Sigma^{(s-1)} + \mu^{(s-1)} \mu^{(s-1)\prime} - \mu^{(s)} \mu^{(s)\prime} \tag{7.4}$$
$$+ (1/s) \left[\phi^{(s)} \phi^{(s)\prime} - \Sigma^{(s-1)} - \mu^{(s-1)} \mu^{(s-1)\prime} \right],$$

and the posterior mode from

$$\bar{\phi}^{(s)} = \begin{cases} \phi^{(s)} & \text{if } p(\phi^{(s)}|Y) > p(\bar{\phi}^{(s-1)}|Y), \\ \bar{\phi}^{(s-1)} & \text{otherwise.} \end{cases} \tag{7.5}$$

The Dynare function `gmhmaxlik` also includes a portion of code that attempts to tune the scale factor c before it simulates the posterior mean, mode, and covariance matrix. During this tuning step, the initial value for ϕ (or θ) also changes as draws are accepted by Monte Carlo procedure. Moreover, once it has estimated the posterior covariance matrix via (7.4), the dynare code optionally uses this estimate to first re-tune the scale factor c and thereafter to make a final attempt of climbing the posterior hill to its peak, i.e.,the mode of the posterior distribution. The tuning of the scale parameter is for this optional case, as well as for the initial tuning case, based on a suitable target for the acceptance rate of the simulation procedure; YADA works with a targeted acceptance rate of 1/4, while the original Dynare function has a target rate of 1/3.

7.4. *YADA Code*

The posterior mode is computed in YADA by the function `PosteriorModeEstimation`. The main inputs for this function are the structures `DSGEModel`, `CurrINI`, and `controls`. The first contains paths to the user files that specify the log-linearized DSGE model, the prior distribution of its parameters, the data, the measurement equations, and any parameter functions that should be dealt with. It also contains information about options for the Kalman filter, the sample to use, names of observed variables, of exogenous variables, of state variables, and names of the state shocks, your choice of optimization routine and parameters to target (transformed or original),

---

[58] For the transformed parameters this means that the rule in equation (8.1) is used. For the original parameters the ratio on the right hand side of the selection rule are replaced with the log posteriors for this parameterization.

the tolerance value, the maximum number of iterations to consider, as well as some other useful features.

The `CurrINI` structure contains data on initialization information needed by YADA. This structure contains non-model related information, while the `DSGEModel` structure contains the model related information. Finally, the `controls` structure holds handles to all the controls on the main GUI of YADA.

7.4.1. VerifyPriorData

Based on the input that `PosteriorModeEstimation` receives, the first task it performs is to check the data in the prior distribution file. This is handled by the function `VerifyPriorData`. Given that the prior distribution data is complete (cf. Section 18.2), this function returns the prior distribution data in various variables. These variables are given by `theta`, `thetaIndex`, `thetaDist`, `LowerBound`, `ModelParameters`, `thetaPositions`, `PriorDist`, `ParameterNames`, and `UniformBounds`.

The vector `theta` contains the initial values of the parameters to be estimated, i.e., θ. The vectors `thetaIndex` and `thetaDist` have the same length as `theta` with integer entries indicating the type of prior distribution that is assumed for each element of `theta`. The difference between these two vectors is that `thetaIndex` indicates the type of transformation that should be applied to obtain ϕ, while `thetaDist` gives the prior distribution. The vector `LowerBound` gives the lower bound specified for the parameters in the prior distribution file. This bound is, for example, used by the parameter transformation function discussed in Section 6.3.

The structure `ModelParameters` has fields given by the parameter names assigned in the prior distribution file; see, e.g., Table 2. Each field is assigned a value equal to the initial value for that parameter. Both estimated and calibrated parameters are given a field in the `ModelParameters` structure. Similarly, the vector structure `thetaPositions` has dimension given by m, the dimension of θ. Each element in the vector structure has a field `parameter` that contains a string with the name of the parameter. The vector structure is constructed such that `thetaPositions(i).parameter` gives the name of the parameter in position i of θ.

The structure `PriorDist` has 11 fields: `beta`, `gamma`, `normal`, `invgamma`, `truncnormal`, `cauchy`, `student`, `uniform`, `logistic`, `gumbel`, and `pareto`. Each such field contains a matrix whose number of rows depends on the number of parameters assigned a given prior distribution. The number of columns is 2 for the normal, uniform, gamma, inverted gamma, Cauchy and Gumbel, while it is 3 for the left truncated normal, the Student-t, the logistic, and the Pareto; the third column holds the lower bound for the left truncated normal, the number of degrees of freedom for the Student-t, and the origin for the Pareto. For the beta distribution, finally, the matrix has 4 columns, where the last two hold the lower and upper bounds. Columns 1 and 2 have the values of prior parameter 1 and 2 that were given in the prior distribution file. The `PosteriorModeEstimation` function later creates 4 new fields in the `PriorDist` structure. These fields are `beta_ab`, `gamma_ab`, `logistic_ab`, and `gumbel_ab`, containing the (a, b) parameters for the beta (eq. (4.15)) and the gamma (eq. (4.7)) distributions, the location, scale and shape parameters (μ, σ, c) for the logistic distribution, and the location and scale parameters for the Gumbel distribution (μ, σ). The functions `MomentToParamStdbetaPDF`, `MomentToParamGammaPDF`, `MomentToParamLogisticPDF` and `MomentToParamGumbelPDF`, respectively, deal with the transformations from mean and standard deviation (and shape parameter c for the logistic) to the needed parameters; cf. Section 4.5.

The structure `ParameterNames` has fields `all`, `calibrated`, `beta`, `gamma`, `normal`, `invgamma`, `truncnormal`, `uniform`, `student`, `cauchy`, `logistic`, `gumbel`, `pareto`, and `estimated`. Each field returns a string matrix with the parameter names. One field to this structure, `additional`, is added by the function `ReadAdditionalParameterNames`. This term holds a string matrix with the names of all the new parameters defined in the file with parameters to update; cf. Section 18.3. Moreover, it extends the string matrix `ParameterNames.calibrated` with any new calibrated parameters that YADA has found in the file with parameters to initialize.

Finally, the matrix `UniformBounds` has dimension $m \times 2$. For all prior distributions but the uniform and the beta each row has 0 and 1. For the uniform and the beta the rows have the lower and the upper bounds.

7.4.2. `logPosteriorPhiDSGE`

Since `csminwel`, `newrat`, and `fminunc` are minimization routines, the function to minimize when transformed parameters are targetted by the optimizer is given by minus the expression within parenthesis on the right hand side of (7.1); this function is called `logPosteriorPhiDSGE` in YADA. Before attempting to minimize this function, YADA allows the user to import alternative initial values for the estimated parameters as well as an initial value for the inverse Hessian. This possibility is supported by the `csminwel` and `newrat` optimizers and may also be skipped.

The files with parameter values to import are standard matlab mat-files and are required to hold certain variables and it is therefore recommended that YADA has created them. One case when this is useful is when optimization has been cancelled by the user who has selected to save the obtained results to disk. These results can then be imported at a later time as initial values and used for optimization. As mentioned above, this possibility requires that either csminwel or newrat are used as optimizers. Another case is during model development and where, say, posterior mode results are available from a previous model version. This file can then be utilized for initial values when optimizing. YADA will compare the parameter names of the selected model to those being imported and the values of the common parameters as well as the inverse Hessian entries from these parameters can then be used.

Once this has been concluded YADA runs a number of checks on the user defined functions. First of all, all user defined Matlab functions are copied to the `tmp` directory to make sure they are visible to Matlab. Second, YADA attempts to run the user defined Matlab functions. The first group contains any functions with additional parameters that the user has included in the model setup; cf. Section 18.3. Given that such functions exist, the order in which they are executed depends on the user input on the DSGE Data tab on the YADA GUI; see Figure 9. If both types of additional parameter files exist, the order is determined by the data in the checkbox "Run file with parameters to initialize before file with parameters to update". The execution of additional parameters updates the `ModelParameters` structure with fields and values.

Given that these files are executed without errors (or that they do not exist), the following step is to check the validity of the measurement equation function; cf. Section 18.4. Assuming this function takes the necessary input and provides the necessary output (without errors), YADA then tries to solve the DSGE model; see Section 3.5. If the model has a unique convergent solution at the initial values (see Section 3.1), YADA proceeds with the final preparations for running the optimization routine. If not, YADA returns an error message, reporting which problem AiM discovered.

The final preparations first involves collecting additional parameters into the `ParameterNames` structure in the field `additional`. Next, the initial values of the additional parameters as well as the initial values of the calibrated parameters are located and stored in the vectors `thetaAdditional` and `thetaCalibrated`, respectively. These two tasks are handled by the functions `ReadAdditionalParameterNames` and `ReadAdditionalParameterValues`. Next, the actual sample to use is determined by running the function `CreateSubSample`. This sub-sample does not take into account that the user may have selected a value for the `StartPeriod` variable different from $t_m = 1$; see Section 5.17. The choice of `StartPeriod` is determined by the choice of "First observation after Kalman filter training sample" on the Settings tab of the YADA dialog.

The last task before the chosen minimization routine is called is to check if the function that calculates the log posterior returns a valid value at the initial parameter values. The `logPosteriorPhiDSGE` function takes 10 input arguments. The first is `phi`, the transformed parameters ϕ. Next, 6 inputs originally created by `VerifyPriorData` are required. They are: `thetaIndex`, `UniformBounds`, `LowerBound`, `thetaPositions`, `thetaDist`, and the structure `PriorDist`. Furthermore, the structure with model parameters `ModelParameters`, the DSGE model structure `DSGEModel`, and the `AIMData` structure are needed. The latter structure is created when the DSGE model is parsed through the `AiMInitialize` function. That function saves

to a mat-file the outputs from the `compute_aim_data` function. When this mat-file is loaded into YADA it creates the `AIMData` structure with fields having names equal to the output variables of `compute_aim_data`. Finally, the variable `OrderQZ` is needed by the Klein (2000) and Sims (2002) DSGE model solvers. Recall that this is a boolean variable which is unity if `ordqz` is a built-in Matlab function (true for version 7 and later) and zero otherwise. Furthermore, the choice of DSGE model solver is determined by the setting in the DSGE Model Data frame on the DSGE Data tab; see Figure 9.

Based on this input the log posterior evaluation function `logPosteriorPhiDSGE` first transforms ϕ into θ by calling `PhiToTheta`. Next, it makes sure that `ModelParameters` is correctly updated for the parameters that are estimated. This is achieved through the function `ThetaToModelParameters`. Apart from `ModelParameters` it needs `theta` and `thetaPositions` to fulfill its task. With `ModelParameters` being updated for the estimated parameters, any remaining parameters are reevaluated next, i.e., the user defined function with parameters to update.

With the parameter point determined, the log-likelihood function is examined and evaluated if the parameter point implies a unique convergent solution for the DSGE model and the state vector is stationary (the largest eigenvalue of F is inside the unit circle). The function `logLikelihoodDSGE` deals with the calculation. The inputs for the function are the three structures `ModelParameters`, `DSGEModel`, and `AIMData`. The function returns three variables: `logLikeValue`, the value of the log-likelihood; `mcode`, indicating if the DSGE model has a unique convergent solution or not; and `status`, a boolean variable, indicating if the F matrix in the state equation (5.2) has all eigenvalues inside the unit circle or not. Given that `mcode` is 1 and that `status` is 0, the value of the log-likelihood is considered valid. If either of these variables returns a different value, the function returns 1000000, otherwise it proceeds with the evaluation of the log of the prior density at $\theta = g^{-1}(\phi)$ through `logPriorDSGE` and, if the prior density value is not a `NaN`, the log of the Jacobian. The latter function is given by `logJacobian`, presented in Section 6.3. If the log prior density value is `NaN`, then `logPosteriorPhiDSGE` again returns 1000000. Otherwise, the function checks if the user has a system prior by examining the `DSGEModel` field `SystemPriorFile`. If such an additional prior exists, it computes the log height of this user written density function and checks if the value is valid or not. Should the value be `NaN`, then `logPosteriorPhiDSGE` once again returns 1000000, otherwise it returns minus the sum of the log-likelihood, the log prior density, the log Jacobian, and the log system prior density. The latter value is always 0 when the model does *not* have a system prior.

In addition to the mandatory `logPost` output variable, the `logPosteriorPhiDSGE` function also supports the optional `logLike` output variable. It is equal to `NaN` when `logPost` returns 1000000, and to `logLikeValue` when all computations could be carried out successfully.

7.4.3. `logPosteriorThetaDSGE`

The function `logPosteriorThetaDSGE` works in a very similar way as `logPosteriorPhiDSGE`. One difference is, of coursem that it takes the input vector `theta` with the original parameters instead of `phi`. Another difference is that the log posterior for the original parameters takes the input variable `ParameterBounds` instead of `UniformBounds`. This matrix has two columns with the lower and upper bounds for the parameters in `theta`. Finally, the output variable `logPost` is equal to the sum of the log likelikhood, the log prior of `theta`, and the log value of any system prior (default is 0). In all other respects, the two log posterior functions are equal.

7.4.4. `logLikelihoodDSGE`

The function `logLikelihoodDSGE` directs the main tasks when using the DSGE model for computing the log-likelihood with a Kalman filter; see, e.g., Section 5.4. The inputs are, as already mentioned, the three structures `ModelParameters`, `DSGEModel`, `AIMData`, and the boolean variable `OrderQZ`.

First, `logLikelihoodDSGE` runs either the `AiMSolver`, the `KleinSolver` or the `SimsSolver` function. Given that it returns an `mcode` equal to unity and AiM is used to solve the model, the `AiMtoStateSpace` function is executed. This provides us with the data to determine F and

Q (B_0) that the Kalman filter needs. Next, the measurement equation function is executed to obtain the A, H, and R matrices for the current value of the parameters. These five matrices are collected into the output variable `Solution`, a structure where the fields are named after the matrices, such as `Solution.A` gives the value of A, `Solution.H` gives H, and so on until `Solution.B0` gives B_0. Once this task is completed, the appropriate Kalman filter function is executed, yielding the outputs `logLikeValue` and `status`. Optionally, the function will also provide the time t values of the log-likelihood in the vector `logLikeT`.

7.4.5. `logPriorDSGE`

The function `logPriorDSGE` computes the log height of the joint prior density function at a given value of θ. It requires the four inputs `theta`, `thetaDist`, `PriorDist`, and `LowerBound`. If the value of log prior density is not a real number, `logPriorDSGE` returns `NaN`.

7.4.6. *System Prior File*

If the user has provided a system prior file, it's full path and name is stored in the field `SystemPriorFile` of the `DSGEModel` structure. This must be a function which takes the six input variables `theta`, `thetaPositions`, `ModelParameters`, `AIMData`, `DSGEModel`, and `OrderQZ`. The last input variable is simply a boolean variable which is true if the built-in matlab function `ordqz` is available; see Warne (2025) concerning the function `SolveDSGEModel`. The solution of the DSGE model is often stored in `DSGEModel.Solution` and may therefore be directly accessed inside the system prior file.

The function should only provide one output variable, which measures the log height of the system prior density. Since the variable name is local to the function it can be called anything, such as `logSP`.

7.4.7. `YADAcsminwel`

Given that the user has chosen to use Christopher Sims' `csminwel` function, the YADA implementation `YADAcsminwel` is utilized to minimize either the function `logPosteriorPhiDSGE` or `logPosterioThetaDSGE`. If the optimization algorithm converges within the maximum number of iterations that the user has selected, YADA makes use of the main output variables from this function. First of all, the vector `phiMode` or `thetaMode` is collected. Furthermore, the value of (minus) the log posterior at the mode is saved into `LogPostDensity`, while the inverse Hessian is located in the variable `InverseHessian`. In case the log posterior of ϕ is used for estimation then the mode of θ is obtained from the parameter transformation function `PhiToTheta`.

If, for some reason, `csminwel` is unable to locate the mode, YADA presents the return code message of `csminwel` indicating what the problem may be.

7.4.8. `YADAnewrat`

When Marco Ratto's `newrat` function has been selected, the YADA implementation `YADAnewrat` is utilized to minimize either the function `logPosteriorPhiDSGE` or `logPosterioThetaDSGE`. Since `newrat` uses an outer product gradient for calculation the Hessian, `newrat` requires values of the log posterior for all time periods in the estimation sample. The additional functions `logPosteriorPhiDSGE4Time` and `logPosterioThetaDSGE4Time` therefore support as a second output argument a vector `logPostT` with all time t values of the log posterior.[59] The input variables are identical to those in `logPosteriorPhiDSGE` and `logPosterioThetaDSGE`.

7.4.9. `YADAgmhmaxlik`

When Dynare's `gmhmaxlik` Monte Carlo based optimization procedure has been chosen, the YADA version `YADAgmhmaxlik` is applied to maximize either minus `logPosteriorPhiDSGE` or minus `logPosterioThetaDSGE`; recall that both these functions are setup for minimization. The Monte Carlo based simulation scheme that the function uses takes input from the posterior

---

[59] The original version of `newrat` in Dynare expects function names ending with `_hh`. This has been changed in `YADAnewrat` to `4Time`.

sampling and the specifics of these inputs are discussed in Sections 8.1 and 8.7.2. The initial scale factor, c, is taken from the *Scale factor for the posterior sampler* selection; see also Figure 4. Moreover, the number is Monte Carlo draws for estimating the posterior covariance matrix in equation (7.4) is equal to the number of posterior draws per chain minus the number of posterior draws discarded as burn-in period. As initial values for the posterior mean and the posterior covariance matrix YADA supplies the function with estimates based on draws from the prior distribution.

The function works in four different steps. First, it tunes the scale factor c such that the acceptance rate is close to the targeted rate of $1/4$. The posterior mode estimate is also updated as in equation (7.5) during this step, but the posterior mean and covariance matrix are not considered. The maximum number of parameter draws during this stage is equal to the minimum of 200,000 and 10 times the number of Monte Carlo draws. Second, based on this tuned scale factor and the initial covariance matrix, the posterior mean and posterior covariance matrix are estimated via (7.4) using the number of selected Monte Carlo draws, with the posterior mode estimate again being updated as in equation (7.5).

Third, the scale factor c is retuned based on the estimated posterior covariance matrix. This step is otherwise the same as the first, with the posterior mode estimate being updated using the rules of the sampler and equation (7.5). Once retuning has finished, the last step involves an attempt to climb the summit of the of the log posterior. The maximum number of parameter draws is the same as during step one and three with the same covariance matrix as in step three, but now all draws are accepted as possible mode candidates and only the test in (7.5) is applied. The scale factor is also retuned during this last step to cool down the system as it comes closer to the summit of the log posterior.

7.4.10. `YADAfminunc*`

YADA has three versions of Matlab's `fminunc` at its disposal. For Matlab versions prior to version 7, an older `fminunc` function is called: it is named `YADAfminunc5` and its original version is dated October 12, 1999. For version 7 and later, `YADAfminunc7` is used, and for Matlab versions prior to 7.5 it is originally dated April 18, 2005, while for Matlab version 7.5 and later it is dated December 15, 2006. The `YADAfminunc*` function attempts to minimize the function `logPosteriorPhiDSGE` or `logPosteriorThetaDSGE`, and if it is successful the vector `phiMode` or `thetaMode`, minus the value of the log posterior at the mode, and the Hessian at the mode are provided. This Hessian is inverted by YADA and the results is stored in the variable `InverseHessian`. In case the log posterior of ϕ is used for estimation then the mode of θ is obtained from the parameter transformation function `PhiToTheta`.

Again, if `YADAfminunc*` fails to locate the posterior mode within the maximum number of iterations, the return code message of `fminunc` is presented.

`YADAfminunc*` is only available in the version of YADA that is exclusive to the New Area-Wide Model team at the European Central Bank. As mentioned in Section 7, the publicly available version of YADA does not include these functions. Instead, diff-files are provided from the YADA website. These files provide the information the user needs to properly edit some files from the Optimization Toolbox such that `fminunc` can be used in YADA. It should be emphasized that the original files from the Optimization Toolbox should *not be edited*, only copies of them that are subject to the names changes required by YADA.

8. Posterior Sampling

8.1. *The Random Walk Metropolis Algorithm*

The Random Walk Metropolis (RWM) algorithm is a special case of the class of Markov Chain Monte Carlo (MCMC) algorithms popularly called Metropolis-Hastings algorithms; see, Hastings (1970) and Chib and Greenberg (1995) for an overview.

The Metropolis version of this MCMC algorithm is based on a symmetric proposal density, i.e., $q(\theta^*, \theta|Y) = q(\theta, \theta^*|Y)$, while the random walk part follows when the proposal density is symmetric around zero, $q(\theta^*, \theta|Y) = q(\theta^* - \theta, 0|Y)$. The random walk version of the algorithm was originally suggested by Metropolis, Rosenbluth, Rosenbluth, Teller, and Teller (1953) and was first used to generate draws from the posterior distribution of DSGE model parameters by Schorfheide (2000) and Otrok (2001).

The description of the RWM algorithm here follows An and Schorfheide (2007) closely.

(1) Compute the posterior mode of ϕ; cf. (7.1) or (7.2). The mode is denoted by $\tilde{\phi}$.
(2) Let $\tilde{\Sigma}$ be the inverse of the Hessian evaluated at the posterior mode $\tilde{\phi}$. YADA actually allows this matrix to be estimated in four different ways and that its correlations are scaled in a joint fashion towards zero.
(3) Draw $\phi^{(0)}$ from $N_m(\tilde{\phi}, c_0^2\tilde{\Sigma})$, where m is the dimension of ϕ, and c_0 is a non-negative constant.
(4) For $s = 1, \ldots, N$, draw φ from the proposal distribution $N_m(\phi^{(s-1)}, c^2\tilde{\Sigma})$, where c is a positive constant. The jump from $\phi^{(s-1)}$ is accepted ($\phi^{(s)} = \varphi$) with probability $\min\{1, r(\phi^{(s-1)}, \varphi|Y)\}$ and rejected ($\phi^{(s)} = \phi^{(s-1)}$) otherwise. The nonnegative function

$$r(\phi, \varphi|Y) = \frac{L(Y; \varphi)p(g^{-1}(\varphi))J(\varphi)}{L(Y; \phi)p(g^{-1}(\phi))J(\phi)}, \tag{8.1}$$

where $J(\phi)$ is the determinant of the Jacobian of $\theta = g^{-1}(\phi)$; cf. Section 6. Furthermore, if the model includes a system prior, then the jump probability in (8.1) is multiplied by $p(\Phi_\omega|g^{-1}(\varphi), h) / p(\Phi_\omega|g^{-1}(\phi), h)$. Notice that if the value of the numerator in (8.1) is greater than the denominator, then the jump from $\phi^{(s-1)}$ to $\phi^{(s)} = \varphi$ is always accepted.

An important difference between the algorithm described by An and Schorfheide (2007) and the one stated above is that An and Schorfheide samples θ directly, while YADA samples ϕ and transforms it into θ with $g^{-1}(\phi)$; see Section 6. The main reason for sampling ϕ rather than θ directly is that the support is $\Phi \equiv \mathbb{R}^m$ for ϕ (the m-dimensional Euclidean space), while the support for θ is typically constrained in some dimensions, i.e., $\Theta \subset \mathbb{R}^m$. Hence, every draw from the proposal density of ϕ is an element of Φ, while all draws of θ need to be verified first. Moreover, this also suggests that the selected proposal density may simply be a better (or not as bad) approximation of the posterior density when sampling ϕ directly rather than θ.

8.1.1. *Student-t Proposal Density*

The above algorithm is based on having a normal proposal density. A simple generalization of this is to consider a Student-t proposal density with d degrees of freedom; see equation (4.25) in Section 4.2.9. This density is also symmetric, but has fatter tails than the normal when d is finite. In step (3) of the RWM algorithm we would then instead draw $\phi^{(0)}$ from $T_m(\tilde{\phi}, c_0^2\tilde{\Sigma}, d)$, while in step (4) we would sample from the proposal density $T_m(\phi^{(s-1)}, c^2\tilde{\Sigma}, d)$. All other computation of the RWM algorithm are unaffected.

8.1.2. *Block Metropolis-Hastings Algorithms*

Further extensions of the basic RWM algorithm for DSGE models are discussed by Herbst and Schorfheide (2016, Chapter 4.4). As the number of parameters grows, the persistence of the Markov chain tends to increase with the result that the efficiency of the posterior simulator decreases. One possibility is therefore to split the parameters into separate groups and treat

the possible updating of the parameters of each group separately. The parameters can either be split into a fixed number of groups called blocks or into a random number of such blocks. In addition, the parameters can be assigned to a particular block for all iterations, randomly assigned to a block initially, or randomly assigned to a block for each iteration; see also Chib and Ramamurthy (2010). One advantage with blocking is that it tends to lower the correlation across iterations of the Markov chain since, e.g., not all parameters are necessarily updated in the same way. A key feature for this to occur is that the blocking procedure is independent of the Markov chain.

YADA supports two approaches to block Metropolis-Hastings posterior simulation. The first blocking algorithm has a fixed number of blocks, $N_B = 2, 3, \ldots, m$, where for each iteration the parameters are randomly assigned to a block through a uniform distribution and, as far as possible, the number of parameters in each block is the same.[60] This algorithm, which is here called fixed-block RWM, is described in Herbst and Schorfheide (2016, Algorithm 7) and the default value for $N_B = \min\{3, m\}$. This means that each parameter is assigned an iid draw from $U(0, 1)$, the parameters are sorted according to the assigned random number, and given to block b based on this ordering. For example, if $N_B = 4$ and $m = 12$, then the three parameters with the smallest assigned random numbers are placed in block 1, the three parameters with the 4th through 6th smallest random numbers in block 2, etc. The proposal density for each block is a multivariate Student-t with a fixed number of degrees of freedom $d_b = 1, 2, \ldots, \infty$ and therefore includes the normal proposal density as a special case. If the number of parameters divided by the number of blocks, m/N_B, is not an integer, the number of parameters per block is automatically adjusted for the last block by YADA.

The second algorithm, called random-block RWM, allows for a random number of blocks in each iteration, $N_{B,i}$, drawn from a discrete uniform distribution over the allowed block size values $\{m_l, m_l + 1, \ldots, m_u\}$, $1 \leq m_l < m_u \leq m$ such that there are at least two possible block sizes, and where uniformity means that each block size value is drawn with probability $1/(m_u - m_l + 1)$. Given this draw, the placement of parameters into blocks is performed like in the fixed-block RWM case, while the same proposal densities is also supported.

Let $\phi_b^{(s)}$ denote the parameters of block b for iteration s of ϕ, $\phi_{<b}^{(s)}$ contains the parameters of all blocks before b for simulation s, while $\phi_{>b}^{(s)}$ similarly contains all the parameters of ϕ of all blocks after b. With m_b being the dimension of ϕ_b, a proposal value for these parameters, denoted by φ_b, is drawn from $T_{m_b}\left(\phi_b^{(s-1)}, c^2 \tilde{\Sigma}_b, d_b\right)$, where $\tilde{\Sigma}_b$ only contains the rows and columns of $\tilde{\Sigma}$ which correspond to the parameters of ϕ_b, whereas $d_b \geq 1$.[61] The degrees of freedon parameter $d_b = d - m + m_b$ when the joint proposal for all m parameters has d degrees of freedom. Since $d_b \geq 1$ should hold, it follows that $d > m - m_b$ is required when selecting d. For the fixed-block RWM algorithm, this means that $d > m - \lfloor m/N_b \rfloor$ (where $\lfloor a \rfloor$ is the floor function for a), while for the random-block RWM $d > m - \lfloor m/m_u \rfloor$. Letting $\varphi = \left(\phi_{<b}^{(s)}, \varphi_b, \phi_{>b}^{(s-1)}\right)$ and $\phi = \left(\phi_{<b}^{(s)}, \phi_b^{(s-1)}, \phi_{>b}^{(s-1)}\right)$, the draw $\phi_b^{(s)} = \varphi_b$ is accepted with probability $\min\{1, r(\phi, \varphi|Y)\}$, and rejected ($\phi_b^{(s)} = \phi_b^{(s-1)}$) otherwise. The value $r(\phi, \varphi|Y)$ is computed from equation (8.1), but with the new definitions of ϕ and φ.

For all block-based RWM algorithms, the calculation time for each iteration increases relative to the standard RWM algorithm since the number of evaluations of the posterior density kernel is linear in the number of blocks. For instance, we may expect that the fixed 3-block RWM takes three times as long as the 1-block RWM for each iteration. It may also be noted that the denominator in equation (8.1) is available from the previous jump probability computation, i.e., from the previous block or from the last block of the previous iteration.

---

[60] The case of $N_B = 1$ is already handled through the basic RWM algorithms with a normal or Student-t proposal density.

[61] In theory, the parameters of the other blocks will also enter the expression for the proposal density. However, given that the joint proposal density is multivariate Student-t with a covariance matrix which does not depend on the draws, the marginal proposal density for each block does not depend on the parameters of the other blocks.

The acceptance rate for the blocking algorithms is based on the full set of parameters. This means that if at least one block has been accepted, then this counts as an acceptance for the iteration. While it would also be possible to count the number of times any parameter block is accepted and compare this with the number of times that a proposal draw is made (equal to the number of blocks times the number of iteration when a fixed number of blocks is used), this alternative acceptance rate is not measured by YADA.

8.1.3. *Numerical Standard Errors*

The expected value of any real valued function $h(\phi)$ is approximated by $N^{-1} \sum_{s=1}^{N} h(\phi^{(s)})$. Hence, the posterior mean and covariance can be calculated directly from the RWM output. Furthermore, the numerical standard error of the posterior mean can be calculated using the Newey and West (1987) estimator.

The numerical standard error of the posterior mean $\bar{\phi}$ is computed as follows. Let $\bar{N} \leq N$ be an integer such that the autocorrelation function for the posterior draws of ϕ tapers off. Consider the matrix

$$\hat{\Sigma}_\phi = \frac{1}{N} \sum_{s=-\bar{N}}^{\bar{N}} \frac{\bar{N} + 1 - |s|}{\bar{N} + 1} \Gamma(s), \tag{8.2}$$

where

$$\Gamma(s) = \frac{1}{N} \sum_{n=s+1}^{N} \left(\phi^{(n)} - \bar{\phi} \right) \left(\phi^{(n-s)} - \bar{\phi} \right)', \quad \text{if } s \geq 0,$$

and $\Gamma(s) = \Gamma(|s|)'$ when $s \leq -1$. The kernel in (8.2), given by $1 - |s| / (\bar{N}+1)$, is a Bartlett kernel; for alternative kernels, see for instance Andrews (1991). The numerical standard error of $\bar{\phi}$ is given by the square root of the diagonal elements of $\hat{\Sigma}_\phi$. The matrix $\hat{\Sigma}_\phi$ has the usual property that it converges to 0 in probability as $N \rightarrow \infty$ when the Markov chain that has generated the $\phi^{(s)}$ sequence is ergodic; see, e.g. Tierney (1994).

8.2. *The Slice Sampling Algorithm*

One drawback with Metropolis-Hastings algorithms is that they require quite a lot of tuning to work well. Tuning involves the selection of a proposal density and its parameterization. If the proposal density is a poor approximation of the posterior distribution, convergence of the sampler may be very slow or, even worse, the sampler may not cover important subsets of the support for the posterior distribution.

The issue of convergence of the MCMC sampler is covered in Section 9 and the tools discussed there apply to any MCMC sampler. Assuming that, for instance, the RWM sampler has converged we may nevertheless find that it is not very efficient. In particular, a large number of draws may be required to achieve convergence due to high serial correlation in the MCMC chain. To examine how efficient the sampling algorithm is, variance ratios, such as the inefficiency factor, can be computed from draws of the individual parameters as well as for the value of the log posterior; see, e.g., Roberts (1996, Section 3.4).[62]

An MCMC sampler that requires less tuning and which relies on standard distributions is the so called *slice sampler*. This sampler is based on the idea that to sample a random variable one can sample *uniformly* from a region under a slice of its density function; Neal (2003). This idea is particularly attractive for models with non-conjugate priors, such as DSGE models, since for the region under the slice to be unique the height of the density need only be determined up to a constant.

---

[62] The inefficiency factor is equal to the ratio between the variance of the draws when autocorrelation is taken into account and the variance under the assumption that the draws are iid. That is, the inefficiency factor is equal to 1 plus 2 times the sum of the autocorrelations. For serially correlated processes this factor is greater than unity, and the larger its value is the more inefficient the sampler is. Roberts (1996) considers the inverse of this ratio, i.e., the efficiency factor, which YADA presents as RNE (relative numerical efficiency); see also Geyer (1992).

To formalize the idea of slice sampling we shall first consider the univariate case. Let $f(x)$ be proportional to the density of x, denoted by $p(x)$. A slice sampling algorith for x follows these steps:

(1) Draw y from $U(0, f(x_0))$, where x_0 is a value of x such that $f(x_0) > 0$. This defines the horizontal slice: $S = \{x : y < f(x)\}$, where by construction $x_0 \in S$.
(2) Find an interval, $I = (L, R)$, around x_0 that contains all, or much, of the slice.
(3) Draw a new point, x_1, uniformly from the part of the slice within I.

Notice that the horizontal slice S is identical to the slice $S^* = \{x : y^* < p(x)\}$, where $y^* \equiv (p(x_0)/f(x_0))y$.

The first step of the sampler involves the introduction of an auxiliary variable, y, which need not be stored once a new value of x has been drawn. It only serves the purpose of determining a lower bound for $f(x)$, i.e., we are only interested in those values of x, on the horizontal axis, for which $f(x)$ is greater than y. The difficult step in the algorithm is the second, where a suitable interval on the horizontal axis needs to be determined.[63] Unless the interval I is identical to S, the third step in the algorithm may require that the function $f(x)$ is evaluated more than once per iteration.

A multivariate case using hyperrectangles is described by Neal (2003, Section 5.1). The three steps given above are still valid, except x is now m-dimensional, and the interval I is replaced by the hyperrectangle $H = (L_1, R_1) \times \cdots \times (L_m, R_m)$. Let σ_i be scale estimates for each variable $i = 1, \ldots, m$ while x_0 is the previous value of x. The multivariate slice sampler based on hyperrectangles can be implemented as follows:

(1) Draw y from $U(0, f(x_0))$;
(2) Randomly position H: let $u_i \sim U(0, 1)$, $L_i = x_{0,i} - \sigma_i u_i$, and $R_i = L_i + \sigma_i$ for $i = 1, \ldots, m$; and
(3) Sample from H, shrinking when points are rejected: (a) let $x_{1,i} = L_i + v_i(R_i - L_i)$, $v_i \sim U(0, 1)$ for $i = 1, \ldots, m$; (b) if $f(x_1) > y$ then exit loop, else let $L_i = x_{1,i}$ if $x_{1,i} < x_{0,i}$ and $R_i = x_{1,i}$, $i = 1, \ldots, m$, otherwise and return to (a) with new H.

Notice that this algorithm does not take dependencies between x_i and x_j into account and, hence, there is room for improvements; see Neal (2003) and the discussion for more details on more elaborate multivariate slice samplers. Convergence properties of slice samplers are considered in Mira and Tierney (2002) and Roberts and Rosenthal (1999), while Roberts and Tweedie (1996) considers convergence for Metropolis-Hastings samplers.

In practise, it is often not suitable to evaluate $f(x)$ since problems with floating-point underflow are likely to appear. To alleviate this issue we instead focus on $\ln(f(x))$, where the auxiliary variable y is replaced with $z = \ln(y) = \ln(f(x_0)) - \epsilon$, and where ϵ is exponentially distributed with mean one. That is, $\epsilon \sim \mathcal{E}(1)$; cf. Section 4.2.3. The slice is now given by $S = \{x : z < \ln(f(x))\}$.[64]

Moreover, the above method shrinks the hyperrectangle in *all* dimensions until a point is found inside the slice, even if the density is relatively flat in some of the dimensions, thereby making shrinkage in these dimensions unnecessary. Neal (2003) has suggested that one way to avoid this is to only shrink in one dimension, determined by the gradient of the density. Specifically, the shrinkage dimension i is chosen such that $(L_i - R_i)|\partial \ln f(x)/\partial x_i|$ is maximized when the gradient is evaluated at the last chosen point (x_0).

The slice sampler that has been implemented in YADA uses the hyperrectangle approach with shrinkage in one dimension only. Let $f(\phi) = L(Y; \phi)p(g^{-1}(\phi))J(\phi)$, $g_i = \partial \ln f(\phi)/\partial \phi_i$, and $\sigma_i = c\sqrt{\tilde{\Sigma}_{ii}}$ for $i = 1, \ldots, m$, where $\tilde{\Sigma}_{ii}$ is the i:th diagonal element of the inverse Hessian at the

---

[63] Draws from a normal distribution can be produced with this algorithm. Specifically, given a value for x draw y uniformly from $[0, \exp(-x^2/2)/\sqrt{2\pi}]$. Define the upper bound of the horizontal slice by $R = \sqrt{-2\ln(y\sqrt{2\pi})}$, while $L = -R$. Finally, draw x uniformly from $[L, R]$.

[64] The exponential distribution with mean μ_ϵ has cdf $F(z|\mu_\epsilon) = \exp(-z/\mu_\epsilon)$. To sample from this distribution we set the cdf equal to $p \sim U(0, 1)$ and invert the expression, so that $z = -\mu_\epsilon \ln(p)$. Notice that when $y \sim U(0, f(x_0))$ it follows directly that $y/f(x_0) \sim U(0, 1)$. Hence, $\ln y - \ln f(x_0) = \ln(p)$, where $p \sim U(0, 1)$.

mode and $c > 0$. The initialization of the slice sampler is performed in the same way as for the RWM algorithm, i.e., steps 1–3 in Section 8.1. For $s = 1, \ldots, N$ the slice sampler continues with:

(4) Let $z = \ln(f(\phi^{(s-1)})) - \epsilon$, where $\epsilon \sim \mathcal{E}(1)$;

(5) Randomly position H: let $u_i \sim U(0, 1)$, $L_i = \phi_i^{(s-1)} - \sigma_i u_i$, and $R_i = L_i + \sigma_i$ for $i = 1, \ldots, m$; and

(6) Sample from H, shrinking when points are rejected: (a) let $\varphi_i = L_i + v_i(R_i - L_i)$, $v_i \sim U(0, 1)$ for $i = 1, \ldots, m$; (b) if $\ln(f(\varphi)) > z$ then $\phi^{(s)} = \varphi$ and exit loop, else let $L_i = \varphi_i$ if $\varphi_i < \phi_i^{(s-1)}$ and $R_i = \varphi_i$ otherwise, where i is given by the element that maximizes $(L_i - R_i)|g_i|$ when g_i is evaluated at $\phi^{(s-1)}$, and return to (a) with new H.

8.3. *Discussion*

The computationally costly part of the slice sampler is the evaluation of the log posterior, while all other steps can easily be vectorized and dealt with very quickly. It is likely that the log posterior will be evaluated more than once during an iteration and, for any fixed number of draws, it therefore takes more time to complete the slice sampler than the single block RWM algorithm (which only requires one evaluation). However, the slice sampler does not require much tuning (choice of σ_i), while the RWM relies on a particular proposal density and its parameterization. Moreover, it seems less likely that draws from the slice sampler suffer from a high autocorrelation than those from the single block RWM and, hence, that fewer draws from the posterior may be required to achieve convergence. In other words, the slice sampler is potentially more efficient than the single block RWM algorithm, at least for models where the proposal density of the RWM is poorly chosen. Hence, there may be a trade-off between sampling efficiency, need for tuning, and the average number of times the log posterior needs to be evaluated; see Neal (2003) and the discussion to the article.

The multiple block RWM algorithms can potentially lead to lower autocorrelation in the Markov chain since potentially larger parameter jumps can occur. The cost is that the log posterior needs to be evaluated as many times as there are blocks per iteration; see Herbst and Schorfheide (2016) for further discussions on this topic.

The choice of scale parameter, $c^2 \tilde{\Sigma}$, under the single and multiple block RWM algorithms is particularly important. If it is too large, a large proportion of the iterations will be rejected ($\phi^{(s)} = \phi^{(s-1)}$), and if the scale parameter is too small, the RWM will accept nearly all proposed moves. In both cases, the RWM will be very inefficient. The studies by Gelman, Roberts, and Gilks (1996b) and Roberts, Gelman, and Gilks (1997) give some theoretical arguments, supported by simulations, for aiming at acceptance rates of the RWM between 15 and 50 percent, with the rate decreasing (and inefficiency increasing) as the dimension of the parameter vector increases. In DSGE models, it is sometimes recommended that the acceptance rate should be around 25-35 percent. Nevertheless, since the "optimal" acceptance rate depends on the dimension of the problem, it is wise to also monitor other properties of the sampler, such as the inefficiency factor, and its convergence properties. The latter aspect is discussed in Section 9.

8.4. *Sequential Monte Carlo Sampling*

Importance sampling has long been one of the main algorithms in economerics and statistics for posterior sampling, but has rarely been applied for DSGE models; see, e.g., Kloek and van Dijk (1978), DeJong et al. (2000), Geweke (1989a, 2005), or An and Schorfheide (2007). A key problem with importance sampling is the selection of a good importance density. The sequential Monte Carlo (SMC) algorithms considered below rely on the construction of importance densities in a sequential fashion, starting from the prior distribrution, and where more weight is given to the likelihood function over the sequence. An interesting approach to the determination of a useful importance density has been suggested by Hoogerheide, Opschoor, and van Dijk (2012), which is based on mixtures of Student-t densities is discussed in Section 8.5.

SMC algorithms were originally designed to deal with filtering problems in nonlinear state-space models. Chopin (2002) showed how to make use of particle filtering methods for posterior

inference in static models. SMC methods have been suggested for DSGE models by Creal (2007), while a survey of their use in economics and finance is provided by Creal (2012). The SMC algorithms in Herbst and Schorfheide (2014, 2016) build on this literature, while the theoretical underpinnings stem from Chopin (2004).

The SMC algorithm with likelihood tempering in YADA is based on Herbst and Schorfheide (2014, 2016) and follows their setup closely.[65] Let τ_n, $n = 1, 2, \ldots, N_\tau$ be a sequence of scalars that increases with n, where $\tau_1 = 0$ and $\tau_{N_\tau} = 1$. This sequence is called a *tempering schedule* and we likewise define a sequence of tempered posteriors as

$$\pi_n(\phi) = \frac{p(Y|\phi)^{\tau_n} p(\phi)}{\int p(Y|\phi)^{\tau_n} p(\phi) d\phi}, \qquad n = 1, 2, \ldots, N_\tau. \tag{8.3}$$

From the perspective of system priors, the prior $p(\phi)$ may also be conditional on Φ_ω, but has here been suppressed for notational convenience; see equation (4.37).

Following the standard SMC terminology, the overall number of particles (parameter draws) is given by N and at any stage the posterior distribution $\pi_n(\phi)$ is represented by a swarm of particles $\{\phi_n^{(s)}, W_n^{(s)}\}_{s=1}^N$, where $W_n^{(s)}$ are the normalized importance weights assigned to each particle $\phi_n^{(s)}$ and which satisfy the condition:

$$\sum_{s=1}^N W_n^{(s)} = N.$$

Embarking from the particles of stage n, $\{\phi_n^{(s)}, W_n^{(s)}\}_{s=1}^N$, the SMC algorithm loops through three steps. Using Chopin's (2004) terminology, the first step is called *correction* and it involves the reweighting of the stage n particles to reflect the density in stage $n + 1$. The second step is called *selection* where a too uneven distribution of particle weights is eliminated by resampling of the particles when deemed necessary. Finally, the third step is called *mutation* where the particles are propagated forward via a Markov transition kernel to adapt the particles to the stage $n + 1$ bridge density $\pi_{n+1}(\phi)$.

Once we have reached the end of the tempering schedule, $n = N_\tau$, the set of particles $\{\phi_{N_\tau}^{(s)}, W_{N_\tau}^{(s)}\}_{s=1}^N$ provide a particle approximation of $\pi(\phi)$, the posterior distribution. With $h(\phi)$ being a vector valued function of ϕ (such as the original parameters θ), the Monte Carlo estimate

$$\bar{h}_{N_\tau, N} = \frac{1}{N} \sum_{s=1}^N h(\phi_{N_\tau}^{(s)}) W_{N_\tau}^{(s)}, \tag{8.4}$$

converges almost surely to $E_\pi[h(\phi)]$, the expectation of h under the posterior $\pi(\phi)$, subject to suitable regularity conditions; see, e.g., Geweke (2005). Notice that the weights need to be used in combination with the parameter draws (like in the standard importance sampling based Monte Carlo estimate of h) since the draws are obtained from the importance density $\pi_{N_\tau-1}(\phi)$ rather than from the approximate posterior density $\pi_{N_\tau}(\phi)$.

An alternative to tempering the likelihood function is to construct a sequence of posteriors by adding observations to the likelihood function; see Del Moral, Doucet, and Jasra (2006) and Durham and Geweke (2014). In this case

$$\pi_n^{(D)}(\phi) = \frac{p(Y_{\tau_n}|\phi) p(\phi)}{\int p(Y_{\tau_n}|\phi) p(\phi) d\phi}, \quad n = 1, \ldots, N_\tau, \tag{8.5}$$

where $Y_{\tau_n} = [y_1 \cdots y_{\tau_n}]$, $\tau_n \in \{0, 1, \ldots, T\}$ with $\tau_n < \tau_{n+1}$ for all $n < N_\tau$, $\tau_1 = 0$, and $\tau_{N_\tau} = T$. Data tempering is particularly attractive in forecasting situations, but is also somewhat less flexible than likelihood tempering since individual observations are not divisible, with $N_\tau \leq T$. Herbst and Schorfheide (2016) point out that this may be important in the early stages

---

[65] One difference is that YADA samples the transformed parameters, another is that YADA also supports a system prior, although the procedures in Herbst and Schorfheide (2014, 2016) extend to such priors. Moreover, YADA supports more efficient resampling algorithms in the selection step, as discussed in Section 8.4.3.

of the SMC sampler where it may be an advantage to add information is small increments. Section 8.4.4 presents a data tempering SMC algorithm based on the approach considered by Durham and Geweke (2014).

An interesting application of SMC sampling with data tempering is Lanne and Luoto (2018), who also suggest running a final nonsequential importance sampling step to be able to more easily assess the numerical accuracy of the point estimates from the SMC sampler. The basic idea is to estimate the parameters of the final importance sampling density, a mixture of Student-t densities, via the Expectation Maximization (EM) algorithm (Dempster, Laird, and Rubin, 1977), using the SMC particles as observations; see also Hoogerheide et al. (2012). Once this importance density has been parameterized, the standard importance sampling algorithm may be applied, including the calculation of numerical standard errors for point estimates; see Geweke (2005).

8.4.1. *A Generic SMC Algorithm with Likelihood Tempering*

The algorithm provided below relies on a number of tuning parameters. Following Herbst and Schorfheide (2014, 2016), let $\{\rho_n\}_{n=1}^{N_\tau}$ be a sequence of zeros and ones that determine if particles are resampled in the selection step. Furthermore, let $\{\zeta_n\}_{n=1}^{N_\tau}$ be a sequence of tuning parameters for the Markov transition density in the mutation step. The adaptive choice of these tuning parameters will be discussed in Section 8.4.2.

(1) **Initialization:** Set $n = 1$ such that $\tau_1 = 0$. Draw the initial particles from the prior: $\phi_1^{(s)} \sim p(\phi)$ and $W_1^{(s)} = 1$, $s = 1, \ldots, N$.

(2) **Recursion:** For $n = 2, 3, \ldots, N_\tau$.

  (a) **Correction:** Reweight the particles from step $n - 1$ by defining
$$\tilde{w}_n^{(s)} = \left[p\big(Y|\phi_{n-1}^{(s)}\big) \right]^{\tau_n - \tau_{n-1}},$$
  and
$$\tilde{W}_n^{(s)} = \frac{\tilde{w}_n^{(s)} W_{n-1}^{(s)}}{(1/N) \sum_{r=1}^N \tilde{w}_n^{(r)} W_{n-1}^{(r)}}, \quad s = 1, 2, \ldots, N.$$

  (b) **Selection:**

  (i) If $\rho_n = 1$, resample the particles via multinomial (simple random) resampling. Let $\{\hat{\phi}_n^{(s)}\}_{s=1}^N$ denote N draws from the multinomial distribution characterized by supporting points and weights $\{\phi_{n-1}^{(s)}, \tilde{W}_n^{(s)}\}_{s=1}^N$ and set $W_n^{(s)} = 1$ for $s = 1, \ldots, N$.

  (ii) If $\rho_n = 0$, let $\hat{\phi}_n^{(s)} = \phi_{n-1}^{(s)}$ and $W_n^{(s)} = \tilde{W}_n^{(s)}$ for $s = 1, \ldots, N$.

  (c) **Mutation:** Propagate the particles $\{\hat{\phi}_n^{(s)}, W_n^{(s)}\}_{s=1}^N$ via M steps of a Metropolis-Hastings algorithm with transition density
$$\phi_n^{(s)} \sim K_n\big(\phi_n | \hat{\phi}_n^{(s)}; \zeta_n\big),$$
  and stationary distribution $\pi_n(\phi)$.

(3) **End:** For $n = N_\tau$, the final importance sampling approximation of $E_\pi[h(\phi)]$ is given by equation (8.4).

This generic algorithm is called *simulated tempering SMC* by Herbst and Schorfheide (2014). Notice that the weights $W_n^{(s)}$ are initialized at $W_1^{(s)} = 1$ for all s. An alternative is to initialize these weights at $1/N$, which implies that terms involving summations of weights in the algorithm need to be adjusted accordingly. The choice of weights for the initialization step above can be motivated from a purely numerical perspective, where for large N the initialization $1/N$ is more likely to lead to numerical inaccuracies or overflow.

One interesting feature of the correction step is that it provides a recursive means to estimate the marginal likelihood. Specifically,

$$\hat{p}_{SMC}(Y) = \prod_{n=2}^{N_\tau} \left(\frac{1}{N} \sum_{s=1}^{N} \tilde{w}_n^{(s)} W_{n-1}^{(s)} \right). \tag{8.6}$$

As pointed out by Herbst and Schorfheide (2016), the calculation of this estimate does not require any additional likelihood evaluations.

Furthermore, the multinomial resampling method in the selection step can be replaced with stratified, systematic, or residual resampling; see Section 8.4.3. Resampling was originally employed by Gordon, Salmond, and Smith (1993) for the Bayesian bootstrap filter, an early particle filter (SMC algorithm), to reduce the effects from sample *degeneracy*—a highly uneven distribution of particle weights—as this part of the selection step adds noise; see Chopin (2004). It also allows for the removal of low weight particles with a high probability and this is very practical as it is preferable that the algorithm is focused on regions with a high probability mass. However, this also means that resampling may produce sample *impoverishment* as the diversity of the particle values is reduced; see also Section 8.4.2 below.

Before we turn our attention to the selection of tuning parameters ρ_n and ζ_n, it should be noted that the tempering schedule, τ_n, also needs to be determined. Herbst and Schorfheide (2014, 2016) suggest to let this schedule be calculated from

$$\tau_n = \left(\frac{n-1}{N_\tau - 1} \right)^\lambda, \quad n = 1, 2, \dots, N_\tau, \tag{8.7}$$

where λ is a *bending* (tempering schedule) parameter. A large value of λ means that the bridge distributions, $\pi_n(\phi)$, will be very similar and close to the prior when n is small and further apart as n becomes larger. A value of $\lambda = 2.1$ is used by Herbst and Schorfheide (2014) in their applications and YADA has therefore selected $\lambda = 2$ as its default value. According to Herbst and Schorfheide (2014), a smaller value for λ means that the information from the likelihood will dominate the prior too quickly, while a too large value leads to some of the bridge distributions having little usefulness and thereby to unnecessary calculations. For example, Creal (2007) uses a linear cooling schedule ($\lambda = 1$), with the effect that the information contained in $p(Y|\phi)^{\tau_2}$ dominates the information contained in the prior. For this reason, Creal (2007) initializes the simulator from a Student-t distribution, centered at the posterior mode. By instead letting $\lambda > 1$, the information from the likelihood can be added more slowly to the bridge distribution.

8.4.2. *Adaptive Choice of Tuning Parameters*

The algorithm in Section 8.4.1 is based on two sets of tuning parameters that affect the selection and the mutation steps, respectively. Both sets of parameters are chosen *adaptively* in YADA, i.e. the parameters for iteration n are selected on the basis of the particles from previous iterations.

Concerning the first set of tuning parameters, ρ_n, we follow Herbst and Schorfheide (2014) and make use of the *effective sample size* (Kong, Liu, and Wong, 1994; Liu and Chen, 1995). It is here given by

$$\text{ESS}_n = \frac{N}{(1/N) \sum_{s=1}^{N} \left(\tilde{W}_n^{(s)} \right)^2}. \tag{8.8}$$

Notice that if all particles have equal weights, then the effective sample size is N. Moreover, we let the tuning parameter ρ_n be determined adaptively such that

$$\hat{\rho}_n = \mathbb{I}\left(\text{ESS}_n < N/k \right),$$

where $k > 1$, while the indicator function $\mathbb{I}(x < y)$ is unity if $x < y$ and zero otherwise. The rule-of-thumb threshold-based value of $N/2$ is suggested by Herbst and Schorfheide (2014, 2016) and is the default value in YADA. Alternative values of k supported by YADA are those such that $100/k$ belongs to the set $\{10, 15, 20, \dots, 85, 90\}$. This set therefore determines the percent of the full sample size (N) where resampling is undertaken during the selection step,

while N/k is the corresponding threshold-based value of the effective sample size. The hyper-parameter k provides a 'crude' tool for balancing the algorithm against the pitfalls of degeneracy or impoverishment by ensuring that resampling takes place but does not occur 'too often'.[66]

For the second set of tuning parameters, the implementation of the mutation step in YADA allows for fixed blocking of parameters with random assignment to blocks (cf. Section 8.1.2), mixed normal transition densities, adaptive estimation of the mean and covariance matrix for the normals, and an adaptive determination of the scaling factor for the covariance matrix. The fixed number of parameter blocks is again denoted by N_B, while the parameters that belong to block b are given by ϕ_b. The parameter $0 < \alpha \le 1$ is the mixing coefficient and M the number of Metropolis-Hastings steps for the particle mutation.

Furthermore, let ϕ_n^* and Σ_n^* be a location vector and covariance matrix which are conformable with ϕ, while $\phi_{n,b}^*$ and $\Sigma_{n,b}^*$ are the partitions of ϕ_n^* and Σ_n^*, respectively, that correspond to the subvector $\phi_{n,b}$ for $b = 1, \dots, N_B$. In practice, we replace ϕ_n^* and Σ_n^* with the importance sampling estimates:

$$\tilde{\phi}_n = \frac{1}{N} \sum_{s=1}^{N} \hat{\phi}_n^{(s)} W_n^{(s)}, \quad \tilde{\Sigma}_n = \frac{1}{N} \sum_{s=1}^{N} (\hat{\phi}_n^{(s)} - \tilde{\phi}_n) W_n^{(s)} (\hat{\phi}_n^{(s)} - \tilde{\phi}_n)'.$$

With c_n being a scaling constant, we let the tuning parameters for the mutation step be given by:

$$\zeta_n = \left[c_n, \phi_n^{*\prime}, \text{vech}\left(\Sigma_n^*\right)' \right]',$$

where vech is the operator which stacks the elements of each column on and below the diagonal. Before running the SMC algorithm, generate a sequence of random partitions $\{B_n\}_{n=2}^{N_\tau}$ of ϕ into N_B equally sized blocks; as in Section 8.1.2. The mutation step of the SMC algorithm now proceeds as follows:

(1) For each particle $s = 1, \dots, N$, step $i = 1, \dots, M$, and block $b = 1, \dots, N_B$, let $\phi_{n,b,i}^{(s)}$ be the parameter value of $\phi_{n,b}^{(s)}$ in the i:th step, initialized with $\phi_{n,b,0}^{(s)} = \hat{\phi}_{n,b}^{(s)}$ and let

$$\phi_{n,-b,i}^{(s)} = \left(\phi_{n,1,i}^{(s)}, \dots, \phi_{n,b-1,i}^{(s)}, \phi_{n,b+1,i-1}^{(s)}, \dots, \phi_{n,N_B,i-1}^{(s)} \right).$$

(2) Generate a proposal draw φ_b from the mixture distribution for step i:

$$\varphi_b \Big| \left(\phi_{n,b,i-1}^{(s)}, \phi_{n,-b,i}^{(s)}, \zeta_n \right) \sim$$
$$\alpha N\left(\phi_{n,b,i-1}^{(s)}, c_n^2 \Sigma_{n,b}^* \right) + \frac{1-\alpha}{2} N\left(\phi_{n,b,i-1}^{(s)}, c_n^2 \text{diag}\left(\Sigma_{n,b}^* \right) \right) \tag{8.9}$$
$$+ \frac{1-\alpha}{2} N\left(\phi_{n,b}^*, c_n^2 \Sigma_{n,b}^* \right),$$

and denote this density by $q\left(\varphi_b | \phi_{n,b,i-1}^{(s)}, \phi_{n,-b,i}^{(s)}, \zeta_n \right)$.

(3) The proposal draw φ_b is accepted with probability p_α such that $\phi_{n,b,i}^{(s)} = \varphi_b$, and rejected with probability $1 - p_\alpha$ and such that $\phi_{n,b,i}^{(s)} = \phi_{n,b,i-1}^{(s)}$. As usual for Metropolis-Hastings algorithms, the probability p_α is given by

$$p_\alpha = \min \left\{ 1, r\left(\phi_{n,b,i-1}^{(s)}, \varphi_b \Big| \phi_{n,-b,i}^{(s)}, Y, \tau_n \right) \right\}, \tag{8.10}$$

---

[66] A more direct approach to combatting sample impoverishment is based on taking the particle values into account when resampling; see, e.g., Doucet and Johansen (2011) for discussions on the *resample-move* algorithm of Gilks and Berzuini (2001) and the *block sampling* algorithm of Doucet, Briers, and Sénécal (2006).

where

$$r\left(\phi_{n,b,i-1}^{(s)}, \varphi_b \Big| \phi_{n,-b,i}^{(s)}, Y, \tau_n\right) = \frac{p\left(Y|\varphi_b, \phi_{n,-b,i}^{(s)}\right)^{\tau_n} p\left(\varphi_b, \phi_{n,-b,i}^{(s)}\right)}{p\left(Y|\phi_{n,b,i-1}^{(s)}, \phi_{n,-b,i}^{(s)}\right)^{\tau_n} p\left(\phi_{n,b,i-1}^{(s)}, \phi_{n,-b,i}^{(s)}\right)}$$
$$\times \frac{q\left(\phi_{n,b,i-1}^{(s)}|\varphi_b, \phi_{n,-b,i}^{(s)}, \zeta_n\right)}{q\left(\varphi_b|\phi_{n,b,i-1}^{(s)}, \phi_{n,-b,i}^{(s)}, \zeta_n\right)}.$$

(4) Let $\phi_{n,b}^{(s)} = \phi_{n,b,M}^{(s)}$ for $b = 1, \ldots, N_B$.

In the event that $N_B = 1$, we may suppress the subscript b, and if $M = 1$ we can likewise suppress i. Notice that if these conditions are met and $\alpha = 1$, the above simplifies to a one-step RWM algorithm with a normal proposal density for each n and s. More generally, with $\alpha = 1$ the proposal density is symmetric such that the $q(\cdot)$ terms cancel out and can be removed from equation (8.10).

The adaptive construction of ζ_n have already in large part been discussed where ϕ_n^* and Σ_n^* are estimated with the importance sampling approximations of the posterior mean and covariance matrix of ϕ. To determine the scaling constant c_n we may also follow Herbst and Schorfheide (2014) and let $\hat{r}_{n-1}(\zeta_{n-1})$ be the average empirical rejection rate based on the mutation step in iteration $n-1$ and across the N_B blocks. To initialize, let $c_2 = c^*$ while

$$c_n = c_{n-1} f\left(1 - \hat{r}_{n-1}(\zeta_{n-1})\right), \quad n = 3, 4, \ldots, N_\tau,$$

where

$$f(x) = 0.95 + 0.10 \frac{\exp\left(16(x - p_\alpha^*)\right)}{1 + \exp\left(16(x - p_\alpha^*)\right)}.$$

The parameter p_α^* is the target acceptance rate and is set equal to 0.25 by Herbst and Schorfheide (2014, 2016). Notice that $f(p_\alpha^*) = 1$ such that the scaling factor is constant when the target acceptance rate has been reached. If the average acceptance rate is below (above) the target rate then the scaling constant is lowered (raised).

8.4.3. *Resampling Algorithms for the Selection Step*

The resampling algorithm considered by Herbst and Schorfheide (2014) for the selection step is called *multinominal resampling* and is discussed in more detail by, for example, Herbst and Schorfheide (2016) and references therein. A number of alternative resampling methods are discussed in this latter reference, such as residual, stratified, and systematic resampling. As pointed out by, for example, Douc, Cappé, and Moulines (2005), the multinomial resampling algorithm produces an unnecessarily large variance of the particles. Moreover, and as emphasized by Hol, Schön, and Gustafsson (2006), ordering of the underlying uniform draws improves the computational speed considerably.

The commonly used alternative resampling algorithms are faster and have a smaller variance of the particles. *Systematic resampling*, introduced by Kitagawa (1996) and also emphasized by Carpenter, Clifford, and Fearnhead (1999), is a commonly used approach as it is very easy to implement, comparatively fast, and, according to Doucet and Johansen (2011), as it often outperforms other sequential resampling schemes. It is a faster version of *stratified resampling* (Kitagawa, 1996), where instead of drawing N uniforms only one is required, while stratification and, simultaneously, sorting is dealt with via the same simple affine function. A drawback with systematic resampling is that it generates cross-sectional dependencies among the particles, which also makes it difficult to establish its theoretical properties; see Chopin (2004). For an overview of sequential resampling schemes see, e.g., Hol et al. (2006), who also discuss theoretical criteria for choosing between multinomial, stratified, systematic and *residual resampling* (suggested by Liu and Chen, 1998); Douc et al. (2005), who also study large sample behavior; and more recently Li, Bolić, and Djurić (2015), who also discusses distributed or parallel algorithms.

8.4.3.1. *Multinominal Resampling*

The basic multinomial resampling algorithm consists in first obtaining N draws u_s from $U(0, 1)$. Next, we compare each such draw to the accumulation of $\tilde{W}_n^{(i)}$ over i, denoted by $\tilde{V}_n^{(i)} = (1/N) \sum_{k=1}^{i} \tilde{W}_n^{(k)}$, and let j_s be the smallest positive integer such that $u_s < \tilde{V}_n^{(j)}$. We now set $\hat{\phi}_n^{(s)} = \phi_{n-1}^{(j_s)}$ for $s = 1, \ldots, N$, yielding N draws from the multinomial distribution in the selection step of the SMC algorithm.

To speed up this resampling algorithm we may sort the uniform random draws u_s from the smallest to the largest. Rather than relying on a separate sorting function, such as `sort` in matlab, one may instead generate N ordered uniform random numbers directly, as in Hol et al. (2006). This means that $\tilde{u}_s = \tilde{u}_{s+1} u_s^{1/s}$, while $\tilde{u}_N = u_N^{1/N}$, where $u_s \sim U(0, 1)$. The efficient multinomial resampling algorithm requires $O(N)$ operations, while sorting unsorted uniform draws u_s and then performing the determination of the resampled indexes requires $O(N \ln(N))$ operations. The basic multinomial resampling algorithm needs $O(N^2)$ operations and is therefore undesirable from a computational perspective.

8.4.3.2. *Stratified Resampling*

Stratified resampling also starts from N draws u_s from $U(0, 1)$ and lets $\tilde{u}_s = (s - 1 + u_s)/N$ for $s = 1, \ldots, N$. It thereafter determines indexes j_s as for the multinomial resampling scheme, but makes use of $\tilde{u}_s$ instead of u_s. An advantage of this resampling approach over the multinomial is that it results in a lower variance due to the stratification.[67] Moreover, with $\tilde{u}_{s+1} \geq \tilde{u}_s$ for $s = 1, \ldots, N - 1$, this resampling approach speeds up the computational time compared with basic multinomial resampling since ordering means that $j_{s+1} \geq j_s$ by construction and the resampling scheme therefore needs $O(N)$ operations.

8.4.3.3. *Systematic Resampling*

The systematic resampling approach is similar to the stratified, but instead of obtaining N draws u_s, it relies on a single draw, $u \sim U(0, 1)$, while $\tilde{u}_s = (s - 1 + u)/N$ for $s = 1, \ldots, N$. With these ordered draws, it proceeds to select j_s as under the basic multinomial resampling.

8.4.3.4. *Residual Resampling*

For residual resampling one first allocates $N_s = \lfloor N\tilde{W}_n^{(s)} \rfloor$ copies of index s for $s = 1, \ldots N$. The remaining number of required indexes (the residual), $\bar{N} = N - \sum_{s=1}^{N} N_s$, is obtained by using multinomial, stratified or systematic resampling. YADA uses the stratified resampling scheme to obtain the $\bar{N}$ additional indexes from $\{1, \ldots, N\}$.

8.4.4. *A Generic SMC Algorithm with Data Tempering*

The generic algorithm provided below relies on a number of tuning parameters. Following Durham and Geweke (2014), let $\{\tau_n\}_{n=1}^{N_\tau}$ be a sequence of integers, subject to $\tau_1 = 0$, $\tau_{n-1} < \tau_n$, $\tau_{N_\tau} = T$, which determine the number of data points to use for iteration $n = 1, 2, \ldots, N_\tau$. Furthermore, let $\{\zeta_n\}_{n=1}^{N_\tau}$ be a sequence of tuning parameters for the Markov transition density in the mutation step. An adaptive choice of the latter tuning parameters has already been discussed in Section 8.4.2, while we shall discuss an adaptive scheme for selecting $\{\tau_n\}_{n=1}^{N_\tau}$ towards the end of the current subsection.

The following generic SMC algorithm with data tempering is based on Durham and Geweke (2014), but is a simplification in one aspect. Namely, the algorithm here is based on a single group, while the algorithm in Durham and Geweke (2014) allows for multiple groups.[68]

---

[67] Let n_s be the possible number of times index s is drawn. Under multinomial resampling, $n_s = 0, 1, \ldots, N$ for all s. By restricting the set of values for n_s to lie closer to $N\tilde{W}_n^{(s)}$, the variance is reduced; see, e.g., Douc et al. (2005) for details.

[68] In principle, this is not an important simplification as the number of independent groups may be thought of as the SMC variant of multiple Markov chains for MCMC, but it simplifies the notation since each particle does not have

(1) **Initialization:** Set $n = 1$ such that $\tau_1 = 0$. Draw the initial particles from the prior: $\phi_1^{(s)} \sim p(\phi)$ for $s = 1, \ldots, N$.

(2) **Recursion:** For $n = 2, 3, \ldots, N_\tau$.

    (a) **Correction:** Compute weights that reflect the density of the particles for iteration n:

$$\tilde{w}_n^{(s)} = \frac{p\left(Y_{\tau_n}|\phi_{n-1}^{(s)}\right)}{p\left(Y_{\tau_{n-1}}|\phi_{n-1}^{(s)}\right)} = p\left(y_{\tau_n}, \ldots, y_{\tau_{n-1}+1}|Y_{\tau_{n-1}}, \phi_{n-1}^{(s)}\right)$$

$$= \prod_{t=\tau_{n-1}+1}^{\tau_n} p\left(y_t|Y_{t-1}, \phi_{n-1}^{(s)}\right)$$

    and the normalized weights

$$\tilde{W}_n^{(s)} = \frac{\tilde{w}_n^{(s)}}{(1/N)\sum_{r=1}^{N}\tilde{w}_n^{(r)}}, \quad s = 1, 2, \ldots, N.$$

    (b) **Selection:** Resample the particles via a suitable resampling algorithm; see Section 8.4.3. Let $\{\hat{\phi}_n^{(s)}\}_{s=1}^{N}$ denote N draws from the selected resampling distribution characterized by supporting points and weights $\{\phi_{n-1}^{(s)}, \tilde{W}_n^{(s)}\}_{s=1}^{N}$, and set $W_n^{(s)} = 1$ for $s = 1, \ldots, N$.

    (c) **Mutation:** Propagate the particles $\{\hat{\phi}_n^{(s)}\}_{s=1}^{N}$ via M steps of a Metropolis-Hastings algorithm with transition density

$$\phi_n^{(s)} \sim K_n\left(\phi_n|\hat{\phi}_n^{(s)}; \zeta_n\right),$$

    and stationary distribution $\pi_n(\phi)$.

(3) **End:** For $n = N_\tau$, the final importance sampling approximation of $E_\pi[h(\phi)]$ is given by equation (8.4) with $W_{N_\tau}^{(s)} = 1$ for $s = 1, \ldots, N$.

Notice that resampling always occurs during the selection step of this algorithm. This means that if $\tau_n = n$ such that exactly one data point is added for each iteration, then the number of iterations $N_\tau = T + 1$. An advantage of such a scheme is that the importance distribution $\pi_{n-1}^{(D)}(\phi)$ approximates the target distribution $\pi_n^{(D)}(\phi)$ quite well as the successive posteriors are close to one another. However, resampling increases the variance of the estimates and reduces the number of distinct particles and should therefore only be undertaken when necessary for preventing degeneracy of the particles.

An adaptive procedure for producing τ_n recursively was therefore suggested by Durham and Geweke (2014). Their procedure involves introducing one data point at a time during the correction step until a suitable stopping criterion is satisfied. The effective sample size in equation (8.8) is a useful measure of degeneracy of the particles and τ_n can therefore be selected as the first incremental time period since τ_{n-1} when the ESS_n falls below a value, such as $N/2$.

Like in Section 8.4.1, the marginal likelihood is also for the data tempering SMC algorithm a direct outcome of the correction step. Since the normalized weights at the end of each iteration are unity for all particles, we now find that the marginal likelihood is estimated by equation (8.6) with $W_{n-1}^{(s)} = 1$.

The mutation step can be adapted as in Section 8.4.2 for a fixed number of Metropolis-Hastings steps M. The only necessary change to the mutation step concerns the determination of the acceptance probability in equation (8.10). Specifically, the following expression for the

---

to be linked to a specific group. The use of multiple groups can of course be important in practise as it affects the actual number of particles and the calculation of certain statistics of interest.

$r(\cdot)$ function should instead be utilized:

$$
r\left(\phi_{n,b,i-1}^{(s)}, \varphi_b \middle| \phi_{n,-b,i}^{(s)}, Y, \tau_n\right) = \frac{p\left(Y_{\tau_n} | \varphi_b, \phi_{n,-b,i}^{(s)}\right) p\left(\varphi_b, \phi_{n,-b,i}^{(s)}\right)}{p\left(Y_{\tau_n} | \phi_{n,b,i-1}^{(s)}, \phi_{n,-b,i}^{(s)}\right) p\left(\phi_{n,b,i-1}^{(s)}, \phi_{n,-b,i}^{(s)}\right)}
$$
$$
\times \frac{q\left(\phi_{n,b,i-1}^{(s)} | \varphi_b, \phi_{n,-b,i}^{(s)}, \zeta_n\right)}{q\left(\varphi_b | \phi_{n,b,i-1}^{(s)}, \phi_{n,-b,i}^{(s)}, \zeta_n\right)}.
$$

Durham and Geweke (2014) suggest to use an adaptive scheme for selecting the number of Metropolis-Hastings steps in each iteration of the mutation step, now denoted by M_n. Their idea is the M_n needs to be larger as the effective sample size gets smaller. This type of adaption is an interesting avenue for improving the diversification of particles in the mutation step through "finesse" rather than "brute force", i.e., using an unnecessary large number of fixed steps in each iteration. However, YADA uses brute force in this regard, although the level of such force, M, is determined by the user.

8.5. An Adaptive Importance-Sampling-Weighted EM Algorithm for Posterior Sampling

As mentioned in Section 8.4, importance sampling (IS) is one of the key algorithms for posterior sampling; see Kloek and van Dijk (1978). An advantage with this approach is that the posterior draws are obtained directly from the importance (candidate) density without any dependence to previous or future draws, while the additional computational requirement concerns the weights attached to the draws. The main problem with IS is the determination of a good candidate density. Ideally, it should match the posterior (target) density very well and cover the part of the parameter space which the target density attaches importance to. Once a good candidate density has been located, we may make use of the practical benefits of IS, such as immediate parallelization of the sampler and straightforward calculation of moments and the marginal likelihood from the output of the sampler via Monte Carlo integration; see, e.g., Geweke (1989a,b, 2005).

The SMC approach provides one way of using ideas from IS, although the posterior draws from SMC are not independent. Hoogerheide et al. (2012) suggest an interesting class of adaptive IS-weighted EM algorithms for estimating the candidate density; see also Baştürk, Grassi, Hoogerheide, Opschoor, and van Dijk (2017) and Baştürk, Grassi, Hoogerheide, and van Dijk (2016). The IS-weighted EM algorithm is discussed first, while the robustified algorithm for making use of this procedure when estimating the parameters of a mixture of Student-t density functions is considered next, before we turn to importance sampling.

8.5.1. The IS Weighted EM Algorithm

Let $f(\phi)$ denote the *target density kernel* of the m-dimensional vector ϕ, while $g(\phi)$ is the *candidate density function*. For notational convenience we have dropped the observed data Y from these expressions. The main issue is that we do not know the distribution of the target density and therefore cannot sample directly from it. By contrast, the distribution of the candidate density is known by the investigator and allows for direct sampling. Under IS, the draws from the candidate density are weighted to reflect the relative heighs of the target and candidate densities and the weights therefore act as a correction factor. Specifically,

$$
\tilde{w}(\phi) = \frac{f(\phi)}{g(\phi)}, \tag{8.11}
$$

such that ϕ-values with a high (low) target kernel height receives a high (low) weight for a fixed candidate density height. For a sample of N draws from the candidate density, $\phi^{(s)}$, it is straightforward to compute N weights $\tilde{w}^{(s)}$ from (8.11).[69]

---

[69] It is possible to normalize the weights by, e.g., dividing them with the average weight, with the effect that the sum of the normalized weights equals N. This may be useful in certain cases for numerical reasons, but is in theory not required.

Hoogerheide et al. (2012) suggest to use a mixture of Student-t densities for the candidate density $g(\phi)$. This mixture is given by

$$g(\phi|\zeta) = \sum_{h=1}^{H} \pi_h p_S(\phi|\mu_h, \Sigma_h, d_h), \qquad (8.12)$$

where $\zeta = \{\pi_h, \mu_h, \Sigma_h, d_h\}_{h=1}^{H}$ is such that $\pi_h \geq 0$, $\sum_{h=1}^{H} \pi_h = 1$, while $p_S(\cdot)$ is the multivariate Student-t density; see equation (4.25). The vectors μ_h are location parameters, the scale matrices Σ_h are assumed to be positive definite, and the degrees of freedom parameters d_h are assumed to obey $d_h \geq 1$ by Hoogerheide et al. (2012), while Baştürk et al. (2017) use the condition $d_h \geq 0.01$.[70]

To estimate the parameters ζ for a fixed number of mixture components H, Hoogerheide et al. (2012) use an IS-weighted EM algorithm. The objective function (log-likelihood) is

$$\frac{1}{N} \sum_{s=1}^{N} \tilde{w}^{(s)} \log[g(\phi^{(s)}|\zeta)], \qquad (8.13)$$

where $\phi^{(s)}$ is a draw from a previous candidate density and $\tilde{w}^{(s)}$ the corresponding weight.

Turning to the motivation for using the objective function in (8.13), the underlying objective is to choose ζ such that the candidate density provides a good approximation of the target density, $\tilde{f}(\phi)$. This is achieved by setting ζ equal to the value which minimizes the Kullback-Leibler divergence or cross-entropy distance; see Kullback and Leibler (1951). It is given by

$$\mathcal{D}_1(\tilde{f} \to g) = \int \tilde{f}(\phi) \log\left[\frac{\tilde{f}(\phi)}{g(\phi|\zeta)}\right] d\phi, \qquad (8.14)$$

which is equivalent to minimizing

$$\begin{aligned}
\mathcal{D}_1(f \to g) &= \int f(\phi) \log\left[\frac{f(\phi)}{g(\phi|\zeta)}\right] d\phi, \\
&= \int f(\phi) \log[f(\phi)] d\phi - \int f(\phi) \log[g(\phi|\zeta)] d\phi.
\end{aligned} \qquad (8.15)$$

Since only the second term on the right hand side of (8.15) depends on ζ, minimization of the Kullback-Leibler divergence is equivalent to maximizing

$$\begin{aligned}
\int f(\phi) \log[g(\phi|\zeta)] d\phi &= E_{f(\phi)}\Big[\log[g(\phi|\zeta)]\Big] \\
&= \int g_0(\phi) \frac{f(\phi)}{g_0(\phi)} \log[g(\phi|\zeta)] d\phi \\
&= E_{g_0(\phi)}\left[\frac{f(\phi)}{g_0(\phi)} \log[g(\phi|\zeta)]\right],
\end{aligned} \qquad (8.16)$$

where $g_0(\phi)$ is a candidate density that has been obtained previously. Equation (8.13) is the sample equivalent to the right hand side of the third line in (8.16).

The EM algorithm of Dempster et al. (1977) can next be employed to maximize (8.13) with respect to ζ. This involves first an expectation step with respect to unobserved data, introduced by Hoogerheide et al. (2012) to fascilitate the computational problem through a so-called data-augmented density, and second maximizing the expected log-likelihood. The latent 'data' are given by the scalars $z_h^{(s)}$ and $w_h^{(s)}$ for $h = 1, \ldots, H$, where $z_h^{(s)} = 1$ if $\phi^{(s)}$ is taken from the h:th Student-t density and zero otherwise with $\Pr[z_h^{(s)} = 1] = \pi_h$. The positive variable $w_h^{(s)}$ is obtained from the well-known decomposition of a Student-t into a conditional normal and a marginal inverted Gamma, i.e., $\phi^{(s)}|z_h^{(s)} = 1, w_h^{(s)} \sim N(\mu_h, w_h^{(s)}\Sigma_h)$, while $w_h^{(s)} \sim IG(a, b)$ with parameters $a = d_h/2$ and $b = 2/d_h$ in equation (4.10), or $IG(s, q)$ with $s = 1$ and $q = d_h$

---

[70] The degrees of freedom parameter of a Student-t distribution need not take on integer values only.

in equation (4.11). The Student-t density is then obtained by integrating out $w_h^{(s)}$; see, e.g., Zellner (1971, p. 388).

Conditional on $\phi^{(s)}$ and $\zeta = \hat{\zeta}$ obtained from a previous density (iteration), Hoogerheide et al. (2012) show that the expectation step of the EM algorithm is based on the following calculations:

$$\tilde{z}_h^{(s)} \equiv E\left[z_h^{(s)}\Big|\phi^{(s)}, \hat{\zeta}\right] = \frac{\hat{\pi}_h p_S\left(\phi^{(s)}|\hat{\mu}_h, \hat{\Sigma}_h, \hat{d}_h\right)}{\sum_{j=1}^{H} \hat{\pi}_j p_S\left(\phi^{(s)}|\hat{\mu}_j, \hat{\Sigma}_j, \hat{d}_j\right)}, \tag{8.17}$$

$$\tilde{v}_h^{(s)} \equiv E\left[\frac{z_h^{(s)}}{w_h^{(s)}}\Big|\phi^{(s)}, \hat{\zeta}\right] = \tilde{z}_h^{(s)} \frac{m + \hat{d}_h}{\rho_h^{(s)} + \hat{d}_h}, \tag{8.18}$$

$$\xi_h^{(s)} \equiv E\left[\log\left(w_h^{(s)}\right)\Big|\phi^{(s)}, \hat{\zeta}\right] = \left[\log(\hat{d}_h/2) - \psi(\hat{d}_h/2)\right]\left(1 - \tilde{z}_h^{(s)}\right)$$
$$+ \left[\log\left((\rho_h^{(s)} + \hat{d}_h)/2\right) - \psi\left((m + \hat{d}_h)/2\right)\right]\tilde{z}_h^{(s)}, \tag{8.19}$$

$$\delta_h^{(s)} \equiv E\left[1/w_h^{(s)}\Big|\phi^{(s)}, \hat{\zeta}\right] = \tilde{v}_h^{(s)} + 1 - \tilde{z}_h^{(s)}, \tag{8.20}$$

where

$$\rho_h^{(s)} = \left(\phi^{(s)} - \hat{\mu}_h\right)'\hat{\Sigma}_h^{-1}\left(\phi^{(s)} - \hat{\mu}_h\right),$$

and $\psi(\cdot)$ is the digamma function; see Section 4.2.10. Notice that all parameters $\hat{\pi}_h$, $\hat{\mu}_h$, $\hat{\Sigma}_h$, and $\hat{d}_h$ are elements of $\hat{\zeta}$ for $h = 1, \ldots, H$.

The maximization step determines a new value of ζ, denoted by $\hat{\zeta}$, as follows:

$$\hat{\mu}_h = \frac{\sum_{s=1}^{N} \tilde{w}^{(s)}\tilde{v}_h^{(s)}\phi^{(s)}}{\sum_{s=1}^{N} \tilde{w}^{(s)}\tilde{v}_h^{(s)}}, \tag{8.21}$$

$$\hat{\Sigma}_h = \frac{\sum_{s=1}^{N} \tilde{w}^{(s)}\tilde{v}_h^{(s)}\left(\phi^{(s)} - \hat{\mu}_h\right)\left(\phi^{(s)} - \hat{\mu}_h\right)'}{\sum_{s=1}^{N} \tilde{w}^{(s)}\tilde{z}_h^{(s)}}, \tag{8.22}$$

$$\hat{\pi}_h = \frac{\sum_{s=1}^{N} \tilde{w}^{(s)}\tilde{z}_h^{(s)}}{\sum_{s=1}^{N} \tilde{w}^{(s)}} \tag{8.23}$$

while $\hat{d}_h$ is obtained by solving the first order condition

$$1 - \psi\left(\hat{d}_h/2\right) + \log\left(\hat{d}_h/2\right) - \frac{\sum_{s=1}^{N} \tilde{w}^{(s)}\xi_h^{(s)}}{\sum_{s=1}^{N} \tilde{w}^{(s)}} - \frac{\sum_{s=1}^{N} \tilde{w}^{(s)}\delta_h^{(s)}}{\sum_{s=1}^{N} \tilde{w}^{(s)}} = 0. \tag{8.24}$$

Once all elements of $\hat{\zeta}$ have been computed, it is straightforward to sample new values of ϕ, denoted by $\phi^{(s)}$, from $g(\phi|\hat{\zeta})$ and compute new weights $\tilde{w}^{(s)}$ from equation (8.11).

8.5.2. *The Mixture of t by IS-Weighted EM Algorithm*

In order to utilize the procedure in Section 8.5.1 for the purpose of estimating a candidate density $g(\phi)$, there are two additional issues to consider. First, how can the algorithm be initialized, i.e., the problem of selecting initial values for $\phi^{(s)}$ and $\hat{\zeta}$. Second, the number of mixing components H needs to be determined. Below, we follow the initialization suggestion from Baştürk et al. (2017) since they provide a robustification of the initialization considered by Hoogerheide et al. (2012). Concerning the second issue, both articles suggest the same bottom-up procedure.

8.5.2.1. *Initialization*

STEP 0: The basic mixture of t by IS-weighted EM (MitISEM) algorithm is initialized by obtaining draws, $\phi^{(s)}$, $s = 1, \ldots, N$, from a naive candidate distribution with density $g^{(i)}(\phi)$. This density is constructed as follows:

(a) Simulate candidate draws $\phi^{(s)} \sim p_S(\phi|\mu, \Sigma, d)$ for $s = 1, \ldots, N$, where $\mu = \tilde{\phi}$ is the mode of the target density and where $\Sigma = \tilde{\Sigma}$ is, for example, the inverse of the Hessian evaluated at the mode; see the RWM algorithm in Section 8.1. The parameter d is selected by the user, with default $d = 1$. Compute the weights $\tilde{w}^{(s)}$ from equation (8.11) based on these draws.

(b) The μ and Σ parameters are updated using the IS weighted EM algorithm, keeping the degrees of freedom parameter fixed. This provides the parameters of the naive candidate density.

Notice that (b) is the additional step included by Baştürk et al. (2017) relative to the initialization suggested by Hoogerheide et al. (2012); see also Baştürk et al. (2016).

8.5.2.2. *Adaption*

STEP 1: Steps (1) and (2) in Hoogerheide et al. (2012) (as well as in Baştürk et al., 2017, and Baştürk et al., 2016) are only applied when $H = 1$ and can therefore jointly be regarded as an adaption step to the initialization. We shall therefore treat them here in that way, before we turn to the iterative part of the algorithm.

(a) Estimate the target distribution's mean and covariance matrix using IS with the N draws and weights from $g^{(i)}(\phi)$, where

$$\bar{\phi} = \frac{\sum_{s=1}^{N} \tilde{w}^{(s)} \phi^{(s)}}{\sum_{s=1}^{N} \tilde{w}^{(s)}}, \quad \bar{\Sigma}_{\phi} = \frac{\sum_{s=1}^{N} \tilde{w}^{(s)} (\phi^{(s)} - \bar{\phi})(\phi^{(s)} - \bar{\phi})'}{\sum_{s=1}^{N} \tilde{w}^{(s)}}.$$

These moments are used as location and scale parameters of the adaptive Student-t distribution $g^{(a)}(\phi)$, where the number of degrees of freedom, d_1, remains fixed at the initial value. Simulate candidate draws $\phi^{(s)}$ from this distribution and compute the corresponding IS weights $\tilde{w}^{(s)}$.

(b) Apply the IS-weighted EM algorithm to the draws, weights and parameters ζ obtained from (1a). This yields a new candidate density $g^{(1)}(\phi)$ with optimized $\zeta = \hat{\zeta}$, where $\hat{\pi}_1 = 1$ since $H = 1$. The optimized ζ includes an estimate of the degrees of freedom parameter, d_1. Simulate candidate draws $\phi^{(s)}$ from this distribution and compute new IS weights $\tilde{w}^{(s)}$ from these draws.

8.5.2.3. *Iterate on the Number of Mixture Components*

STEP 2: For $H = 1, 2, \ldots, H^*$ and given the current mixture of H components with $\zeta = \{\mu_h, \Sigma_h, d_h, \pi_h\}_{h=1}^{H}$ and with draws and weights $\phi^{(s)}$ and $\tilde{w}^{(s)}$, take the share α, with $0 < \alpha < 1$, of the sample of draws $\phi^{(s)}$ that correspond to the highest weights. With these draws and weights calculate new parameters μ_{H+1} and Σ_{H+1} using the IS estimates of the mean and covariance, while d_{H+1} and π_{H+1} are selected by the user and where π_h for $h = 1, \ldots, H$ are scaled down proportionally such that $\sum_{h=1}^{H} \pi_h = 1 - \pi_{H+1}$; these two parameters default to $d_{H+1} = 1$ and $\pi_{H+1} = 0.1$. Given the candidate draws $\phi^{(s)}$ and weights $\tilde{w}^{(s)}$, apply the IS-weighted EM algorithm to estimate ζ for $H + 1$ mixing components. Simulate a new sample of draws and weights $\phi^{(s)}, \tilde{w}^{(s)}$ from this density, denoted by $g^{(H+1)}(\phi)$.

8.5.2.4. *Stopping Criterion*

STEP 3: Compute the coefficient of variation (CoV) of the IS weights, i.e., the standard deviation of these weights divided by their mean. Stop the algorithm when this coefficient has converged and otherwise go back to Step 2.

8.5.3. *Discussion of the MitISEM Algorithm*

Hoogerheide et al. (2012) consider the *relative change* of the CoV as their convergence criterion and regard a value less than 10 percent as being below the tolerance level. This is also the default value considered by Baştürk et al. (2017) in their implementation for R of the MitISEM algorithm.

An additional convergence criterion allowed for in YADA is based on the value of the objective function in (8.13). To avoid potential numerical issues, a common normalization of the IS weights can be considered, such as dividing the IS weights $\tilde{w}^{(s)}$ with the sum of the IS weights based on the draws from the optimized candidate density for $H = 1$ in Step 1b, i.e., based on the draws taken from $g^{(1)}(\phi)$. The relative change of the objective function based on this common normalization for $H \geq 1$ is optionally used as an *additional* criterion in YADA. When selected by the user, the algorithm will then also stop if either (i) the objective function has decreased when comparing its value for $H + 1$ and H, with the effect that the number of mixture components H is selected by YADA, or (ii) if a positive value of the relative change less than the tolerance level is recorded, with the consequence that $H + 1$ is selected. One important reason for also using the relative change of the objective function as a convergence criterion is, of course, that this is the function the MitISEM algorithm is supposed to maximize, a property which is not guaranteed to be fulfilled by examining the relative change of the coefficient of variation for the weights.

To robustify Step 2 of the algorithm, Hoogerheide et al. (2012) consider three different values of α, $\alpha = 0.01, 0.05, 0.10$, and select the value for which the resulting mixture of $H + 1$ components has the lowest CoV value. Moreover, if during Step 2 a scale matrix Σ_h becomes (nearly) singular, this component is removed from the mixture for the selected α. Similarly, if a weight π_h is close to zero, this component is also removed. As a consequence of these two checks, the algorithm may in principle converge also for one component even if the criterion based on the normalized objective function in (8.13) is not utilized when checking for convergence.

The MitISEM algorithm benefits from the possibility of parallelizing several parts; see Baştürk et al. (2016) for discussions on this topic in connection with their implementation of the algorithm for the R software. YADA can make use of the parallel computing toolbox and allows for parallel computations where deemed advantageous. First of all, the computation of IS weights is the most costly part for DSGE models since it involves determining the log-likelihood for each parameter value using the Kalman filter. It is straightforward to use parallel calculations for this and a by-product of computing the weights is the determination of the t-densities required by equation (8.17) of the expectation step. Next, all equations of the expectation step (8.17)–(8.20) can be vectorized (expressed as $N \times H$ matrices) once the t-densities are available and therefore do not necessarily benefit from using parallel for-loops. Similarly, equations (8.21) and (8.23) can be expressed as matrices, while this is feasible for each h in equation (8.22), while a parallel for-loop can be used over the number of mixture components H. The last maximization step in (8.24) can be partly vectorized (fourth and fifth term on the left hand side), while solving the first order condition can be done in parallel over the number of mixture components, leading to a gain from parallelization which is proportional to the minimum of H and the number of available CPU (or GPU) cores (workers). Finally, YADA computes draws from the Student-t distribution as matrices such that parallelization over the draws is unlikely to be significantly less time-consuming.

To be specific about how to vectorize the expectation step, let P_S be an $N \times H$ matrix where element (s, h) is given by $p_S(\phi^{(s)}|\hat{\mu}_h, \hat{\Sigma}_h, \hat{d}_h)$. Similarly, let ρ be an $N \times H$ matrix where element (s, h) is given by $\rho_h^{(s)}$. We also collect in $H \times 1$ vectors $\hat{\pi}$ and $\hat{d}$ the estimates of the mixture weights π_h and the degrees of freedom d_h parameters that are used as input for the expectation step. Let $\odot$ and $\oslash$ denote the Hadamard element-by-element product and division, respectively. The matrix version of equation (8.17) is now

$$\tilde{z} = \left[P_S \odot \iota_N \hat{\pi}' \right] \oslash \left[P_S \hat{\pi} \iota_H' \right],$$

an $N \times H$ matrix, where for $H = 1$ we find that $\tilde{z} = \iota_N$. Next, the matrix version of (8.18) is given by

$$\tilde{v} = \left[\tilde{z} \odot (m + \iota_N \hat{d}') \right] \oslash \left[\rho + \iota_N \hat{d}' \right].$$

For equation (8.19) we have that

$$\xi = \left[\iota_N \left(\log(\hat{d}/2) - \psi(\hat{d}/2) \right)' \right] \odot \left(1 - \tilde{z} \right)$$
$$+ \left[\log\left((\rho + \iota_N \hat{d}')/2 \right) - \iota_N \psi\left([m + \hat{d}]/2 \right)' \right] \odot \tilde{z},$$

while equation (8.20) can be expressed as

$$\delta = \tilde{v} + 1 - \tilde{z},$$

completing the vectorization of the expectation step.

Concerning the vectorization of the maximization step, we let $\tilde{w}$ denote the $N \times 1$ vector of IS weights, while column h of, for instance, $\tilde{z}$ is given by $\tilde{z}_h$ and the $m \times N$ matrix Φ has columns $\phi^{(s)}$. We now find that the matrix version of equation (8.21) is given by

$$\hat{\mu} = \left[\Phi\left(\tilde{w}\iota_H' \odot \tilde{v} \right) \right] \oslash \left[\iota_m \iota_N' \left(\tilde{w}\iota_H' \tilde{v} \right) \right].$$

Similarly, the matrix version of the right hand side of equation (8.22) is equal to

$$\hat{\Sigma}_h = \left(1/\iota_N'(\tilde{w} \odot \tilde{z}_h) \right) \left[\left(\Phi - \hat{\mu}_h \iota_N' \right) \odot \iota_m \left(\tilde{w} \odot \tilde{v}_h \right)' \right] \left(\Phi - \hat{\mu}_h \iota_N' \right)'.$$

To vectorize equation (8.23) we have

$$\hat{\pi} = (1/\iota_N' \tilde{w}) \left[\iota_N' \left(\tilde{w}\iota_H' \odot \tilde{z} \right) \right]'.$$

Concerning the use of vectorization for equation (8.24), we note that the sum of the fourth and fifth terms on the left hand side is given by $-\tilde{w}'(\xi_h + \delta_h)/\iota_N' \tilde{w}$.[71]

8.5.4. Importance Sampling

The candidate density produced by the MitISEM algorithm can in principle be used by many posterior sampling algorithms, including Metropolis-Hastings and other MCMC-based algorithms. In YADA, MitISEM is "only" used to estimate a candidate density for importance sampling.

Geweke (2005, Section 4.2.2) provides a number of useful asymptotic results for the importance sampler; see also Geweke (1989a,b). In particular, with $h(\phi)$ being a vector valued function of ϕ, he shows that the Monte Carlo estimate:

$$\bar{h}^{(N)} = \frac{\sum_{s=1}^{N} \tilde{w}^{(s)} h\left(\phi^{(s)} \right)}{\sum_{s=1}^{N} \tilde{w}^{(s)}}, \tag{8.25}$$

converges almost surely to $\bar{h} = E_{\tilde{f}}[h(\phi)]$, subject to suitable regularity conditions. Furthermore, if the IS weights are bounded, then asymptotic normality is also established, allowing for the calculation of numerical standard errors of the estimates. Specifically,

$$\sqrt{N}\left(\bar{h}^{(N)} - \bar{h} \right) \xrightarrow{d} N\left(0, \Omega_h \right),$$

where a consistent estimate of Ω_h is given by

$$\bar{\Omega}_h^{(N)} = \frac{\sum_{s=1}^{N} \tilde{w}^{(s)2} \left(h\left(\phi^{(s)} \right) - \bar{h}^{(N)} \right) \left(h\left(\phi^{(s)} \right) - \bar{h}^{(N)} \right)'}{\left[\sum_{s=1}^{N} \tilde{w}^{(s)} \right]^2}.$$

In addition, Geweke (2005) shows that the marginal likelihood can be consistently estimated from the mean of the IS weights. That is,

$$\hat{p}_{IS}\left(Y \right) = \bar{w}^{(N)} = \frac{1}{N} \sum_{s=1}^{N} \tilde{w}^{(s)}. \tag{8.26}$$

---

[71] It may be noted that equation (8.24) can be fully vectorized and expressed as:

$$1 - \psi(\hat{d}/2) + \log(\hat{d}/2) - (1/\iota_N' \tilde{w})(\xi + \delta)' \tilde{w} = 0.$$

The vector $\hat{d}$, which solves this first order condition, can be determined with the `fsolve` function in matlab. However, this function is only available from the optimization toolbox, and the simpler `fzero` function, which is supplied with matlab, supports scalars but not vectors. For this reason, YADA does not fully vectorize (8.24), but loops over h and vectorizes the fourth and the fifth terms on the left hand side as mentioned above.

Moreover, the numerical standard error of this estimate is given by $\sqrt{\bar{\tau}^{(N)2}/N}$, where

$$\bar{\tau}^{(N)2} = \frac{1}{N} \sum_{s=1}^{N} \left(\tilde{w}^{(s)} - \bar{w}^{(N)} \right)^2,$$

is a consistent estimate of τ^2, the variance in the zero-mean asymptotic normal distribution of $\hat{p}_{IS}(Y)$. It is interesting to note that the CoV convergence measure, used for the stopping criterion of the MitISEM algorithm, is directly related the marginal likelihood estimate under importance sampling and its standard deviation.

Notice also that the calculation of numerical standard errors does not require any autocorrelation correction since the draws from the candidate density are uncorrelated; for details, see Geweke (2005, Section 8.2.2). This also opens up for introducing parallel computations when producing the IS draws from the final candidate density, as well as for all the densities used within the MitISEM algorithm; see Baştürk et al. (2016) for discussions on this topic.

8.6. *Credible Sets and Confidence Intervals*

The draws $\phi^{(s)}$, $s = 1, \ldots, N$ from the RWM, slice sampling, the SMC and IS algorithms are all from the joint posterior density $p(\phi|Y)$, with two of these approaches also involving IS weights. The individual elements of $\phi_i^{(s)}$, $i = 1, \ldots, m$ (with m being the dimension of ϕ) are draws from the marginal posterior density $p(\phi_i|Y)$; see, e.g., Gelfand and Smith (1990).

Let $0 < \alpha < 1$, $p(\phi)$ be a continuous probability density function of the random variable ϕ having support Φ. A $100(1 - \alpha)$ percent credible set (region) $C_\alpha \subseteq \Phi$ is defined such that

$$\int_{C_\alpha} p(\phi)d\phi = 1 - \alpha.$$

The credible set is generally not unique; see, e.g., Bernardo and Smith (2000, Chapter 5.1) or Geweke (2005, Chapter 2.5). For example, we may select $C_\alpha = (\phi_l, \phi_u)$, $\phi_l < \phi_u$ such that $\Pr[\phi < \phi_l] = 1 - \Pr[\phi > \phi_u] = \alpha/2$, i.e., an equal tails credible interval. One advantage of this choice is that it is always unique. At the same time, a disadvantage is that it is generally not the shortest possible credible set in terms of distance from the maximum to the minimum, or even the set with the highest probability density values.

The highest probability density (HPD) credible set is a popular choice in Bayesian inference. Such a set, $C_\alpha^{(HPD)}$, is defined with respect to $p(\phi)$ such that

(i) $\int_{C_\alpha^{(HPD)}} p(\phi)d\phi = 1 - \alpha$; and

(ii) $p(\phi) \geq p(\varphi)$ for all $\phi \in C_\alpha^{(HPD)}$ and $\varphi \notin C_\alpha^{(HPD)}$, except possibly for a subset of Φ having probability zero.

If $p(\phi)$ is unimodal and symmetric, then the HPD credible region is typically equal to the equal tails credible interval. This is, for example, true when $p(\phi)$ is Gaussian. However, when $p(\phi)$ is skewed, then the the equal tails credible interval is not equal to the HPD set. At the same time, while the equal tails credible interval is unique the HPD credible set need not be; for instance, when ϕ is uniformly distributed.

The HPD set may be estimated directly from the posterior draws. Let $\phi_i^{(s)}$ be ordered as $\phi_{i,j}$, where $j = 1, \ldots, N$ and $\phi_{i,j} \leq \phi_{i,j+1}$ for all $j = 1, \ldots, N-1$. Furthermore, let $[(1-\alpha)N]$ denote the closest integer and define the interval $R_j(N) = (\phi_{i,j}, \phi_{i,j+[(1-\alpha)N]})$ for $j = 1, \ldots, N - [(1-\alpha)N]$. Provided that the posterior density of ϕ_i is unimodal and that the sample of draws is ergodic, it is shown by Chen and Shao (1999, Theorem 2) that

$$R_{j^*}(N) = \min_{1 \leq j \leq N - [(1-\alpha)N]} R_j(N), \tag{8.27}$$

converges almost surely to $C_\alpha^{(HPD)}$. Since j^* need not be unique, Chen and Shao suggest using the lowest value of j satisfying (8.27) to obtain a unique estimate.

Furthermore, since the HPD is not unique to transformations of ϕ_i, such as $h(\phi_i)$, one needs to estimate the HPD on the transformation. Moreover, the transformation may also be a function

of ϕ (and the data), HPD's for such transformations should also be estimated directly from the transformed values.

8.7. *YADA Code*

8.7.1. `NeweyWestCovMat`

The function `NeweyWestCovMat` computes the numerical standard error of any sampled real valued vector using the Newey and West (1987) corrected covariance matrix. The only required input argument is `PostSample`, a matrix of dimension $N \times m$, with N being the number of sample points and m the dimension of the vector. The function also accepts the input `BarN`, representing $\bar{N}$ in (8.2). If this input is either missing or assigned an empty value, the function sets $\bar{N} = [N^{(1/2.01)}]$, unless $[N^{(1/2.01)}] < 100$ and $N > 200$ when $\bar{N} = 100$ is selected. This ensures that $\lim_{N \to \infty} \bar{N} = +\infty$, while $\lim_{N \to \infty} \bar{N}^2/N = 0$; see, e.g., Geweke (2005, Theorem 4.7.3). The output is given by `StdErr`, an $m \times m$ covariance matrix such that the numerical standard errors are available as the square root of the diagonal elements.

8.7.2. `DSGERWMPosteriorSampling` & `DSGEFixedBlockingRWMPosteriorSampling` & `DSGERandomBlockingRWMPosteriorSampling`

The functions `DSGERWMPosteriorSampling` and `DSGERWMStudentPosteriorSampling` handle the actual run of the (single block) RWM algorithm with the normal and the Student-t proposal densities, respectively while the functions `DSGEFixedBlockingRWMPosteriorSampling` for the normal proposal and `DSGEFixedBlockingRWMStudentPosteriorSampling` for the Student-t proposal take care of the fixed blocking RWM algorithm for the corresponding proposal densities. Finally, `DSGERandomBlockingRWMPosteriorSampling` handles the random block RWM posterior sampling with a normal proposal, while `DSGERandomBlockingRWMStudentPosteriorSampling` does the same but with a Student-t proposal density. All these functions use the same inputs as the function `PosteriorModeEstimation`, i.e., `DSGEModel`, `CurrINI`, `controls`. One important difference relative to the posterior mode estimation function is that some fields in the `DSGEModel` structure are ignored in favor of values saved to disk while running the posterior mode estimation routine. In particular, YADA stores data about the prior distribution and sample dates information to disk and `DSGERWMPosteriorSampling` makes sure that the same prior and dates are used when sampling from the posterior as when estimating the posterior mode.

Before starting up the RWM algorithm, the functions perform a number of tests. First, they attempt to execute the additional parameter functions that are present. If they run without giving any errors, the measurement equation function is executed next and, thereafter, YADA attempts to solve the DSGE model at the posterior mode estimate. Given that all checks return positive results, the log posterior function `logPosteriorPhiDSGE` is executed for the posterior mode estimate of ϕ.

The RWM algorithm in YADA has been designed to be flexible and to avoid computing things more often than necessary. The *Posterior Sampling* frame on the Options tab is displayed in Figure 4 for MCMC samplers. First, the choice of posterior sampler is provided: (1) the RWM algorithm with a normal proposal, (2) the slice sampler, (3) the RWM with a Student-t proposal, (4) fixed block RWM algorithm with a normal proposal, (5) fixed block RWM algorithm with a Student-t proposal, (6) random block RWM with a normal proposal, (7) random block RWM with a Student-t proposal, (8) Sequential Monte Carlo with likelihood tempering, or (9) Sequential Monte Carlo with data tempering. Notice that the posterior sampling frame on the Options tab changes when the user has selected SMC rather than one of the MCMC algorithms; see Figure 6. We will return to the SMC selections below when discussing the implementations of SMC with likelihood or data tempering in YADA.

For the MCMC algorithms, the number of degrees of freedom is also selected provided the algorithm requires it, the number of fixed blocks for the fourth and fifth algorithm, and the lower and upper number of blocks for the random blocking algorithms. Next, the total number of draws from the posterior per sampling chain can be selected as well as the number of sample

FIGURE 4: The Posterior Sampling frame on the Options tab in YADA for MCMC sampling algorithms.
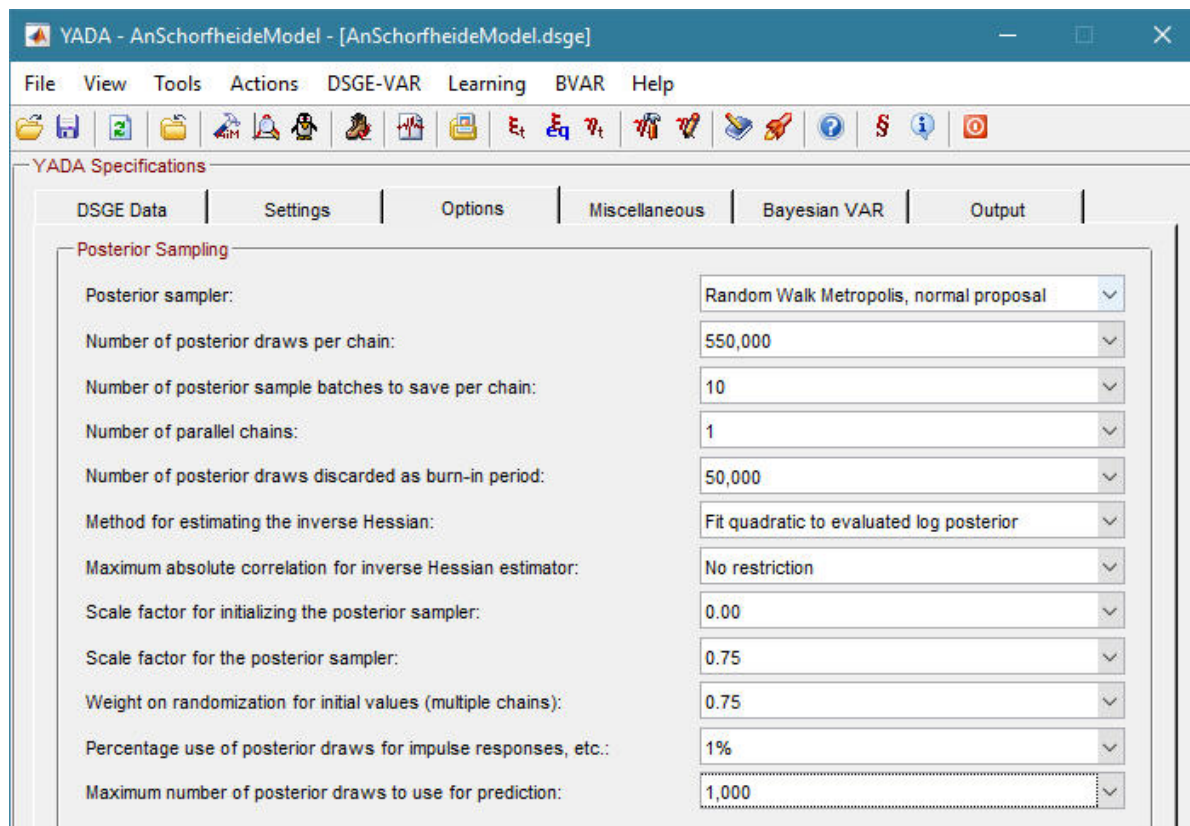
batches per chain. YADA always stores the data to disk when a sample batch has been completed. The number of draws in each batch depends on the total number of draws per chain and the number of sample batches per chain. This allows the user to abort a run and later on restart the posterior sampler from the last saved position.

Furthermore, the number of sampling chains can be selected as well as the length of the burn-in period for the sampler. The draws obtained during the burn-in period are later on discarded from the total number of posterior draws per chain, although they will still be saved to disk.

The selections thereafter turn to the proposal density. First, the method for estimating the inverse Hessian at the posterior mode can be selected. YADA makes use of the output from the selected optimization routine by default. Alternatively, YADA can fit a quadratic to the log posterior kernel evaluated in a symmetric interval around the posterior mode for each parameter and from these estimates construct the diagonal of the inverse Hessian; cf. Section 7.2. In addition, when the option "Transform conditional standard deviations for modified Hessian to marginal using correlations from Hessian" on the *Miscellaneous* tab is check marked, then these estimates are scaled up accordingly. For both possibilities, the correlation structure is thereafter taken from the inverse Hessian that the optimization routine provides. Third, a finite difference estimator can be applied. Here, the step length is determined by the user and this selection is located on the *Miscellaneous* tab. Finally, a user specified parameter covariance matrix for the proposal density is also supported. Such a matrix may, for instance, be estimated using old draws from the posterior distribution. YADA supports this feature from the *View* menu, but any user defined matrix is also allowed provided that it is stored in a mat-file and that the matrix has the name `ParameterCovarianceMatrix`.

The estimator of the inverse Hessian can be influenced through a parameter that determines its maximum absolute correlation. This parameter is by default set to 1 (no restriction), but

values between 0.95 and 0 can also be selected. This parameter interacts with the largest absolute correlation for the inverse Hessian such that the off-diagonal elements of the new inverse Hessian are given by the old off-diagonal elements times the minimum of 1 and the ratio between the desired maximum absolute correlation and the estimated maximum absolute correlation. Hence, if the desired maximum absolute correlation is greater than the estimated maximum absolute correlation then the inverse Hessian is not affected. On the other hand, if the ratio between these correlations is less than unity, then all correlations are scaled towards zero by with the ratio. At the extreme, all correlations can be set to zero by selecting a maximum absolute correlation of zero.

The following two selections concern the c_0 and c scale parameters for the initial density and for the proposal density, respectively. The selection of the c parameter influences greatly the sample acceptance probability. If you consider this probability to be too low or high, then changing c will often help; see, e.g., Adolfson, Lindé, and Villani (2007c). The c_0 parameter gives the user a possibility to influence the initial value $\phi^{(0)}$. For instance, if $c_0 = 0$, then $\phi^{(0)} = \tilde{\phi}$, i.e., the posterior mode.

The next parameter in the *Posterior Sampling* frame is only used under multiple sampling chains. The weight on randomization refers to the weight given to a randomly drawn ϕ, determined as in the case of the single chain but with $c_0 = 4$, relative to the posterior mode when setting up $\phi^{(0)}$. Hence, if the weight on randomization is 1, then each sampling chain uses $\phi^{(0)}$ from $N_m(\tilde{\phi}, 16\tilde{\Sigma})$ and if the weight on randomization is 0, then each sampling chain starts from the posterior mode. This means that the weight on randomization is identical to c_0 except that it is restricted to the 0-4 interval. Since multiple chains are used to check, for instance, convergence related issues, it is not recommended to start all chains from the posterior mode.

The following parameter on this frame is the percentage use of posterior draws for impulse responses, variance decompositions, etc. It allows the user to apply a fraction of the available posterior draws when computing the posterior distribution of such a function. The case of less than 100 percent may be particularly relevant when preliminary results are desired or when the user is mainly concerned with point estimates, such as the posterior mean. Among the stored posterior draws the parameter values used are obtained by either using a fixed interval (default) or by drawing randomly from the available draws using a uniform distribution. The length of the fixed interval is the maximum possible while ensuring that the desired share of parameter draws is feasible. The option "Randomize draws from posterior distributions" on the *Miscellaneous* tab determines which case is selected.

The final parameter on the Posterior Sampling frame determines the maximum number of draws from the posterior that will be used in prediction exercises. When comparing the number of posterior draws minus the length of the burn-in period to the desired maximum number of draws to use in such exercises YADA selects the smallest of these two numbers. As above for the less than 100 percent share of the draws case, if fewer than the maximum available are used, the user can choose between the largest fixed interval between draws (default) or uniform draws from the posterior draws.

One additional user determined parameter influences how the RWM algorithm is executed. This parameter in located on the *DSGE Posterior Sampling* frame on the Settings tab; see Figure 5. If the checkbox *Overwrite old draws* is check marked, then previous draws will be overwritten. Conversely, when this box is not checked, then old draws will be used when available. This allows the user to recover from a previously aborted run of the RWM algorithm provided that the number of sample batches is greater than 1 and that at least one batch was saved to disk.

There is also another case when YADA can recover previously saved posterior draws. Given that the checkbox *Overwrite old draws* is check marked and the posterior draws are only obtained from one sampling chain, YADA will check if, for your current selection about the number of sample batches, posterior draws exist on disk such that the number of posterior draws is lower than what you have currently selected. For instance, suppose you have selected to save 10 sample batches per chain and that you currently consider 100,000 posterior draws. YADA

FIGURE 5: The DSGE Posterior Sampling frame on the Settings tab in YADA.
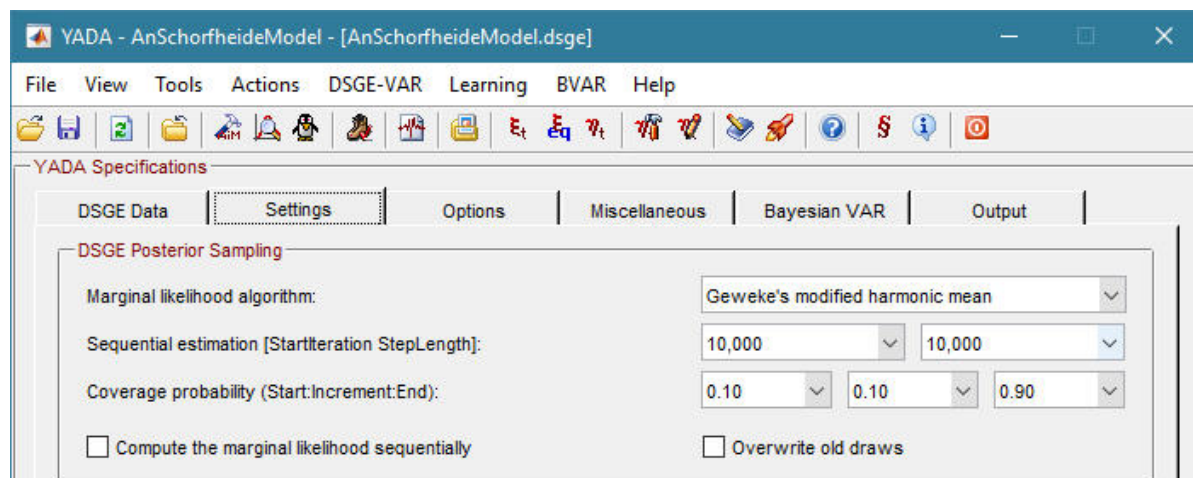
will then check if you have previous run the posterior sampler with less than 100,000 draws for 10 sample batches. The highest such number of posterior draws will then be considered as a candidate. Supposing that you have already run the posterior sampler for 75,000 draws with 10 sample batches, YADA will ask you if you would like to make use of these 75,000 draws.

When the RWM algorithm has finished, YADA first allows for (but does not force) sequential estimation of the marginal likelihood. The alternative estimators are discussed in Section 10. The choice of algorithm is stated on the *DSGE Posterior Sampling* frame, where certain other parameters that are needed by the marginal likelihood estimation function can also be selected.

Before one of the posterior sampling functions has completed its mission it sends the results to a function that writes a summary of them to file. This file is finally displayed for the user.

8.7.3. DSGESlicePosteriorSampling

The function DSGESlicePosteriorSampling handles the actual run of the slice sampler. It uses the same inputs as DSGERWMPosteriorSampling, i.e., DSGEModel, CurrINI, controls. Moreover, it behaves in essentially the same way as the RWM function with the exception of the actual sampling part. Moreover, the RWM algorithm keeps track of the acceptance rate, while the slice sampler counts the number of times the log posterior is computed. In all other respects, the functions perform the same tasks except that the slice sampler will never call the Chib and Jeliazkov (2001) marginal likelihood estimator function; see Section 10.4.4.

8.7.4. ExponentialRndFcn

The function ExponentialRndFcn computes random draws from an exponential distribution. The function takes 1 required input variables, mu, the mean of the distribution. An optional input variable for total number of draws, NumDraws, is also accepted. The default value for this integer is 1. The algorithm used is discussed in footnote 64 on page 128.

The function provides one output variable, z, a matrix with row dimension equal to the length of mu and column dimension equal to NumDraws.

8.7.5. DSGESMCLikelihoodTemperingPosteriorSampler & DSGESMCDataTemperingPosteriorSampler

The function DSGESMCLikelihoodTemperingPosteriorSampler handles the execution of the sequential Monte Carlo with likelihood tempering posterior sampler, while the data tempering sampler is handled by DSGESMCDataTemperingPosteriorSampler. These two functions use the same inputs as DSGERWMPosteriorSampling, i.e., DSGEModel, CurrINI, controls. As mentioned above, the *Posterior Sampling* frame has different controls under the SMC sampler than under the MCMC samplers; see Figure 6.

FIGURE 6: The Posterior Sampling frame on the Options tab in YADA for SMC sampling algo-
rithms.
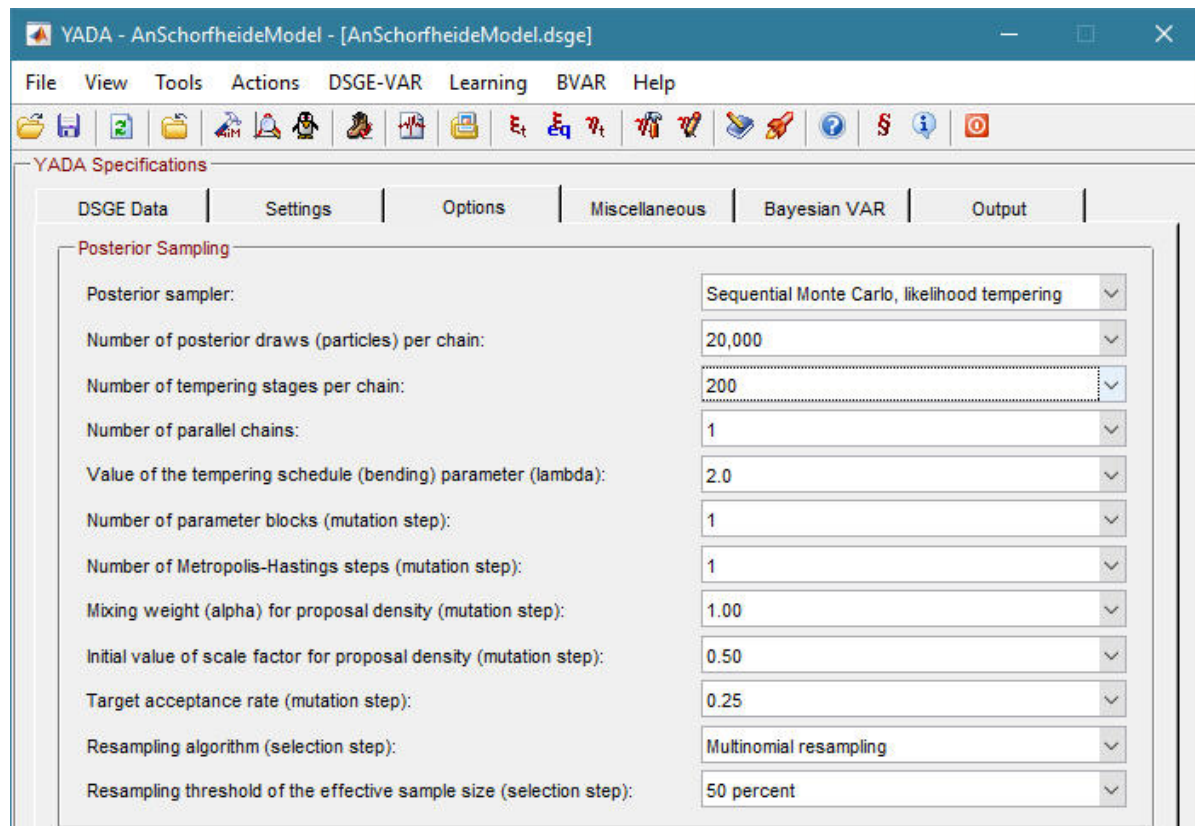
The number of posterior draws is the same as for the MCMC samplers, but the text part now makes clear that this number is also the number of particles. Next, the number of temper-ing stages can be selected under likelihood tempering, while data tempering uses an adaptive scheme for determining this number, which is given by N_τ in Section 8.4. YADA allows for inte-ger values between 100 and 10,000, with default equal to 100. The number of chains controls appears thereafter and has the same functionality as for the MCMC samplers. The value of the tempering schedule parameters, or bending parameter, denoted by λ, can be selected on the following control provided that the selected SMC algorithm is with likelihood tempering. The default value is 2, but values from 1 to 4 are also allowed for.

The following five controls make it possible to tune the mutation step of the SMC algorithm. First of all, the number of fixed parameter blocks, N_B, can be selected where integer values between 1 and 100 are possible, and its default value is 1. If this value exceeds the dimension of ϕ, then YADA sets N_B equal to the length of the parameter vector. The number of Metropolis-Hastings steps, M, can be determined from 1 to 100, with default being 1. Next, the mixing weight α for constructing the proposal density is selected by the user. Value between 0.60 and 1.0 are supported, where the default is 1.

The fourth tuning parameter for the mutation step is the initial value of the scale factor, denoted by c^*. It is possible to set this parameter to values from 0.05 to 10 with the default value of 0.30. Finally, the last tuning parameter of the mutation step that the user can determine is the target acceptance rate. It is here possible to select values between 0.20 and 0.50, with the default value being 0.25, i.e., a target acceptance rate of 25 percent.

The second to last control on this frame that the user can work with is the resampling al-gorithm for the selection step. When the effective sample size is below the threshold (N/k), the multinomial, stratified, systematic, or residual resampling algorithm is used. The default

choice is multinomial resampling. Note that resampling always occurs under the data tempering algorithm as discussed above in Section 8.4.4.

Finally, the resampling threshold for the effective sample size can be selected, with values between 10 and 90 percent of the number of posterior draws (particles, N) being accepted. A value of $N/2$, as in Herbst and Schorfheide (2014, 2016), corresponds to 50 percent and is the default value in YADA. Note that under data tempering, this control is renamed "Recursion threshold of the effective sample size" and is used in the correction step to determine how many new data points should be added for the current recursion.

8.7.6. `MultinomialResampling`

The function `MultinomialResampling` determines the resampling of the particles based on draws from a multinomial distribution during the selection step of the SMC with likelihood or data tempering algorithm. It takes one required input given by the vector `Weights` and one optional integer input `N`. The latter integer is by default equal to the lenght of `Weights` and determines the length of the output vector `Indexes`. This latter vector gives the index positions to use for the resampling scheme of the parameter vectors as well as of the functions that depend on the selected parameter vectors, such as the log-likelihood and the log prior values.

8.7.7. `StratifiedResampling`

The function `StratifiedResampling` determines the index positions of the resampled particles during the selection step based on stratified resampling. It takes the same inputs and provides the same output type as the `MultinomialResampling` function.

8.7.8. `SystematicResampling`

The function `SystematicResampling` determines the index positions of the resampled particles during the selection step based on systematic resampling. It takes the same inputs and provides the same output type as the `MultinomialResampling` function.

8.7.9. `ResidualResampling`

The function `ResidualResampling` determines the index positions of the resampled particles during the selection step based on residual resampling. It takes the same inputs and provides the same output type as the `MultinomialResampling` function.

8.7.10. `MixedDistributionDraw`

The function `MixedDistributionDraw` provides a draw from the proposal density during the Metropolis-Hastings step of the mutations. The function takes seven required input variable: `phi`, `phiTilde`, `CholSigma`, `CholDiagSigma`, `alpha`, `cstar`, and `NumParam`. These input variables correspond in equation (8.9) to the parameters: $\phi_{n,b,i-1}^{(s)}$, $\phi_{n,b}^*$, the Choleski decomposition of $\Sigma_{n,b}^*$, the Choleski decomposition of $\mathrm{diag}(\Sigma_{n,b}^*)$, the mixing weight α, the scale factor c_n, and the dimension of the parameter block b, respectively.

As output it gives `phiProp`, the proposal draw φ_b from the mixed normal distribution.

8.7.11. `DSGEISMitISEMPosteriorSampler`

The function `DSGEISMitISEMPosteriorSampler` handles running the importance sampler (IS) based on the MitISEM algorithm discussed in Section 8.5. It uses the same input variables as the function `DSGERWMPosteriorSampling`, i.e., `DSGEModel`, `CurrINI`, and `controls`. The *Posterior Sampling* frame has a few different controls under IS based on MitISEM than under the MCMC or SMC samplers; see Figure 7.

The number of posterior draws per chain (or sample) has the same functionality as for the MCMC and SMC samplers, but these draws are now taken independently from the candidate density. The number of sample batches is disabled since the draws are taken independently and there is no reason for storing draws sequentially. The number of parallel chains (or samples) serves the same functionality as for the MCMC and SMC samplers, while the tolerance level for

Figure 7: The Posterior Sampling frame on the Options tab in YADA for IS based on the MitISEM algorithms.
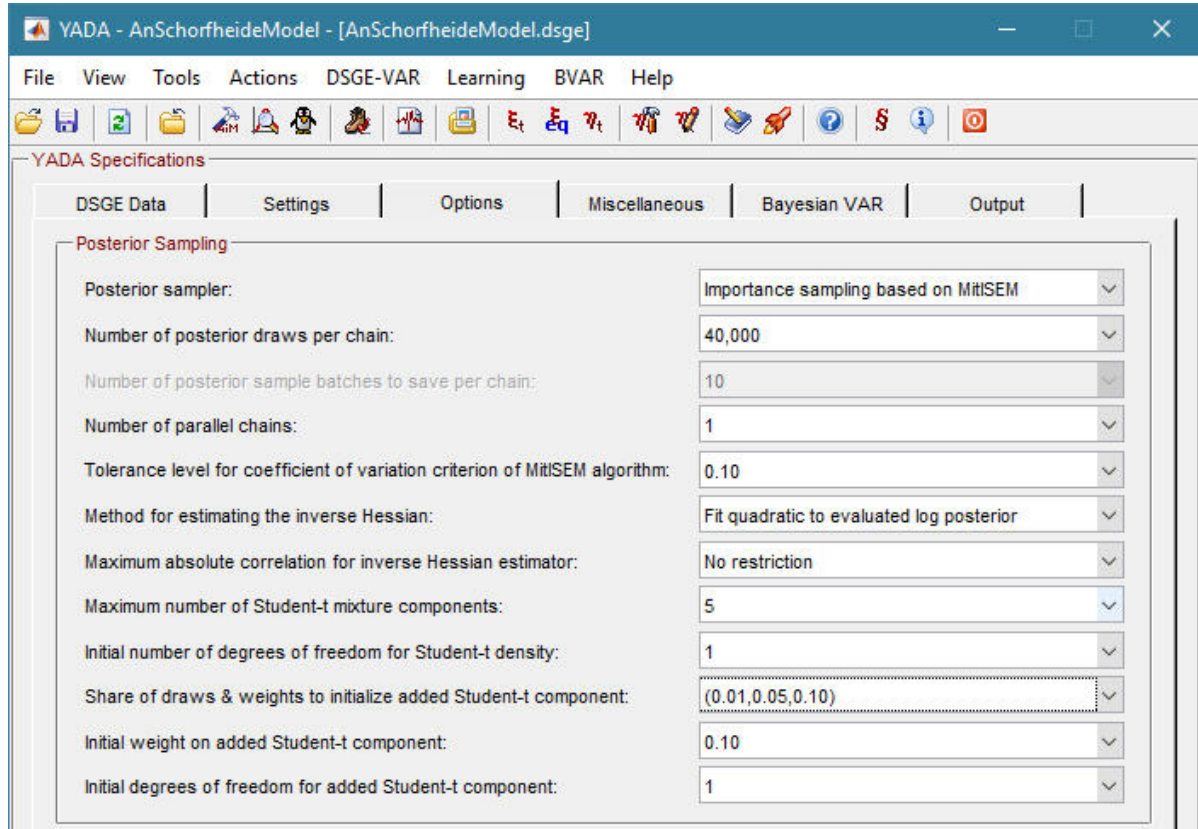
coefficient of variation criterion of MitISEM algorithm checks if the relative change of the ratio between the standard deviation of the IS weights and the mean is such that the estimation of the candidate density has converged. The following two options are shared with the MCMC samplers and are used to initialize MitISEM in Step 0 with respect to $\tilde{\Sigma}$.

The maximum number of Student-t mixture components determined H^* in Step 2, while the initial number of degrees of freedom for Student-t density as the following tuning parameter is used in Step 0 and 1b. The share of draws and weights to initialize added Student-t component that comes thereafter determines the parameter α in Step 2, where multiple values are allowed for and the best α value is selected based on the lowest coefficient of variation, as in for instance Hoogerheide et al. (2012).

The tuning parameter initial weight on added Student-t component determines the initial value of π_{H+1} in Step 2, while initial degrees of freedom for added Student-t component sets the initial value for d_{H+1} in the same step of the algorithm.

It may be noted that when the user opts to also make use of the additional converges criterion based on the relative change of the normalized objective function in (8.13), the tolerance level is given by value set in the Optimization frame on the Options tab of YADA, directly below the Posterior Sampling frame on the same tab.

9. Markov Chain Monte Carlo Convergence

This section is concerned with assessing the convergence of the MCMC sampler. A fundamental problem of inference from Markov chain simulation is that there will always be areas of the posterior (target) distribution that have not been covered by the finite chain. As the simulation progresses the ergodic property of the Markov chain causes it eventually to cover all the target distribution but, in the short run, the simulations cannot, generally, tell us about areas where they have not been. As pointed out by, e.g., Gelman (1996), this is a general problem whenever convergence is slow, even in a distribution that has a single mode; see, also, Gelman and Rubin (1992).

This section is divided into two parts. The first deals with tools for evaluating convergence using a single chain, while the following is concerned with multiple chains. For a review of various tools for evaluating MCMC convergence see, e.g., Cowles and Carlin (1996).

9.1. *Single Chain Convergence Statistics*

The simplest tool for assessing if a MCMC chain has converged or not is to view graphs of the raw draws; see, e.g., Adolfson, Laséen, Lindé, and Villani (2008b). If the draws are trending this is a strong indication that the sampler has not converged. In fact, the reason for the trending may be that the DSGE model is misspecified. The raw draws may also be graphed directly from multiple chains to check if they cover the same region of the parameter space.

Next, sequential estimates of various location parameters may be examined. YADA allows for sequential estimation of the posterior mean and the posterior median. If the sequential estimates are trending this is indicative of poor convergence. Furthermore, sequential estimates of the marginal likelihood may also be studied; see Section 10.

A common tool in time series analysis for studying parameter stability is the partial sum or cusum. Yu and Mykland (1998) proposed to use this tool for evaluating convergence. Like the sequential estimates of the posterior mean, median, and marginal likelihood, the cusum estimates rely on first determining a burn-in period. Let $S(X)$ be a chosen summary statistic of the N post burn-in posterior draws. With $\hat{\mu}$ being the average of $S(X)$ for all N draws, the observed cusum is

$$\hat{C}_i = \sum_{j=1}^{i} \big(S(X_j) - \hat{\mu}\big), \quad i = 1, \dots, N. \tag{9.1}$$

The cusum path plot is obtained by plotting $\{\hat{C}_i\}$ against $i = 1, \dots, N$. If N is very large it may be practical to plot the statistic against, say, $i = N_0, 2N_0, \dots, N$ instead for some suitable integer N_0.

The cusum statistic in (9.1) is zero for $i = N$. In YADA the value of $\hat{\mu}$ is added to $\hat{C}_i$ and the summary statistics are either the log posterior ($\ln L(Y; g^{-1}(\phi)) + \ln p(g^{-1}(\phi)) + \ln J(\phi)$), the original parameters (θ), or the transformed parameters (ϕ). Moreover, YADA calculates moving window cusum paths for a fixed window size of $N_1 = N/10$, i.e.,

$$\bar{C}_i = \sum_{j=i+1-N_1}^{i} \big(S(X_j) - \hat{\mu}\big), \quad i = N_1, \dots, N, \tag{9.2}$$

where, again, $\hat{\mu}$ is added to $\bar{C}_i$.

A separated partial means test for a single MCMC chain has been suggested by Geweke; see, e.g., Geweke (2005, Theorem 4.7.4). Let N be the number of draws and suppose that $N_p = N/2p$ and p are positive integers. For instance, with $N = 10,000$ and $p = 5$ we have that $N_5 = 1,000$. Define the p separated partial means:

$$\hat{S}_{j,p}^{(N)} = \frac{1}{N_p} \sum_{m=1}^{N_p} S\big(\phi^{(m+N_p(2j-1))}\big), \quad j = 1, \dots, p, \tag{9.3}$$

where S is some summary statistic of the transformed parameters ϕ (such as the original parameters θ). Let $\hat{\tau}_{j,p}$ be the Newey and West (1987) numerical standard error for $j = 1, \dots, p$.

Define the $(p-1)$ vector $\hat{S}_p^{(N)}$ with typical element $\hat{S}_{j+1,p}^{(N)} - \hat{S}_{j,p}^{(N)}$ and the $(p-1) \times (p-1)$ tridiagonal matrix $\hat{V}_p^{(N)}$ where

$$\hat{V}_{j,j}^{(N)} = \hat{\tau}_{j,p}^2 + \hat{\tau}_{j+1,p}^2, \quad j = 1, \ldots, p-1$$

and

$$\hat{V}_{j,j+1}^{(N)} = \hat{V}_{j+1,j}^{(N)} = \hat{\tau}_{j+1,p}^2, \quad j = 1, \ldots, p-1.$$

The statistic

$$G_p^{(N)} = \hat{S}_p^{(N)\prime} \left[\hat{V}_p^{(N)} \right]^{-1} \hat{S}_p^{(N)} \xrightarrow{\text{d}} \chi^2(p-1), \tag{9.4}$$

as $N \to \infty$ under the hypothesis that the MCMC chain has converged with the separated partial means being equal.

9.2. Multiple Chain Convergence Statistics

One approach for monitoring convergence using draws from multiple MCMC chains is to use analysis of variance. The approach outlined below is based on Brooks and Gelman (1998), which generalizes the ideas in Gelman and Rubin (1992); see also Gelman (1996) and Gelman et al. (2004).

For a univariate scalar summary S we may assume that we have N draws from M chains. Let S_{ij} denote draw i from chain j. We may then define the average of S for chain j as $\bar{S}_j = (1/N) \sum_{i=1}^N S_{ij}$, while the overall average is $\bar{S} = (1/M) \sum_{j=1}^M \bar{S}_j$. The between-chain variance B and the within-chain variance W are now given by

$$B = \frac{N}{M-1} \sum_{j=1}^M \left(\bar{S}_j - \bar{S} \right)^2, \tag{9.5}$$

and

$$W = \frac{1}{M(N-1)} \sum_{j=1}^M \sum_{i=1}^N \left(S_{ij} - \bar{S}_j \right)^2. \tag{9.6}$$

The between-chain variance B contains a factor of N because it is based on the variance of the within-chain means, $\bar{S}_j$, each of which is an average of N draws S_{ij}.

From the two variance components two estimates of the variance of S in the target distribution, Σ_S, can be constructed. First

$$\hat{\Sigma}_S = \frac{N-1}{N} W + \frac{1}{N} B, \tag{9.7}$$

is an unbiased estimate of the variance under the assumption of stationarity, i.e., when the starting points of the posterior draws are actually draws from the target distribution. Under the more realistic assumption that the starting points are overdispersed, then (9.7) is an *overestimate* of the variance of S.

Second, for any finite N, the within-chain variance W in (9.6) should *underestimate* the variance of S. In the limit as $N \to \infty$, both $\hat{\Sigma}_S$ and W approach Σ_S, but from opposite directions. Accounting for sampling variability of the estimator $\bar{S}$ yields a pooled posterior variance estimate of $\hat{V} = \hat{\Sigma}_S + B/(MN)$.

To monitor convergence of the posterior simulation Gelman and Rubin (1992) therefore suggest estimating the ratio of the upper and the lower bounds of the variance of S through:

$$\hat{R} = \sqrt{\frac{\hat{V}}{W}} = \sqrt{\frac{N-1}{N} + \frac{(M+1)B}{MNW}} = \sqrt{\frac{(M+1)}{M} \frac{\hat{\Sigma}_S}{W} - \frac{N-1}{MN}}. \tag{9.8}$$

As the simulation converges, the *potential scale reduction factor* in (9.8) declines to 1. This means that the M parallel Markov chains are essentially overlapping.

The scalar summary S can here be individual parameters of the DSGE model. A multivariate version of the potential scale reduction factor is suggested by Brooks and Gelman (1998). Now S is, e.g., a vector of all the model parameters, with B and W being covariance matrix estimators of the between-chain and within-chain covariance. The multivariate potential scale reduction

factor (MPSRF) is now:

$$\hat{R} = \sqrt{\frac{N-1}{N} + \frac{M+1}{M}\lambda_1}, \qquad (9.9)$$

where λ_1 is the largest eigenvalue of the positive definite matrix $(1/N)W^{-1}B$; see Brooks and Gelman (1998, Lemma 2). The MPSRF in (9.9) declines towards 1 as the simulation converges. Gelman et al. (2004) suggests that values for $\hat{R}$ less than 1.1 may be regarded as an indication that the MCMC sampler has converged.

In addition to monitoring the MPSRF in (9.9), Brooks and Gelman (1998, p. 447) suggest to monitor the determinants of W and $\hat{V}$. This allows the user to also check if both the within-chain covariance matrix W and the pooled covariance matrix $\hat{V}$ stabilize as functions of N. Since the determinant can be a very small number in models with a large number of parameters and where some are highly correlated, YADA instead computes the trace.

9.3. *YADA Code*

9.3.1. CUSUM

The function CUSUM computes the cusum paths in equation (9.1) and (9.2) for the values of the log posterior from the MCMC output, the draws of the original parameters (θ), or of the transformed parameters (ϕ). The function takes 4 input variables: X, NumBurnin, PlotType, and CurrINI. The matrix X has dimension NumIter × NumStat, where NumIter is the total number of draws from the posterior, and NumStat is the number of summary statistics. The integer NumBurnin gives the number of burn-in draws from the MCMC chain, while the string vector PlotType can be either log posterior, original parameters, or transformed parameters. Finally, the structure CurrINI contains initialization information.

The function provides 2 output arguments: CUSUMPost and CUSUMAverage. The former has dimension (NumIter − NumBurnin) × NumStat and holds the cusum statistic in (9.1) plus the mean of the input statistics over the post-burn-in sample. The second output variables has dimension (0.9NumIter − NumBurnin) × NumStat with the moving window cusum statistic in (9.2).

9.3.2. SeparatedPartialMeansTest

The function SeparatedPartialMeansTest calculates the separated partial means test in (9.4). The function needs 4 input variables: PostSample, p, PlotType, and CurrINI. The matrix PostSample has dimension NumIter × NumParam, where NumIter is now the number of post-burn-in period draws from the posterior while NumParam is the number of summary statistics. The integer p gives the number of separated partial means to evaluate. The last two input arguments are the same as in the case of the CUSUM function.

As output the function gives a matrix GewekeStat that has dimension NumParam × 2. The first column holds the values for the separated partial means test for the NumParam summary statistics, and the second column the asymptotic p-values based on applying the $\chi^2(p-1)$ distribution.

9.3.3. MultiANOVA

The function MultiANOVA computes the multivariate potential scale reduction factor in (9.9) as well as the determinants of $\hat{V}$ and W sequentially. The function takes 5 input variables: NumChains, ComputeSequential, DSGEModel, CurrINI, and controls. The last three input variables are also used by the functions for estimating the posterior mode and running the random walk Metropolis algorithm. The first input simply gives the number of parallel MCMC chains to examine (M in Section 9.2), while the second variables is boolean and takes on the value 1 if the output should be calculated sequentially and 0 otherwise. The output statistics are calculated from posterior draws of the original parameters θ, which are loaded from disk, one chain at a time.

As output the function gives two variables: DoneCalc and MPSRF. The first is a boolean variable which indicates if the calculations were finished (1) or not (0). The matrix MPSRF has 4

columns, where the first holds the number of draws used in the calculation, the second the $\hat{R}$ value, the third the determinant of $\hat{V}$, while the fourth holds the determinant of W. The rows correspond to the sample sizes used for the sequential estimates, as can be read from the first column.

10. Computing the Marginal Likelihood

The Bayesian approach to model comparisons is based on the posterior odds ratio. This criterion can be applied regardless of whether the "true" model is included in the set of models or not. The posterior odds ratio between two models, denoted by m_1 and m_2, is given by

$$\frac{\Pr(m_1|Y)}{\Pr(m_2|Y)} = \frac{p(Y|m_1)}{p(Y|m_2)} \times \frac{\Pr(m_1)}{\Pr(m_2)}.$$

The posterior odds ratio on the left hand side is therefore equal to the Bayes factor, being the ratio of marginal likelihoods, times the prior odds ratio; see Kass and Raftery (1995) for a survey on model comparisons based on posterior probabilities. If the models are given equal prior probabilities then the posterior odds ratio is simple equal to the Bayes factor.

The posterior model probabilities can be calculated for all models m_i, $i = 1, \ldots, M$, once the marginal likelihood has been evaluated for all M models. In that case, the posterior model probabilities are given by

$$\Pr(m_i|Y) = \frac{p(Y|m_i)\Pr(m_i)}{\sum_{j=1}^{M} p(Y|m_j)\Pr(m_j)}.$$

If the "true" model is among the models being studied, then the posterior model probabilities can be used to choose a preferred specification. One simple selection criterion is to choose the model which has highest posterior probability. In fact, even in situations where *all* the models being evaluated are misspecified, this criterion may be used. For instance, Phillips (1996) and Fernández-Villaverde and Rubio-Ramírez (2004), show that posterior odds asymptotically favor the model that is closest to the "true" data generating process in the Kullback-Leibler sense; see also Gelfand and Dey (1994).

In order to compare models with posterior odds the marginal likelihood, $p(Y|m_i)$, needs to be computed. The four methods that are supported in YADA for computing the value of this density will be discussed next. One of them is an approximation method, called the Laplace approximation, which only requires having estimates of the model parameters at the posterior mode, while the other three are based on draws from the posterior distribution. These sampling based methods differ in that two of them can be used for different MCMC methods, while the third is specifically constructed for Metropolis-Hastings methods.

The discussion below is based on the assumption that the model does not have a system prior. In the event that the model indeed has such a prior, then the estimators below need to be adjusted by the normalizing constant of the system prior, i.e., by $p(\Phi_\omega|h)$. Using the terms in equation (4.37), the posterior density can be decomposed as

$$p(\theta|Y, \Phi_\omega, h) = \frac{p(Y|\theta)p(\Phi_\omega|\theta, h)p(\theta)}{p(Y|\Phi_\omega, h)p(\Phi_\omega|h)}. \tag{10.1}$$

The numerator on the right hand side represents the posterior kernel and is given by the product of the likelihood function, the system prior $p(\Phi_\omega|\theta, h)$, and the marginal prior of the parameters. The first term in the denominator, $p(Y|\Phi_\omega, h)$, is the marginal density of the data, i.e. the marginal likelihood we are interested in. The second term in the denominator is the marginal likelihood of the system prior, which typically is not known but needs to be estimated as well. This can be achieved by replacing the likelihood function in the expressions below by the conditional system prior $p(\Phi_\omega|\theta, h)$, and then proceed with the estimators below.[72]

When estimating the marginal likelihood of the data using the posterior kernel (numerator) in (10.1), the resulting marginal likelihood gives an estimate of $p(Y, \Phi_\omega|h)$, the denominator of this equation. To obtain an estimate of what is often be the object of interest, $p(Y|\Phi_\omega, h)$, we then need to divide its value by the marginal likelihood of the system prior. Except for the Laplace approximation, this step is not performed by YADA.

---

[72] In practise, the marginal likelihood of the system prior is estimated with good precision using the Laplace approximation based on a finite difference estimator of the inverse Hessian.

10.1. *The Laplace Approximation*

The Laplace approximation (or the Laplace-Metropolis estimator when posterior draws are used to estimate the quantities it needs) of the log marginal likelihood, $\ln p(Y)$, was originally suggested by Tierney and Kadane (1986); see, also, Raftery (1996) for discussions. It requires an estimate of the posterior mode of the parameters and of the inverse Hessian at the mode. The Laplace estimator of the marginal likelihood makes use of a normal approximation and YADA considers only the transformed parameters ϕ. With the inverse Hessian evaluated at the mode being given by $\tilde{\Sigma}$ and $\tilde{\phi}$ being the posterior mode of the transformed parameters, the Laplace approximation of the log marginal likelihood is given by:

$$\ln \hat{p}_L(Y) = \ln L\big(Y; g^{-1}(\tilde{\phi})\big) + \ln p\big(g^{-1}(\tilde{\phi})\big) + \ln J\big(\tilde{\phi}\big) + \frac{m\ln(2\pi) + \ln\big|\tilde{\Sigma}\big|}{2}, \qquad (10.2)$$

where m is the dimension of ϕ. The third term on the right hand side approximates $-\ln p(\tilde{\phi}|Y)$ with $O(T^{-1})$ accuracy and, hence, the expression in (10.2) is a reflection of Bayes theorem through what Chib (1995) calls the basic marginal likelihood identify.

Notice that since YADA calculates the posterior mode by minimizing minus the log of the posterior, the inverse Hessian does not need to be multiplied by minus 1 in (10.2). For the Bayes factor, i.e., the ratio of the marginal likelihoods of two models, the relative error in (10.2) was shown by Tierney and Kadane (1986) to be $O(T^{-1})$ for regular statistical models.

It is also interesting to note that if the inverse Hessian of the log posterior for the θ parameter is approximated through the delta method, then the Laplace approximation of the log marginal likelihood using $\bar{\theta} = g^{-1}(\tilde{\phi})$ is identical to the expression on the right hand side of (10.2). This follows directly from noting that the determinant of the inverse Hessian is now $J(\tilde{\phi})^2|\tilde{\Sigma}|$, while the log posterior is $\ln L(Y; \bar{\theta}) + \ln p(\bar{\theta})$.

10.2. *Modified Harmonic Mean Estimators*

Harmonic mean estimators are based on the identity:

$$p(Y) = \left[\int \frac{f(\theta)}{p(Y|\theta)p(\theta)} p(\theta|Y)d\theta\right]^{-1}, \qquad (10.3)$$

where $f(\theta)$ is a proper probability density function such that $\int f(\theta)d\theta = 1$; see Gelfand and Dey (1994).[73] Given a choice for $f(\theta)$, the marginal likelihood $p(Y)$ can then be estimated using:

$$\hat{p}_H(Y) = \left[\frac{1}{N}\sum_{s=1}^{N}\frac{f\big(\theta^{(s)}\big)}{L\big(Y; \theta^{(s)}\big)p\big(\theta^{(s)}\big)}\right]^{-1}, \qquad (10.4)$$

where $\theta^{(s)}$ is a draw from the posterior distribution and N is the number of draws. As noted by, e.g., An and Schorfheide (2007), the numerical approximation is efficient if $f(\theta)$ is selected such that the summands are of equal magnitude. It can be shown that the harmonic mean estimator is consistent but not unbiased. In fact, due to Jensen's equality it is upward biased.

10.2.1. *Truncated Normal Weighting Function*

Geweke (1999) suggested to use the density of a truncated multivariate normal distribution in (10.4). That is, for $0 < p < 1$

$$f(\theta) = \frac{\exp\big[-(1/2)(\theta - \bar{\theta})'\Sigma_\theta^{-1}(\theta - \bar{\theta})\big]}{p(2\pi)^{m/2}|\Sigma_\theta|^{1/2}}\Big\{(\theta - \bar{\theta})'\Sigma_\theta^{-1}(\theta - \bar{\theta}) \leq \chi_p^2(m)\Big\}, \qquad (10.5)$$

where $\bar{\theta}$ is the mean and Σ_θ the covariance matrix from the output of the posterior simulator, i.e., $\bar{\theta} = N^{-1}\sum_{s=1}^{N}\theta^{(s)}$ and $\Sigma_\theta = N^{-1}\sum_{s=1}^{N}\theta^{(s)}\theta^{(s)\prime} - \bar{\theta}\bar{\theta}'$. The expression $\{a \leq b\}$ is 1 if true

---

[73] The identify follows from noticing that $p(\theta|Y) = p(Y|\theta)p(\theta)/p(Y)$ so that the right hand side of (10.3) is equal to $[(1/p(Y))\int f(\theta)d\theta]^{-1}$.

and 0 otherwise, while $\chi^2_p(m)$ is the $100p$ percentile value of the χ^2 distribution with m degrees of freedom; see also Geweke (2005, Section 8.2.4 and Theorem 8.1.2).

10.2.2. *Truncated Elliptical Weighting Function*

The accuracy of the harmonic mean estimator depends on the degree of overlap between the numerator (weight function) and denominator (posterior kernel) in (10.4). When the posterior density of θ is far from Gaussian, the Gaussian weighting function is less likely to work well. First of all, the height of the posterior density can be very low at the mean, especially when it is multimodal. Second, the truncated normal is often a poor local approximation to a non-Gaussian posterior density. Third, the likelihood can get close to zero in the interior of the parameter space. To deal with these three issues, Sims, Waggoner, and Zha (2008) (SWZ) have suggested an alternative weighting function based on a truncated elliptical distribution.

Let $\tilde{\theta}$ be the posterior mode and the scaling matrix $\tilde{\Sigma}_\theta = N^{-1} \sum_{s=1}^N (\theta^{(s)} - \tilde{\theta})(\theta^{(s)} - \tilde{\theta})'$. Next, let the nonnegative scalar r be given by

$$r = \sqrt{(\theta - \tilde{\theta})' \tilde{\Sigma}_\theta^{-1} (\theta - \tilde{\theta})}. \tag{10.6}$$

SWZ now define the elliptical density for θ as follows:

$$g(\theta) = \frac{\Gamma(m/2)}{2\pi^{m/2} |\tilde{\Sigma}_\theta|^{1/2}} \frac{h(r)}{r^{m-1}}, \tag{10.7}$$

where m is the dimension of θ, and $h(r)$ is a density function that does not depend on m which is defined for nonnegative r and is to be estimated.[74]

Let $r^{(s)}$ be the value of r when $\theta = \theta^{(s)}$, i.e., it represents the value of r for the posterior draws $s = 1, \ldots, N$. Suppose that $h(r)$ have a support on $[a, b]$ and be defined as

$$h(r|a, b, c) = \frac{cr^{c-1}}{b^c - a^c}. \tag{10.8}$$

The parameters a, b, and c are chosen as follows. Let Q_x be the x percentile of the $r^{(s)}$ values, ordered from the smallest to the largest, for $x = 1, 10, 90$. The values b and c are selected such that the probability of $r \leq Q_{10}$ from $h(r|0, b, c)$ is equal to 0.1 and the probability of $r \leq Q_{90}$ from $h(r|0, b, c)$ is equal to 0.9. This means that

$$b = \frac{Q_{90}}{0.9^{1/c}}, \quad c = \frac{\ln(1/9)}{\ln(Q_{10}/Q_{90})}.$$

Furthermore, the value of $a = Q_1$ to keep $h(r)$ bounded above.

In order to truncate the elliptical distribution, SWZ suggested using iid draws from $g(\theta)$. These are constructed as

$$\theta^{(i)} = \frac{r}{\|x\|} \tilde{\Sigma}_\theta^{1/2} x + \tilde{\theta}, \quad i = 1, \ldots, M, \tag{10.9}$$

where $\tilde{\Sigma}_\theta^{1/2}$ is the Choleski factor of $\tilde{\Sigma}_\theta$, $\|x\| = \sqrt{x'x}$ is the Euclidean norm, $x \sim N_m(0, I_m)$, and r is a random draw from $h(r)$. The latter draws are obtained as

$$r = \left[(b^c - a^c) p \right]^{1/c}, \quad p \sim U(0, 1).$$

These draws are obtained by utilizing the cdf $H(r) = r^c / (b^c - a^c)$ and inverting it at $H(r) = p$.

Let $k(\theta|Y) = L(Y; \theta)p(\theta)$, i.e., the kernel of the posterior density. Since the SWZ approach is based on a nonzero value for $h(r)$, the weight function $f(\theta)$ is effectively bounded above. Hence, the upper bound truncation on the ratio $f(\theta)/k(\theta|Y)$ can be implemented by a lower bound truncation of the posterior kernel itself. Specifically, let L be a positive number and Θ_L be the region defined by

$$\Theta_L = \{ \theta : k(\theta|Y) > L \}. \tag{10.10}$$

---

[74] The normal, Student t, logistic, and Laplace distributions are examples of elliptical distributions; see, e.g., Fang, Kotz, and Ng (1990) and Landsman and Valdez (2003).

The weighting function $f(\theta)$ is now chosen to be truncated elliptical with density such that its support is Θ_L. If q_L is the probability that random draws from the elliptical lies in Θ_L, then

$$f(\theta) = \frac{\{\theta \in \Theta_L\}}{q_L} g(\theta), \tag{10.11}$$

where $\{\theta \in \Theta_L\}$ is 1 if true and 0 otherwise. SWZ suggest that a good choice of L is a value such that 90 percent of the posterior draws lie in Θ_L.

To summarize, the SWZ appoach to estimating a suitable weighting function is as follows:

(1) Simulate N draws from the posterior density $p(\theta|Y)$ and record the minimum and maximum values of the posterior kernel, denoted by $k_{\min}$ and $k_{\max}$, respectively, and let $L \in (k_{\min}, k_{\max})$.

(2) Simulate M iid draws from the elliptical distribution in (10.9) and compute the proportion of these draws, denoted by $\hat{q}_L$, that belong to Θ_L. This estimate has a binomial distribution and its accuracy depends on the number of iid draws from $g(\theta)$. SWZ note that if $\hat{q}_L < 1.0e - 06$ then the estimate is unreliable since 3 or 4 standard deviations will include the zero value. As a rule of thumb they suggest $\hat{q}_L > 1.0e - 05$.

SWZ also take note of the fact that the computation of q_L gives a practical mechanism to evaluate how much of the overlap $f(\theta)$ and $k(\theta|Y)$ share. In particular, if q_L is too small the weight function and the posterior kernel have very little overlap and the estimated marginal likelihood is likely to be misleading.[75]

10.2.3. *Transformed Parameters*

Since YADA works internally with the transformed ϕ parameters, the expression for the modified harmonic mean estimator of the marginal likelihood is slightly different. Specifically,

$$\hat{p}_H(Y) = \left[\frac{1}{N} \sum_{s=1}^{N} \frac{f(\phi^{(s)})}{L(Y; g^{-1}(\phi^{(s)})) p(g^{-1}(\phi^{(s)})) J(\phi^{(s)})} \right]^{-1}, \tag{10.12}$$

where $f(\phi)$ is either the truncated normal or the truncated elliptical for the transformed ϕ parameters.

The numerical standard error can easily be computed for the modified harmonic mean estimator of the (log of the) marginal likelihood. Let

$$m(\phi) = \frac{f(\phi)}{k(\phi|Y)}, \tag{10.13}$$

where $k(\phi|Y) = L(Y; g^{-1}(\phi)) p(g^{-1}(\phi)) J(\phi)$ is the posterior kernel of $p(\phi|Y)$. To handle possible numerical issues we introduce a constant c which guarantees that a rescaling of $m(\phi)$ is bounded for all ϕ. An example of such a numerical problem is that the log posterior kernel is equal to, say, -720 for some ϕ. While $\exp(-720)$ is a small positive number, computer software such as matlab states that $1/\exp(-720)$ is infinite.[76]

One suitable choice is often $c = \max_\phi k(\phi|Y)$, i.e., the largest posterior kernel value over all posterior draws of ϕ. For any suitable choice of c, it follows that

$$n(\phi) = \frac{f(\phi)}{\exp(\ln k(\phi|Y) - \ln(c))}. \tag{10.14}$$

The rescaling needs to be accounted for when computing the marginal likelihood. Let $\bar{n}$ be the sample mean of $n(\phi)$. It follows that the log marginal likelihood based on the modified

---

[75] SWZ note that for any sequence of posterior draws $\theta^{(s)}$, $s = 1, \ldots, N$, increasing the variance of $f(\theta)$ means that the value of $f(\theta)/k(\theta|Y)$ tends to decrease. As a consequence, the estimated marginal likelihood is artifically inflated, irrespective of whether the tail of the distribution represented by the weight function is truncated or not. By requiring q_L to be computed, the estimate of this proportion goes to zero as the variance of $f(\theta)$ becomes too large or too small.

[76] In practise, it is important that rescaled log posterior kernel values are bounded from below by about -700 and from above by approximately 700. This guarantees that the exponential function yields finite values.

harmonic mean estimator is given by

$$\ln \hat{p}_H(Y) = \ln(c) - \ln(\bar{n}). \tag{10.15}$$

The $n(\phi)$ values can now be used to compute a numerical standard error of the log marginal likelihood based on the modified harmonic mean estimator. The numerical standard error of $n(\phi)$ can be computed from the Newey and West (1987) estimator in (8.2) by replacing ϕ with $n(\phi)$. The delta method can thereafter be applied such that the numerical standard error of the log marginal likelihood is equal to the numerical standard error of $n(\phi)$ divided by $\bar{n}$.

10.3. *The Chib and Jeliazkov Estimator*

The Chib and Jeliazkov (2001) estimator of the marginal likelihood starts from the so called marginal likelihood identity

$$p(Y) = \frac{L(Y;\theta)p(\theta)}{p(\theta|Y)} = \frac{L(Y;g^{-1}(\phi))p(g^{-1}(\phi))J(\phi)}{p(\phi|Y)}. \tag{10.16}$$

This relation holds for any value of ϕ (and θ), but a point with high posterior density should preferably be selected, e.g., the posterior mode $\tilde{\phi}$.

The numerator of (10.16) can be determined directly once the posterior mode has been found. The denominator, however, requires a numerical approximation. Hence,

$$\hat{p}_{CJ}(Y) = \frac{L(Y;g^{-1}(\tilde{\phi}))p(g^{-1}(\tilde{\phi}))J(\tilde{\phi})}{\hat{p}(\tilde{\phi}|Y)}, \tag{10.17}$$

where $\hat{p}(\tilde{\phi}|Y)$ remains to be calculated.

Based on the definition of $r(\phi, \varphi|Y)$ in equation (8.1), let

$$\alpha(\phi, \varphi|Y) = \min\{1, r(\phi, \varphi|Y)\}. \tag{10.18}$$

Let $q(\phi, \tilde{\phi}|Y)$ be the proposal density for the transition from ϕ to $\tilde{\phi}$. For the random walk Metropolis algorithm we have used the multivariate normal density as the proposal, i.e.,

$$q(\phi, \tilde{\phi}|Y) = (2\pi)^{-m/2} \left|c^2\tilde{\Sigma}\right|^{-1/2} \exp\left\{-\frac{1}{2c^2}(\tilde{\phi} - \phi)'\tilde{\Sigma}^{-1}(\tilde{\phi} - \phi)\right\}. \tag{10.19}$$

This density is symmetric, i.e., $q(\phi, \tilde{\phi}|Y) = q(\tilde{\phi}, \phi|Y)$.[77] The posterior density at the mode can now be approximated by

$$\begin{aligned}
\hat{p}(\tilde{\phi}|Y) &= \frac{N^{-1}\sum_{s=1}^{N}\alpha(\phi^{(s)}, \tilde{\phi}|Y)q(\phi^{(s)}, \tilde{\phi}|Y)}{J^{-1}\sum_{j=1}^{J}\alpha(\tilde{\phi}, \phi^{(j)}|Y)} \\
&= \frac{N^{-1}\sum_{s=1}^{N}q(\phi^{(s)}, \tilde{\phi}|Y)}{J^{-1}\sum_{j=1}^{J}\alpha(\tilde{\phi}, \phi^{(j)}|Y)},
\end{aligned} \tag{10.20}$$

where $\phi^{(s)}$, $s = 1, \ldots, N$ are sampled draws from the posterior distribution with the RWM algorithm, while $\phi^{(j)}$, $j = 1, \ldots, J$, are draws from the proposal density (10.19). The second equality stems from the fact that $\alpha(\phi^{(s)}, \tilde{\phi}|Y) = 1$ for all $\phi^{(s)}$ when $\tilde{\phi}$ is the posterior mode, i.e., the transition from $\phi^{(s)}$ to $\tilde{\phi}$ is always accepted by the algorithm. Hence, the Chib and Jeliazkov estimator of $p(\tilde{\phi}|Y)$ is simply the sample average of the proposal density height for the accepted draws relative to the posterior mode, divided by the sample average of the acceptance probability, evaluated at the posterior mode.

The parameter J is always equal to N in YADA. In contrast with the modified harmonic mean estimator of the marginal likelihood, the Chib and Jeliazkov estimator requires J additional draws. When J is large and the parameter space is high dimensional, the Chib and Jeliazkov

---

[77] Notice that $\alpha(\phi, \varphi|Y) = \min\{1, r(\phi, \varphi|Y)q(\varphi, \phi|Y)/q(\phi, \varphi|Y)\}$ in Chib and Jeliazkov (2001); see, e.g., Section 2.1, above equation (7).

estimator will be considerably slower to compute since the log posterior function on the right hand side of equation (7.1) needs to be evaluated an additional J times.

The numerical standard error of the log of the marginal likelihood estimate $\hat{p}(\tilde{\phi}|Y)$ in (10.20) can be computed from the vectors

$$h^{(s,j)} = \begin{bmatrix} h_1^{(s,j)} \\ h_2^{(s,j)} \end{bmatrix} = \begin{bmatrix} q(\phi^{(s)}, \tilde{\phi}|Y) \\ \alpha(\tilde{\phi}, \phi^{(j)}|Y) \end{bmatrix}.$$

The average of $h^{(s,j)}$ is denoted by $\hat{h}$. This means that

$$\ln \hat{p}(\tilde{\phi}|Y) = \ln \hat{h}_1 - \ln \hat{h}_2,$$

and the numerical standard error of the log marginal likelihood estimate can be calculated from the sample variance of $h^{(s,j)}$ via the delta method. The sample variance of the latter can be computed using the Newey and West (1987) estimator.

10.4. *YADA Code*

Geweke's (1999) modified harmonic mean estimator of the marginal likelihood with a truncated normal density as weighting function is computed with `MargLikeModifiedHarmonic`. This function can also be used to estimate the marginal likelihood sequentially. The nature of the sequential estimation is quite flexible, where a starting value and an incremental value for the sequence can be selected on the Settings tab. By default, YADA sets the starting value to 100 and the increment value to 100. For a posterior sample with 10000 draws, this means that the marginal likelihood is estimated for the sample sizes $100, 200, \dots, 9900, 10000$. The selection of sequential estimation sample is determined on the *DSGE Posterior Sampling* frame on the Settings tab; cf. Figure 5.

Similarly, the modified harmonic mean estimator of the log marginal likelihood with a truncated elliptical density as weighting function suggested by Sims et al. (2008) is computed with the function `MargLikeSWZModifiedHarmonic`. It can also perform a sequential estimation of the log marginal likelihood.

The function `MargLikeChibJeliazkov` calculates the Chib and Jeliazkov (2001) estimator of the marginal likelihood as well as its numerical standard error. Like the modified harmonic mean estimator function, `MargLikeModifiedHarmonic`, the calculations can be performed sequentially using the same sample grid.

The Laplace approximation is calculated by `MargLikeLaplace`. It is only run towards the end of the posterior mode estimation routine and should only be viewed as a quick first order approximation when comparing models.

10.4.1. `MargLikeLaplace`

The function `MargLikeLaplace` takes 3 inputs. First, it requires the value of the log posterior at the mode of ϕ, `LogPost`, second minus (the inverse of) the Hessian at the mode, `Hessian`, and third `IsInverse`, a boolean variable that takes the value 1 if `Hessian` is the inverse Hessian and 0 otherwise. Based on equation (10.2) the log marginal likelihood is calculated and provides as the output `LogMarg`.

10.4.2. `MargLikeModifiedHarmonic`

The function `MargLikeModifiedHarmonic` takes 5 inputs. First of all, `PostSample`, an $N \times m$ matrix with N being the number posterior draws that are used. This values is often smaller than the total number of posterior draws that have been computed either since burn-in draws are skipped or since the log marginal likelihood is computed from a subsample of the available posterior draws (or both). Next, the values of the log posterior kernel are needed. These are assumed to be given by the N dimensional vector `LogPost`. Third, the function accepts a boolean variable `ComputeSequential` that is 1 if the marginal likelihood should be estimated sequentially and 0 otherwise. Fourth, a vector with coverage probabilities are needed. This vector, denoted by `CovProb`, can be empty or contain numbers between 0 and 1. Finally, the function

requires the structure `DSGEModel` with model related information. This structure should contain the fields `SequentialStartIterationValue` and `SequentialStepLengthValue`, where the former gives the starting value of the sequential estimates and the latter gives the increment value. In case `CovProb` is empty, the structure should also include the fields `CovStartValue`, `CovIncValue`, and `CovEndValue`. These fields determine the starting probability value, the increment and the upper bound of the coverage probability p in (10.5); cf. the DSGE Posterior Sampling frame on the Settings tab in Figure 5.

The output of `MargLikeModifiedHarmonic` is given by the variables `LogMargs`, `CovProb`, and `NWStdErr`. The dimension of the first output variable is given by the number of successful computations of the marginal likelihood for the given coverage probabilities times the number of coverage probabilities plus 1. The first column of this matrix contains the number of draws used for the computations, while the remaining columns contains the marginal likelihood values for each given coverage probability. The second output argument is simply the vector of coverage probabilities that was used by the function. Finally, the third output variable is a vector with the numerical standard errors of the estimated log marginal likelihood for each coverage probability at the full sample estimates using the Newey and West (1987) estimator; see equation (8.2).

10.4.3. `MargLikeSWZModifiedHarmonic`

The function `MargLikeSWZModifiedHarmonic` needs at nine input variables to complete its mission. These are: `PostSample`, `LogPost`, `TotalDraws`, `PValue`, `ComputeSequential`, `ModeData`, `lambda`, `DSGEModel`, and `CurrINI`. Four of these variables are identical to the same names inputs for `MargLikeModifiedHarmonic`. The integer `TotalDraws` is the number of parameter draws that could be used and is often greater than N; see the explanation in 10.4.2. The integer `PValue` lies between 0 and 100 and is the percent of log posterior kernel values that are greater than a lower bound that is defined through its value. The structure `ModeData` has fields with names that include data from the posterior mode estimation of the DSGE or DSGE-VAR model. The scalar `lambda` is the DSGE-VAR hyperparameter λ (see Section 15) and is empty for DSGE models. The structure `CurrINI` is discussed above.

The results from the calculations are provided in the four output variables `LogMargs`, `qL`, `LBound`, and `NWStdErr`. The first is a matrix with rows equal to the number of successful computations of the log marginal likelihood and two columns. The first column holds the number of parameter draws used, and the second the estimated log marginal likelihood at that number of parameter draws. The scalar `qL` is the fraction of iid draws from the elliptical distribution such that the log posterior kernel values at these draws are greater than `LBound`, the lower bound for the log posterior kernel implied by `PValue` when all the N posterior draws are used. Finally, the variable `NWStdErr` is the numerical standard error, based on the Newey and West (1987) estimator, of the estimated log marginal likelihood value for all the N parameter draws.

10.4.4. `MargLikeChibJeliazkov`

The function `MargLikeChibJeliazkov` needs 17 input arguments. The first two are the matrix `PostSample` and the vector `LogPost` that are also used by `MargLikeModifiedHarmonic`. Next, the posterior mode $\tilde{\phi}$ and the inverse Hessian at the posterior mode $\tilde{\Sigma}$ are needed, along with the scale factor c that is used by the proposal density. These inputs are denoted by `phiMode`, `SigmaMode`, and `c`, respectively. Furthermore, the function takes the inputs `logPostMode` (the value of the log posterior at the mode), `NumBurnin` (number of burn-in draws), and the boolean variable `ComputeSequential` that is also used by `MargLikeModifiedHarmonic`.

The remaining 9 input arguments are the last 9 inputs for the `logPosteriorPhiDSGE` function, used to compute the $\alpha(\tilde{\phi}, \phi^{(j)}|Y)$ term in the denominator of equation (10.20). The function `MargLikeChibJeliazkov` always sets $J = N$ in (10.20).

The output matrix is given by `LogMargs`. The first column gives the number of draws used for estimating the marginal likelihood, the second column the estimated value of the log marginal likelihood, while the numerical standard error of the log marginal likelihood is provided in the

third column. The standard error is computed using the Newey and West (1987) estimator, with $\bar{N} = N^{(1/2.01)}$.

11. Analysing the Properties of a DSGE Model

There are several ways that you can evaluate an estimated DSGE model in YADA through the behavior of its economic shocks. In this section I will consider a number of tools that are designed for this purpose. Namely, the estimated economic shocks (the η_t's), historical forecast error decompositions, impulse response functions, conditional variance decompositions, forecast error variance decompositions, conditional correlations and correlation decompositions, historical observed variable decompositions, and parameter scenarios. We then turn to the issue if a VAR model can uncover the economic shocks and measurement errors of the DSGE model, as well as local identification via the rank of Fisher's information matrix.

11.1. *Estimation of the Economic Shocks*

In Section 5.5 the smooth estimates of the state shocks for the state equation of the Kalman filter were given in equation (5.26). Let us assume that B_0 has full column rank so that $B_0'B_0$ is invertible. This assumption means that there are no redundant shocks in the DSGE model.[78]

For a given sequence of smoothly estimated state shocks, $v_{t|T}$, smooth estimates of the economic shocks are now given by:

$$\eta_{t|T} = \left(B_0'B_0\right)^{-1}B_0'v_{t|T} = B_0'r_{t|T}, \quad t = 1,\ldots,T, \tag{11.1}$$

through equation (5.27) with $Q = B_0B_0'$. The covariance matrix of the smooth estimates of the economic shocks is determined from the covariance matrix of the smooth estimates of the state shocks and is given by $B_0'N_{t|T}B_0 \leq I_q$, where equality holds if η_t is observable at T. The conditional covariance matrix of η_t is therefore given by

$$E\left[(\eta_t - \eta_{t|T})(\eta_t - \eta_{t|T})'|\mathcal{Y}_T\right] = I_q - B_0'N_{t|T}B_0, \quad t = 1,\ldots,T. \tag{11.2}$$

Similarly, update estimates of the economic shocks can be computed directly from update estimates of the state shocks. From equation (5.29) we therefore have that

$$\eta_{t|t} = \left(B_0'B_0\right)^{-1}B_0'v_{t|t} = B_0'H\Sigma_{y,t|t-1}^{-1}\left(y_t - y_{t|t-1}\right) = B_0'r_{t|t}, \quad t = 1,\ldots,T. \tag{11.3}$$

The covariance matrix of the update estimates of the economic shocks is also determined from the covariance matrix of the state shocks and is equal to $B_0'H\Sigma_{y,t|t-1}^{-1}H'B_0 \leq I_q$, where equality holds if η_t is observable at t. Accordingly, the conditional covariance matrix for η_t is

$$E\left[(\eta_t - \eta_{t|t})(\eta_t - \eta_{t|t})'|\mathcal{Y}_t\right] = I_q - B_0'H\Sigma_{y,t|t-1}^{-1}H'B_0, \quad t = 1,\ldots,T. \tag{11.4}$$

11.1.1. *Limit Covariance Matrices of Update and Smooth Estimates of the Structural Shocks*

The population covariance matrix of the update estimates of the structural shocks, $\eta_{t|t}$, is given by $B_0'H\Sigma_{y,t|t-1}^{-1}H'B_0$ and therefore varies over time based on the one-step-ahead covariance matrix of of the observed variables, which in turn varies over time with the one-step-ahead covariance matrix for the state variables. It is shown in Section 11.5 that if $n = q$, $R = 0$, and F has all eigenvalues inside the unit circle, then an asymptote to $P_{t|t-1}$ is given by $P_1 = B_0B_0'$. Letting $\Sigma_{y,1}$ denote the asymptote of $\Sigma_{y,t|t-1}$ and neglecting the fact that the asymptote for $P_{t|t-1}$ need not be unique, it follows that the limiting covariance matrix of the updated structural shocks is I_n since

$$\Sigma_{y,1}^{-1} = \left(B_0'H\right)^{-1}\left(H'B_0\right)^{-1}. \tag{11.5}$$

For this particular case it therefore follows that the update estimates of the structural shocks are uncorrelated *once* the Kalman filter recursions have converged to an asymptote for the state covariance matrix. However, if the number of unique measurement errors and the number of structural shocks is greater than the number of observed variables, then the asymptote for the covariance matrix of the structural shocks, is not diagonal with unit diagonal entries. Although it is very tempting to compare a sample estimate of the covariance matrix for the update estimate

---

[78] YADA actually checks this by trying to find column of zeros in B_0. Such columns may appear when a certain shock is forced to have no effect on the variables in the AiM model file, but the shock has not been deselected in YADA.

of these shocks to the identity matrix, it is generally not a useful exercise. Instead, one may compare such a sample estimate to either an asymptote of the population covariance matrix or, better yet, to the sample average of $B_0' H \Sigma_{y,t|t-1}^{-1} H' B_0$.

The issue is, in fact, even more problematic once we turn to the smooth estimates of the structural shocks. The covariance matrix of these estimates is given by $B_0' N_{t|T} B_0$. Provided that an asymptote of $N_{t|T}$ exists, it is given by

$$N_k = N_0 + L' N_{k-1} L, \quad k = 1, 2, \ldots,$$

where $k = T - t$,

$$N_0 = H \Sigma_{y,1}^{-1} H',$$

and

$$L = F(I_r - P_H), \quad P_H = P_1 H (H' P_1 H + R)^{-1} H'.$$

Under the assumptions that $n = q$, $R = 0$, and the eigenvalues of F lie inside the unit circle we know that equation (11.5) holds. It is now straightforward to show that $B_0' N_0 B_0 = I_n$, while $L B_0 = 0$. As a consequence, it holds that $B_0' N_k B_0 = I_n$ for $k \geq 0$.

Concerning the autocovariances for the smooth estimates of the structural shocks, it holds that

$$E\left[\eta_{t|T} \eta_{t+1|T}'\right] = B_0' (F - K_t H')' N_{t+1,T} B_0.$$

Hence, the smooth estimates of these shocks are generally correlated over time. Given that an asymptote of this covariance matrix exists, it satisfies $B_0' L' N_{k-1} B_0$. It now follows that if $n = q$ and $R = 0$, this first order autocovariance matrix is zero since $L B_0 = 0$.

11.1.2. *Observation Weights under Standard Initialization*

The dependence of the smooth estimates of the shocks on the observed variables and the initial conditions may be investigated by using the weighting results in Section 5.9. Since $\eta_{t|T} = B_0' r_{t|T}$ for $t = 1, 2, \ldots, T$, it can directly be established that

$$\eta_{t|T} = \sum_{\tau=1}^{T} \alpha_\tau(\eta_{t|T}) z_\tau + \beta_0(\eta_{t|T}) \xi_{1|0}, \quad t = 1, \ldots, T. \tag{11.6}$$

The $q \times n$ matrices with weights on the observed variables are given by

$$\alpha_\tau(\eta_{t|T}) = B_0' \alpha_\tau(r_{t|T}), \quad \tau = 1, \ldots, T, \tag{11.7}$$

while the $q \times r$ weighting matrix on the initial condition is

$$\beta_0(\eta_{t|T}) = B_0' \beta_0(r_{t|T}). \tag{11.8}$$

Recalling that the forecast error, $y_t - y_{t|t-1}$, is equal to $z_t - H' \xi_{t|t-1}$, it can be seen that the update estimates of the shock depend on the observed variables and the initial conditions according to

$$\eta_{t|t} = \sum_{\tau=1}^{t} \alpha_\tau(\eta_{t|t}) z_\tau + \beta_0(\eta_{t|t}) \xi_{1|0}, \quad t = 1, \ldots, T. \tag{11.9}$$

The $q \times n$ matrices with weights on the observed variables are equal to

$$\alpha_\tau(\eta_{t|t}) = \begin{cases} -B_0' H \Sigma_{y,t|t-1}^{-1} H' \alpha_\tau(\xi_{t|t-1}) & \text{if } \tau = 1, \ldots, t-1, \\ B_0' H \Sigma_{y,t|t-1}^{-1} & \text{if } \tau = t. \end{cases} \tag{11.10}$$

The weights on the initial condition are now given by

$$\beta_0(\eta_{t|t}) = -B_0' H \Sigma_{y,t|t-1}^{-1} H' \beta_0(\xi_{t|t-1}). \tag{11.11}$$

11.1.3. *Observation Weights under Diffuse Initialization*

It was noted in Sections 5.14.2 and 5.15.2 that under diffuse initialization of the Kalman filter the smooth estimates of the state shocks for the initialization sample ($t = 1, \ldots, d$) are given by

$$v_{t|T} = Q r_{t|T}^{(0)},$$

where

$$r_{t|T}^{(0)} = \begin{bmatrix} I_r & 0 \end{bmatrix} \tilde{r}_{t|T} = J_r' \tilde{r}_{t|T}.$$

Given the relationship between the economic shocks and the state shocks in (11.1), we therefore find that smooth estimates of the economic shocks for the initialization sample are

$$\eta_{t|T} = B_0' J_r' \tilde{r}_{t|T}.$$

Concerning the observation weights of the smoothly estimated economic shocks, equation (11.6) remains valid, but the $q \times n_t$ matrices with weights for $t = 1, \ldots, d$ are now

$$\alpha_\tau(\eta_{t|T}) = B_0' J_r' \alpha_\tau(\tilde{r}_{t|T}), \quad \tau = 1, \ldots, T, \tag{11.12}$$

where the weights on the extended innovation vector $\tilde{r}_{t|T}$ are located in Section 5.16.3. The $q \times r$ weighting matrices on the initial state value are similarly given by

$$\beta_0(\eta_{t|T}) = B_0' J_r' \beta_0(\tilde{r}_{t|T}).$$

In order to determine the observation weights for the update estimates of the economic shocks, we need to take the same three cases into account as for the state variable estimates. In the event that $\Sigma_{\infty,t|t-1}$ has full rank n_t we know from equation (5.85) that for $t = 1, \ldots, d$:

$$r_{t|t} = r_{\infty,t|t} = H \Sigma_{\infty,t|t-1} z_t - H \Sigma_{\infty,t|t-1} H' \xi_{t|t-1}.$$

Similarly, if the rank of $\Sigma_{\infty,t|t-1}$ is zero, equation (5.88) implies that

$$r_{t|t} = r_{*,t|t} = H \Sigma_{*,t|t-1} z_t - H \Sigma_{*,t|t-1} H' \xi_{t|t-1}.$$

Finally, if the rank of this matrix is less than n_t and greater than or equal to unity, such that the univariate filtering routine is utilized, then equations (5.142)–(5.143) along with the definitions of $\bar{F}_t$ and $\bar{G}_t$ in equation (5.154) means that

$$r_{t|t} = J_r' \tilde{u}_{t,0} = J_r' \bar{F}_t z_t - J_r' \bar{G}_t \xi_{t|t-1}.$$

This can be expressed more compactly as

$$r_{t|t} = \tilde{F}_t z_t - \tilde{G}_t \xi_{t|t-1}, \qquad t = 1, \ldots, d, \tag{11.13}$$

where

$$\tilde{F}_t = \begin{cases} H \Sigma_{\infty,t|t-1}, & \text{if rank}\left(\Sigma_{\infty,t|t-1}\right) = n_t, \\ H \Sigma_{*,t|t-1}, & \text{if rank}\left(\Sigma_{\infty,t|t-1}\right) = 0, \\ J_r' \bar{F}_t, & \text{otherwise}, \end{cases}$$

and

$$\tilde{G}_t = \begin{cases} H \Sigma_{\infty,t|t-1} H', & \text{if rank}\left(\Sigma_{\infty,t|t-1}\right) = n_t, \\ H \Sigma_{*,t|t-1} H', & \text{if rank}\left(\Sigma_{\infty,t|t-1}\right) = 0, \\ J_r' \bar{G}_t, & \text{otherwise}. \end{cases}$$

For the $r \times r$ observation weight matrices of the update innovations we therefore find that

$$\alpha_\tau(r_{t|t}) = \begin{cases} -\tilde{G}_t \alpha_\tau(\xi_{t|t-1}), & \text{if } \tau = 1, \ldots, t-1, \\ \tilde{F}_t, & \text{if } \tau = t, \end{cases}$$

while the $r \times r$ matrices with weights on the initial state are

$$\beta_0(r_{t|t}) = -\tilde{G}_t \beta_0(\xi_{t|t-1}).$$

The update estimate of the economic shocks over the initialization sample may now be computed directly from the update innovations according to:

$$\eta_{t|t} = B_0' r_{t|t}, \qquad t = 1, \ldots, d.$$

It therefore follows that the $q \times n_t$ weighting matrices of the update estimates of the economic shocks are given by

$$\alpha_\tau(\eta_{t|t}) = B_0' \alpha_\tau(r_{t|t}), \qquad \tau = 1, \ldots, t, \qquad (11.14)$$

while the $q \times r$ matrices with weights on the initial state are

$$\beta_0(\eta_{t|t}) = B_0' \beta_0(r_{t|t}).$$

11.1.4. *Simulation Smoother*

Since the dimension of η_t is typically lower than the dimension of v_t, i.e., $q < r$, a more efficient algorithm of the simulation smoother from Section 5.10 would take this account. This can directly be achieved by replacing v_t by η_t in the definition of ω and by letting $\Omega = (I_T \otimes \mathrm{diag}[R, I_q])$.

Furthermore, the covariance matrix R will typically have reduced rank. We may therefore apply the decomposition $R = S\Lambda S'$, where S is $n \times n_R$, $\mathrm{rank}[R] = n_R \leq n$, $\Lambda = \mathrm{diag}[\lambda_1, \ldots, \lambda_{n_R}]$ are the non-zero eigenvalues of R, ordered from the largest to the smallest, while $S'S = I_{n_R}$. We then let $w_t = R^{1/2} \varsigma_t$, with $R^{1/2} = S\Lambda^{1/2}$, while ς_t has covariance matrix I_{n_R}. Finally, we replace w_t with ς_t in the definition of ω, with the consequence that $\Omega = I_{T(n_R+q)}$.

It should be emphasized that if η_t (or w_t) is observable at t, such that the right hand side of (11.2) (or (5.35)) is zero, then the covariance matrix of the simulation smoother for the economic shocks is zero. This may happen when $q + n_R = n$. Assuming that $n_R = 0$, this can easily be checked by inspecting if $B_0' N_{T|T} B_0 = I_q$. In addition, if a diagonal element of the covariance matrix in (11.2) is zero, then the corresponding economic shock can be observed from the data and the parameter values; a similar conclusion about the measurement errors can be drawn by inspecting the right hand side of (5.35).

To improve the efficiency of the simulation smoother we may use antithetic variables; see Section 5.10. Given our redefinition of ω as being a vector of dimension $T(n_R + q)$ of $N(0,1)$ random variables, we can now provide some additional antithetic variables that allow the simulation sample to be *balanced for scale*. Recall first that

$$\tilde{\omega}^{(i)} = E[\omega|\mathcal{Y}_T] + \omega^{(i)} - E[\omega|\mathcal{Y}_T^{(i)}],$$

where $\omega^{(i)}$ is a draw from $N(0, \Omega)$, while the anithtetic variable

$$\tilde{\omega}^{(-i)} = 2E[\omega|\mathcal{Y}_T] - \tilde{\omega}^{(i)}$$
$$= E[\omega|\mathcal{Y}_T] - \omega^{(i)} + E[\omega|\mathcal{Y}_T^{(i)}],$$

is equiprobable with the draw $\tilde{\omega}^{(i)}$. Making use of this variable for the simulation smoother means that the simulation sample is balanced for location.

Define $c = \omega^{(i)\prime} \omega^{(i)}$ so that $c \sim \chi^2(T(n_R + q))$. This means that $p = \Pr[\chi^2(T(n_R + q)) < c] = F(c)$ gives the probability that a random variable with the same distribution as c is less than the value of c. A value from the $\chi^2(T(n_R + q))$ distribution which is equiprobable with c is therefore $d = F^{-1}(1 - p)$. Following Durbin and Koopman (2012, Chapter 11.4.3), a third antithetic variable for $\tilde{\omega}^{(i)}$ is therefore

$$\bar{\omega}^{(i)} = E[\omega|\mathcal{Y}_T] + \sqrt{d/c} \left(\tilde{\omega}^{(i)} - E[\omega|\mathcal{Y}_T] \right),$$

while a fourth antithetic is

$$\bar{\omega}^{(-i)} = E[\omega|\mathcal{Y}_T] + \sqrt{d/c} \left(\tilde{\omega}^{(-i)} - E[\omega|\mathcal{Y}_T] \right).$$

These two vectors have the same distribution as $\tilde{\omega}^{(i)}$ and $\tilde{\omega}^{(-i)}$, i.e., $N(E[\omega|\mathcal{Y}_T], C)$, where $C = Cov[\omega|\mathcal{Y}_T]$ does not depend on $\mathcal{Y}_T$.

11.2. *Historical Forecast Error Decomposition*

A common tool in the analysis of the structural VAR model is the decomposition of the forecast errors for the observed variables into the underlying structural shocks. For such models, the forecast errors are linear combinations of the structural shocks. In state-space models a forecast error decomposition becomes more delicate since the forecast errors not only depend on the structural, economic shocks, but also on measurement errors and prediction errors of the un-observed state variables. Since the prediction errors of these variables depend on the economic shocks and on the measurement errors, one cannot really speak about a unique decomposition since one may further decompose the variable prediction errors.

Let $\varepsilon_{t+h} = y_{t+h} - y_{t+h|t}$ denote the h-step ahead forecast error of the observed variables when we condition on the parameters. Furthermore, notice that $y_{t+h} = y_{t+h|T}$ provided that $t+h \leq T$. From the measurement error estimation equation (5.34) and the multistep forecasting equation (5.37) we thus have that for any h such that $t + h \leq T$:

$$\varepsilon_{t+h} = H'\big(\xi_{t+h|T} - \xi_{t+h|t}\big) + w_{t+h|T}, \quad t = 1, \ldots, T - h. \tag{11.15}$$

By making use of equation (5.40) we can rewrite the difference between the smoothed and the h-step forecasted state vector on the right hand side of (11.15) as

$$\xi_{t+h|T} - \xi_{t+h|t} = F^h\big(\xi_{t|T} - \xi_{t|t}\big) + \sum_{i=0}^{h-1} F^i v_{t+h-i|T}.$$

Substituting this expression into (11.15) and noticing that $v_{t+h-i|T} = B_0 \eta_{t+h-i|T}$, we obtain the following candidate of a historical forecast error decomposition

$$\varepsilon_{t+h} = H' F^h\big(\xi_{t|T} - \xi_{t|t}\big) + H' \sum_{i=0}^{h-1} F^i B_0 \eta_{t+h-i|T} + w_{t+h|T}, \quad t = 1, \ldots, T - h. \tag{11.16}$$

Unless the state vector can be uniquely recovered from the observed variables[79] the first term on the right hand side is non-zero. It measures the improvement in the projection of the state vector when the full sample is observed relative to the partial sample. As the forecast horizon h increases, this term converges towards zero. It may be argued that the choice of time period t for the state vector is here somewhat arbitrary. We could, in principle, decompose this term further until we reach period 1. However, the choice of state variable period t for the forecast error is reasonable since this is the point in time when the forecasts are made. Moreover, shifting it back to, say, period 1 would mean that the historical forecast error decomposition would include estimates of the economic shocks that are based not only on the full sample, but also on the $\mathcal{y}_t$ information set and, furthermore, the timing of those shocks would be for time periods $\tau = 2, \ldots, t$, and would therefore be shocks with time period prior to $t + 1$, the first time period of the forecast period.

Apart from the timing of the terms, the decomposition in (11.16) also has the advantage that the dynamics of the state variables enter the second moving average term that involves only the economic shocks, while the measurement errors in the third term do not display any serial correlation. We may regard this as a "model consistent" decomposition in the sense that only the state variables display dynamics and the measurement errors are independent of the state variables. Hence, serial correlation in the forecast errors should stem from the shocks that affect the dynamics, i.e., the economic shocks.

11.3. *Impulse Response Functions*

11.3.1. *Impulse Responses For Unanticipated Shocks in the Linear Model*

The responses of the observed variables y_{t+h} from shocks to η_t can easily be calculated through the state-space representation and the relationship between the state shocks and the economic shocks. Suppose that $\eta_t = e_j$ and zero thereafter, with e_j being the j:th column of I_q. Hence,

---

[79] This would be the case if the state vector could be expressed as a linear function of the observed variables (and the deterministic) by inverting the measurement equation.

we consider the case of a one standard deviation impulse for the j:th economic shock. From the state equation (5.2) the responses in ξ_{t+h} for $h \geq 0$ are:

$$\text{resp}\big(\xi_{t+h}|\eta_t = e_j\big) = F^h B_0 e_j, \quad h \geq 0. \tag{11.17}$$

If the model is stationary, then the responses in the state variables tend to zero as h increases.

From the measurement equation (5.1) we can immediately determine the responses in the observed variables from changes to the state variables. These changes are here given by equation (11.17) and, hence, the responses of the observed variables are:

$$\text{resp}\big(y_{t+h}|\eta_t = e_j\big) = H' F^h B_0 e_j, \quad h \geq 0. \tag{11.18}$$

Again, the assumption that the state variables are stationary implies that the responses of the observed variables tend to zero as the response horizon h increases.

11.3.2. *Impulse Responses when the Zero Lower Bound may be Binding*

The impulse responses in the previous section were computed under the implicit assumption that the model does not have any anticipated shocks. Such shocks are allowed for when dealing with a nonlinearity such as the zero lower bound (ZLB). The general solution of the linear DSGE model subject to such a condition was discussed in detail in Section 3.4 and involves allowing for anticipated shocks which, when estimated, depend on which value the unanticipated shocks take. This means that the impulse responses under the ZLB are different from those in the purely linear model provided that this lower bound is binding over the response horizon.

Recall that the anticipated shocks are collected into A_t, a $(T+1)$-dimensional vector with entries $\alpha_{t+\tau|t}$ for $\tau = 0, 1, \ldots, T$. The integer T corresponds to the number of time periods that the ZLB may be binding. We will compute the impulse responses from period t until period $t + h$, where h may differ from T. Below it is assumed that the ZLB is either not binding in period $t-1$ or, somewhat more generally, that $A_{t-1} = 0$.

Impulse responses from a shock to η_t can now be computed as follows. Consider the following unanticipated shock in period t:

$$\eta_t = ce_j, \tag{11.19}$$

where e_j as in Section 11.3.1 above is a q-dimensional vector with unity in its j:th position and zeros elsewhere. The constant c is typically unity for impulse responses, but we will here allow it to take on other values since it will influence the impulse reponses through its effect on the size of the anticipated shocks, and this effect is not expected to be proportional to c. Moreover, we will measure impulse reponses based on two η_t vectors, one with c being different from zero and one when it is equal to zero. The difference between the paths of the state (observable) variables using these two shock values constitutes their response to the shock in (11.19).

Specifically, the paths for the state variables can be calculated using the stochastic simulation algorithm in Section 3.4.6 with $\bar{S} = 1$. This means that a sequence of anticipated shocks $A_{t+i}^{(c)}$, $i = 0, 1, \ldots, h$, based on η_t satisfying (11.19) in period t and being zero for all $t + i$ (with $i \geq 1$) is computed with the forward-back shooting algorithm, first for c being nonzero and then for c being equal to zero. With $\xi_t = z_t$, the impulse response functions for the state variables are given by the difference between these two paths:

$$\text{resp}\big(\xi_{t+i}|\eta_t = ce_j\big) = z_{t+i}^{(c)} - z_{t+i}^{(0)}, \quad i = 0, 1, \ldots, h, \tag{11.20}$$

where $z_{t+i}^{(c)}$ is the simulated value of the state variables at $t + i$ for a given value of c. Similarly, using the measurement equation the responses of the observable variables are

$$\text{resp}\big(y_{t+i}|\eta_t = ce_j\big) = H'\big(z_{t+i}^{(c)} - z_{t+i}^{(0)}\big), \quad i = 0, 1, \ldots, h. \tag{11.21}$$

These impulse response functions depend on the selected shock (j), the size of the shock (c), and the location of the state variables before the shock ($\xi_{t-1} = z_{t-1}$). The latter effect stems from the impact the location of the state variables at $t-1$ has on values of the anticipated shocks. On the one hand, if we assume that the model is in steady-state at $t-1$ ($\xi_{t-1} = 0$), it is unlikely that the ZLB will be binding in some time periods from t and onwards unless c is sufficiently large (in absolute terms). On the other hand, if we assume that the nominal interest

rate is close to the lower bound at $t - 1$, then also a small size of the shock can lead to the ZLB becoming binding over the horizon for which impulse responses are computed.

To see how the impulse responses can be built, consider first period t. From equation (3.29) we have for the extended vector of state variables that

$$Y_t^{(c)} = \begin{bmatrix} z_t^{(c)} \\ \hat{r}_t^{(c)} \end{bmatrix} = P \begin{bmatrix} ce_j \\ z_{t-1} \\ \hat{r}_{t-1} \\ A_t^{(c)} \end{bmatrix},$$

where $A_t^{(c)}$ is the vector of anticipated shocks obtained from the forward-back shooting algorithm when (11.19) is taken into account. We therefore find that the initial response of Y_t is given by

$$Y_t^{(c)} - Y_t^{(0)} = P \begin{bmatrix} ce_j \\ 0 \\ 0 \\ A_t^{(c)} - A_t^{(o)} \end{bmatrix}.$$

In case $A_t^{(c)} = A_t^{(0)} = 0$ such that the ZLB is not binding in time period t and not projected to be binding over the horizon until $t + T$, then the initial responses in (11.17) and (11.20) are equal when $c = 1$, and proportional when $c \neq 1$ and nonzero.

Moving ahead, for period $t + i$ with $i \geq 1$ we find from equation (3.29) that

$$Y_{t+i}^{(c)} = \begin{bmatrix} z_{t+i}^{(c)} \\ \hat{r}_{t+i}^{(c)} \end{bmatrix} = P \begin{bmatrix} 0 \\ z_{t+i-1}^{(c)} \\ \hat{r}_{t+i-1}^{(c)} \\ A_{t+i}^{(c)} \end{bmatrix}.$$

Accordingly, the dynamic impulse responses for the state variables under the ZLB are given by

$$Y_{t+i}^{(c)} - Y_{t+i}^{(0)} = P \begin{bmatrix} 0 \\ z_{t+i-1}^{(c)} - z_{t+i-1}^{(0)} \\ \hat{r}_{t+i-1}^{(c)} - \hat{r}_{t+i-1}^{(0)} \\ A_{t+i}^{(c)} - A_{t+i}^{(0)} \end{bmatrix} = P \begin{bmatrix} 0 \\ Y_{t+i-1}^{(c)} - Y_{t+i-1}^{(0)} \\ A_{t+i}^{(c)} - A_{t+i}^{(0)} \end{bmatrix}, \quad i = 1, \ldots, h.$$

It may finally be noted that if $A_{t+i}^{(c)} = A_{t+i}^{(0)} = 0$ for all $i \geq 0$, then all the impulse responses in (11.17) and (11.20) are equal at when $c = 1$, and proportional when $c \neq 1$ and nonzero.

11.4. *Conditional Variance Decompositions*

The forecast error variance decomposition is derived from the historical forecast error decomposition in equation (11.16). The h-step ahead forecast error using data until period T can be expressed as:

$$\varepsilon_{T+h} = H'F^h(\xi_T - \xi_{T|T}) + H' \sum_{i=0}^{h-1} F^i B_0 \eta_{T+h-i} + w_{T+h}, \quad h = 1, 2, \ldots, H. \tag{11.22}$$

If we condition the forecast error ε_{T+h} on the state projection error (first term) and the measurement error (third term), the conditional h-step ahead forecast error variance is given by the

variance of the second term on the right hand side. That is,

$$V_h = \sum_{i=0}^{h-1} H' F^i B_0 B_0' (F')^i H \tag{11.23}$$
$$= V_{h-1} + R_{h-1} R_{h-1}',$$

where $R_i = H' F^i B_0$, $V_0 = 0$, and $h \geq 0$. This forecast error variance is identical to the forecast error variance that we obtain when a VAR model is written on state-space form. It is therefore analogous to a "variance decomposition" that is calculated from the impulse response functions in (11.18).

The conditional forecast error variance decomposition can be expressed as the $n \times q$ matrix

$$v_h = \left[\sum_{i=0}^{h-1} \left(R_i R_i' \odot I_n \right) \right]^{-1} \left[\sum_{i=0}^{h-1} \left(R_i \odot R_i \right) \right], \tag{11.24}$$

where $\odot$ is the Hadamard (element-by-element) product. With $e_i^{(n)}$ being the i:th column of I_n, the share of the h-step ahead conditional forecast error variance of the i:th observed variable that is explained by the j:th economic shock is given by $e_i^{(n)'} v_h e_j^{(q)}$.

YADA can also handle conditional variance decompositions for levels variables. To illustrate how this is achieved assume for simplicity that all observed variables are expressed as first differences so that the levels are obtained by accumulating the variables. This means that the h-step ahead forecast error for the levels is the accumulation of the error in (11.22), i.e.,

$$\bar{\varepsilon}_{T+h} = H' \sum_{j=1}^{h} F^j \left(\xi_T - \xi_{T|T} \right) + H' \sum_{j=1}^{h} \sum_{i=0}^{j-1} F^i B_0 \eta_{T+j-i} + \sum_{j=1}^{h} w_{T+j}, \quad h = 1, 2, \ldots, H. \tag{11.25}$$

The conditional h-step ahead forecast error for the levels variables is the second term on the right hand side of (11.25). This can be expressed as

$$\bar{\varepsilon}_{T+h}^{(c)} = \sum_{j=1}^{h} \sum_{i=0}^{j-1} R_i \eta_{T+j-i} = \sum_{j=0}^{h-1} R_j^* \eta_{T+h-j}, \tag{11.26}$$

where $R_j^* = \sum_{i=0}^{j} R_i$. It therefore follows that the conditional forecast error variance for the levels of the observed variables is

$$V_h^* = \sum_{j=0}^{h-1} R_j^* R_j^{*'} \tag{11.27}$$
$$= V_{h-1}^* + R_{h-1}^* R_{h-1}^{*'},$$

where $V_0^* = 0$. We can then define the levels variance decomposition as in equation (11.24) with R_j^* instead of R_i (and summing over $j = 0, 1, \ldots, h-1$).

By collecting the products $R_i R_i'$ into one group and all other product into a second group and dividing both sides of equation (11.27) by h, it can be rewritten as:

$$\bar{V}_h = V_h^* / h$$
$$= \sum_{i=0}^{h-1} \left(\frac{h-i}{h} \right) R_i R_i' + \sum_{m=1}^{h-1} \sum_{i=0}^{m-1} \left(\frac{h-m}{h} \right) \left(R_i R_m' + R_m R_i' \right). \tag{11.28}$$

Taking the limit of $\bar{V}_h$ as the forecast horizon approaches infinity we obtain an finite expression of the long-run forecast error covariance. We here find that

$$
\begin{aligned}
\lim_{h\to\infty} \bar{V}_h &= \sum_{i=0}^{\infty} R_i R_i' + \sum_{m=1}^{\infty} \sum_{i=0}^{m-1} \left(R_i R_m' + R_m R_i' \right) \\
&= \left(\sum_{i=0}^{\infty} R_i \right) \left(\sum_{i=0}^{\infty} R_i \right)' \\
&= H' \left(I_r - F \right)^{-1} B_0 B_0' \left(I_r - F' \right)^{-1} H.
\end{aligned}
\tag{11.29}
$$

Hence, if we divide the h-step ahead forecast error covariance matrix by h and take the limit of this expression, we find that the resulting long-run covariance matrix is equal to the cross product of the accumulated impulse responses.

These results allow us to evaluate how close the forecast error covariance matrix at the h-step horizon is to the long-run forecast error covariance matrix. The ratio between the l:th diagonal element in (11.28) and in (11.29) is an indicator of such convergence. A value close to unity can be viewed as long-run convergence at forecast horizon h, while a very large or very small value indicates a lack of convergence.

We can also use the result in (11.29) to calculate the long-run conditional forecast error variance decomposition. Letting $R_{lr} = H'(I_r - F)^{-1} B_0$, we find that

$$
v_{lr} = \left[R_{lr} R_{lr}' \odot I_n \right]^{-1} \left[R_{lr} \odot R_{lr} \right],
\tag{11.30}
$$

provides such a decomposition.

11.5. *Forecast Error Variance Decompositions*

The forecast error covariance matrix for the h-step ahead forecast of the observed vector y_{t+h} conditional on $\mathcal{Y}_t$ and a fixed value of the parameters θ is given in equation (5.39). It can be seen from this equation that this covariance matrix is time-varying. Although this is of interest when we wish to analyse the forecast errors at a particular point in time, the time-variation that the state-space model has introduced is somewhat artificial since the covariance matrix $P_{t+h|t}$ depends only on the choice of $t = 1$ and not on the time t information $\mathcal{Y}_t$. For this reason we may wish to consider an "unconditional" forecast error covariance matrix, where the value of (5.39) no longer depends on the chosen initialization period.

Assume that a unique asymptote of the forecast error covariance matrix $P_{t+h|t}$ exists and let it be denoted by P_h for $h = 0, 1, \ldots$. By equation (5.38) it follows that

$$
P_h = F P_{h-1} F' + Q, \quad h \geq 1.
\tag{11.31}
$$

Similarly, from the expression for $P_{t|t}$ in equation (5.10) we deduce that P_0 satisfies

$$
P_0 = P_1 - P_1 H \left[H' P_1 H + R \right]^{-1} H' P_1.
\tag{11.32}
$$

Let $h = 1$ in equation (11.31) and substitute for P_0 from (11.32). We then find that the asymptote P_1 must satisfy

$$
P_1 = F P_1 F' - F P_1 H \left[H' P_1 H + R \right]^{-1} H' P_1 F' + Q.
\tag{11.33}
$$

Given that we can solve for a unique asymptote P_1, all other P_h matrices can be calculated using (11.31) and (11.32).

The assumptions that (i) F has all eigenvalues inside the unit circle, and (ii) Q and R are positive semidefinite, are sufficient for the *existence* of an asymptote, P_1, that satisfies (11.33); see, e.g., Proposition 13.1 in Hamilton (1994). Let the asymptote for the Kalman gain matrix in (5.9) be denoted by K, where

$$
K = F P_1 H \left[H' P_1 H + R \right]^{-1}.
$$

The assumptions (i) and (ii) also imply that all the eigenvalues of $L = F - K H'$ lie on or inside the unit circle.

In fact, if we replace (ii) with the stronger assumption that either Q or R is positive definite, then the asymptote P_1 is also *unique*; see Proposition 13.2 in Hamilton (1994). This stronger assumption is in the case of DSGE models often not satisfied since the number of economic shocks tends to be lower than the number of state variables (Q singular) and not all observed variables are measured with error (R singular). Nevertheless, from the proof of Proposition 13.2 in Hamilton it can be seen that the stronger assumption about Q, R can be replaced with the assumption that all the eigenvalues of L lie inside the unit circle. From a practical perspective this eigenvalue condition can easily be checked once an asymptote P_1 has been found; see also Harvey (1989, Chapter 3.3).

The expression in (11.33) is a discrete algebraic *Riccati equation* and we can therefore try to solve for P_1 using well known tools from control theory. The matrix Q is typically singular for DSGE models since there are usually fewer economic shocks than state variables. Moreover, the matrix R is not required to be of full rank. For these reason, YADA cannot *directly* make use of the function `dare` from the *Control System Toolbox* in Matlab or the procedures discussed by Anderson, Hansen, McGrattan, and Sargent (1996). Instead, YADA uses a combination of iterations (with Σ_ξ as an initial value) and eigenvalue decompositions, where a solution to (11.33) is attempted in each iteration using the `dare` function for a reduction of P_1. The details on this algorithm are presented in Section 11.6.

Prior to making use of such a potentially time consuming algorithm it makes sense to first consider a very simple test. From equation (11.31) is can be seen that if $FP_0F' = 0$ then $P_1 = Q$. This test can be performed very quickly and when successful save considerable computing time. It may be noted that the condition $FP_0F' = 0$ means that the rows of F are orthogonal to the columns of P_0. This means that if F has full rank r and $P_1 = Q$, then $P_0 = 0$.

For state-space models with $R = 0$, $n \leq r$, and $n \leq q$, it is straightforward to show that L is given by

$$L = F (I_r - P_H) ,$$

where $P_H = P_1 H(H'P_1 H)^{-1} H'$ is idempotent with rank n, while $I_r - P_H$ is also idempotent but with rank $n - r$, i.e., the latter matrix has $n - r$ eigenvalues equal to unity and r eigenvalues equal to zero. Accordingly, L is equal to F times an $r \times r$ idempotent matrix of rank $n - r$. Furthermore, if L has a unit eigenvalue, then $I_r - L$ is singular.

A likely source for a unit eigenvalue of L is that the state equation has a variable in levels, but its measurement is in first differences. For instance, the An and Schorfheide model in Section 2.1 gives a measurement for Δy_t, which is mapped to the state variables $\hat{y}_t$, $\hat{y}_{t-1}$, and $\hat{z}_t$. With $\hat{y}_{t-1}$ being a definition and appearing as the last element in the vector of state variables, the corresponding L matrix has a unit root. Moreover, when the last column and row of L are removed the resulting matrix has all eigenvalues inside the unit circle. If instead the row and column of L corresponding the $\hat{y}_t$ are deleted, the derived matrix remains singular. In the case of the Smets and Wouters model, described in Section 2.4, the L matrix has four unit roots and these may be linked to having first difference measurements of real GDP, real private consumption, real investments, and real wages, while these variables appear in levels in the state equations.

From the expression in equation (5.14) we find that an asymptote P_1 satisfying the Riccati equation in (11.33) with $R = 0$ also satisfies

$$P_1 = LP_1L' + Q, \tag{11.34}$$

when $R = 0$. For the case when $P_1 = Q$ is a solution to (11.33), it follows by equation (11.34) that $LQL' = 0$ must hold.

Specifically, for the case when $n = q$ and $R = 0$, a solution to the Riccati equation (11.33) is always given by $P_1 = Q = B_0 B_0'$. For this case, $H'B_0$ is a square nonsingular matrix and

$$L = F \left(I_r - B_0 (H'B_0)^{-1} H' \right).$$

It is clear from this expression that $LB_0 = 0$ such that $LB_0 B_0' L' = 0$. Provided that F and L both have all eigenvalues inside the unit circle, it follows that the asymptote $P_1 = Q$ is unique.

The asymptotic h-step-ahead forecast error covariance matrix for the observed variables is now given by:

$$\Sigma_{\varepsilon_h} = H'P_hH + R, \quad h \geq 1. \tag{11.35}$$

The h-step ahead forecast error for the observed variables when the forecasts are performed based on the observations in period T can be expressed as:

$$\varepsilon_{T+h} = H'\left(\xi_{T+h} - \xi_{T+h|T}\right) + w_{T+h}, \quad h = 1, 2, \ldots. \tag{11.36}$$

But we can also derive an alternative asymptotic forecast error variance decomposition from (11.36). The forecast error of the state variables can — as we have already seen in, for example, equations (5.36) and (5.40) — be expressed as

$$\xi_{T+h} - \xi_{T+h|T} = F^h\left(\xi_T - \xi_{T|T}\right) + \sum_{i=0}^{h-1} F^i B_0\eta_{T+h-i},$$

where we have made use of $v_t = B_0\eta_t$. If ξ_T is observable at T, then $P_0 = 0$ and $P_1 = Q$, thus making the computations particularly simple since we do not need to use the Riccati equation solver. It therefore also follows that the reason why we need to use this solver is because we need to determine the sources behind the state variable forecast error at T.

In fact, by substituting the above decomposition of the forecast error for the state variables at T into (11.36) and taking expectations, the asymptote is

$$\Sigma_{\varepsilon_h} = H'F^h P_0(F')^h H + V_h + R, \tag{11.37}$$

where V_h is given in equation (11.23). The forecast error variance decompositions in YADA are based on evaluating this expression for fixed parameter values as well as for draws from the prior or the posterior distribution. In the cases of such draws, it should be kept in mind that the average of Σ_{ε_h} over all parameter draws is *not* the h-step ahead forecast error covariance matrix of y_{T+h} given the data, but one part of it. The remainder of the full forecast error covariance is due to parameter uncertainty; see, e.g., equation (12.5) in Section 12.1.

The decomposition in (11.37) can also be derived by recursively substituting for P_h based on equation (11.31) into equation (11.35). Although the decomposition in (11.37) requires P_0 to be determined, we may opt to leave this term otherwise undetermined, i.e., the P_0 matrix is not decomposed into the shares of the individual economic shocks. For the impact of the individual shocks on the forecast error covariances we may focus on the second term, while the third term gives us the share of the measurement errors.[80]

The forecast error variance decomposition can now be performed using equation (11.37) in combination with the tools developed in Section 11.4. Specifically, the $n \times (2 + q)$ matrix

$$\sigma_{\varepsilon_h} = \left[\Sigma_{\varepsilon_h} \odot I_n\right]^{-1} \left[\text{diag}\left(H'F^h P_0(F')^h H\right) \quad \text{diag}(R) \quad \sum_{i=0}^{h-1}\left(R_i \odot R_i\right)\right], \tag{11.38}$$

gives the h-step-ahead forecast error variance decompositions for the n observed variables (rows) in terms of the joint impact of historical state variable uncertainty, the individual q structural shocks, and the joint impact of the measurement errors. Notice that we do not attempt to separate the influence of the individial state variables or of the individual measurement errors in the decomposition but simply take them as two independent sources of forecast uncertainty.

The long-run forecast error variance decomposition can also be calculated using the above relations. First, we note that if the unique asymptote P_1 exists, then all P_h exist and are unique. Second, $\lim_{h\to\infty} P_h = \Sigma_\xi$, where Σ_ξ is obtained by solving the Lyapunov equation (5.15), so that the long-run forecast error covariance of the observed variables is simply the contemporaneous covariance matrix. The long run forecast error covariance due to measurement errors is R, while the long run forecast error covariance matrix due to economic shock j is given by $H'\Sigma_\xi^{(j)}H$, where:

$$\Sigma_\xi^{(j)} = F\Sigma_\xi^{(j)}F' + B_{0j}B_{0j}', \quad j = 1, \ldots, q, \tag{11.39}$$

---

[80] One justification for skipping the influence of economic shocks on P_0 is that these shocks must have occurred before the forecast were made and therefore concern the impact of not being able to observed the current state at the time when the forecast is produced.

i.e., the solution to the Lyapunov equation when all shocks except shock j is zero. Summing the q shocks terms we obtain:

$$\Sigma_\xi = \sum_{j=1}^{q} \Sigma_\xi^{(j)}.$$

This is easily verified by summing both sides of equation (11.39) over j and noting that

$$\sum_{j=1}^{q} B_{0j} B_{0j}' = B_0 B_0'.$$

Accordingly,

$$\lim_{h\to\infty} V_h = H' \Sigma_\xi H.$$

Finally, the state variable term in (11.37) converges to zero when h becomes very large provided that either all eigenvalues of F are inside the unit circle or all observed variables are stationary. The latter is actually already assumed since the covariance matrix is taken to be finite. Accordingly, we have that

$$\lim_{h\to\infty} \Sigma_{\varepsilon_h} = H' \Sigma_\xi H + R, \tag{11.40}$$

Furthermore, the long-run forecast error variance decomposition is given by the $n \times (q+1)$ matrix:

$$\sigma_{\varepsilon_{lr}} = \left[\left(H' \Sigma_\xi H + R \right) \odot I_n \right]^{-1} \left[\mathrm{diag}(R) \quad \mathrm{diag}\left(H' \Sigma_\xi^{(1)} H\right) \quad \cdots \quad \mathrm{diag}\left(H' \Sigma_\xi^{(q)} H\right) \right]. \tag{11.41}$$

The asymptotic forecast error variance decompositions for levels variables can also be calculated by YADA. As in Section 11.4 this is illustrated by assuming that all observed variables are expressed as first differences so that the levels are obtained by accumulating the variables. The h-step-ahead forecast error is given in equation (11.25). Substituting for the conditional h-step-ahead forecast error in equation (11.26), and taking expectations we obtain

$$\Sigma_{\varepsilon_h}^* = H' F_h^* P_0 F_h^{*\prime} H + V_h^* + hR, \tag{11.42}$$

where

$$F_h^* = F_{h-1}^* + F^h, \quad h = 1, 2, \ldots,$$

$F_0^* = 0$, and V_h^* is given in equation (11.27). We can now define levels forecast error variance decompositions as in equation (11.38)

$$\sigma_{\varepsilon_h}^* = \left[\Sigma_{\varepsilon_h}^* \odot I_n \right]^{-1} \left[\mathrm{diag}\left(H' F_h^* P_0 F_h^{*\prime} H\right) \quad \mathrm{diag}(hR) \quad \sum_{i=0}^{h-1} \left(R_i^* \odot R_i^*\right) \right]. \tag{11.43}$$

The long-run levels forecast error covariance decomposition can be determined as in equation (11.28) for the conditional long-run variance decomposition. This means that

$$\lim_{h\to\infty} \frac{1}{h} \Sigma_{\varepsilon_h}^* = R_{lr} R_{lr}' + R, \tag{11.44}$$

where the term involving state variable uncertainty converges to zero since the first term on the right hand side of (11.42) converges to a constant as h increases.[81]

11.6. *The Riccati Equation Solver Algorithm in YADA*

The algorithm used by YADA to (try to) solve the Riccati equation (11.33) for the forecast error variances uses a combination of iterative and non-iterative techniques. Let the Riccati equation be given by

$$P = FPF' - FPH\left[H'PH + R\right]^{-1} H'PF' + Q, \tag{11.45}$$

---

[81] Specifically, the state variable uncertainty term of (11.42) converges to

$$\lim_{h\to\infty} H' F_h^* P_0 F_h^{*\prime} H = H' F \left(I_r - F\right)^{-1} P_0 \left(I_r - F'\right)^{-1} F' H.$$

When the state variable uncertainty term is divided by h the corresponding expression converges to zero as h becomes very large.

where Q and R are positive semidefinite and P is used instead of P_1. I will discuss the main ingredients of the algorithm below, where each iteration follows the same steps. It is assumed that $P = Q$ has already been tested and rejected.

First, a positive semidefinite value of P is required. This value is used to evaluate the right hand side of (11.45), yielding a new value for P that we shall explore.[82] Since the new value of P may have reduced rank, we first use an eigenvalue decomposition such that for the $r \times r$ positive semidefinite matrix P

$$P = N\Lambda N',$$

where N is $r \times s$ such that $N'N = I_s$, while Λ is an $s \times s$ diagonal matrix with the non-zero eigenvalues of P. Substituting this expression for P into (11.45), premultiplying both sides by N' and postmultiplying by N, we obtain the new Riccati equation

$$\Lambda = A\Lambda A' - A\Lambda B\left[B'\Lambda B + R\right]^{-1}B'\Lambda A' + C, \tag{11.46}$$

where $A = N'FN$, $B = N'H$, and $C = N'QN$.

Next, if the matrix $B'\Lambda B + R$ in (11.46) has reduced rank, YADA performs a second eigenvalue decomposition. Specifically,

$$B'\Lambda B + R = D\Gamma D',$$

where D is $n \times d$ such that $D'D = I_d$, while Γ is a $d \times d$ diagonal matrix with the non-zero eigenvalues of $B'\Lambda B + R$. Replacing the inverse of this matrix with $D(B^{*'}\Lambda B^* + R^*)^{-1}D$, with $B^* = BD$ and $R^* = D'RD$, the Riccati equation (11.46) can be rewritten as

$$\Lambda = A\Lambda A' - A\Lambda B^*\left[B^{*'}\Lambda B^* + R^*\right]^{-1}B^{*'}\Lambda A' + C. \tag{11.47}$$

When the matrix $B'\Lambda B + R$ in (11.46) has full rank, YADA sets $D = I_n$.

YADA now tries to solve for Λ in (11.47) using `dare` from the *Control System Toolbox*; see Arnold and Laub (1984) for details on the algorithm used by `dare`. If `dare` flags that a unique solution to this Riccati equation exists, Λ^*, then YADA lets $P = N\Lambda^*N'$. When the call to `dare` does not yield a unique solution, YADA instead compares the current P to the previous P. If the difference is sufficiently small it lets the current P be the solution. Otherwise, YADA uses the current P as input for the next iteration.

11.7. Conditional Correlations and Correlation Decompositions

The basic idea behind conditional correlations is to examine the correlation pattern between a set of variable conditional on one source of fluctuation at a time, e.g., technology shocks. Following the work by Kydland and Prescott (1982) the literature on real business cycle models tended to focus on matching unconditional second moments. This was critized by several economists since a model's ability to match unconditional second moments well did not imply that it could also match conditional moments satisfactorily; see, e.g., Galí (1999).

We can compute conditional correlations directly from the state-space representation. Let column j of B_0 be denoted by B_{0j}, while the j:th economic shock is $\eta_{j,t}$. The covariance matrix for the state variables conditional on only shock j is given by

$$\Sigma_\xi^{(j)} = F\Sigma_\xi^{(j)}F' + B_{0j}B_{0j}', \tag{11.48}$$

where $\Sigma_\xi^{(j)} = E[\xi_t\xi_t'|\eta_{j,t}]$. We can estimate $\Sigma_\xi^{(j)}$ at θ by either solving (11.48) analytically through the vec operator, or numerically using the doubling algorithm discussed in Section 5.3. The conditional correlation for the observed variables can thereafter be calculated from the conditional covariance matrix

$$\Sigma_y^{(j)} = H'\Sigma_\xi^{(j)}H. \tag{11.49}$$

As an alternative to conditional population moments we can also consider simulation methods to obtain estimates of conditional sample moments. In that case we can simulate a path for the state variables conditional on only shock j being non-zero, by drawing T values for $\eta_{j,t}$ and

---

[82] In the event that $H'PH + R$ has reduced rank, its "inverse" is replaced by $S(S'[H'PH + R]S)^{-1}S'$ where S is obtained from the eigenvalue decomposition $H'PH + R = S\Gamma S'$.

letting

$$\xi_t^{(s)} = F\xi_{t-1}^{(s)} + B_{0j}\eta_{j,t}, \quad t = 1,\ldots,T. \tag{11.50}$$

where $\xi_0^{(s)}$ is drawn from $N(0,\Sigma_\xi^{(j)})$. The conditional sample correlations for simulation s can now be computed from the covariance matrix:

$$\hat{\Sigma}_y^{(j,s)} = \frac{1}{T}\sum_{t=1}^{T} H'\xi_t^{(s)}\xi_t^{(s)\prime}H, \quad s = 1,\ldots,S. \tag{11.51}$$

By repeating the simulations S times we can estimate the distribution of the conditional sample correlations for a given θ.

Correlation decompositions have been suggested by Andrle (2010b) as a means of decomposing autocorrelations for pairs of variables in a linear model. These decompositions relate the conditional correlations to the correlations by weighting them with the ratio of the conditional variances and the variances. From equation (5.42) we know that the state-space model provides us with

$$\Sigma_y(h) = \begin{cases} H'\Sigma_\xi H + R, & \text{if } h = 0, \\ H'F^h\Sigma_\xi H, & \text{otherwise.} \end{cases} \tag{11.52}$$

In addition, the covariance matrix of the state variables is related to the conditional covariance matrices in (11.48) linearly with

$$\Sigma_\xi = \sum_{j=1}^{q} \Sigma_\xi^{(j)}.$$

This means that

$$\Sigma_y^{(j)}(h) = \begin{cases} H'\Sigma_\xi^{(j)}H, & \text{if } h = 0, \\ H'F^h\Sigma_\xi^{(j)}H, & \text{otherwise,} \end{cases} \tag{11.53}$$

while

$$\Sigma_y(h) = \sum_{j=1}^{q} \Sigma_y^{(j)}(h) + \mathbb{I}(h=0)R.$$

The indicator function $\mathbb{I}(h=0)$ is unity if $h = 0$ and zero otherwise.

To simplify notation let us consider the case when the model does not have any measurement errors , i.e., when $R = 0$. Let $\Sigma_{y_k,y_l}(h)$ denote element (k,l) of $\Sigma_y(h)$. It follows that the correlation between $y_{k,t}$ and $y_{l,t-h}$ is given by

$$\rho_{y_k,y_l}(h) = \frac{\Sigma_{y_k,y_l}(h)}{\sqrt{\Sigma_{y_k,y_k}(0)\Sigma_{y_l,y_l}(0)}} = \sum_{j=1}^{q} \frac{\Sigma_{y_k,y_l}^{(j)}(h)}{\sqrt{\Sigma_{y_k,y_k}(0)\Sigma_{y_l,y_l}(0)}}, \tag{11.54}$$

where $\Sigma_{y_k,y_k}(0)$ is the variance of $y_{k,t}$. Assuming that all conditional variances $\Sigma_{y_k,y_k}^{(j)}(0)$ are positive, Andrle (2010b) notes that the right hand side of (11.54) can be rewritten as

$$\begin{aligned} \rho_{y_k,y_l}(h) &= \sum_{j=1}^{q} \sqrt{\frac{\Sigma_{y_k,y_k}^{(j)}(0)\Sigma_{y_l,y_l}^{(j)}(0)}{\Sigma_{y_k,y_k}(0)\Sigma_{y_l,y_l}(0)}} \frac{\Sigma_{y_k,y_l}^{(j)}(h)}{\sqrt{\Sigma_{y_k,y_k}^{(j)}(0)\Sigma_{y_l,y_l}^{(j)}(0)}} \\ &= \sum_{j=1}^{q} \omega_{y_k}^{(j)}\omega_{y_l}^{(j)}\rho_{y_k,y_l}^{(j)}(h). \end{aligned} \tag{11.55}$$

In other words, the correlation between $y_{k,t}$ and $y_{l,t-h}$ is equal to the sum over all shocks j of the product of the shares of the standard deviations of y_k and y_l due to shock j ($\omega_{y_k}^{(j)}$ times $\omega_{y_l}^{(j)}$) and the conditional correlation between the variables ($\rho_{y_k,y_l}^{(j)}(h)$). The two standard deviation shares are non-negative (positive if all the conditional variances are positive) and sum to at

most unity over the different shocks (unity when $R = 0$) and can be interpreted as the weights needed to obtain the correlations from the conditional correlations.

In case one of the conditional variances is zero for some shock, we let all terms be zero for that shock in (11.55). Moreover, to deal with non-zero measurement errors we add the correlation $R_{y_k,y_l} / \sqrt{\Sigma_{y_k,y_k}(0)\Sigma_{y_l,y_l}(0)}$ to the expressions for $\rho_{y_k,y_l}(0)$ that involve sums of conditional covariances. Finally, the population moments used in the above expressions can be substituted for simulated sample moments, such as (11.51), to decompose the sample correlations.

11.8. *Historical Observed Variable Decompositions*

Given the smooth estimates of the economic shocks (11.1) and the measurement errors (5.34) we can also calculate a historical decomposition of the observed variables. Specifically, we know that

$$y_t = A'x_t + H'\xi_{t|T} + w_{t|T}.$$

The estimated share of y_t due to measurement errors is therefore simply $w_{t|T}$, while the shares due to the various economic shocks need to be computed from the smoothed state variables and the state equation. Specifically, we have that

$$\xi_{t|T} = F^t \xi_{0|T} + \sum_{i=0}^{t-1} F^i B_0 \eta_{t-i|T}. \tag{11.56}$$

To estimate the impact of $\xi_{0|T}$ at $t = 1$, we compute the term $F\xi_{0|T}$ as a residual of $\xi_{1|T} - B_0\eta_{1|T}$. For $t = 2, \ldots, T$, the initial state term at t can be computed recursively by multiplying the $t - 1$ initial state term by F. Similar recursive calculations can be performed for the individual shocks, with the $t = 1$ decomposition being obtained from $B_0\eta_{1|T}$, while the decomposition at t is obtained from $B_0\eta_{t|T}$ plus F times the decomposition at $t - 1$.

From (11.56) we can decompose the smooth estimates of the state variables into terms due to the q economic shocks. Substituting this expression into the measurement equation based on smoothed estimates, we obtain

$$y_t = A'x_t + H'F^t\xi_{0|T} + \sum_{i=0}^{t-1} H'F^i B_0 \eta_{t-i|T} + w_{t|T}, \quad t = 1, \ldots, T. \tag{11.57}$$

Accordingly, it is straightforward to decompose the observed variables into terms determined by (i) the deterministic variables, (ii) the initial state estimate ($\xi_{0|T}$), (iii) the q economic shocks, and (iv) the measurement errors.

The decomposition in (11.57) can also be generalized into decompositions for all possible subsamples $\{t_0 + 1, \ldots, T\}$, where $t_0 = 0, 1, \ldots T - 1$. In the decomposition above the choice is $t_0 = 0$. The generalization into an arbitrary t_0 gives us:

$$y_t = A'x_t + H'F^{t-t_0}\xi_{t_0|T} + \sum_{i=0}^{t-t_0-1} H'F^i B_0 \eta_{t-i|T} + w_{t|T}, \qquad t = t_0 + 1, \ldots, T. \tag{11.58}$$

This provides a decomposition of the observed variables y_t into (i) deterministic variables, (ii) the estimated history of the state until t_0 ($\xi_{t_0|T}$), (iii) the q economic shocks from $t_0 + 1$ until t, and (iv) the measurement error. This decomposition is called *sample-split* in YADA, where t_0 is the last time period before and $t_0 + 1$ the first time period after the split. This may be of interest when the analysis focuses on structural shocks that have occurred after a certain point in time, while the shocks that are estimated for earlier time periods may be grouped together.

Alternative to a fixed t_0-based decomposition, we may consider a decomposition based on $t - \tau$, where $\tau \geq 1$ is a fixed lag. Specifically,

$$y_t = A'x_t + H'F^\tau\xi_{t-\tau|T} + \sum_{i=0}^{\tau-1} H'F^i B_0 \eta_{t-i|T} + w_{t|T}, \qquad t = \tau, \ldots, T. \tag{11.59}$$

The second term on the right hand side may be further decomposed into shocks that are estimated for periods up to $t - \tau$ as well as the impact from the estimated initial condition as in equation (11.56). The full decomposition in (11.59) is called *recent & past* in YADA, where τ is

the number of lagged time periods until the past is reached from t and all time periods after $t - \tau$ are recent relative to t, the current time period. This decomposition may be of interest when the analysis wishes to focus on the very recent shocks for each time period and those having occurred in the past may simply be grouper together.

11.9. Parameter Scenarios

Parameter scenarios are used to examine the impact that changes in some parameters has on the behavior of the variables or on the economic shocks. Let θ_b be the baseline value of the parameter vector and θ_a the alternative value. The baseline value can, for example, be the posterior mode estimate of θ.

Assuming that the model has a unique and convergent solution at both θ_b and at θ_a, YADA provides two approaches for parameter scenario analysis. The first is to calculate smooth estimates of the economic shocks under the two parameter vectors. The path for the observed variables are, in this situation, the same for both parameter vectors.[83]

The second approach takes the economic shocks and measurement errors based on θ_b as given and calculates the implied observed variables from the state-space representation with the parameter matrices A, H, F and B_0 determined by the alternative θ_a vector.[84] This path can then be compared with the actual data for the observed variables.

In addition, to changes in the *estimated parameters*, θ, YADA also allows for changes to any *calibrated parameters* when conducting parameter scenarios. Such an analysis is particularly useful as it allows the model builder to examine how sensitive the estimates of the unobservables (state variables and structural economic shocks) are to the assumptions about the calibrated parameters when the estimated parameters may be fixed at, say, the posterior mode values. In this sense, parameter scenarios may be regarded as both a scenario and a sensitivity tool.

11.10. Controllability and Observability

Two properties that are typically of interest when working with state-space models are *controllability* and *observability*. The first concept stems from control theory where the issue is if the control variables of the model can be used to manipulate the state variables to particular values. Formally we may say that a system with internal state ξ is controllable if and only if the system states can be changed by changing the input (η). Similarly, a system with initial state ξ_{t_0} is observable if and only if this state can be determined from the system output y_t that has been observed through the interval $t_0 < t < t_1$. If the initial state cannot be determined, the system is said to be unobservable.

For time-invariant state-space models it is well known that the system (5.1)-(5.2) is controllable if and only if the $r \times rq$ matrix

$$C = \begin{bmatrix} B_0 & FB_0 & \cdots & F^{r-1}B_0 \end{bmatrix}, \tag{11.60}$$

has rank r. Similarly, the state-space model is observable if and only if the $r \times nr$ matrix

$$O = \begin{bmatrix} H & F'H & \cdots & \left(F'\right)^{r-1} H \end{bmatrix}, \tag{11.61}$$

has rank r; see, e.g., Harvey (1989, p. 115).

We generally do not expect DSGE model to be controllable or observable since r is expected to be substantially larger than n or q. Nevertheless, the ranks of the matrices C and O are informative about the degree of controllability and observability.

---

[83] Alternatively, one may wish to compare smooth estimates of the state variables under the two parameter vectors.

[84] One alternative to the second approach is to simulate the model under θ_a by drawing the economic shocks and the measurement errors from their assumed distribution a large number of times, compute the implied path for the observed variables, and then compare, say, the average of these paths to the actual data for the observed variables.

To address the issue if the economic shocks and measurement errors of the state-space representation can be uncovered from a VAR representation of the observed variables, Fernández-Villaverde, Rubio-Ramírez, Sargent, and Watson (2007) provide a simple condition for checking this.

To cast equations (5.1) and (5.2) into their framework we first rewrite the measurement error as:

$$w_t = \Phi\omega_t,$$

where $R = \Phi\Phi'$ while $\omega_t \sim N(0, I)$. The matrix Φ is of dimension $n \times m$, with $m = \text{rank}(R) \leq n$.

Substituting for ξ_t from the state equation into the measurement equation we get:

$$\begin{aligned} y_t &= A'x_t + H'F\xi_{t-1} + H'B_0\eta_t + \Phi\omega_t \\ &= A'x_t + H'F\xi_{t-1} + D\varphi_t, \end{aligned} \tag{11.62}$$

where the residual vector $\varphi_t = [\eta_t' \ \omega_t']'$ while $D = [H'B_0 \ \Phi]$ is of dimension $n \times (q + m)$.

The state equation can likewise be expressed as:

$$\xi_t = F\xi_{t-1} + B\varphi_t, \tag{11.63}$$

where $B = [B_0 \ 0]$ is an $r \times (q + m)$ matrix.

The state-space representation has a VAR representation when φ_t can be retrieved from the history of the observed variables. The first condition for this is that D has rank $q + m$ such that a Moore-Penrose inverse $D^+ = (D'D)^{-1}D'$ exists. A necessary condition for the existence of this inverse is clearly that $n \geq q + m$, i.e., that we have at least as many observed variables as there are economic shocks and unique measurement errors.

Assuming D^+ exists we can write φ_t in (11.62) as a function of y_t, x_t, and ξ_{t-1}. Substituting the corresponding expression into the state equation and rearranging terms yields

$$\xi_t = \left(F - BD^+H'F\right)\xi_{t-1} + BD^+\left(y_t - A'x_t\right). \tag{11.64}$$

If the matrix $G = F - BD^+H'F$ has all eigenvalues inside the unit circle, then the state variables are uniquely determined by the history of the observed (and the exogenous) variables. The state vector ξ_t can therefore be regarded as known, and, moreover, this allows us to express the measurement equation as an infinite order VAR model. Accordingly, the economic shocks and the measurement errors are uniquely determined by the history of the observed data (and the parameters of the DSGE model). The eigenvalue condition is called a "poor man's invertibility condition" by Fernández-Villaverde et al. (2007). The problem of uncovering the economic shocks from a VAR concerns the issue of fundamentalness (when they can be recovered from a VAR) and non-fundamentalness (when they cannot be recovered from a VAR) of economic models.

The poor mans's invertibility condition is only sufficient for fundamentalness and it has been applied in a number studies; see, e.g., Leeper, Walker, and Yang (2013), Schmitt-Grohé (2010), and Sims (2012). The sufficiency property means that an economic model can support fundamentalness although the invertibility condition fails. An example is provided by Franchi and Paruolo (2012),[85] who also derive both necessary and sufficient conditions for fundamentalness under the assumption that $n = q + \text{rank}[R]$, i.e., the number of shocks and measurement errors is equal to the number of observed variables (the square case).

Franchi and Paruolo (2012) provide a formal guide for checking if a state-space model is invertible or not. In its general form, this guide involves checking properties of the inverse of $I - Gz$, where z is a complex number, and can therefore be messy to handle in practise when the model has multiple unstable eigenvalues of G. However, Franchi and Paruolo state simple cases when the poor man's invertibility condition is necessary as well as sufficient. Once such case which is particularly important concerns the eigenvalues of F. Corollary 5.3 in Franchi and Paruolo here posits that if all the eigenvalues of F are stable (inside the unit disc), then

---

[85] The example is technically correct, but economically obscure since it states that the *change* in (growth rate of) consumption is proportional to the *level* of labor income.

the state-space model is invertible if and only if G is stable.[86] Many DSGE models satisfy the requirement that F is stable, thereby justifying the use of the poor man's invertibility condition for checking for fundamentalness in the square case.

11.12. *Fisher's Information Matrix*

It is well known that when the vector of parameters θ is estimated with maximum likelihood, then the inverse of Fisher's information matrix is the asymptotic covariance matrix for the parameters. If this matrix has full rank when evaluated at the true parameter values, the parameters are said to be locally identified; cf. Rothenberg (1971).

Since DSGE models are typically estimated with Bayesian methods, identification problems can likewise be viewed through the behavior of the Hessian of the log-posterior distribution. However, such problems can be dealt with by changing the prior such that the log-posterior has more curvature. Still, use of prior information to deal with identification problems is unsatisfactory. One way to examine how much information there is in the data about a certain parameter is to compare the plots of the prior and the posterior distributions. If these distributions are very similar, then it is unlikely that the data is very informative about this particular parameter.

The comparison between prior and posterior distributions require that we have access to draws from the posterior. Since drawing from the posterior may be very time consuming, it may be useful to consider an alternative approach. In this respect, Fisher's information matrix may also be useful when considering identification issues from a Bayesian perspective. This approach has been investigated in a series of articles by Nikolay Iskrev; see Iskrev (2008, 2010).[87]

Using standard results from matrix differential algebra (see Magnus and Neudecker, 1988) it has been shown by Klein and Neudecker (2000) that with $\tilde{y}_t = y_t - y_{t|t-1}$ the second differential of the log-likelihood function in (5.18) can be written as:

$$d^2 \ln L\big(\mathcal{Y}_T; \theta\big) = \frac{1}{2} \sum_{t=1}^{T} \text{tr}\bigg\{ \Sigma_{y,t|t-1}^{-1} \big(d\Sigma_{y,t|t-1}\big) \Sigma_{y,t|t-1}^{-1} \big(d\Sigma_{y,t|t-1}\big) \bigg\} - \sum_{t=1}^{T} \big(d\tilde{y}_t\big)' \Sigma_{y,t|t-1}^{-1} \big(d\tilde{y}_t\big) +$$

$$- \sum_{t=1}^{T} \text{tr}\bigg\{ \Sigma_{y,t|t-1}^{-1} \big(d\Sigma_{y,t|t-1}\big) \Sigma_{y,t|t-1}^{-1} \big(d\Sigma_{y,t|t-1}\big) \Sigma_{y,t|t-1}^{-1} \tilde{y}_t \tilde{y}_t' \bigg\} +$$

$$+ 2 \sum_{t=1}^{T} \text{tr}\bigg\{ \Sigma_{y,t|t-1}^{-1} \big(d\Sigma_{y,t|t-1}\big) \Sigma_{y,t|t-1}^{-1} \big(d\tilde{y}_t\big) \tilde{y}_t' \bigg\} - \sum_{t=1}^{T} \text{tr}\bigg\{ \Sigma_{y,t|t-1}^{-1} \big(d^2\tilde{y}_t\big) \tilde{y}_t' \bigg\}.$$

Taking the expectation of both sides with respect to θ, the Lemma in Klein and Neudecker implies that the last two terms on the right hand side are zero. Moreover, with $E[\tilde{y}_t \tilde{y}_t'; \theta] = E[\Sigma_{y,t|t-1}; \theta]$, the above simplifies to

$$E_\theta\Big[d^2 \ln L\big(\mathcal{Y}_T; \theta\big)\Big] = -\frac{1}{2} \sum_{t=1}^{T} E_\theta\bigg[\big(d\text{vec}\big(\Sigma_{y,t|t-1}\big)\big)' \Big[\Sigma_{y,t|t-1}^{-1} \otimes \Sigma_{y,t|t-1}^{-1}\Big] d\text{vec}\big(\Sigma_{y,t|t-1}\big) \bigg] +$$

$$- \sum_{t=1}^{T} E_\theta\Big[\big(d\tilde{y}_t\big)' \Sigma_{y,t|t-1}^{-1} d\tilde{y}_t \Big]$$

The matrix $\Sigma_{y,t|t-1}$ is a differentiable function of the parameters θ such that

$$d\text{vec}\big(\Sigma_{y,t|t-1}\big) = \frac{\partial \text{vec}\big(\Sigma_{y,t|t-1}\big)}{\partial \theta'} d\theta.$$

Similarly, we let

$$d\tilde{y}_t = \frac{\partial \tilde{y}_t}{\partial \theta'} d\theta.$$

---

[86] Another example when the poor man's invertibility condition is both necessary and sufficient is when the model is controllable and observable; see Franchi and Paruolo (2012, Corollary 5.4). However, we generally do not expect DSGE models to satisfy these properties.

[87] See also Beyer and Farmer (2004), Canova and Sala (2009), Consolo, Favero, and Paccagnini (2009), Komunjer and Ng (2011), and Bonaldi (2010) for discussions of identifiability issues in DSGE models.

Collecting these results, the Fisher's information matrix may be expressed as:

$$-E_\theta\left[\frac{\partial^2 \ln L\left(\mathcal{Y}_T; \theta\right)}{\partial\theta\partial\theta'}\right] = \sum_{t=1}^{T} E_\theta\left[\left(\frac{\partial\tilde{y}_t}{\partial\theta'}\right)'\Sigma_{y,t|t-1}^{-1}\frac{\partial\tilde{y}_t}{\partial\theta'}\right] +$$

$$+ \frac{1}{2}\sum_{t=1}^{T} E_\theta\left[\left(\frac{\partial\mathrm{vec}\left(\Sigma_{y,t|t-1}\right)}{\partial\theta'}\right)'\left[\Sigma_{y,t|t-1}^{-1}\otimes\Sigma_{y,t|t-1}^{-1}\right]\frac{\partial\mathrm{vec}\left(\Sigma_{y,t|t-1}\right)}{\partial\theta'}\right].$$

$$(11.65)$$

The partial derivatives of $\tilde{y}_t$ and $\Sigma_{y,t|t-1}$ with respect to A, H, R, F, Q (the reduced form parameters) can be determined analytically. The form depends on how the initial conditions for the state variables relate to the parameters; see Zadrozny (1989, 1992) for details. The step from reduced form parameters to θ is explained by Iskrev (2008).

Instead of making use of these analytic results, YADA currently computes numerical derivatives of $\tilde{y}_t$ and $\Sigma_{y,t|t-1}$ with respect to θ.

11.13. *A Rank Revealing Algorithm*

In practise it may be difficult to assess the rank of the information matrix. In most cases we may expect that the determinant is positive but perhaps small. One approach to disentangling the possible parameters that are only weakly identified is to compute the correlations based on the information matrix. If two parameters are highly correlated in absolute terms, it may be difficult to identify both. However, it should be kept in mind that the information matrix can be of full rank also when the correlation between some pairs of parameters is close or equal to unity. Moreover, the matrix can have less than full rank also when all correlation pairs are less than unity. Hence, the information content in such correlations is of limited value when it comes the determining the rank of the information matrix.

As pointed out by Andrle (2010a), a natural tool for examining the rank properties of any real valued matrix is the *singular value decomposition*; see, e.g., Golub and van Loan (1983). For symmetric matrices the decomposition simplifies to the eigenvalue decomposition. The determinant of a square matrix is equal to the product of its eigenvalues and, hence, the eigenvectors of the smallest eigenvalues of the information matrix give the linear combinations of the parameters that are the most difficult to identify. Provided that an eigenvalue is very small and its eigenvector has large weights on a pair of parameters, then the correlation between these two parameters will be large in absolute terms. However, if the eigenvector has large weights for more than two parameters, then the correlations between pairs of these parameters need not be large. This suggests that correlations are potentially unreliable for determining weakly identified parameters, but also that the eigenvectors of the smallest eigenvalues do not suffer from such a problem.

In fact, Andrle (2010a) has suggested a heuristic procedure, based on Q-R factorization with column pivoting (Golub and van Loan, 1983, Section 6.4), that sorts the parameters from the "most" identified to the "least" identified. For a positive definite matrix A, the rank revealing approach may be applied as follows:[88]

(1) Compute the singular value decomposition of $A = USV'$ ($= VSV'$ since A is symmetric) where U and V are orthogonal ($U'U = I$ and $V'V = I$). It is assumed that the singular values in S are sorted from the largest to the smallest.

(2) Calculate the Q-R factorization with column pivoting on V', i.e.,

$$V'P = QR,$$

where P is a permutation matrix, Q is orthogonal ($Q'Q = I$), and R is upper triangular.

(3) The last column of the permutation matrix P is a vector with unity in position j and zeros elsewhere, linking column j of A to the smallest singular value. Discard row and column j of A and return to (1) until A is a scalar.

---

[88] Note that the algorithm may also be applied when A has reduced rank or when it is not a square matrix; see Andrle (2010a) for details on how it then needs to be modified.

The first column that is removed from A is the least identified and the last to remain is the most identified. To keep track of the column numbers from the original $m \times m$ matrix A, we use the vector $v = [1, 2, \ldots, m]$ as the column number indicator. Now, element j of v determines the column number c_j of A that is linked to the smallest singular value. In step (3) we therefore discard element j from v and row and column c_j from A before returning to step (1).

As an alternative to performing the Q-R factorization with column pivoting on V', one may calculate the factorization directly on A.[89] In that case, the singular value decomposition can be skipped and only the last two steps need be performed. This latter option is the preferred approach to parameter ordering in YADA. The singular value decomposition is nevertheless conducted for the full A matrix since, as discussed above, this may provide useful information about which parameters can be linked to the large and to the small eigenvalues, respectively.

11.14. *Monte Carlo Filtering*

The tools we have discussed above share the condition that the underlying value of θ is such that the DSGE model has a unique and convergent solution; see Section 3 for further details. At the same time, indeterminacy and no solution are also interesting properties that are worthwhile to investigate further; see, e.g., the discussions in Burmeister (1980), Clarida, Galí, and Gertler (2000), Lubik and Schorfheide (2004, 2007b), and Beyer and Farmer (2007). One approach to analysing the sensitivity of the stability properties of a DSGE model to its parameters, suggested by Ratto (2008), is to apply *Monte Carlo filtering*.

The basic of idea of Monte Carlo filtering is to first generate S draws of the parameters from their prior distribution. We may denote these draws $\theta^{(s)}$, where $s = 1, 2, \ldots, S$. Second, the characteristics of the model that we wish to investigate are examined for each such draw. In the case of stability, we assign a draw $\theta^{(s)}$ to the group $\mathcal{S}$ if the model has a unique and convergent solution at $\theta^{(s)}$, and to the group $\bar{\mathcal{S}}$ otherwise. This gives us N values of θ that belong to $\mathcal{S}$ and $\bar{N}$ values that belong to $\bar{\mathcal{S}}$, with $N + \bar{N} = S$.

In order to locate the parameters of the DSGE model that mostly drive the model to the target behavior under $\mathcal{S}$, the empirical marginal distributions $F_N(\theta_i|\mathcal{S})$ and $F_{\bar{N}}(\theta_i|\bar{\mathcal{S}})$ can be compared for each parameter θ_i. If the two distributions are significantly different, then Ratto suggests that θ_i is a key factor in driving the model behavior. In this case, there will be subsets of values in its predefined range which are more likely to fall under $\mathcal{S}$ than under $\bar{\mathcal{S}}$. Similarly, if the two distributions are not significantly different, then θ_i is not important. In that case, any value in its predefined range is either likely to fall into either group or all values in its support imply the target behavior.[90]

Provided that N and $\bar{N}$ are both larger than zero (and sufficiently large), Ratto (2008) suggests to apply to Kolmogorov-Smirnov test when examining if the two distribution are different or not; see, e.g., Rao (1973, Chapter 6f) for details. For the cumulative distribution functions of θ_i, the statistic is

$$D_{N,\bar{N}} = \sup_{\theta_i} \left| F_N(\theta_i|\mathcal{S}) - F_{\bar{N}}(\theta_i|\bar{\mathcal{S}}) \right|. \tag{11.66}$$

The asymptotic behavior of the test statistic is given by

$$\sqrt{\frac{N\bar{N}}{N + \bar{N}}} D_{N,\bar{N}} \implies \sup_{t \in [0,1]} |B(t)| = K, \tag{11.67}$$

---

[89] In practise, a factorization based on A seems to yield the same ordering of the columns as a factorization based on $S^{1/2}V'$.

[90] An example of this latter case is trivially when the parameter concerns a first order autoregression and its support is given by $(-1, 1)$.

where $\Rightarrow$ denotes weak convergence, $B(t)$ is a Brownian bridge,[91] and K has a Kolmogorov distribution. The cdf of K is given by

$$\Pr(K \le x) = 1 - 2 \sum_{i=1}^{\infty} (-1)^{i-1} \exp\left(-2i^2 x^2\right). \tag{11.68}$$

This means that the test value on the left hand side of (11.67) is simply substituted for x in (11.68), where the infinite sum is approximated by an finite sum based on the exponential term being sufficiently close to zero. An approximate p-value for the Kolmogorov-Smirnov test that the two distributions are equal can therefore be computed from such a truncation.[92]

When parameters which are important for the target behavior have been located, a graphical illustration may prove useful. It is well known that certain parameters of the monetary policy reaction function are candidates for yielding, e.g., a unique convergent solution or indeterminacy. In the case of the An and Schorfheide model in Section 2.1, the parameter ψ_1 can produce such behavior. Scatter plot of pairs of parameters may provide useful information about the regions over which the model has a unique convergent solution, indeterminacy, and when there is no convergent solution.

11.15. *Moments of the Observed Variables*

The population autocovariances conditional on a given value of θ (and the selected model) have already been presented in Section 5.8. In this section we shall discuss how one can compute the moments of the observed variables conditional only on the model. To this end, let $p(\theta)$ be a proper density function of the DSGE model parameters. From a notational point of view I have suppressed additional variables from the density, but it should nevertheless be kept in mind that it could either be the prior or the posterior density of the model parameters.

Recall that the population mean of the observed variables conditional on θ and the exogenous variables is:

$$E\left[y_t | x_t, \theta\right] = A' x_t = \mu_{y|\theta} x_t.$$

The population mean conditional on the exogenous variables (and the model) is therefore given by

$$E\left[y_t | x_t\right] = \mu_y x_t = \left(\int_{\theta \in \Theta} \mu_{y|\theta} p(\theta) d\theta\right) x_t, \tag{11.69}$$

where Θ is the domain of the parameters and μ_y is an $n \times k$ matrix. In the event that A is calibrated, then $\mu_{y|\theta} = \mu_y$ for all $\theta \in \Theta$.

Turning next to the covariance matrix of y_t conditional on θ and the exogenous variables, we have from equation (5.42) that

$$E\left[\left(y_t - \mu_{y|\theta} x_t\right)\left(y_t - \mu_{y|\theta} x_t\right)' | x_t, \theta\right] = H' \Sigma_\xi H + R = \Sigma_{y|\theta}.$$

From the law of iterated expectations and by utilizing the mean expansion

$$y_t - \mu_y x_t = y_t - \mu_{y|\theta} x_t + \left(\mu_{y|\theta} - \mu_y\right) x_t,$$

---

[91] Recall that a Brownian bridge, $B(t)$, is defined from a Wiener process, $W(t)$, over the unit interval $t \in [0,1]$ such that

$$B(t) = W(t) - tW(1),$$

where $W(t)$ is normally distributed with mean 0, variance t, and have stationary and independent increments.

[92] An equivalent formulation of equation (11.68) is

$$\Pr(K \le x) = \frac{\sqrt{2\pi}}{x} \sum_{i=1}^{\infty} \exp\left(-(2i-1)^2 \pi^2 / (8x^2)\right).$$

This expression is better suited when x is small. The mean of the Kolmogorov distribution is $\mu_K = \sqrt{\pi/2} \ln(2) \approx 0.868731$, while the variance is $\sigma_K^2 = (\pi^2/12) - \mu_K^2 \approx 0.260333^2$. For discussions on computational issues, see, e.g., Marsaglia, Tsang, and Wang (2003).

the population covariance matrix for y_t conditional only on the exogenous variables can be shown to satisfy

$$E\left[\left(y_t - \mu_y x_t\right)\left(y_t - \mu_y x_t\right)'|x_t\right] = E\left[E\left[\left(y_t - \mu_y x_t\right)\left(y_t - \mu_y x_t\right)'|x_t, \theta\right]\right]$$
$$+ E\left[\left(\mu_{y|\theta} - \mu_y\right) x_t x_t' \left(\mu_{y|\theta} - \mu_y\right)'|x_t\right].$$

This may be written more compactly as

$$\Sigma_y = E\left[\Sigma_{y|\theta}|x_t\right] + C\left[\mu_{y|\theta} x_t|x_t\right]. \tag{11.70}$$

That is, the covariance matrix of y_t (conditional on the exogenous variables) is equal to the mean of the covariance matrix conditional on the parameters and the covariance matrix of the mean conditional on the parameters. If $\mu_{y|\theta} = \mu_y$ for all $\theta \in \Theta$, then the second term is zero. Similar expressions can be determined for all the autocovariances as well.

To keep notation as simple as possible, I assume that $n = 1$ and that $x_t = 1$ when examining higher central moments. Concerning the third central moment, the law of iterated expectations and the mean expansion can be applied to show that

$$E\left[\left(y_t - \mu_y\right)^3\right] = E\left[E\left[\left(y_t - \mu_{y|\theta}\right)^3|\theta\right]\right] + E\left[\left(\mu_{y|\theta} - \mu_y\right)^3\right] + 3C\left[\sigma_{y|\theta}^2, \mu_{y|\theta}\right]. \tag{11.71}$$

Hence, skewness[93] of y_t is equal to the mean of skewness of y_t conditional on the parameters plus skewness of the mean conditional on the parameters plus three times the covariance between the conditional variance and the conditional mean.[94] It can now be deduced that if $\mu_{y|\theta} = \mu_y$ for all $\theta \in \Theta$, then the second and third term on the right hand side of (11.71) are both zero. In addition, the third term is zero if $\sigma_{y|\theta}^2 = \sigma_y^2$ for all $\theta \in \Theta$, but this case is only expected to occur in state-space models where the matrices (H, R, F, B_0) are fully determined from calibrated parameters.

For the log-linearized DSGE model with normally distributed structural shocks and measurement errors we know that $y_t|\theta$ has zero skewness since it is normally distributed. If the distribution of the conditional mean is skewed, then y_t inherits the skewness from the conditional mean but its skewness is also affected by the covariance between the conditional variance and the conditional mean. Since both these conditional moments may be non-linear functions of θ we typically do not know the sign of the third term on the right hand side in equation (11.71).

The fourth central moment of y_t can likewise be determined from the law of iterated expectations and the mean expansion. This gives us

$$E\left[\left(y_t - \mu_y\right)^4\right] = E\left[E\left[\left(y_t - \mu_{y|\theta}\right)^4|\theta\right]\right] + E\left[\left(\mu_{y|\theta} - \mu_y\right)^4\right]$$
$$+ 4C\left[E\left[\left(y_t - \mu_{y|\theta}\right)^3|\theta\right], \mu_{y|\theta}\right] + 6E\left[\sigma_{y|\theta}^2\left(\mu_{y|\theta} - \mu_y\right)^2\right]. \tag{11.72}$$

Hence, kurtosis[95] is equal to the expectation of conditional kurtosis plus kurtosis of the conditional mean plus 4 times the covariance between conditional skewness and the conditional mean plus 6 times the expected value of the product between the conditional variance and the square of the conditional mean in deviation from the mean. If $\mu_{y|\theta} = \mu_y$ for all $\theta \in \Theta$, then the last three terms on the right hand side of (11.72) are zero, and kurtosis of y_t is given by the

---

[93] Skewness is usually defined as the third moment of the standardized random variable $(y_t - \mu_y)/\sigma_y$. This means that skewness is equal to the ratio between the third central moment of the random variable y_t and the standard deviation of y_t to the power of three.

[94] Since $E[\sigma_y^2(\mu_{y|\theta} - \mu_y)] = 0$, the third term on the right hand side in equation (11.71) comes from the relationship

$$E\left[\sigma_{y|\theta}^2\left(\mu_{y|\theta} - \mu_y\right)\right] = E\left[\left(\sigma_{y|\theta}^2 - \sigma_y^2\right)\left(\mu_{y|\theta} - \mu_y\right)\right].$$

[95] Kurtosis is usually defined as the fourth moment of the standardized random variable $(y_t - \mu_y)/\sigma_y$. This means that kurtosis is actually equal to the fourth central moments divided by the square of the variance.

mean conditional kurtosis. Furthermore, the third term is zero when $y_t|\theta$ is symmetric so that conditional skewness is zero.

Since $y_t|\theta$ is normal for the state-space model with normal measurement errors and structural shocks, it follows from the properties of the normal distribution[96] that conditional kurtosis of y_t is given by

$$E\left[\left(y_t - \mu_{y|\theta}\right)^4 |\theta\right] = 3\sigma_{y|\theta}^4.$$

Since the normal distribution is symmetric we also know that the third term in (11.72) is zero. Hence, kurtosis of y_t is determined by the mean of conditional kurtosis of $y_t|\theta$, kurtosis of the conditional mean, and the mean of the product between the conditional variance and the square of the conditional mean in deviation from the mean.

11.16. *Prior and Posterior Predictive Checks: Bayesian Specification Analysis*

In order to assess to plausibility of a model, prior (Box, 1980) and posterior (Rubin, 1984, Meng, 1994, and Gelman, Meng, and Stern, 1996a) predictive checks have been suggested when an explicit alternative model is not available. The checks were first based on the idea that a useful model should be able to reliably predict features of the observed data. This could be achieved by assessing how likely the features are by simulating new data with the model and parameter draws from the prior (or posterior), yielding a distribution for the data feature and compare the feature using the observed data to this distribution, i.e. a *p*-value type calculation. The idea has since Box's original article been extended from functions of the observables only to also include the parameters, such as forecast error variances. The functions of interest are called *discrepancies* by Gelman et al. (1996a) and *features* by Faust and Gupta (2012). In the terminology of Faust and Gupta, functions of the observables only are referred to as *descriptive features*, while functions of the observables and the parameters are called *structural features*.

Geweke (2005, Chapter 8.3) has suggested that prior predictive checks takes place when the econometrician is considering alternative variants of the prior distribution, the conditional density of the observables, and (possibly) the conditional density of the vector of interest (a function of the observables and the parameters). Posterior predictive analysis, on the other hand, is in Geweke opinion of interest after the econometrician has conditioned on the observed data and is considering changes to the complete model. That is, Geweke considers prior predictive checks as a tool when creating a complete model, and posterior predictive checks when improving or modifying a complete model. He also collectively refers to them as *Bayesian specification analysis*.

Prior predictive analysis is based on drawing parameters from the prior distribution, simulating data with the conditional density of the observables, and computing the features function using the simulated data and for structural features also the parameter draw. This generates a distribution of the feature, which can be compared with a value for the descriptive feature using the observed data, and to a distribution for the structural feature when using the observed data for each parameter draw. Similarly, the posterior predictive analysis uses the posterior draws of

---

[96] From, e.g., (Zellner, 1971, Appendix A) we have that the even central moments of $z \sim N(\mu, \sigma^2)$ can be expressed as

$$E\left[(z-\mu)^{2r}\right] = \frac{2^r \sigma^{2r}}{\sqrt{\pi}}\Gamma(r+1/2), \quad r = 1, 2, \ldots.$$

Since $r + 1/2 > 1$ it is straightforward to show that the gamma function (see Section 4.2.2) can be expressed as:

$$\Gamma(r+1/2) = \prod_{j=0}^{r-1}(r-j-1/2)\Gamma(1/2),$$

where $\Gamma(1/2) = \sqrt{\pi}$. Hence, the even central moments of z of can also be written as

$$E\left[(z-\mu)^{2r}\right] = \sigma^{2r}\prod_{j=0}^{r-1}\left(2(r-j)-1\right), \quad r = 1, 2, \ldots$$

where I have used the fact that $\prod_{j=0}^{r-1}(r-j-1/2) = (1/2^r)\prod_{j=0}^{r-1}(2(r-j)-1)$.

the parameters instead of the prior draws and is for this reason computationally more expensive than the prior predictive analysis.

Prior and posterior predictive check are both non-Bayesian in nature since the idea of using a p-value is inherently not Bayesian. It may be kept in mind that prior and posterior predictive checks violate the *likelihood principle*, which states that all relevant experimental information for making decisions about the parameters after observing the data is contained in the likelihood function; see, e.g., Zellner (1971, p. 14) or Berger (1985). In addition, posterior predictive checks uses the data twice since the posterior is first estimated using the data. As Geweke (2010) shows, this blunts the evidence against the model from such checks. If an alternative model is readily available, then using standard model comparison tools is preferable. For a nonstandard defense of posterior predictive checks in the case of DSGE modelling, see Faust and Gupta (2012).

Among the possible descriptive features, denoted by $h(\mathcal{Y}_T)$, that we may use when examining DSGE models, sample variances and covariances naturally come to mind. A predictive check of $h(\mathcal{Y}_T^{(o)})$, where $\mathcal{Y}_T^{(o)}$ denotes the observed data, is conducted by simulating data with the model for S parameter draws from the prior or the posterior distribution. For each such sample, the function $h(\mathcal{Y}_T^{(s)})$ is evaluated for $s = 1, \ldots, S$. This yields an empirical distribution of the descriptive feature, $F_h(c)$, with the probability that $h(\mathcal{Y}_T) \leq c$. When large values of the feature are considered unlikely, Box (1980) suggested for the prior predictive checks to compute the p-value

$$1 - F_h\big(\mathcal{Y}_T^{(o)}\big),$$

i.e., the probability of observing $h(\mathcal{Y}_T)$ being greater than the realized value in repeated sampling under the assumption that the data are generated by the model and its prior.

In the case of a prior predictive check of a descriptive feature for a DSGE model, $\mathcal{Y}_T^{(s)}$ is obtained by:

(1) Draw the parameter vector $\theta^{(s)}$ from $p(\theta)$;
(2) Solve the DSGE model and calculate the state-space matrices (A, H, R, F, B_0) for $\theta^{(s)}$;
(3) Provided that the state variables are stationary, draw initial values ξ_1 from $N(0, \Sigma_\xi)$, where Σ_ξ is the population covariance matrix of the state variables in equation (5.15), draw structural shocks $\eta_t^{(s)}$ from $N(0, I_q)$, and measurment errors $w_t^{(s)}$ from $N(0, R)$ for t=1, ..., T;
(4) Calculate $y_t^{(s)}$ recursively from the state-space model in (5.1) and (5.2) for $t = 1, \ldots, T$;
(5) Compute $h(\mathcal{Y}_T^{(s)})$ and save it.

Once we have S values of a descriptive feature, the observed value of the feature can be compared with the empirical distribution from the simulated data. For a posterior predictive check of a descriptive feature, step (1) is replaced with $\theta^{(s)}$ being a draw from $p(\theta|\mathcal{Y}_T^{(o)})$. Notice that this means that $F_h(c)$ depends on the observed data, thus explaining how the observed data comes in twice for such predictive checks.

Turning to structural features, the object of interest is given by $h(\mathcal{Y}_T, \theta)$, i.e. the feature is not only a function of the observed variables but also of the structural parameters. Examples of such features include one-step-ahead sample forecast error variances, sample correlation decompositions, and shock decompositions. In this case we can compare the distribution of the structural feature for the observed data and the parameter draws to the distribution of the simulated data and the parameter draws. A prior predictive check of a structural feature for a DSGE model can be determined from the following:

(i) Draw the parameter vector $\theta^{(s)}$ from $p(\theta)$;
(ii) Compute $h(\mathcal{Y}_T^{(o)}, \theta^{(s)})$;
(iii) Simulate $\mathcal{Y}_T^{(s)}$ with $\theta^{(s)}$, i.e., steps (2)–(4) above;
(iv) Compute $h(\mathcal{Y}_T^{(s)}, \theta^{(s)})$;
(v) Save the pair $\{h(\mathcal{Y}_T^{(o)}, \theta^{(s)}), h(\mathcal{Y}_T^{(s)}, \theta^{(s)})\}$.

For a posterior predictive check of a structural feature, step (i) is replaced with $\theta^{(s)}$ being a draw from $p(\theta|\mathcal{Y}_T^{(o)})$.

The S pairs $\{h(\mathcal{Y}_T^{(o)}, \theta^{(s)}), h(\mathcal{Y}_T^{(s)}, \theta^{(s)})\}$ may now be shown in a scatter plot with the values for the observed data on the horizontal axis and those for the simulated data on the vertical axis. The share of points above the 45 degree line gives an estimate of the p-value, i.e., the probability that the structural feature for the simulate data is greater than the feature for the observed data.

11.16.1. *Structural Features: One-Step-Ahead Forecasts*

The within-sample one-step-ahead forecast error sample covariance matrix for a given value of the parameters is calculated as follows. First, the one-step-ahead forecast errors are given by:

$$\tilde{\varepsilon}_t = y_t - A'x_t + H'\xi_{t|t-1}, \quad t = 1, \dots, T.$$

The within-sample forecast error covariance matrix is thereafter given by

$$\tilde{\Sigma}_y = \frac{1}{T} \sum_{t=1}^{T} \tilde{\varepsilon}_t \tilde{\varepsilon}_t'.$$

11.17. *Permanent Shock to a Target Variable*

A target variable, such as an inflation target, is usually connected with only one equation of a DSGE model, the monetary policy rule. In this section we discuss the case where the target variable, π_t^*, is originally equal to zero and then shifts permanently to, say, $\pi_t^* = \pi^*$. Such a permanent shock allows us to analyse the dynamic and long-run responses to the state and observed variables of the model. One particularly interesting case is, as suggested, a permanent shock to the inflation target and the response variables of interest are real GDP growth or the output gap, but other cases could also be envisaged, such as changed targets for fiscal policy variables. One may also look at accumulated responses in order to study, e.g., the sacrifice ratio; see, e.g., Fuhrer (1994), Ascari and Ropele (2012a,b) and Andrle and Beneš (2013).

The structural form of the DSGE model can be expressed as in equation (3.2)

$$H_{-1}z_{t-1} + H_0 z_t + H_1 E_t[z_{t+1}] = D\eta_t.$$

Let the target variable be denoted by π_t^* and suppose the location of the corresponding model variable π_t in the p-dimensional vector z_t is given by j, such that

$$\pi_t = e_j' z_t,$$

and where e_j is the j:th column of I_p. Furthermore, let the policy equation of the model be the i:th equation, i.e. given by

$$e_i'\Big(H_{-1}z_{t-1} + H_0 z_t + H_1 E_t[z_{t+1}]\Big) = e_i' D\eta_t.$$

The model now needs to be expanded by one equation that determines the behavior of π_t^*. We shall consider this variable to be exogenous and driven by an unanticipated permanent shock

$$\pi_t^* = \pi_{t-1}^* + \pi^* \eta_{\pi^*, t},$$

where $\eta_{\pi^*, t} \sim N(0, 1)$ is the unanticipated shock which has a permanent effect on the target variable, and where π^* is a parameter that needs to be calibrated. The assumption that this shock is unanticipated means that we shall consider the case when the next period value is always assumed to be 0.

The objective is now to rewrite the policy equation such that it includes $\pi_\tau - \pi_\tau^*$ instead of π_τ for $\tau = t - 1, t, t + 1$. Note that

$$e_i'H_{-1}e_j, \quad e_i'H_0 e_j, \quad e_i'H_1 e_j,$$

are the coefficients on π_{t-1}, π_t and $E_t[\pi_{t+1}]$, respectively, in the policy equation. It therefore follows that this equation is now

$$e_i'H_{-1}\big(z_{t-1} - e_j\pi_{t-1}^*\big) + e_i'H_0\big(z_t - e_j\pi_t^*\big) + e_i'H_1 E_t\big(z_{t+1} - e_j\pi_{t+1}^*\big) = e_i' D\eta_t.$$

Notice that this formulation of the policy equation rules out the possibility that the target enters with some other coefficients than the negative of those on the corresponding model variable; an example is the policy rule in Aruoba and Schorfheide (2011). Still, several papers use this assumption which justifies its validity here; as in Del Negro and Eusepi (2011), Del Negro and Schorfheide (2013) and Del Negro et al. (2015). In YADA, users have the option of entering such a target variable into their model directly and simply run the standard impulse response analysis instead.

Notice that $E_t[\pi^*_{t+1}] = \pi^*_t$ given the assumed properties of the permanent shock. Hence, the revised policy equation becomes

$$e'_i H_{-1}(z_{t-1} - e_j \pi^*_{t-1}) + e'_i H_0 z_t - e'_i(H_0 + H_1)e_j \pi^*_t + e'_i H_1 E_t(z_{t+1}) = e'_i D\eta_t.$$

This means that the solution of the extended system and the revised policy equation is unaffacted as far as the z_t variables are concerned. In other words, we can work with the system in equation (3.7), i.e.

$$S_0 z_t = S_1 z_{t-1} + D\eta_t.$$

The position of the policy equation i remains the same in (3.7) as in the structural form, and the location of π_t in z_t is also unaffected. The rewritten policy equation is therefore given by

$$e'_i S_0 z_t - e'_i(H_0 + H_1)e_j \pi^*_t = e_i S_1(z_{t-1} - e_j \pi^*_{t-1}) + e'_i D\eta_t,$$

where we have made use of the fact that $S_1 = -H_{-1}$. In terms of the modified system, the first p equations are therefore given by

$$S_0 z_t - e_i e'_i(H_0 + H_1)e_j \pi^*_t = S_1 z_{t-1} - e_i e'_i S_1 e_j \pi^*_{t-1} + D\eta_t.$$

Combining these equations with the one for the target, we obtain the following system of $p + 1$ equations

$$\begin{bmatrix} S_0 & -e_i e'_i(H_0 + H_1)e_j \\ 0 & 1 \end{bmatrix} \begin{bmatrix} z_t \\ \pi^*_t \end{bmatrix} = \begin{bmatrix} S_1 & -e_i e'_i S_1 e_j \\ 0 & 1 \end{bmatrix} \begin{bmatrix} z_{t-1} \\ \pi^*_{t-1} \end{bmatrix} + \begin{bmatrix} D & 0 \\ 0 & \pi^* \end{bmatrix} \begin{bmatrix} \eta_t \\ \eta_{\pi^*,t} \end{bmatrix},$$

or

$$S^*_0 z^*_t = S^*_1 z^*_{t-1} = D^* \eta^*_t. \tag{11.73}$$

Notice that S^*_0 has full rank $p + 1$ since S_0 has full rank p and is therefore invertible.

We can now solve the extended DSGE model with the permanent shock and the target variable and obtain

$$\xi^*_t = F^* \xi^*_{t-1} + B^*_0 \eta^*_t,$$

where $\xi^*_t = z^*_t$, $F^* = (S^*_0)^{-1} S^*_1$, while $B^*_0 = (S^*_0)^{-1} D$. The effects from a permanent negative shock to $\eta^*_t = -[0 \ 1]' = \eta^*$ can now be traced with impulse responses, such that

$$\text{resp}(\xi^*_{t+h}|\eta^*_t = \eta^*) = \sum_{j=0}^{h} (F^*)^j B^*_0 \eta^*, \quad h = 0, 1, \ldots, h^*. \tag{11.74}$$

From this expression it is straightforward to compute the responses of the observed variables, where we only need account for the increased dimension of the vector with state variables. This means that the revised measurement equation is

$$y_t = A' x_t + H^{*'} \xi^*_t,$$

where $H^{*'} = [H' \ 0]$ now has an extra column with n zeros. Furthermore, accumulated responses can also be computed by accumulation the responses in (11.74).

Finally, it is also worth pointing out that the three solvers directly supported in YADA (AiM, Klein and Sims) may be used to obtain the S_0 and S_1 from F, B_0 and H_i for $i = -1, 0, 1$; see equations (3.4)–(3.6).

The impulse responses are handled by the function DSGEImpulseResponseFcn, historical forecast error decompositions by DSGEHistDecompFcn, while the variance decompositions are calculated by the function DSGEVarianceDecompFcn for the original data and for the levels by DSGEVarianceDecompLevelsFcn. Since the variance decompositions require that a Riccati equation can be solved, the code includes the function RiccatiSolver. The conditional correlations are performed by the functions DSGEConditionalCorrsTheta that can deal with both population-based and sample-based correlations, while correlation decompositions are carried out by DSGECorrelationDecompTheta.

The conditional variance decompositions are handled by DSGECondVarianceDecompFcn, while output on estimates of unobserved variables and observed variable decompositions is provided by the function CalculateDSGEStateVariables. The levels of the conditional variance decompositions are handled by DSGECondLevVarianceDecompFcn, while the levels of the impulse response functions are taken care of by DSGELevImpulseResponseFcn. Parameter scenarios for fixed parameter values are handled by the function DSGEParameterScenariosTheta. The function DSGEtoVARModel checks the "poor man's invertibility condition" of Fernández-Villaverde et al. (2007), i.e., if all the eigenvalues of the matrix on lagged states in equation (11.64) lie inside the unit circle. In addition, it checks if all the eigenvalues of the state variable transition matrix, F, lie inside the unit circle. When this latter condition is satisfied along with $n = q + \text{rank}[R]$ (the square case), the poor man's invertibility condition is not only sufficient, but also necessary for the existence of an infinite order VAR representation of the DSGE model.

Annualizations of the conditional variance decompositions are computed by the function DSGECondAnnVarianceDecompFcn, while impulse responses for annualized data is calculated directly from the output of DSGEImpulseResponseFcn. With s being the data frequency, this typically involves summing s consecutive impulse responses provided that the variable is annualized by summing s consecutive observations. For response horizons prior to period $s-1$ all responses from period 0 until the response horizon are summed. We thereafter return to Fisher's information matrix from Section 11.12. The function used for this purpose is DSGEInformationMatrix which can estimate the information matrix for any selected value of θ. The observation weight computations for the forecast, update and smooth projections of the state variables and economic shocks are handled by the function DSGEObservationWeightsTheta. The singular value decomposition and Q-R factorization with column pivoting to determine "identification patterns" from a matrix is taken care of by DSGEIdentificationPatterns. The Monte Carlo filtering checks are performed via the function MonteCarloFiltering. The computations for performing the prior and posterior predictive checks for the one-step-ahead forecast error covariances are handled by DSGEOneStepAheadPredictiveChecks.

11.18.1. DSGEImpulseResponseFcn

The function DSGEImpulseResponseFcn is used to calculate the responses in the state variables and the observed variables from the economic shocks. It provides as output the structure IRStructure using the inputs H, F, B0, and h. The $r \times n$ matrix H is given by the measurement equations, while F and B0 are obtained from the DSGE model solution as determined by the function AiMtoStateSpace. The last input h is a positive integer denoting the maximum horizon for the impulse responses.

The output structure IRStructure has two fields Ksi and Y. The fields contain the responses of the state variables and of the observed variables, respectively. These are provided as 3D matrices. The first dimension is the number of states (r) for the field Ksi and observed variables (n) for Y, the second dimension is the number of shocks (q), and the third is the number of responses plus one ($h + 1$). First instance, IRStructure.Y(:,:,i+1) hold the results for response horizon i.

11.18.2. `DSGELevImpulseResponseFcn`

The function `DSGELevImpulseResponseFcn` is used to calculate the accumulated responses in the state variables and the levels responses of the observed variables from the economic shocks. It provides as output the structure `IRStructure` using the inputs `H`, `F`, `B0`, `AccMat`, and `h`. The $r \times n$ matrix `H` is given by the measurement equations, while `F` and `B0` are obtained from the DSGE model solution as determined by the function `AiMtoStateSpace`. The matrix `AccMat` is an $n \times n$ diagonal 0-1 matrix. It is used to accumulate the responses in the observed variables provided that they are viewed as being in first differences. The last input `h` is a positive integer denoting the maximum horizon for the impulse responses.

The output structure `IRStructure` has the same dimensions as for the original data function `DSGEImpulseResponseFcn`. The fields contain the responses of the state variables (`Ksi`) and of the observed variables (`Y`), respectively. While the responses in the state variables are pure accumulations of the response function in (11.17), the levels response for the observed variables are only accumulated for those variables which are viewed as being in first differences. Specifically, with C being the 0-1 diagonal matrix `AccMat`, the levels responses for the observed variables are given by

$$\text{resp}\left(y_{t+h}^L | \eta_t = e_j\right) = C \cdot \text{resp}\left(y_{t+h-1}^L | \eta_t = e_j\right) + H'F^h B_0 e_j, \quad h \geq 1,$$

where $\text{resp}(y_t^L | \eta_t = e_j) = H' B_0 e_j$. Observed variables are viewed by YADA as being in first differences based on the user defined input in the *Data Construction File*; cf. Section 18.5.

11.18.3. `CalculateDSGEStateVariables`

The function `CalculateDSGEStateVariables` provides output on estimates of various unobserved variables. To achieve this it needs 5 inputs: `theta`, `thetaPositions`, `ModelParameters`, `DSGEModel`, and `ObsVarDec`. The first three inputs are discussed in some detail in connection with the prior file handling function `VerifyPriorData`, while the structure `DSGEModel` is discussed in connection with the posterior mode estimation function `PosteriorModeEstimation`; cf. Section 7.4. The 6th input variable, `ObsVarDec`, is a boolean variable that determines if the function should compute the observed variable decompositions or not. This input is optional and defaults to 0 if not supplied. The last two input variables are related to recursive smooth estimation of state variables, state shocks, and measurement errors. The boolean variable `RecEst` is unity if recursive estimation should be performed and zero otherwise. The string vector `HeaderStr` is shown in the window title of the wait dialog that is displayed during recursive estimation. Both these input variables are optional and are equal to 0 and an empty string, respectively, by default.

The only required output from the function is the structure `StateVarStructure`. In addition, the function can provide output on `status`, the `mcode` output from the DSGE model solving function that have been used, and `kalmanstatus`, the `status` output from the `KalmanFilter` function. The latter two outputs are only taken into account by YADA when initial parameter estimates are given to `CalculateDSGEStateVariables`.

The structure `StateVarStructure` has at most 27 fields. First of all, the fields `Y` and `X` hold the data matrices for the observed variables and the exogenous variables for the actual sample used by the Kalman filter. Furthermore, the field `TrainSample` hold a boolean variable which reports if a training sample was used or not when computing the log-likelihood through the Kalman filter. Furthermore, the output on the state variable estimates are given through the fields `Ksitt1` (forecast), `Ksitt` (update), and `KsitT` (smooth), while the forecasted observed variables are held in the field `Yhat`. Next, the field `lnLt` stores the vector with sequential log-likelihood values, i.e., the left hand side of equation (5.19).

The smooth estimates of the economic shocks are located in the field `etatT`. The matrix stored in this field has the same number of rows as there are shocks with a non-zero impact on at least one variable. This latter issue is determined by removing zero columns from the estimated B_0 matrix. The columns that are non-zero are stored as a vector in the field `KeepVar`. Update estimates of the economic shocks are located in `etatt`. The smooth innovation population

covariance matrices $N_{t|T}$ in equation (5.25) are located in the field `NtMat`, an $r \times r \times T$ matrix. These covariance matrices are used to compute the average population covariance matrix for the smoothed economic shocks, i.e., $(1/T) \sum_{t=1}^{T} B_0' N_{t|T} B_0$. If the model contains measurement errors, then the smoothed estimates of the non-zero measurement errors are located in the field `wtT`, while names of equations for these non-zero measurement errors are stored as a string matrix in the field `wtNames`. At the same time, all estimated measurement errors are kept in a matrix in the field `wthT`. Similarly, update estimates of the measurement errors are found in `wtt` and `wtht`.

Given that the boolean input `ObsVarDec` is unity, the historical observed variable decompositions are calculated. The field `XiInit` contains a matrix with typical column element given by $H' F^t \xi_{0|T}$. Similarly, the field `etaDecomp` holds a 3D matrix with the contributions to the observed variables of the non-zero economic shocks. For instance, the contribution of shock i for observed variable j can be obtained for the full sample as `etaDecomp(j,:,i)`, a $1 \times T$ vector. The historical decompositions for the state variables are similarly handled by the fields `XietaInit` and `XietaDecomp`.

Recursive estimates of the state variables, the state shocks, and the measurement errors are computed when the boolean variable `RecEst` is unity. The field `recursive` is then setup and defined as a vector (or structure array) such that `recursive(t).KsitT` holds the t:th recursive values of the smoothed state variables. Similarly, state shocks and measurement errors are located in `recursive(t).etatT` and `recursive(t).wtT`, respectively, for the t:th recursive smooth estimate.

The structure `StateVarStructure` also has 5 fields with parameter matrices `A`, `H`, and `R` from the measurement equation, and `F` and `B0` from the state equation. The last field is given by `MaxEigenvalue`, which, as the name suggests, holds the largest eigenvalue (modulus) of the state transition matrix F.

11.18.4. DSGESimulationSmootherTheta

The function `DSGESimulationSmootherTheta` calculates the distribution of unobserved variables conditional on the data and the parameters of the DSGE model. The input variables are given by: `theta`, `thetaPositions`, `ModelParameters`, `VarType`, `EstStr`, `DSGEModel`, `controls`, and `CurrINI`. The first three inputs are discussed above while (as mentioned above) the structura `DSGEModel` is discussed in connection with the posterior mode estimation function; cf. Section 7.4. The structures `CurrINI` and `controls` are also discussed in some details there, while additional details are given in Warne (2025).

Since a log-linearized DSGE model (and a state-space model) has three types of unobserved variables, the input variable `VarType` is an integer that takes a value that reflects which type should be studied: it is 1 if the simulation smoother should examine state variables; it is 2 if economic shocks should be analysed and, finally, it is equal to 3 when the user is interested in measurement errors. The input `EstStr` is a string vector that indicates which takes of parameter values are considered, i.e., initial values of posterior mode values.

The function provides 2 required output variables and 2 optional. The required output variables are the structure `StateVarStructure` and `TotalPaths`, while the optional outputs are the same as for the function `CalculateDSGEStateVariables`. The `TotalPaths` variable is an integer that is equal to the number of total paths that were drawn from the conditional distribution of the selected unobservable. Apart from a set of common field names, the structure `StateVarStructure` has field names that reflect the chosen value for `VarType`. The common fields are `Quantiles` with the quantile data and `DateVector` with the numeric dates. In the event that output should be produced for state variables, the fields `KsitT`, `PtT`, and `StateVariableNames` are included. When the conditional distribution of the economic shocks is studied the fields are `etatT` and `ShockNames`. Finally, for the measurement errors the fields are given by `wtT`, `VariableNames`, and `ErrorIndicator`. The common field `Quantiles` is a vector structure with common field `percent` and an additional field that is either `KsitT`, `etatT` or `wtT`.

11.18.5. DSGEHistDecompFcn

The function `DSGEHistDecompFcn` calculates the historical forecast error decomposition of the h-step ahead forecast errors as described in equation (11.16). The inputs of the function are given by Y, X, A, H, F, B0, Ksitt, KsitT, etatT, wtT, and h. As before, Y and X are the data on the observed variables and the exogenous variables, respectively, A and H are given by the measurement equation, while F and B0 are obtained from the `AiMtoStateSpace` function regarding the state equation. Furthermore, Ksitt and KsitT are the updated and smoothed state variables that are prepared by the `StateSmoother` function. The input etatT is the smoothed estimate of the economic shocks in equation (11.1), wtT is the smoothed estimate of the measurement error in (5.34), while h is the forecast horizon h.

The function supplies the structure `HistDecompStructure` as output. This structure has 5 fields: `epstth`, an $n \times (T - h)$ matrix with the forecast errors for the observed variables; `KsiErr`, an $n \times (T - h)$ matrix with the state projection error in the first term of the right hand side of equation (11.16); `etathT`, $n \times (T - h) \times q$ matrix with the shares of the q economic shocks in the second term of the equation;[97] `wthT`, an $n \times (T - h)$ matrix with the smoothed measurement errors; and `KeepVar`, a vector with index values of the columns of B_0 that are non-zero.

11.18.6. DSGEConditionalCorrsTheta

The function `DSGEConditionalCorrsTheta` can calulate either population-based or sample-based conditional correlations as described in Section 11.7. It takes 10 input variables: `theta`, `thetaPositions`, `ModelParameters`, `NumPaths`, `EstType`, `DSGEModel`, `CurrINI`, `SimulateData`, `FirstPeriod`, `LastPeriod`. The structures `thetaPositions`, `ModelParameters`, `DSGEModel`, and `CurrINI` have all been discussed above. The vector `theta` holds the values for the parameters as usual. The integer `NumPaths` is equal to the number of simulations and is used only if sample-based conditional correlation should be calculated. The text string `EstType` indicates if posterior mode or initial parameter values are used. The boolean variable `SimulateData` is 1 (0) if sample-based (population-based) conditional correlations should be computed. Finally, the integers `FirstPeriod` and `LastPeriod` marks the sample start and end point when the sample-based approach should be used.

The function provides one required and one optional output. The required output variable is `CondCorr`, a structure with fields `Mean`, `Quantiles`, and `ShockNames`. When the field `Quantiles` is not empty, then it has length equal to the number of quantiles, and each sub-entry has fields `percent` and `Mean`. The former stores the percentile value of the distribution, while the latter stores the conditional correlations at that percentile. The optional output variable is `status` that indicates if the solution to the DSGE model is unique or not. The value is equal to the variable `mcode` given by either the function `AiMSolver`, the function `KleinSolver`, or `SimsSolver`.

11.18.7. DSGECorrelationDecompTheta

The function `DSGECorrelationDecompTheta` computes the correlation decompositions for the observed variables and the state variables. It takes 6 input variables: `theta`, `thetaPositions`, `ModelParameters`, `VarStr`, `DSGEModel`, and `CurrINI`. All these input variables have been discussed above except for `VarStr`. It is a string that supports the values `'Observed Variables'` and `'State Variables'`.

The function yields one required and one optional output variable. The requires variable is `CorrDec`, a structure with 7 fields. The first field, Y, an $n(n + 1)/2 \times (2h + 1) \times (q + 1)$ matrix with the decompositions of the observed variable correlations over the horizons $-h$ until h into the q shocks and the measurement error. Similarly, the second field, Xi, is an $r(r + 1)/2 \times (2h + 1) \times q$ matrix with the correlation decompositions for the state variables. The third field, `AutoCovHorizon`, is an integer with the value of h, the autocorrelation horizon.

The `ShockNames` and the `ShockGroupNames` fields are string matrices where the rows hold the names of the shocks and the shock groups, respectively, where the number of rows is q for the

---

[97] This means that the part of the h-step ahead forecast error that is due to economic shock j is obtained from `etathT(:,:,j)`, an $n \times (T - h)$ matrix.

shocks and g for the shock groups, with $q \geq g$. The `ShockGroups` field is a vector of dimension q with integers that map each shock to a certain shock group, while the `ShockGroupColors` field is a $g \times 3$ matrix, where each row gives the color as an RGB triple for a shock group. The RGB triple holds values between 0 and 1, representing the combination of red, green and blue, and this scale can be translated into the more common 8-bit scale that is used to represent colors with integer values between 0 and 255.

The optional output variable is `mcode`, determined by either the function `AiMSolver`, the function `KleinSolver`, or the function `SimsSolver`. It is only used when `theta` is given by the initial parameter values.

11.18.8. DSGEParameterScenariosTheta

The function `DSGEParameterScenariosTheta` calculates the parameter scenario for two values of the parameter vector, the baseline value and the alternative value. It takes 10 input variables: `DSGEModel`, `theta`, `thetaScenario`, `thetaPositions`, `ModelParameters`, `FirstPeriod`, `LastPeriod`, `BreakPeriod`, `CopyFilesToTmpDir`, and finally `CurrINI`. The structures `DSGEModel`, `ModelParameters`, `thetaPositions` and `CurrINI` have all been discussed above. The vector `theta` holds the baseline values of the parameters, while `thetaScenario` holds the alternative (scenario) values of the parameters. The integers `FirstPeriod` and `LastPeriod` simply indicate the first and the last observation in the estimation sample (not taking a possible training sample for the state variables into account). The integer `BreakPeriod` indicates the position in the sample (taking the training sample into account) where the parameters change, while the boolean `CopyFilesToTmpDir` indicates if certain files should be copied to the `tmp` directory of YADA or not.

The function provides 8 required output variables. These are: `Status`, a boolean that indicates if all calculations were completed successfully or not. Next, the function gives the actual path for the observed variables in the matrix `Y`, as well as the matrix `YScenario`, holding the alternative paths. As mentioned in Section 11.9, these paths are based on feeding the smooth estimates of the economic shocks (and measurement errors) based on the baseline parameters into the state-space model for the alternative parameters. Next, the function gives two matrices with smooth estimates of the economic shocks: `OriginalShocks` and `ScenarioShocks`. The former holds the values of the economic shocks under the baseline parameter values, while the latter gives the values of the economic shocks under the alternative parameter values. Similarly, two matrices with state variable estimates are provided: `OriginalStates` and `ScenarioStates`, where the former holds the smooth estimates of the state variables for the baseline parameter values, while the latter matrix holds the implied state variables for the alternative parameter values. That is, when the state equation is applied to the combination of the smoothly estimated economic shocks under the baseline parameter values along and the F and B_0 matrices for the alternative parameter values. Finally, the function provides a vector with positive integers, `KeepShocks`, signalling which of the economic shocks have a non-zero influence on the variables of the DSGE model.

11.18.9. DSGEtoVARModel

The function `DSGEtoVARModel` is used to check if the state-space representation of the DSGE model satisfies the "poor man's invertibility condition" of Fernández-Villaverde et al. (2007). The function takes 4 inputs: `H`, `R`, `F`, and `B0`. These are, as before, the matrices H and R from the measurement equation, and the matrices F and B_0 of the state equations; see, e.g., the details on `DSGEImpulseResponseFcn`.

As output the function provides `status`, `EigenValues`, and `Fstatus`. The integer `status` is unity if the eigenvalues of the G matrix in equation (11.64) are less than unity in absolute terms, and 0 if some eigenvalues is on or outside the unit circle. In the event that the number of economic shocks and unique measurement errors exceeds the number of observed variables ($n < q + \text{rank}[R]$) `status` is equal to -1, while it is equal to -2 when the opposite holds ($n > q + \text{rank}[R]$). The vector `EigenValues` provides the modulus of the eigenvalues from the invertibility condition when `status` is non-negative. Finally, the boolean variable `Fstatus` is

unity if all eigenvalues of the F matrix of the state-space model are less than unity in absolute term, and zero otherwise. Note than when `Fstatus` is unity and the number of observed variables is equal to the number of sources of noise ($n = q + \text{rank}[R]$), then status provides both a necessary and sufficient statement about if the DSGE model can be rewritten as an infinite order VAR model or not.

11.18.10. DSGEInformationMatrix

The function `DSGEInformationMatrix` is used to estimate Fisher's information matrix stated in equation (11.65). To achieve this 6 input variables are required: `theta`, `thetaPositions`, `ModelParameters`, `ParameterNames`, `DSGEModel`, and `CurrINI`. The first 3 variables are identical to the input variables with the same names in the `CalculateDSGEStateVariables` function. The 4th input is a string matrix with the names of the estimated parameters. The last two input variables are structures that have been mentioned above; see also Section 7.4.

The function provides the output variable `InformationMatrix` which is an estimate of the right hand side in (11.65) with the partial derivatives $\partial \breve{y}_t / \partial \theta'$ and $\partial \text{vec}(\Sigma_{y,t|t-1}) / \partial \theta'$ replaced by numerical partials. By default, each parameter change is equal to 0.1 percent of its given value. If the model cannot be solved at the new value of θ, YADA tries a parameter change of 0.01 percent. Should YADA also be unsuccessful at the second new value of θ, estimation of the information matrix is aborted.

11.18.11. DSGEIdentificationPatterns

The function `DSGEIdentificationPatterns` computes the eigenvectors and eigenvalues of a square (symmetric) matrix and attempts to order the columns of the input matrix through Q-R factorization with column pivoting; see Section 11.12. The function takes one input variable, the matrix `InformationMatrix`, which is typically positive semidefinite.

Three output variables are determined from the input: `EigenVectorMatrix`, `EigenValues`, and `ParameterOrdering`. The first two variables are taken from the singular value decomposition of `InformationMatrix`, where the first variable is given by V in $USV' = A$ (where A is given by `InformationMatrix`) and the second is equal to the diagonal of S. The third variable is a matrix with two columns. The first column contains the column ordering based on the Q-R factorization with column pivoting of V', while the second column contains the ordering based on the same algorithm applied directly to `InformationMatrix`. When printing output on parameter ordering, YADA currently only writes the ordering based on the `InformationMatrix`.

11.18.12. MonteCarloFiltering

The function `MonteCarloFiltering` is used to check the DSGE model solution property for a given set of draws from the prior distribution. The 6 input variables are given by: `thetaDraws`, `thetaPositions`, `ModelParameters`, `AIMData`, `DSGEModel`, and `CurrINI`. The last five input variables have been discussed above, while the first input variable is a matrix of dimension $p \times n_d$ where p is the dimension of θ, the vector of parameters to estimate, and n_d is the number of draws from the prior distribution of θ.

As output the function provides the n_d-dimensional vector `SolutionCode`, where a unit value indicates a unique and convergent solution of the model, a value of 2 that there is indeterminacy, and a value of 0 that there is no convergent solution.

11.18.13. DSGEOneStepAheadPredictiveChecks

The function `DSGEOneStepAheadPredictiveChecks` is used to perform the computations needed for the prior and posterior predictive checks of the one-step-ahead forecast error covariances of the DSGE model. The 10 input variables are: `theta`, `thetaPostSample`, `thetaPositions`, `ModelParameters`, `SelectedParameters`, `IsPosterior`, `TotalDraws`, `CurrChain`, `DSGEModel`, and `CurrINI`. The `theta` input is like above fixed values for the original vector of estimated parameter. The matrix `thetaPostSample` with `NumDraws` rows and `NumParam` columns contains the parameter draws from the prior or the posterior distribution. The variables `thetaPositions`, `ModelParameters`, `TotalDraws`, `DSGEModel`, and `CurrINI` have all been discussed above. the

vector `SelectedParameters` contains the entries in the `theta` vector that can change from one parameter draw to the next. The boolean variable `IsPosterior` is one if the parameter draws are taken from the posterior distribution and zero if they are taken from the prior. The input variable `CurrChain` is an integer that indicates the MCMC chain number for posterior draws and is zero otheriwse.

The only output variable provided by the function is called `DoneCalc`, which is a boolean that takes the value 1 if the calculations have been performed and 0 otherwise.

11.18.14. DSGEObservationWeightsTheta

The function `DSGEObservationWeightsTheta` is used to estimate the observation weights and the associated decompositions that were discussed in Section 5.9 for the state variables and in Section 11.1 for the economic shocks. The computations require 6 input variables: `theta`, `thetaPositions`, `ModelParameters`, `VarType`, `DSGEModel`, and `CurrINI`. The first 3 variables are identical to the input variables with the same names in the `CalculateDSGEStateVariables` function. The 4th variable is an integer which indicates the type of output to prepare. The supported values are: (1) decompositions for state variables; (2) decompositions for economic shocks; (3) weights for state variables; and (4) weights for economic shocks. The last two input variables are structures that have been mentioned above; see also Section 7.4.

The function provides the output variable `StateDecomp`, a structure whose fields depend on the value for `VarType`. In addition, the function can provide output on `status`, the `mcode` output from the DSGE model solving function, and `kalmanstatus`, the `status` output from the Kalman filter.

11.18.15. DSGECondVarianceDecompFcn

The function `DSGECondVarianceDecompFcn` computes the conditional forecast error variance decomposition in (11.24). The function needs 4 inputs: `H`, `F`, `B0`, and `h`. These are exactly the same as those needed by `DSGEImpulseResponseFcn`.

As output the function provides the 3D matrix `FEVDs`. This matrix has dimension $n \times q \times (h+1)$, with n being the number of observed variables, q the number of economic shocks, and h the forecast horizon. The last third dimensional $n \times q$ matrix is the long-run conditional variance decomposition.

11.18.16. DSGECondLevVarianceDecompFcn

The function `DSGECondLevVarianceDecompFcn` computes the conditional forecast error variance decomposition in (11.24), but where R_i is partly an accumulation of R_{i-1}. Specifically, let C denote a diagonal 0-1 matrix, where a diagonal element is 1 if the corresponding variable should be accumulated and 0 otherwise. The R_i matrices are here calculated according to

$$R_i = CR_{i-1} + H'F^i B_0, \quad i = 1, 2, \ldots,$$

while $R_0 = H'B_0$. This allows YADA to compute levels effects of observed variables that only appear in first differences in the y_t vector, e.g., GDP growth. At the same time, variables that already appear in levels, e.g., the nominal interest rate, are not accumulated. The function needs 5 inputs: `H`, `F`, `B0`, `AccMat`, and `h`. These are identical to the inputs accepted by `DSGELevImpulseResponseFcn`.

As output the function provides the 3D matrix `FEVDs`. This matrix has dimension $n \times q \times (h+1)$, with n being the number of observed variables, q the number of economic shocks, and h the forecast horizon. As above, the last third dimensional $n \times q$ matrix is the long-run conditional variance decomposition for the levels.

11.18.17. DSGECondAnnVarianceDecompFcn

The function `DSGECondAnnVarianceDecompFcn` calculates the conditional variance decomposition in (11.24), but where the R_i matrices need to take annualization information into account.

In particular, let N denote a diagonal 0-1 matrix, while s is the frequency of the data, e.g., $s = 4$ ($s = 12$) for quarterly (monthly) data. The R_i matrices are now given by:

$$R_i = N \left(\sum_{j=1}^{\min\{i,s\}-1} H'F^{i-j}B_0 \right) + H'F^i B_0, \quad i = 0, 1, 2, \ldots.$$

A diagonal element of N is unity if the corresponding observed variable should be annualized by adding the current and previous $s-1$ observations, and zero otherwise.

The function needs 6 inputs: H, F, B0, AccMat, h, and AccHorizon. The first five are identical to the inputs accepted by DSGECondLevVarianceDecompFcn, with the exception of AccMat which is now given by the matrix A from the equation above. The final input is AccHorizon which corresponds to the integer s above.

As output the function provides the 3D matrix FEVDs. This matrix has dimension $n \times q \times h$, with n being the number of observed variables, q the number of economic shocks, and h the forecast horizon.

11.18.18. DSGEVarianceDecompFcn & DSGEVarianceDecompLevelsFcn

The function DSGEVarianceDecompFcn computes all the forecast error variance decompositions for the original variables, while DSGEVarianceDecompLevelsFcn takes care of the levels. Both functions accept the same input variables and return the same output variables.

They require the 9 input variables, H, F, B0, R, h, DAMaxIter, DAConvValue, RicMaxIter, and RicConvValue. The first three and the fifth inputs are the same as those required by DSGEImpulseResponseFcn, while the input R is simply the covariance matrix of the measurement errors. The last four inputs concern the maximum number of iterations (DAMaxIter, RicMaxIter) and the tolerance level (DAConvValue, RicConvValue) when calling the functions DoublingAlgorithmLyapunov and RiccatiSolver, respectively. The values for these inputs can be determined by the user on the *Settings* tab for the doubling algorithm and on the *Miscellaneous* tab for the Riccati solver.

As output the functions give FEVDs, LRVD, status, RiccatiResults, and UniquenessCheck. Provided that all potential calls to RiccatiSolver gave valid results, the matrix FEVDs is $n \times (n+2) \times h$, while LRVD is $n \times (n+1)$. The first 3D matrix has the $n \times (n+2)$ variance decomposition matrix for horizon i in FEVDs(:,:,i). The first column contain the shares due to state variable uncertainty, the second column the shares due to the n potential measurement errors, while the remaining q columns have the shares due to the economic shocks. The matrix LRVD is structured in the same way, except that there is no state variable uncertainty.

The status variable is obtained from the same named output variable of the RiccatiSolver function, while RiccatiResults is a 1×2 vector with information about from the Riccati solver for the overall forecast error variance calculation. The first value gives the number of iterations used by the algorithm, while the second gives the value of the convergence measure. Finally, the UniquenessCheck scalar records the largest eigenvalue of L, where K is the asymptote of the Kalman gain matrix for the overall forecast error variance calculation.

11.18.19. RiccatiSolver

The function RiccatiSolver tries to solve the Riccati equation (11.33) iteratively and through the use of the Matlab function dare from the *Control System Toolbox*. It requires 7 inputs. They are F, H, Q, R, P1, MaxIter, and ConvValue. As before F is the state transition matrix, H the mapping from the state variables to the observed, Q is the covariance matrix of the state shocks, and R the covariance matrix of the measurement errors. The matrix P1 is the initial value for the covariance matrix P_1. YADA always sets this value equal to the covariance matrix of the state variables, given the assumptions made about the shocks. Finally, MaxIter is the maximum number of iterations that can be used when attempting to solve the Riccati equation, while ConvValue is the tolerance level. As in the case of the function DoublingAlgorithmLyapunov, the convergence criterion is given by the Matlab norm function applied to the change in P_1,

unless the call to `dare` indicates a solution of the Riccati equation. The details of the algorithm are given in Sections 11.5 and 11.6.

The Riccati solver function gives three required output variables `P1`, `status`, and `NumIter`, as well as one optional variable, `TestValue`. The first required variable is the solution candidate for P_1. The solution is considered as valid by YADA if the iterations have converged within the maximum number of iterations. If so, the variable `status` is assigned the value 0, otherwise it is 1 unless infinite values or NaN's were located. This last case results in a value of 2 for `status`. The third output `NumIter` is simply the number of iterations used, while the fourth (optional) variable gives the value of the convergence criterion for the last iteration.

12. A Bayesian Approach to Forecasting with DSGE Models

12.1. *Unconditional Forecasting with a State-Space Model*

To analyse the predictions of a set of future observations $y_{T+1}, \ldots, y_{T+h}$ conditional on the data that can be observed at time T and a path for future values of the exogenous variables, we need to determine its predictive distribution. Letting this distribution be denoted by $p(y_{T+1}, \ldots, y_{T+h}|x_{T+1}, \ldots, x_{T+h}, \mathcal{Y}_T)$, we note that it can be described in terms of the distribution for the future observations conditional on the data and the parameters and the full posterior distribution. That is,

$$
p(y_{T+1}, \ldots, y_{T+h}|x_{T+1}, \ldots, x_{T+h}, \mathcal{Y}_T) = \int_{\theta \in \Theta} p(y_{T+1}, \ldots, y_{T+h}|x_{T+1}, \ldots, x_{T+h}, \mathcal{Y}_T; \theta) \\
\times p(\theta|\mathcal{Y}_T) d\theta,
\tag{12.1}
$$

where Θ is the support of θ and where, for convenience, we have neglected to include an index for the model in the above expressions.

From a Bayesian perspective, it may be noticed that for a given model there is no uncertainty about the predictive density and, thus, there is no uncertainty about a point or a density forecast which is determined from it. This can be seen in equation (12.1) where posterior parameter uncertainty is integrated out and what remains is a deterministic function of the data and the model. In practise, numerical methods typically need to be applied, but the induced simulation uncertainty can be controlled by the econometrician.[98]

The problem of numerically determining the predictive distribution is examined in the context of a cointegrated VAR by Villani (2001). Although the full predictive distribution cannot be obtained analytically, a procedure suggested by Thompson and Miller (1986) may be applied.

Their procedure is based on a double simulation scheme, where S draws of θ from its full posterior are first obtained. In the second stage, prediction paths are simulated for $y_{T+1}, \ldots, y_{T+h}$ conditional on the data and θ. From the conditional predictive distribution we have that

$$
p(y_{T+1}, \ldots, y_{T+h}|x_{T+1}, \ldots, x_{T+h}, \mathcal{Y}_T; \theta) = \prod_{i=1}^{h} p(y_{T+i}|x_{T+i}, \mathcal{Y}_{T+i-1}; \theta).
\tag{12.2}
$$

The right hand side of equation (12.2) is obtained by the usual conditioning formula and by noting that y_{T+i} conditional on $x_{T+i}, \mathcal{Y}_{T+i-1}$ and the parameters is independent of x_{t+j} for all $j > i$. Moreover, the density for this conditional expression is for the state-space model given by a multivariate normal with mean $y_{T+i|T+i-1}$ and covariance $(H'P_{T+i|T+i-1}H + R)$.

The approach suggested by Thompson and Miller (1986) may be implemented for the state-space model as follows. For a given draw of θ from the posterior distribution, the DSGE model is solved and the matrices A, H, R, F, and Q are calculated. A value for y_{T+1} is now generated by drawing from the normal distribution with mean $y_{T+1|T}$ and covariance matrix $(H'P_{T+1|T}H + R)$ using the expressions in Section 5.

For period $T + 2$ we treat the draw y_{T+1} as given. This allows us to compute $y_{T+2|T+1}$ and $(H'P_{T+2|T+1}H + R)$ and draw a value of y_{T+2} from the multivariate normal using these values as the mean and the covariance. Proceeding in the same way until we have drawn y_{T+h} we have obtained one possible future path for $y_{T+1}, \ldots, y_{T+h}$ conditional on $x_{T+1}, \ldots, x_{T+h}, \mathcal{Y}_T$ and the given draw θ from the posterior distribution.

We may now continue and draw a total of P paths for $y_{T+1}, \ldots, y_{T+h}$ conditional on this information. Once all these paths have been drawn, we pick a new value of θ from its posterior and recalculate everything until a total of PS draws from the density in (12.2) have been drawn.

---

[98] This may be contrasted with a frequentist approach to forecasting where a point or a density forecast is conditioned on the unknown parameters of the model, i.e., it is based on the first density term on the right hand side of (12.1). Once the unknown θ is replaced with a point estimate, the resulting point or density forecast is subject to the estimation uncertainty inherent to the selected estimator and sample period and which cannot be influenced by the econometrician. At the same time, the "true" predictive density, based on the "true" values of the parameters, is deterministic but remains unknown; see Geweke and Whiteman (2006) for discussions on Bayesian forecasting.

Alternatively, and as suggested by Adolfson, Lindé, and Villani (2007d), we can directly utilize the state-space form. Specifically, for a given draw of θ from its posterior distribution, the period T state vector can be drawn from $N(\xi_{T|T}, P_{T|T})$, where $\xi_{T|T}$ and $P_{T|T}$ are obtained from the final step of the Kalman filter computations; cf. Section 5. Next, a sequence of future states $\xi_{T+1}, \ldots, \xi_{T+h}$ can be simulated from the state equation (5.2) given draws of the state shocks $v_{T+1}, \ldots, v_{T+h}$. The latter are drawn from the normal distribution with mean zero and covariance matrix $Q = B_0 B_0'$. Next, vectors of measurement errors $w_{T+1}, \ldots, w_{T+h}$ are drawn from a normal distribution with mean zero and covariance matrix R. Adding the sequence of state variables and the measurement errors a path for $y_{T+1}, \ldots, y_{T+h}$ is obtained via the measurement equation (5.1). For the given value of θ, we can now generate P paths of the observed variables and by repeating this for S draws from the posterior distribution of θ and total of PS paths from the predictive density of $y_{T+1}, \ldots, y_{T+h}$ may be obtained.

The Adolfson, Lindé, and Villani (2007d) approach is faster than the first approach since the underlying computations are more direct. For this reason, the Adolfson, Lindé, and Villani approach has been implemented in YADA. Moreover, this procedure highlights the fact that the uncertainty in the forecasts stems from four sources: parameter uncertainty (θ), uncertainty about the current state (ξ_T), uncertainty about future shocks (v), and measurement errors (w).

Based on the *law of iterated expectations* the forecast uncertainty for y_{T+i} can be decomposed ("Rao-Blackwellization") as follows:

$$C(y_{T+i}|\mathcal{Y}_T) = E_T\left[C(y_{T+i}|\mathcal{Y}_T; \theta)\right] + C_T\left[E(y_{T+i}|\mathcal{Y}_T; \theta)\right], \qquad (12.3)$$

where E_T and C_T denote the expectation and covariance with respect to the posterior of θ at time T and where, for notational simplicity, the sequence of exogenous variables $x_{T+1}, \ldots, x_{T+h}$ has been suppressed from the expressions. Adolfson, Lindé, and Villani (2007d) show that the first term on the right hand side of (12.3) is given by

$$E_T\left[C(y_{T+i}|\mathcal{Y}_T; \theta)\right] = E_T\left[H'F^i P_{T|T}(F^i)'H\right] + E_T\left[H'\left(\sum_{j=0}^{i-1} F^j Q(F')^j\right)H\right] + E_T[R], \quad (12.4)$$

providing the uncertainties regarding the current state, the future shocks, and the measurement errors. Similarly, for the second term in (12.3) we have that

$$C_T\left[E(y_{T+i}|\mathcal{Y}_T; \theta)\right] = C_T\left[A'x_{T+i} + H'F^i \xi_{T|T}\right], \qquad (12.5)$$

which thus reflects the influence of parameter uncertainty on forecast uncertainty.

To simplify the expression for the shock uncertainty term, consider the difference equation

$$\bar{\Sigma}_\xi^{(i)} = Q + F\bar{\Sigma}_\xi^{(i-1)} F', \quad i = 1, \ldots, h,$$

where the $r \times r$ matrix $\bar{\Sigma}_\xi^{(i)}$ is initialized at $\bar{\Sigma}_\xi^{(0)} = 0$. It is now straightforward to show that

$$\bar{\Sigma}_\xi^{(i)} = \sum_{j=0}^{i-1} F^j Q(F')^j.$$

This expression allows for fast computation of the shock uncertainty term since loops over j are not required.

It is interesting to note that *not* all terms in equation (12.3) necessarily increase as the forecast horizon increases. For example, the uncertainty due to the state at time T is positive semidefinite at $i = 1$ when some state variable are unobserved and have an impact on the observed variables; cf. first term on the right hand side of (12.4). As $i \to \infty$, the state uncertainty term converges to zero when all eigenvalues of F are inside the unit circle. Hence, there exists some finite forecast horizon i beyond which the state uncertainty factor is decreasing. Similarly, the parameter uncertainty term may also have this property; cf. equation (12.5). For example, if all elements of A are known, then parameter uncertainty is entirely due to variation in $F^i \xi_{T|T}$ across different parameter draws from the posterior, and the uncertainty of this term will decrease beyond some forecast horizon i when all eigenvalue of F are inside the unit circle. In

fact, the only term that becomes larger and larger for finite forecast horizons is the shock uncertainty term (second term on the right hand side of equation 12.4) and, hence, we expect this term to dominate the forecast error covariance matrix of the observed variables at and beyond some finite forecast horizon. Moreover, the sum of the state, shock, and measurement error uncertainty terms on the right hand side of equation (12.4) are increasing as the forecast horizon increases and we therefore expect the overall forecast error uncertainty in (12.3) to increase with the horizon as well, at least beyond some finite horizon when the share due to parameter uncertainty is sufficiently small.[99]

12.2. *Conditional Forecasting with a State-Space Model*

Conditional forecasting concerns forecasts of endogenous variables conditional on a certain path and length of path for some other endogenous variables; see, e.g., Waggoner and Zha (1999). While use of such conditioning assumptions may at first seem to be of limited interest, one important forecasting situation that should be kept in mind is when real-time data vintages[100] are used by the forecaster. The values for all observed variables for period T, the last "historical" time period, have often not been released by the statistical authority yet and are therefore missing from the relevant data vintage, i.e., the data set is *unbalanced*. Accordingly, some of the time T values need to be forecasted and the forecasts of these variables need to take into account that values for other variables are available for the same time period.

In this section I will discuss conditional forecasting as it has been implemented in YADA for a DSGE model. Specifically, it is assumed that the conditioning information satisfies *hard conditions*, i.e., a particular path, rather than *soft conditions* (a range for the path). Furthermore, YADA covers three ways of handling the need to control the shock processes such that the conditioning information is satisfied. First, the user can select which economic shocks should be controlled directly. Second, the Waggoner and Zha (1999) approach is supported. In this case, the economic shocks are drawn from a distribution that ensures that the conditioning information is met; see also Robertson, Tallman, and Whiteman (2005). Finally, a mixed case is also supported where a subset of the shocks are drawn from a distribution which is consistent with the conditioning assumptions, while another subset of shocks are standard normal. The second and third methods are only available in the version of YADA that is exclusive to the Modelling Unit at Sveriges Riksbank and the NAWM team within the Directorate General Research of the ECB.

Let K_1 and K_{2j} be known $n \times q_m$ matrices with $q_m \leq \min\{n, q\}$ such that $\text{rank}(K_1) = q_m$ and $j = 1, \ldots, g - 1$. Furthermore, consider the following relation:

$$z_{T+i} = K_1' y_{T+i} + \sum_{j=1}^{i-1} K_{2j}' y_{T+i-j} + u_T, \quad i = 1, \ldots, g. \tag{12.6}$$

The specification in equation (12.6) is general enough to satisfy our purposes. In the special case where $K_{2j} = 0$ and $u_T = 0$ the vector z_{T+i} is determined directly from y_{T+i}, e.g., one particular observed variable. Although such a specification covers many interesting cases it does not allow us to handle the case when y includes the real exchange rate and the first differences of the domestic and foreign prices, but where z is the nominal exchange rate. Let p_t and p_t^* denote the domestic and foreign prices, respectively, while s_t denotes the nominal exchange rate. We may then let K_1 be defined such that $K_1' y_{T+i} = (s_{T+i} + p_{T+i}^* - p_{T+i}) + \Delta p_{T+i} - \Delta p_{T+i}^*$, whereas $K_{2j} = K_2$ for all j and $K_2' y_{T+i-j} = \Delta p_{T+i-j} - \Delta p_{T+i-j}^*$ and $u_T = p_T - p_T^*$. Another interesting case which requires K_{2j} to vary with j is when the conditioning assumptions involve annual inflation and the observed variables have quarterly inflation.

---

[99] For more detailed discussion on forecasting with DSGE models in practise, see Adolfson et al. (2007d), Christoffel, Coenen, and Warne (2011), and Del Negro and Schorfheide (2013).

[100] See, for instance, Croushore and Stark (2001) and Croushore (2011a,b) for further details on forecasting with real-time data vintages.

12.2.1. *Direct Control of the Shocks*

To keep the values in z_{T+i} fixed over the given horizon, the first method we consider requires that a subset of the economic shocks are adjusted to take on certain values. The selection of shocks is defined by the user while the values are calculated by taking equation (12.6) into account. The selection of economic shocks is determined by the $q \times q_m$ matrix M where $q > q_m$ and $\text{rank}(M) = q_m$. Let $M_\perp$ be the $q \times (q - q_m)$ orthogonal matrix, i.e., $M'_\perp M = 0$. It now follows that $N = [M_\perp \ M]$ is a full rank $q \times q$ matrix while

$$N'\eta_t = \begin{bmatrix} M'_\perp \\ M' \end{bmatrix} \eta_t = \begin{bmatrix} \eta_t^{(q-q_m)} \\ \eta_t^{(q_m)} \end{bmatrix}. \tag{12.7}$$

The shocks $\eta_t^{(q_m)}$ will be adjusted over the time interval $t = T+1, \ldots, T+g$ to ensure that (12.6) is met for all forecast paths of the observed variables over this time interval.

Let $\bar{M} = M(M'M)^{-1}$ while $\bar{M}_\perp = M_\perp(M'_\perp M_\perp)^{-1}$.[101] We can now express the state equation (5.2) as

$$\xi_t = F\xi_{t-1} + B_0\bar{M}_\perp \eta_t^{(q-q_m)} + B_0\bar{M}\eta_t^{(q_m)}. \tag{12.8}$$

Turning first to period $T + 1$ we know that if we substitute for y_{T+1} using the measurement equation (5.1) and the rewritten state equation (12.8), the conditioning on the vector z_{T+1} in (12.6) implies that:

$$z_{T+1} = K'_1 A' x_{T+1} + K'_1 w_{T+1} + K'_1 H' F\xi_T + K'_1 H' B_0\bar{M}_\perp \eta_{T+1}^{(q-q_m)} + K'_1 H' B_0\bar{M}\eta_{T+1}^{(q_m)} + u_T. \tag{12.9}$$

Provided that the $q_m \times q_m$ matrix $K'_1 H' B_0 \bar{M}$ has full rank, the economic shocks $\eta_{T+1}^{(q_m)}$ can be uniquely specified such that a fixed value for z_{T+1} is obtained.

With $\eta_{T+1}^{(q_m)}$ being computed such that (12.9) holds, it follows from the state equation (12.8) that the state vector at $T + 1$ conditional on the path for z_{T+i} is determined by

$$\xi_{T+1}^{(q_m)} = F\xi_T + B_0\bar{M}_\perp \eta_{T+1}^{(q-q_m)} + B_0\bar{M}\eta_{T+1}^{(q_m)},$$

while the measurement equation implies that y_{T+1} is given by:

$$y_{T+1}^{(q_m)} = A' x_{T+1} + H'\xi_{T+1}^{(q_m)} + w_{T+1}.$$

We may now continue with period $T + 2$ where the shocks $\eta_{T+2}^{(q_m)}$ can, for given parameter values, be determined from z_{T+2}, x_{T+2}, w_{T+2}, $\xi_{T+1}^{(q_m)}$, $\eta_{T+2}^{(q-q_m)}$, $y_{T+1}^{(q_m)}$, and u_T. In fact, it is now straightforward to show that the values for the economic shocks which guarantee that the conditioning path $z_{T+1}, \ldots, z_{T+g}$ is always met are:

$$\eta_{T+i}^{(q_m)} = \left(K'_1 H' B_0 \bar{M}\right)^{-1} \left[z_{T+i} - K'_1 A' x_{T+i} - K'_1 w_{T+i} - K'_1 H' F\xi_{T+i-1}^{(q_m)} \right.$$
$$\left. - K'_1 H' B_0 \bar{M}_\perp \eta_{T+i}^{(q-q_m)} - \sum_{j=1}^{i-1} K'_{2j} y_{T+i-j}^{(q_m)} - u_T \right], \quad i = 1, \ldots, g, \tag{12.10}$$

while the states and the observed variables evolve according to:

$$\xi_{T+i}^{(q_m)} = F\xi_{T+i-1}^{(q_m)} + B_0\bar{M}_\perp \eta_{T+i}^{(q-q_m)} + B_0\bar{M}\eta_{T+i}^{(q_m)}, \quad i = 1, \ldots, g, \tag{12.11}$$

and

$$y_{T+i}^{(q_m)} = A' x_{T+i} + H'\xi_{T+i}^{(q_m)} + w_{T+i}, \quad i = 1, \ldots, g, \tag{12.12}$$

while $\xi_T^{(q_m)} = \xi_T$.

For $i > g$ there are not any direct restrictions on the possible paths for the observed variables other than that the state vector at $T + g$ needs to be taken into account.

The procedure described here makes it straightforward to calculate conditional predictive distributions. For a given draw of θ from its posterior distribution, the period T state vector ξ_T

---

[101] In many situations we will have that $\bar{M} = M$ since M is typically a 0-1 matrix with only one unit element per column. Similarly, we can always select $M_\perp$ such that $\bar{M}_\perp = M_\perp$.

is drawn from $N(\xi^{(z)}_{T|T+g}, P^{(z)}_{T|T+g})$. The determination of the mean and the covariance matrix of the state vector that utilizes the conditioning assumptions is discussed in Section 12.2.4. The default behavior in YADA is to set $\xi^{(z)}_{T|T+g} = \xi_{T|T}$ and $P^{(z)}_{T|T+g} = P_{T|T}$ and, hence, to ignore the conditioning assumptions. The preferred choice of initial state distribution can be selected on the *Miscellaneous* tab.

Next, the economic shocks $\eta^{(q-q_m)}_{T+i}$ are drawn from $N(0, M'_\perp M_\perp)$ and w_{T+i} from $N(0, R)$. Based on the conditioning assumptions, the shocks $\eta^{(q_m)}_{T+i}$ are calculated from (12.10), the state vector from (12.11), and the observed variables from (12.12) in a sequential manner until $i = g + 1$, when the economic shocks $\eta^{(q_m)}_{T+i}$ are drawn from $N(0, M'M)$ until $i = h$. This provides one path for $y_{T+1}, \ldots, y_{T+h}$. Given the value of θ we may next repeat this procedure yielding P paths. By considering S draws of θ we can calculate a total of PS sample paths for the observed variables that take the conditioning into account.

To evaluate the forecast error covariances let us first combine the measurement and the state equation such that y_{T+i} is expressed as a function of the parameters, x_{T+i}, ξ_T, w_{T+i}, and the sequence of economic shocks $\eta_{T+i}, \ldots, \eta_{T+1}$. This relationship is:

$$y_{T+i} = A'x_{T+i} + H'F^i\xi_T + H' \sum_{j=0}^{i-1} F^j B_0 \eta_{T+i-j} + w_{T+i}, \quad i = 1, \ldots, h. \tag{12.13}$$

For periods $T + 1$ to $T + g$ we can stack the equations as

$$
\begin{bmatrix} y_{T+g} \\ y_{T+g-1} \\ \vdots \\ y_{T+1} \end{bmatrix} = \begin{bmatrix} A'x_{T+g} \\ A'x_{T+g-1} \\ \vdots \\ A'x_{T+1} \end{bmatrix} + \begin{bmatrix} H'F^g \\ H'F^{g-1} \\ \vdots \\ H'F \end{bmatrix} \xi_T + \begin{bmatrix} w_{T+g} \\ w_{T+g-1} \\ \vdots \\ w_{T+1} \end{bmatrix} +
$$

$$
+ \begin{bmatrix} H'B_0 & H'FB_0 & \cdots & H'F^{g-1}B_0 \\ 0 & H'B_0 & & H'F^{g-2}B_0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & & H'B_0 \end{bmatrix} \begin{bmatrix} \eta_{T+g} \\ \eta_{T+g-1} \\ \vdots \\ \eta_{T+1} \end{bmatrix},
$$

or

$$Y_{T+g} = X_{T+g} + G\xi_T + W_{T+g} + DN_{T+g}. \tag{12.14}$$

The conditioning relations in equation (12.6) can also be stacked for the time periods $T + 1$ until $T + g$. Here we find that

$$
\begin{bmatrix} z_{T+g} \\ z_{T+g-1} \\ \vdots \\ z_{T+1} \end{bmatrix} = \begin{bmatrix} K'_1 & K'_{21} & \cdots & K'_{2g-1} \\ 0 & K'_1 & & K'_{2g-2} \\ \vdots & & \ddots & \\ 0 & 0 & & K'_1 \end{bmatrix} \begin{bmatrix} y_{T+g} \\ y_{T+g-1} \\ \vdots \\ y_{T+1} \end{bmatrix} + \begin{bmatrix} u_T \\ u_T \\ \vdots \\ u_T \end{bmatrix},
$$

or

$$Z_{T+g} = K'Y_{T+g} + U_T. \tag{12.15}$$

Let us now substitute for Y_{T+g} from (12.14) into the conditioning assumptions (12.15). Moreover, define the stacked decomposition of the structural shocks

$$N_{T+g} = (I_g \otimes \bar{M}_\perp)N^{(q-q_m)}_{T+g} + (I_g \otimes \bar{M})N^{(q_m)}_{T+g},$$

where $N^{(q_m)}_{T+g} = [\eta^{(q_m)'}_{T+g} \cdots \eta^{(q_m)'}_{T+1}]'$ and $N^{(q-q_m)}_{T+g}$ is defined similarly. Rearranging terms we obtain

$$K'D(I_g \otimes \bar{M})N^{(q_m)}_{T+g} = Z_{T+g} - K'X_{T+g} - K'G\xi_T - K'W_{T+g} - U_T - K'D(I_g \otimes \bar{M}_\perp)N^{(q-q_m)}_{T+g}.$$

Provided that the $q_m g \times q_m g$ matrix $K'D(I_g \otimes \bar{M})$ is invertible, this gives us a stacked version of equation (12.10). In fact, the matrix in question is invertible when the $q_m \times q_m$ dimensional matrix $K_1' H' B_0 \bar{M}$ has full rank which, consistent with the discussion above, guarantees that a unique solution for $N_{T+g}^{(q_m)}$ exists. Collecting these results, the stacked vector of shocks is given by

$$N_{T+g} = \mu_{N,T+g} + \left(I_{qg} - (I_g \otimes \bar{M}) \left[K'D(I_g \otimes \bar{M}) \right]^{-1} K'D \right) (I_g \otimes \bar{M}_\perp) N_{T+g}^{(q-q_m)}, \qquad (12.16)$$

where

$$\mu_{N,T+g} = (I_g \otimes \bar{M}) \left[K'D(I_g \otimes \bar{M}) \right]^{-1} k_{T+g},$$
$$k_{T+g} = Z_{T+g} - U_T - K'(X_{T+g} + G\xi_T + W_{T+g}).$$

With this in mind, it is straightforward to show that conditional on $\mathcal{Y}_T$, Z_{T+g} and θ the mean of the stacked shock vector N_{T+g} is

$$\tilde{\mu}_{N,T+g} = (I_g \otimes \bar{M}) \left[K'D(I_g \otimes \bar{M}) \right]^{-1} \left(Z_{T+g} - K'(X_{T+g} + G\xi_{T|T+g}^{(z)}) - U_T \right). \qquad (12.17)$$

It follows that the conditional forecast of Y_{T+g} can be written as

$$E\left[Y_{T+g} | \mathcal{Y}_T, Z_{T+g}; \theta \right] = X_{T+g} + G\xi_{T|T+g}^{(z)} + D\tilde{\mu}_{N,T+g}. \qquad (12.18)$$

Hence, premultiplication of both sides by K' we find that the conditioning assumptions are satisfied by the conditional mean predictions. Notice that the expectation is here also taken with respect to M, i.e., the selection of shocks used the ensure that the conditioning assumptions are satisfied. For notational convenience, however, it has been left out of the conditioning information.

Furthermore, it can also be shown that the covariance matrix of the observed variable for fixed θ is given by

$$C\left(Y_{T+g} | \mathcal{Y}_T, Z_{T+g}; \theta \right) = \tilde{D}GP_{T|T+g}^{(z)}G'\tilde{D}' + \tilde{D}(I_g \otimes R)\tilde{D}' + \tilde{D}D(I_g \otimes \bar{M}_\perp M_\perp')D'\tilde{D}', \qquad (12.19)$$

where

$$\tilde{D} = I_{ng} - D(I_g \otimes \bar{M}) \left[K'D(I_g \otimes \bar{M}) \right]^{-1} K'.$$

Premultiplication of the covariance matrix in (12.19) by K' or postmultiplication by K we find that the resulting expression is zero. Hence, the uncertainty about the conditioning assumption is zero. Moreover, the first term of the right hand side of (12.19) gives the share of the forecast error covariances for fixed θ due to state variable uncertainty, the second term yielding the measurement error share, while the third provides the shock uncertainty share.

For forecast horizons i beyond the conditioning horizon g we have that

$$y_{T+i} = A'x_{T+i} + H'F^i\xi_T + \sum_{j=0}^{i-g-1} H'F^j B_0 \eta_{T+i-j} + H'F^{i-g}\bar{B}N_{T+g} + w_{T+i}, \quad i = g+1, \ldots, h, \quad (12.20)$$

where $\bar{B}$ is an $r \times gq$ matrix given by

$$\bar{B} = \begin{bmatrix} B_0 & FB_0 & \cdots & F^{g-1}B_0 \end{bmatrix}.$$

It therefore follows that

$$E\left[y_{T+i} | \mathcal{Y}_T, Z_{T+g}; \theta \right] = A'x_{T+i} + H'F^i\xi_{T|T+g}^{(z)} + H'F^{i-g}\bar{B}\tilde{\mu}_{N,T+g}, \qquad (12.21)$$

for $i = g + 1, \ldots, h$. Moreover, the forecast error covariance matrix for these horizons can be shown to be given by

$$C\big(y_{T+i}|\mathbf{y}_T, Z_{T+g}; \theta\big) = H'F^{i-g}\tilde{G}P^{(z)}_{T|T+g}\tilde{G}'\big(F'\big)^{i-g}H + H'F^{i-g}\tilde{K}\big(I_g \otimes R\big)\tilde{K}'\big(F'\big)^{i-g}H +$$

$$+ R + \sum_{j=0}^{i-g-1} H'F^j B_0 B_0'\big(F'\big)^j H + \tag{12.22}$$

$$+ H'F^{i-g}\bar{B}\tilde{N}\big(I_g \otimes \bar{M}_\perp M_\perp'\big)\tilde{N}'\bar{B}'\big(F'\big)^{i-g}H,$$

for $i = g + 1, \ldots, h$, and where

$$\tilde{K} = \bar{B}\big(I_g \otimes \bar{M}\big)\big[K'D\big(I_g \otimes \bar{M}\big)\big]^{-1}K',$$
$$\tilde{G} = F^g - \tilde{K}G, \tag{12.23}$$
$$\tilde{N} = I_{qg} - \big(I_g \otimes \bar{M}\big)\big[K'D\big(I_g \otimes \bar{M}\big)\big]^{-1}K'D.$$

The first term on the right hand side of (12.22) represents state uncertainty for fixed parameters, the second and third give the measurement error uncertainty, while the last two provide a measure of shock uncertainty. Moreover, it is worth pointing out that when all the eigenvalues of F are less than unity in absolute terms the first, second, and fifth terms on the right hand side converge to zero as $i \to \infty$ so that the forecast error covariance matrix approaches the covariance matrix of the observed variables for fixed parameters.

As in equation (12.3) we now find that the forecast error covariance matrix for the observed variables can be decomposed as

$$C\big(y_{T+i}|\mathbf{y}_T, Z_{T+g}\big) = E_T\big[C\big(y_{T+i}|\mathbf{y}_T, Z_{T+g}; \theta\big)\big] + C_T\big[E\big(y_{T+i}|\mathbf{y}_T, Z_{T+g}; \theta\big)\big], \tag{12.24}$$

for $i = 1, \ldots, h$, and where E_T and C_T denote the expectation and covariance with respect to the posterior of θ at time T.[102] The first term on the right hand side gives us a decomposition of the forecast uncertainty into state, measurement error, and shock uncertainty, while the second term provides a measure of parameter uncertainty.

It is also interesting to look at the case when the distribution of the conditional forecasts is recentered using some values of the observed variables over the forecast horizon. In practise, the recentering values may be projected values of the observed variables from an institution that provides forecasts. They can also be actual values when the forecast study looks at past or real time data and where actual values are taken from later vintages than the vintage available at T.

An approach to such recentering is to first compute the smooth estimates of the state variables, the structural shocks and the measurement errors based on the centering values. Let these values over the conditioning horizon be given by $\bar{Y}_{T+g}$, the smooth estimates of the structural shocks and the measurement errors over the conditioning horizon be $\bar{W}_{T+g}$ and $\bar{N}_{T+g}$, respectively, while the smooth estimate at T is $\bar{\xi}_T$. In terms of the notation in equation (12.14) we obtain the following:

$$\bar{Y}_{T+g} = X_{T+g} + G\bar{\xi}_T + \bar{W}_{T+g} + D\bar{N}_{T+g}.$$

Since $g \le h$, it is assumed that the recentering data covers periods up to $T + h$ such that the smooth estimates of the state variables, structural shocks and measurement errors are based on data up to $T + h$. This ensures that the recentering of periods $T + g + 1$ until $T + h$ is consistent with the recentering over the conditioning sample.

The recentered variables are obtained by adding $\bar{Y}_{T+g}$ to Y_{T+g} and subtracting the mean in equation (12.18). This provides us with

$$\tilde{Y}_{T+g} = X_{T+g} + G\big(\xi_T + \bar{\xi}_T - \xi^{(z)}_{T|T+g}\big) + W_{T+g} + \bar{W}_{T+g} + DN_{T+g} + D\big(\bar{N}_{T+g} - \tilde{\mu}_{N,T+g}\big),$$

where $\xi_T \sim N(\xi_{T|T+g}^{(z)}, P_{T|T+g}^{(z)})$. Letting $\tilde{\xi}_T = \xi_T + \bar{\xi}_T - \xi_{T|T+g}^{(z)}$ it follows that $\tilde{\xi}_T \sim N(\bar{\xi}_T, P_{T|T+g}^{(z)})$. The value of N_{T+g} needs to be adjusted for the recentering in order for $\tilde{Y}_{T+g}$ to satisfy the conditioning restrictions:

$$Z_{T+g} = K'\tilde{Y}_{T+g} + U_T.$$

Some uplifting and refreshing algebra later we find that the controlled shocks are now given by

$$N_{T+g}^{(q_m)} = \left[K'D\left(I_g \otimes \bar{M}\right)\right]^{-1} \left(k_{T+g}^* - K'D\left(I_g \otimes \bar{M}_\perp\right)\right) N_{T+g}^{(q-q_m)},$$

$$k_{T+g}^* = Z_{T+g} - U_T - K'\left(X_{T+g} + G\tilde{\xi}_T + W_{T+g} + \bar{W}_{T+g}\right) - K'D\left(\bar{N}_{T+g} - \tilde{\mu}_{N,T+g}\right).$$

The mean of the shocks N_{T+g} is equal to

$$\mu_{N,T+g}^* = \left(I_g \otimes \bar{M}\right)\left[K'D\left(I_g \otimes \bar{M}\right)\right]^{-1} \bar{k}_{T+g}^*,$$

$$\bar{k}_{T+g}^* = Z_{T+g} - U_T - K'\left(X_{T+g} + G\bar{\xi}_T + \bar{W}_{T+g}\right) - K'D\left(\bar{N}_{T+g} - \tilde{\mu}_{N,T+g}\right),$$

with the effect that the predicted mean of $\tilde{Y}_{T+g}$ is

$$
\begin{aligned}
E\left[\tilde{Y}_{T+g}|\mathcal{Y}_T, Z_{T+g}; \theta\right] = {} & X_{T+g} + G\bar{\xi}_T + \bar{W}_{T+g} + D\left(\bar{N}_{T+g} - \tilde{\mu}_{N,T+g}\right) + \\
& + D\left(I_g \otimes \bar{M}\right)\left[K'D\left(I_g \otimes \bar{M}\right)\right]^{-1} \bar{k}_{T+g}^*.
\end{aligned}
\tag{12.25}
$$

Premultiplying both sides by K', it follows that the mean prediction indeed satisfies the conditioning restrictions. Also, the deviation of this mean from $\bar{Y}_{T+g}$ depends on the difference between the mean value of the shocks with and without recentering. This deviation is given by

$$\mu_{N,T+g}^* - \tilde{\mu}_{N,T+g} = \left(I_g \otimes \bar{M}\right)\left[K'D\left(I_g \otimes \bar{M}\right)\right]^{-1} K'\left[G\left(\xi_{T|T+g}^{(z)} - \bar{\xi}_T\right) - \bar{W}_{T+g} - D\left(\bar{N}_{T+g} - \tilde{\mu}_{N,T+g}\right)\right].$$

It can be seen from equation (12.17) that $\tilde{\mu}_{N,T+g}$ reflects only the mean of the shocks in $N_{T+g}^{(q_r)}$ without recentering.

For horizons $\tau = T + g + 1, \ldots, T + h$, the recentering method of the forecasts implies that we should add the smooth estimate of the measurement error in period τ to the projected value of y_τ and add the smooth estimate of the structural shock in period τ to the corresponding period shock value.

12.2.2. *Control of the Distribution of the Shocks*

The second method for ensuring that the conditioning assumption in equation (12.6) is satisfied relies on selecting appropriate first and second moments for the distribution of the economic shocks. The approach, suggested by Waggoner and Zha (1999) for Bayesian VAR models, is based on the observation that the conditioning assumption imposes linear restrictions on the shocks. The mean of the shocks can be selected such that the restrictions are satisfied on average, while the deviation of the shocks from their mean should be orthogonal to the restrictions.

To see how this idea can be used in the state-space framework, let us substitute for Y_{T+g} from (12.14) into (12.15) and rearrange terms. This gives us the following $q_m g$ equations that the distribution of the shocks must satisfy if the conditioning assumptions are to hold:

$$K'DN_{T+g} = k_{T+g}, \tag{12.26}$$

where $k_{T+g} = Z_{T+g} - U_T - K'(X_{T+g} + G\xi_T + W_{T+g})$. As in Waggoner and Zha (1999), the distribution of the economic shocks N_{T+g} conditional on the restriction (12.26) is normal with mean $\mu_{N,T+g}$ and *idempotent* covariance matrix $\Sigma_{N,T+g}$, where the state-space model gives us

$$
\begin{aligned}
\mu_{N,T+g} &= D'K\left(K'DD'K\right)^{-1}k_{T+g}, \\
\Sigma_{N,T+g} &= I_{qg} - D'K\left(K'DD'K\right)^{-1}K'D.
\end{aligned}
\tag{12.27}
$$

Notice that the common inverse in these expressions exists when the $q_m \times q$ matrix $K_1'H'B_0$ has rank q_m. This is similar to the requirement in (12.9) that the $q_m \times q_m$ matrix $K_1'H'B_0\bar{M}$ is invertible.

The Waggoner and Zha approach to conditional forecasting with a DSGE model may now proceed as follows. For a given draw of θ from its posterior distribution, the period T state vector ξ_T is drawn from $N(\xi_{T|T+g}^{(z)}, P_{T|T+g}^{(z)})$. Next, the measurement errors w_{T+i} are drawn from $N(0, R)$ for $i = 1, \ldots, h$. The economic shocks η_{T+i} can now be drawn from $N(\mu_{N,T+g}, \Sigma_{N,T+g})$ for $i = 1, \ldots, g$ and from $N(0, I_q)$ for $i = g + 1, \ldots, h$. Given the drawn economic shocks, the state vector ξ_{T+i} can be simulated for $i = 1, \ldots, h$ using the state equation (5.2). With paths for the state vector and the measurement errors we can next calculate a path for the observed variables $y_{T+1}, \ldots, y_{T+h}$ via the measurement equation (5.1). Repeating these steps P times for each one of the S different draws of θ from its posterior distribution, gives us PS paths of the observed variables that satisfy the conditioning assumptions and that make up the conditional predictive distribution of $y_{T+1}, \ldots, y_{T+h}$.

For each given value of θ the mean prediction for the observed variables is given by:

$$E\left[y_{T+i}|\mathcal{Y}_T, Z_{T+g}; \theta\right] = A'x_{T+i} + H'\xi_{T+i|T}^{(*)}, \quad i = 1, \ldots, h.$$

The value for the state vector $\xi_{T+i|T}^{(*)}$ evolves according to:

$$\xi_{T+i|T}^{(*)} = \begin{cases} F\xi_{T+i-1|T}^{(*)} + B_0\bar{\mu}_{T+i} & \text{if } i = 1, \ldots, g, \\ F\xi_{T+i-1|T}^{(*)} & \text{otherwise,} \end{cases}$$

where $\xi_{T|T}^{(*)} = \xi_{T|T+g}^{(z)}$. The vector $\bar{\mu}_{T+i}$ is determined from $\mu_{N,T+g}$ by setting $k_{T+g} = \bar{k}_{T+g}$. The latter value is obtained by setting W_{T+g} and ξ_T equal to their means conditional on the sample data and the parameters, i.e., to zero and $\xi_{T|T+g}^{(z)}$, respectively. This gives us

$$\bar{\mu}_{N,T+g} = D'K\left(K'DD'K\right)^{-1}\bar{k}_{T+g} = \begin{bmatrix} \bar{\mu}_{T+g} \\ \vdots \\ \bar{\mu}_{T+1} \end{bmatrix},$$

and where $\bar{k}_{T+g} = Z_{T+g} - U_T - K'(X_{T+g} + G\xi_{T|T+g}^{(z)})$.

In order to determine the population moments of the predictive distribution as well as the decomposition of the forecast error variances into state, shock, measurement error, and parameter uncertainty we can start from equation (12.14) and make use of the moments in equation (12.27). This means that we can express the stacked system as

$$Y_{T+g} = X_{T+g} + G\xi_T + W_{T+g} + D\mu_{N,T+g} + D\Sigma_{N,T+g}^{1/2}\gamma_{T+g},$$

where $\gamma_{T+g} = N(0, I_{g(q-q_m)})$ and independent of W_{T+g} and ξ_T, while $\Sigma_{N,T+g}^{1/2}$ is treated as a $qg \times (q - q_m)g$ matrix that satisfies $\Sigma_{N,T+g} = \Sigma_{N,T+g}^{1/2}\Sigma_{N,T+g}^{1/2\prime}$. It follows that

$$E\left[Y_{T+g}|\mathcal{Y}_T, Z_{T+g}; \theta\right] = X_{T+g} + G\xi_{T|T+g}^{(z)} + D\bar{\mu}_{N,T+g}. \tag{12.28}$$

Premultiplying both sides of this equation by K' we find that the conditioning assumptions are satisfied by the conditional mean predictions.

The covariance matrix of the forecast errors over the conditioning horizon can now be shown to be given by

$$C\left(Y_{T+g}|\mathcal{Y}_T, Z_{T+g}; \theta\right) = \bar{D}GP_{T|T+g}^{(z)}G'\bar{D}' + \bar{D}\left(I_g \otimes R\right)\bar{D}' + D\Sigma_{N,T+g}D', \tag{12.29}$$

where

$$\bar{D} = I_{ng} - DD'K\left(K'DD'K\right)^{-1}K'.$$

The first term represents state variable uncertainty for fixed parameters, the second term gives the measurement error uncertainty, and the third provides the shock uncertainty for fixed parameters. Notice also that premultiplication of the right hand side of (12.29) by K' and postmultiplication by K gives a zero covariance matrix.

For forecast horizons i beyond the conditioning horizon g it is straightforward to show via equation (12.20) that

$$E[y_{T+i}|\mathcal{Y}_T, Z_{T+g};\theta] = A'x_{T+i} + H'F^i\xi^{(z)}_{T|T+g} + H'F^{i-g}\bar{B}\bar{\mu}_{N,T+g}, \quad i = g+1,\dots,h. \quad (12.30)$$

Furthermore, the forecast error covariance matrix for fixed parameters is given by

$$C(y_{T+i}|\mathcal{Y}_T, Z_{T+g};\theta) = H'F^{i-g}\bar{G}P^{(z)}_{T|T+g}\bar{G}'(F')^{i-g}H + H'F^{i-g}\bar{K}(I_g \otimes R)\bar{K}'(F')^{i-g}H +$$

$$+ R + \sum_{j=0}^{i-g-1} H'F^j B_0 B_0'(F')^j H + H'F^{i-g}\bar{B}\Sigma_{N,T+g}\bar{B}'(F')^{i-g}H, \quad (12.31)$$

for $i = g+1,\dots,h$, and where

$$\bar{K} = \bar{B}D'K(K'DD'K)^{-1}K',$$

$$\bar{G} = F^g - \bar{K}G.$$

The first term on the right hand side of (12.31) represents state variable uncertainty for fixed parameters, the second and third give the measurement error uncertainty, while the last two provide a measure of shock uncertainty. Moreover, it is worth pointing out that when all eigenvalues of F are inside the unit circle, then the first, second, and fifth terms on the right hand side converge to zero as $i \to \infty$ so that the forecast error covariance matrix approaches the covariance matrix of the observed variables for fixed parameters. In addition, the full decomposition into state variable, measurement error, shock, and parameter uncertainty can be performed as in equation (12.24) using the results in equations (12.28)–(12.31) as input.

Recentering is for this conditioning method achieved by adding $\bar{Y}_{T+g}$ to Y_{T+g} and subtracting $E[Y_{T+g}|\mathcal{Y}_T, Z_{T+g};\theta]$ in equation (12.28). The recentered variables are therefore given by

$$\tilde{Y}_{T+g} = X_{T+g} + G\tilde{\xi}_{T+g} + W_{T+g} + \bar{W}_{T+g} + DN_{T+g} + D(\bar{N}_{T+g} - \bar{\mu}_{N,T+g}). \quad (12.32)$$

As a consequence, the mean of the distribution of N_{T+g} needs to be adjusted in order for $\tilde{Y}_{T+g}$ to satisfy the conditioning restrictions. Some algebra later it can be shown that the mean of the distribution of N_{T+g} is now equal to

$$\tilde{\mu}_{N,T+g} = D'K(K'DD'K)^{-1}\tilde{k}_{T+g},$$

where

$$\tilde{k}_{T+g} = Z_{T+g} - U_T - K'(X_{T+g} + G\tilde{\xi}_{T+g} + W_{T+g} + \bar{W}_{T+g}) - K'D(\bar{N}_{T+g} - \bar{\mu}_{N,T+g}),$$

while the covariance matrix of N_{T+g} remains unchanged. It can now be shown that the conditional mean prediction of $\tilde{Y}_{T+g}$

$$E[\tilde{Y}_{T+g}|\mathcal{Y}_T, Z_{T+g};\theta] = X_{T+g} + G\bar{\xi}_T + \bar{W}_{T+g} + D(\bar{N}_{T+g} - \bar{\mu}_{N,T+g})$$

$$+ DD'K(K'DD'K)^{-1}[Z_{T+g} - U_T \quad (12.33)$$

$$- K'(X_{T+g} + G\bar{\xi}_T + \bar{W}_{T+g}) - K'D(\bar{N}_{T+g} - \bar{\mu}_{N,T+g})].$$

Premultiplying both sides by K', we immediately find that the mean value satisfies the conditioning assumptions. Also, the deviation of the mean prediction from the centered values $\tilde{Y}_{T+g}$ is determined by the difference between the mean of $\tilde{\mu}_{N,T+g}$ and $\bar{\mu}_{N,T+g}$. This difference is given by

$$E[\tilde{\mu}_{N,T+g}] - \bar{\mu}_{N,T+g} = D'K(K'DD'K)^{-1}K'[G(\xi^{(z)}_{T|T+g} - \bar{\xi}_T) - \bar{W}_{T+g} - D(\bar{N}_{T+g} - \bar{\mu}_{N,T+g})].$$

12.2.3. *Control of the Distribution of a Subset of the Shocks*

To guarantee that the conditioning assumptions in equation (12.6) are met, a third approach builds on a mixture of the two previous methods. That is, a subset of the shocks are used to control the distribution of the shocks needed for the assumptions to be met, while the remaining shocks are free, i.e., have their usual normal distribution. In case the subset of shocks used to control the distribution is equal to the full set of shocks, then the mixed method is identical

to the method in Section 12.2.2. At the other extreme, if this subset has the same dimension as the number of conditioning assumptions then the mixed case is equal to the method in Section 12.2.1. Hence, provided that the number of conditioning assumptions is less than the number of shocks minus one ($q_m < q - 1$) the third method is different from the other two if $q_r = q - 1$.[103]

Recall that $q_m \leq \min\{q, n\}$ and assume now that $q_m \leq q_r \leq q$ shocks are normally distributed with a particular mean and covariance matrix which guarantees that the conditioning assumptions are satisfied, while $q - q_r$ shocks are zero mean normal. Let M be a $q \times q_r$ matrix with rank q_r such that $\eta_t^{(q_r)} = M'\eta_t$. That is, M gives q_r linear combinations of the shocks that will be used to ensure that the conditioning assumptions are satisfied. Typically, this means that q_r specific shocks among the q available shocks are selected for that purpose. Similarly, let $\eta_t^{(q-q_r)}$ be the remaining shocks which are determined as $M'_\perp \eta_t$, where $M_\perp$ is a $q \times (q - q_r)$ matrix with rank $q - q_r$. Moreover, $M'_\perp M = 0$, while

$$\eta_t = \bar{M}\eta_t^{(q_r)} + \bar{M}_\perp \eta_t^{(q-q_r)},$$

where $\bar{M} = M(M'M)^{-1}$ and $\bar{M}_\perp = M_\perp(M'_\perp M_\perp)^{-1}$. We shall also assume that $q_r > q_m$ so that the shocks here are different from those in Section 12.2.1, while $\eta_t^{(q-q_r)} \sim N(0, M'_\perp M_\perp)$ are the $q - q_r$ "free" shocks.

The stacked vector with shocks N_{T+g} can therefore be expressed as

$$N_{T+g} = \left(I_g \otimes \bar{M}\right)N_{T+g}^{(q_r)} + \left(I_g \otimes \bar{M}_\perp\right)N_{T+g}^{(q-q_r)}, \tag{12.34}$$

where $N_{T+g}^{(q_r)} = [\eta_{T+g}^{(q_r)\prime} \cdots \eta_{T+1}^{(q_r)\prime}]'$ and $N_{T+g}^{(q-q_r)}$ is defined analogously using the $\eta_{T+i}^{(q-q_r)}$ shocks. The $q_m g$ restrictions on the $N_{T+g}^{(q_r)}$ vector are now given by

$$K'D\left(I_g \otimes \bar{M}\right)N_{T+g}^{(q_r)} = k_{T+g}^{(q_r)}, \tag{12.35}$$

where

$$k_{T+g}^{(q_r)} = Z_{T+g} - U_T - K'\left(X_{T+g} + G\xi_T + W_{T+g} + D\left(I_g \otimes \bar{M}_\perp\right)N_{T+g}^{(q-q_r)}\right).$$

This may be compared with the restrictions in equation (12.26) under the pure Waggoner and Zha approach. If $q_r = q$, then $M = I_q$ and $M_\perp$ is empty so that the expressions in (12.35) and (12.26) are equivalent.

To satisfy the restrictions in (12.35), the $q_r g$ shocks in $N_{T+g}^{(q_r)}$ conditional on the $(q - q_r)g$ shocks $N_{T+g}^{(q-q_r)}$ are normally distributed with mean and idempotent covariance matrix given by

$$\begin{aligned}
\mu_{N,T+g}^{(q_r)} &= \left(I_g \otimes \bar{M}'\right)D'K\left(K'D\left(I_g \otimes \bar{M}\bar{M}'\right)D'K\right)^{-1}k_{T+g}^{(q_r)}, \\
\Sigma_{N,T+g}^{(q_r)} &= I_{q_r g} - \left(I_g \otimes \bar{M}'\right)D'K\left(K'D\left(I_g \otimes \bar{M}\bar{M}'\right)D'K\right)^{-1}K'D\left(I_g \otimes \bar{M}\right).
\end{aligned} \tag{12.36}$$

Notice that the common inverse in these expressions exists when the $q_m \times q_r$ matrix $K'_1 H' B_0 M$ has rank q_m. This is similar to the requirement in (12.9) that the corresponding matrix has full rank q_m.

The mixed approach to conditional forecasting with a DSGE model can now be implemented as follows. For a given draw of θ from its posterior distribution, the period T state vector ξ_T is drawn from $N(\xi_{T|T+g}^{(z)}, P_{T|T+g}^{(z)})$. The measurement errors w_{T+i} are drawn from $N(0, R)$ for $i = 1, \ldots, h$. The economic shocks $N_{T+g}^{(q-q_r)}$ are drawn from $N(0, [I_g \otimes M'_\perp M_\perp])$, while $N_{T+g}^{(q_r)}$ is drawn from $N(\mu_{N,T+g}^{(q_r)}, \Sigma_{N,T+g}^{(q_r)})$. Finally, if $g < h$, then η_{T+i} is drawn from $N(0, I_q)$ for $i = g + 1, \ldots, h$. Given these draws a path for the observed variables is simulated through the

---

[103] The third method can also be different from the direct control of the shocks method if each shock is mapped to a specific conditioning assumption under the direct control case and the number of conditioning assumptions, q_m, varies of the prediction sample. This means that q_r can remain fixed, while q_m is actually time varying. Furthermore, this remains true when $q_r = \max_{i=\{1,\ldots,g\}} q_{m,T+i}$.

state-space representation. Repeating these steps P times for each one of the S draws of θ from its posterior distribution gives us PS paths for the observed variables that satisfy the conditioning assumptions and that make up a sample from the conditional predictive distribution of $y_{T+1}, \ldots, y_{T+h}$.

The determination of the mean and covariances of the predictive distribution of the observed variables for $T+1, \ldots, T+h$ conditional on (12.35) can proceed as in Section 12.2.2. First, the stacked system can be expressed as

$$Y_{T+g} = X_{T+g} + G\xi_T + W_{T+g} + D\big(I_g \otimes \bar{M}_\perp\big)N_{T+g}^{(q-q_r)}$$
$$+ D\big(I_g \otimes \bar{M}\big)\mu_{N,T+g}^{(q_r)} + D\big(I_g \otimes \bar{M}\big)\Sigma_{N,T+g}^{(q_r)1/2}\gamma_{T+g}^{(q_r)},$$

where $\Sigma_{N,T+g}^{(q_r)1/2}$ is a $q_r g \times (q_r - q_m)g$ matrix that satisfies $\Sigma_{N,T+g}^{(q_r)1/2}\Sigma_{N,T+g}^{(q_r)1/2\prime} = \Sigma_{N,T+g}^{(q_r)}$, while $\gamma_{T+g}^{(q_r)}$ is $N(0, I_{g(q_r - q_m)})$ and independent of W_{T+g}, ξ_T, and $N_{T+g}^{(q-q_r)}$. We therefore find that

$$E\big[Y_{T+g}|\mathcal{Y}_T, Z_{T+g};\theta\big] = X_{T+g} + G\xi_{T|T+g}^{(z)} + D\big(I_g \otimes \bar{M}\big)\bar{\mu}_{N,T+g}^{(q_r)}, \qquad (12.37)$$

where

$$\bar{\mu}_{N,T+g}^{(q_r)} = \big(I_g \otimes \bar{M}'\big)D'K\big(K'D\big(I_g \otimes \bar{M}\bar{M}'\big)D'K\big)^{-1}\bar{k}_{T+g}^{(q_r)},$$
$$\bar{k}_{T+g}^{(q_r)} = Z_{T+g} - U_T - K'\left(X_{T+g} + G\xi_{T|T+g}^{(z)}\right).$$

Premultiplication of both sides of equation (12.37) by K' we find that the conditioning assumptions are satisfied by the conditional mean predictions.

The covariance matrix of the forecast errors over the conditioning horizon can after some algebra be shown to be equal to

$$C\big(Y_{T+g}|\mathcal{Y}_T, Z_{T+g};\theta\big) = \bar{D}GP_{T|T+g}^{(z)}G'\bar{D}' + \bar{D}\big(I_g \otimes R\big)\bar{D}' + \bar{D}D\big(I_g \otimes \bar{M}_\perp M'_\perp\big)D'\bar{D}'$$
$$+ D\big(I_g \otimes \bar{M}\big)\Sigma_{N,T+g}^{(q_r)}\big(I_g \otimes \bar{M}'\big)D', \qquad (12.38)$$

where

$$\bar{D} = I_{ng} - D\big(I_g \otimes \bar{M}\bar{M}'\big)D'K\big(K'D\big(I_g \otimes \bar{M}\bar{M}'\big)D'K\big)^{-1}K'.$$

The first term on the right hand side of (12.38) represents state variable uncertainty for a fixed value of θ, the second measurement error uncertainty, while the last two terms provide the shock uncertainty share. Notice that premultiplication of the covariance matrix by K' or postmultiplication by K yields a zero matrix.

The above expressions have been derived under the assumptions that $q_m < q_r < q$. However, they are also valid when $q_r = q_m$ or $q_r = q$. In the former case we find that $\Sigma_{N,T+g}^{(q_m)} = 0$ in equation (12.36), since $K'D(I_g \otimes \bar{M})$ is now a $q_m g \times q_m g$ invertible matrix. Moreover, this implies that the $\bar{\mu}_{N,T+g}^{(q_m)}$ term in equation (12.37) is equal to $\tilde{\mu}_{N,T+g}$ in (12.17) and, hence, that the conditional mean in (12.37) is equal to the conditional mean in (12.18). Furthermore, $\bar{D}$ in equation (12.38) is equal to $\tilde{D}$ in equation (12.19) so that the covariance matrices in these two equations are equal. In other words, with $q_r = q_m$ we find that the conditioning approach with a distribution for a subset of the shocks corresponds to direct control of the shocks. Similarly, with $q_r = q$ we have that $M = I_q$ and it therefore follows that the control of the distribution for a subset of the shocks method is identical to the control of the distribution of the shocks approach in Section 12.2.2. From this prespective we may therefore regard the distribution for a subset of the shocks conditioning approach as a generalization of the previously two discussed conditioning methods.

For forecast horizons i beyond the conditioning horizon g it can be shown via equation (12.20) that

$$E\big[y_{T+i}|\mathcal{Y}_T, Z_{T+g};\theta\big] = A'x_{T+i} + H'F^i\xi_{T|T+g}^{(z)} + H'F^{i-g}\bar{B}\big(I_g \otimes \bar{M}\big)\bar{\mu}_{N,T+g}^{(q_r)}, \quad i = g+1, \ldots, h. \qquad (12.39)$$

Provided that all the eigenvalues of F are inside the unit circle, it follows that the conditional mean forecast converges to the steady state values as $i \to \infty$. Furthermore, the forecast error covariance matrix for a fixed value of θ is

$$
\begin{aligned}
C\left[y_{T+i} | \mathcal{Y}_T, Z_{T+g}; \theta\right] = {}& H'F^{i-g}\bar{G}P^{(z)}_{T|T+g}\bar{G}'(F')^{i-g}H + H'F^{i-g}\bar{K}(I_g \otimes R)\bar{K}'(F')^{i-g}H \\
& + R + H'F^{i-g}\bar{B}(I_g \otimes \bar{M})\Sigma^{(q_r)}_{N,T+g}(I_g \otimes \bar{M}')\bar{B}'(F')^{i-g}H \\
& + H'F^{i-g}\bar{B}\bar{N}(I_g \otimes \bar{M}_\perp M'_\perp)\bar{N}'\bar{B}'(F')^{i-g}H \\
& + \sum_{j=0}^{i-g-1} H'F^j B_0 B'_0 (F')^j H,
\end{aligned}
\tag{12.40}
$$

for $i = g+1, \ldots, h$, and where

$$
\begin{aligned}
\bar{K} &= \bar{B}(I_g \otimes \bar{M}\bar{M}')D'K \left(K'D(I_g \otimes \bar{M}\bar{M}')D'K\right)^{-1} K', \\
\bar{G} &= F^g - \bar{K}G, \\
\bar{N} &= I_{qg} - \left(I_g \otimes \bar{M}\bar{M}'\right)D'K \left(K'D(I_g \otimes \bar{M}\bar{M}')D'K\right)^{-1} K'D.
\end{aligned}
$$

The first term on the right hand side of (12.40) represents state variable uncertainty for fixed θ, the second and third terms give the measurement error uncertainty share, while the last three terms provide a measure of shock uncertainty. Notice also that when all the eigenvalues of F are inside the unit circle, then the first, second, fourth, and fifth terms on the right hand side converge to zero as $i \to \infty$ and, hence, that the forecast error covariance matrix of the observed variables for fixed θ approaches the covariance matrix of these variables. As above, a full decomposition of the forecast uncertainty into state variable, measurement error, shock, and parameter uncertainty can be performed as in equation (12.24) using the results in equations (12.37)–(12.40) as input.

The recentered case can also be of interest under the mixed approach. Following the same setup as in Section 12.2.2, it can now be shown that the recentered variables are given by

$$
\begin{aligned}
\tilde{Y}_{T+g} = {}& X_{T+g} + G\tilde{\xi}_T + W_{T+g} + \bar{W}_{T+g} + D(I_g \otimes \bar{M})N^{(q_r)}_{T+g} \\
& + D(I_g \otimes \bar{M})\left[\bar{N}^{(q_r)}_{T+g} - \bar{\mu}^{(q_r)}_{N,T+g}\right] + D(I_g \otimes \bar{M}_\perp)\left[N^{(q-q_r)}_{T+g} + \bar{N}^{(q-q_r)}_{T+g}\right],
\end{aligned}
\tag{12.41}
$$

where $\tilde{\xi}_T$, as above, is normal with mean $\bar{\xi}_T$ and covariance matrix $P^{(z)}_{T|T+g}$. To ensure that $\tilde{Y}_{T+g}$ satisfies the conditioning restrictions, the mean of $N^{(q_r)}_{T+g}$ is now adjusted such that

$$
\bar{\mu}^{(q_r)}_{N,T+g} = \left(I_g \otimes \bar{M}'\right)D'K \left(K'D(I_g \otimes \bar{M}\bar{M}')D'K\right)^{-1} \tilde{k}^{(q_r)}_{T+g}
$$

where

$$
\begin{aligned}
\tilde{k}^{(q_r)}_{T+g} = {}& Z_{T+g} - U_T - K'\left(X_{T+g} + G\xi_T + W_{T+g} + \bar{W}_{T+g}\right) \\
& - K'D(I_g \otimes \bar{M})\left[\bar{N}^{(q_r)}_{T+g} - \bar{\mu}^{(q_r)}_{N,T+g}\right] - K'D(I_g \otimes \bar{M}_\perp)\left[N^{(q-q_r)}_{T+g} + \bar{N}^{(q-q_r)}_{T+g}\right],
\end{aligned}
$$

while the covariance matrix is equal to the expression in equation (12.36). The conditional mean of the recentered variables can now be expressed as

$$
\begin{aligned}
E\left[\tilde{Y}_{T+g} | \mathcal{Y}_T, Z_{T+g}; \theta\right] = {}& X_{T+g} + G\bar{\xi}_T + \bar{W}_{T+g} \\
& + D(I_g \otimes \bar{M})\left[\bar{N}^{(q_r)}_{T+g} - \bar{\mu}^{(q_r)}_{N,T+g}\right] + D(I_g \otimes \bar{M}_\perp)\bar{N}^{(q-q_r)}_{T+g} \\
& + D(I_g \otimes \bar{M}\bar{M}')D'K \left(K'D(I_g \otimes \bar{M}\bar{M}')D'K\right)^{-1} \\
& \times \left[Z_{T+g} - U_T - K'\left(X_{T+g} + G\bar{\xi}_T + \bar{W}_{T+g}\right)\right. \\
& - K'D(I_g \otimes \bar{M})\left[\bar{N}^{(q_r)}_{T+g} - \bar{\mu}^{(q_r)}_{N,T+g}\right] \\
& \left. - K'D(I_g \otimes \bar{M}_\perp)\bar{N}^{(q-q_r)}_{T+g}\right].
\end{aligned}
\tag{12.42}
$$

FIGURE 8: Some forecast options on the Miscellaneous tab in YADA.
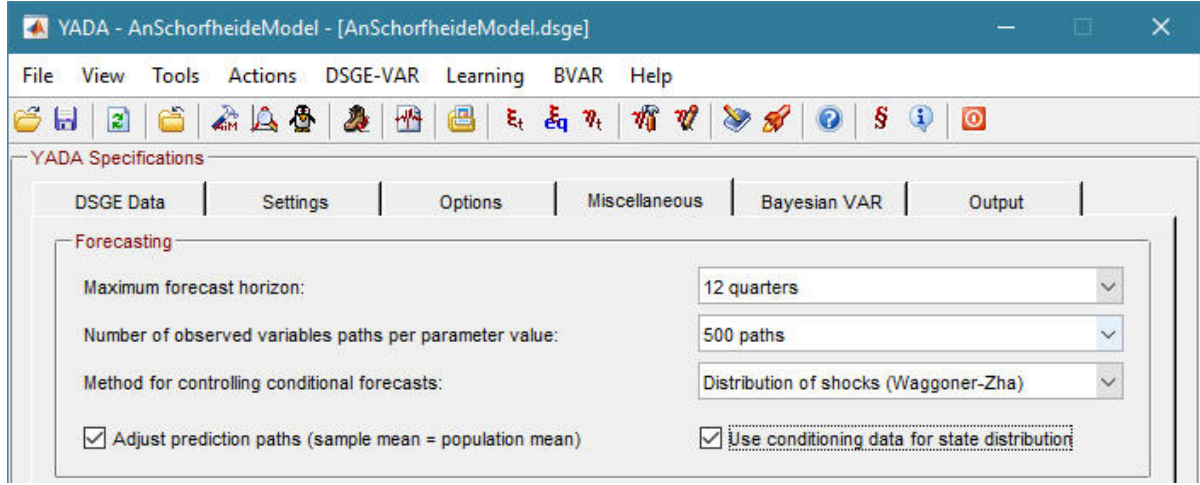
From this expression it can be seen that the mean of the recentered variables satisfies the conditioning restrictions. Furthermore, the difference between the mean prediction and the recentered path is determined by the difference between the mean value of the q_r shocks $N^{(q_r)}_{T+g}$ with and without recentering. This, is turn, is given by

$$E\left[\tilde{\mu}^{(q_r)}_{N,T+g}\right] - \bar{\mu}^{(q_r)}_{N,T+g} = (I_g \otimes \bar{M}')D'K\left(K'D(I_g \otimes \bar{M}\bar{M}')D'K\right)^{-1}K'$$
$$\times \left(G\left(\xi^{(z)}_{T|T+g} - \bar{\xi}_T\right) - \bar{W}_{T+g} - D\left[\bar{N}_{T+g} - (I_g \otimes \bar{M})\bar{\mu}^{(q_r)}_{N,T+g}\right]\right).$$

As in the previous two cases, for horizons $\tau = T + g + 1, \ldots, T + h$, this form of recentering the forecasts implies that we should add the smooth estimate of the measurement error in period τ to the projected value of y_τ and add the smooth estimate of the structural shock in period τ to the corresponding period shock value.

12.2.4. *Smooth Estimation of the State Variables using the Conditioning Assumptions*

The procedures discussed above require that we have access to the mean and the covariance matrix for the state vector at time T. As mentioned in Section 12.2.1 the default behavior in YADA is to set $\xi^{(z)}_{T|T+g} = \xi_{T|T}$ and $P^{(z)}_{T|T+g} = P_{T|T}$, i.e., to ignore the conditioning assumptions when estimating the mean and covariance matrix for the distribution of ξ_T using the data. Below I shall discuss how these moments can be estimated such that the conditioning assumptions are taken into account. At this stage it may be noted that the method for estimating the state variables using the conditioning assumptions can be simplified substantially when these assumptions at time t are linear combinations of the observed variables at time t, i.e., when $K_{2j} = 0$ for all j and $u_T = 0$ in equation (12.6). For such conditioning assumptions we may directly apply the Kalman filtering and smoothing routines for missing observations; see Harvey (1989, Chapter 3.4.7) and (Durbin and Koopman, 2012, Chapters 2.7 and 4.10).

The selection of method for representing the mean vector and covariance matrix of the state variables at time T is handled on the *Miscellaneous* tab in YADA, where check marking the option "Use conditioning data for state distribution" means that the approach described below is used rather than the quick and dirty default method; cf. Figure 8.

The relationship between the conditioning assumptions and the observed variables in (12.6) suggests an autoregressive relationship of maximum order g. Provided that $T \geq g$, this equation can be rewritten as follows by allowing the initial condition to be time varying

$$z_t = K_1'y_t + \sum_{j=1}^{g-1} K_{2j}'y_{t-j} + u_{t-g}, \quad t = T + 1, \ldots, T + g. \tag{12.43}$$

This can be expressed more compactly as

$$
z_t = \begin{bmatrix} K_1' & K_{21}' & \cdots & K_{2g-1}' \end{bmatrix} \begin{bmatrix} y_t \\ y_{t-1} \\ \vdots \\ y_{t-g+1} \end{bmatrix} + u_{t-g} \tag{12.44}
$$

$$
= K^{*\prime} Y_t + u_{t-g}.
$$

In order to derive a state-space representation for z_t we begin by rewriting the measurement equation for the observed variables as:

$$
y_t = A'x_t + \begin{bmatrix} H' & I_n \end{bmatrix} \begin{bmatrix} \xi_t \\ w_t \end{bmatrix} \tag{12.45}
$$

$$
= A'x_t + H^{*\prime} \xi_t^*.
$$

Hence, we are now treating the measurement errors as state variables. This implies that the state equation is given by

$$
\begin{bmatrix} \xi_t \\ w_t \end{bmatrix} = \begin{bmatrix} F & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \xi_{t-1} \\ w_{t-1} \end{bmatrix} + \begin{bmatrix} B_0 \eta_t \\ w_t \end{bmatrix},
$$

or

$$
\xi_t^* = F^* \xi_{t-1}^* + v_t^*. \tag{12.46}
$$

The state shocks v_t^* have dimension $r+n$ and are normally distributed with zero mean and covariance matrix

$$
Q^* = \begin{bmatrix} B_0 B_0' & 0 \\ 0 & R \end{bmatrix}.
$$

The next step is to stack both the measurement and state equation such that we can substitute for Y_t in (12.44). For the measurement equation we obtain

$$
\begin{bmatrix} y_t \\ y_{t-1} \\ \vdots \\ y_{t-g+1} \end{bmatrix} = \begin{bmatrix} A'x_t \\ A'x_{t-1} \\ \vdots \\ A'x_{t-g+1} \end{bmatrix} + \begin{bmatrix} H^{*\prime} & 0 & \cdots & 0 \\ 0 & H^{*\prime} & & 0 \\ \vdots & & \ddots & \\ 0 & 0 & & H^{*\prime} \end{bmatrix} \begin{bmatrix} \xi_t^* \\ \xi_{t-1}^* \\ \vdots \\ \xi_{t-g+1}^* \end{bmatrix},
$$

or more compactly

$$
Y_t = X_t + \begin{bmatrix} I_g \otimes H^{*\prime} \end{bmatrix} \Xi_t^*, \quad t = T+1, \ldots, T+g. \tag{12.47}
$$

Substituting for Y_t in (12.44) we get the following measurement equation for z_t

$$
z_t = \begin{bmatrix} K^{*\prime} & I_{qm} \end{bmatrix} \begin{bmatrix} X_t \\ u_{t-g} \end{bmatrix} + K^{*\prime} \begin{bmatrix} I_g \otimes H^{*\prime} \end{bmatrix} \Xi_t^* \tag{12.48}
$$

$$
= \tilde{A}'X_t^* + \tilde{H}'\Xi_t^*, \quad t = T+1, \ldots, T+g.
$$

For the state equation we need to make sure that the shocks to the stacked state equation are serially uncorrelated. The simplest way to achieve the stacking under this requirement is

$$
\begin{bmatrix} \xi_t^* \\ \xi_{t-1}^* \\ \xi_{t-2}^* \\ \vdots \\ \xi_{t-g+1}^* \end{bmatrix} = \begin{bmatrix} F^* & 0 & \cdots & 0 & 0 \\ I_{r+n} & 0 & \cdots & 0 & 0 \\ 0 & I_{r+n} & & 0 & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & & I_{r+n} & 0 \end{bmatrix} \begin{bmatrix} \xi_{t-1}^* \\ \xi_{t-2}^* \\ \xi_{t-3}^* \\ \vdots \\ \xi_{t-g}^* \end{bmatrix} + \begin{bmatrix} v_t^* \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix},
$$

or, with the compact expression,

$$\Xi_t^* = \tilde{F}\Xi_{t-1}^* + V_t^*, \quad t = T+1, \ldots, T+g. \tag{12.49}$$

It remains to define proper initial conditions for the state vector $\Xi_{T+1|T}^*$ and the conditional covariance matrix $\tilde{P}_{T+1|T}$ of the stacked state system. First of all, we note that $\Xi_{T+1|T}^* = \tilde{F}\Xi_{T|T}^*$, while $\tilde{P}_{T+1|T} = \tilde{F}\tilde{P}_{T|T}\tilde{F}' + \tilde{Q}$, where $\tilde{Q}$ has dimension $g(r+n)$ and is equal to

$$\tilde{Q} = \begin{bmatrix} Q^* & 0 \\ 0 & 0 \end{bmatrix}.$$

Second, to utilize the information $\mathcal{Y}_T$ optimally we let

$$\Xi_{T|T}^* = \begin{bmatrix} \xi_{T|T}^* \\ \xi_{T-1|T}^* \\ \vdots \\ \xi_{T-g+1|T}^* \end{bmatrix}, \quad \text{where } \xi_{T-i|T}^* = \begin{bmatrix} \xi_{T-i|T} \\ w_{T-i|T} \end{bmatrix}, \quad i = 0, 1, \ldots, g-1.$$

That is, we use the within-sample smooth estimates of the state vector and the measurement errors. For the conditional covariance matrix we likewise use smooth estimates for initialization. In this case, the block diagonal elements of $\tilde{P}_{T|T}$ are given by

$$P_{T-i|T}^* = \begin{bmatrix} P_{T-i|T} & 0 \\ 0 & R \end{bmatrix}, \quad i = 0, 1, \ldots, g-1.$$

The off-diagonal elements of $\tilde{P}_{T|T}$ for row and column block (i, j) are set equal to $(F^*)^{j-i}P_{T-j+i|T}^*$ if $g-1 \geq j > i \geq 0$, and to $P_{T-i+j|T}^{*\prime}(F^{*\prime})^{i-j}$ if $g-1 \geq i > j \geq 0$.

With the initial conditions as well as the measurement and state equation for the conditioning system z in (12.48) and (12.49) we can apply the Kalman filter to obtain $(\Xi_{t|t-1}^*, \tilde{P}_{t|t-1})$ for $t = T+1, \ldots, T+g$. To speed up the computation, the smoother need only be applied to generate $(\Xi_{t|T+g}^*, \tilde{P}_{t|T+g})$ for $t = T+g-1, T+g$. This means that $\xi_{T|T+g}^{(z)}$ is obtained from element $(g-1)(r+n)+1$ until $(g-1)(r+n)+r$ of $\Xi_{T+g-1|T+g}^*$, while $P_{T|T+g}^{(z)}$ is given by the rows and columns of $\tilde{P}_{T+g-1|T+g}$ based on the same range integers.

12.3. *Modesty Statistics for the State-Space Model*

Conditional forecasting experiments may be subject to the well known Lucas (1976) critique. Leeper and Zha (2003) introduced the concept of *modest* policy interventions along with a simple metric for evaluating how unusual a conditional forecast is relative to the unconditional forecast. Their idea has been further developed by Adolfson et al. (2005). I shall present three modesty statistics, two univariate and one multivariate, for the case when fixed shock values are used to ensure that the conditioning assumptions are satisfied. Similarly, two univariate and one multivariate modesty statistic are provided for the case when the Waggoner and Zha shock distribution or the control of the distribution of a subset of the shocks is applied.

The general idea behind the modesty statistics is to compare the conditional and the unconditional forecast. The forecasts are subject to uncertainty concerning which shocks will hit the economy during the prediction period. For the conditional forecasts some shocks have to take on certain values over the conditioning period to ensure that forecasts are consistent with the conditioning information. If these restricted shocks behave as if they are drawn from their distributions, then the conditioning information is regarded as modest. But if the behavior of the shocks over the conditioning period is different from the assumed, then the agents in the economy may be able to detect this change. In this case, the conditioning information need no longer be modest and might even be subject to the famous Lucas (1976) critique.

12.3.1. *Modesty Analysis: Direct Control of the Shocks*

Within the context of the state-space representation of the DSGE model, the univariate statistic suggested by Leeper and Zha (2003) is based on setting $\eta_{T+i}^{(q-q_m)} = 0$ and $w_{T+i} = 0$ for $i = 1, \ldots, g$ in equations (12.10)–(12.12). The alternative univariate statistic suggested by Adolfson et al. (2005) does not force these shocks to be zero over the conditioning horizon.

For both approaches the difference between the conditional and the unconditional forecasts at $T + g$ for a given θ is:

$$
\begin{aligned}
\Phi_{T,g}(\bar{\eta}) &= y_{T+g}(\bar{\eta}; \theta) - E\left[y_{T+g} | \mathcal{Y}_T; \theta\right] \\
&= H'F^g\left(\xi_T - \xi_{T|T}\right) + H'\sum_{j=0}^{g-1} F^j B_0\left(\bar{M}\eta_{T+g-j}^{(q_m)} + \bar{M}_\perp \eta_{T+g-j}^{(q-q_m)}\right) + w_{T+g},
\end{aligned}
\tag{12.50}
$$

where $\bar{\eta} = \{\eta_t^{(q_m)}, \eta_t^{(q-q_m)}\}_{t=T+1}^{T+g}$. Under the Leeper and Zha approach the measurement errors and the other shocks $\{\eta_t^{(q-q_m)}\}_{t=T+1}^{T+g}$ are set to zero, while under the Adolfson et al. approach these shocks and errors are drawn from their distributions.

For the latter approach we have that the forecast error variance at $T + g$ is

$$
\Omega_{T+g} = H'P_{T+g|T}H + R,
\tag{12.51}
$$

where as already shown in equation (5.38)

$$
P_{T+i|T} = FP_{T+i-1|T}F' + B_0 B_0', \quad i = 1, \ldots, g.
$$

Under the hypothesis that ξ_T is can be observed at T, the matrix $P_{T|T} = 0$. This assumption is used by Adolfson et al. (2005) and is also imposed in YADA.

The assumption that the state vector in T can be observed at T in the modesty analysis is imposed to make sure that it is consistent with the assumptions underlying the DSGE model. That is, the agents know the structure of the model, all parameters, and all past and present shocks. Hence, there cannot be any state uncertainty when evaluating the current state. This also has an implication for equation (12.50) where we set $\xi_T = \xi_{T|T}$.

The multivariate modesty statistic can now be defined as:

$$
\mathcal{M}_{T,g}(\bar{\eta}) = \Phi_{T,g}(\bar{\eta})'\Omega_{T+g}^{-1}\Phi_{T,g}(\bar{\eta}).
\tag{12.52}
$$

Under the hypothesis that the conditioning shocks are modest, i.e., $\{\eta_t^{(q_m)}\}_{t=T+1}^{T+g}$ can be viewed as being drawn from a multivariate standard normal distribution, this statistic is $\chi^2(n)$. Instead of using the chi-square as a reference distribution for the multivariate modesty statistic, one may calculate the statistic in (12.52) using the shocks $\eta = \{\eta_t\}_{t=T+1}^{T+g}$ in (12.52) that are drawn from the $N_q(0, I_q)$ distribution and thereafter compute the tail probability $\Pr[\mathcal{M}_{T,g}(\eta) \geq \mathcal{M}_{T,g}(\bar{\eta})]$ to determine if the conditioning information is modest.

One univariate statistic suggested by Adolfson et al. is the following

$$
\mathcal{M}_{T,g}^{(i)}(\bar{\eta}) = \frac{\Phi_{T,g}^{(i)}(\bar{\eta})}{\sqrt{\Omega_{T+g}^{(i,i)}}}, \quad i = 1, \ldots, n,
\tag{12.53}
$$

where $\Phi_{T,g}(\bar{\eta})$ is calculated with the other shocks and the measurement errors drawn from a normal distribution. This statistic has a standard normal distribution under the assumption of modest conditioning shocks.

For the alternative Leeper-Zha related statistic we let $\Phi_{T,g}(\bar{\eta})$ be computed for zero measurement errors and other shocks, while

$$
\Omega_{T+g} = H'P_{T+g|T}H,
$$
$$
P_{T+i|T} = FP_{T+i-1|T}F' + B_0\bar{M}M'B_0'.
$$

The alternative $\Phi_{T,g}^{(i)}(\bar{\eta})$ and $\Omega_{T+g}^{(i,i)}$ values may now be used in equation (12.53).

12.3.2. *Modesty Analysis: Control of the Distribution of the Shocks*

The difference between the conditional and unconditional mean forecasts at $T + g$ for a given θ is:

$$\Phi_{T,g}(N_{T+g}) = y_{T+g}(N_{T+g}; \theta) - E\left[y_{T+g}|\mathcal{Y}_T; \theta\right]$$

$$= H'F^g\left(\xi_T - \xi_{T|T}\right) + H'\sum_{j=0}^{g-1} F^j B_0 \eta_{T+g-j} + w_{T+g}, \tag{12.54}$$

where N_{T+g} is drawn from the $N(\mu_{N,T+g}, \Sigma_{N,T+g})$ distribution. The forecast error variance at $T + g$ is, as above, given by Ω_{T+g} in equation (12.51). This means that the multivariate modesty statistic for the Waggoner and Zha drawn shocks is

$$\mathcal{M}_{T,g}(N_{T+g}) = \Phi_{T,g}(N_{T+g})'\Omega_{T+g}^{-1}\Phi_{T,g}(N_{T+g}). \tag{12.55}$$

Like in the case with fixed shock values, this statistic can be compared to a reference statistic for determining the tail probability $\Pr[\mathcal{M}_{T,g}(\eta) \geq \mathcal{M}_{T,g}(N_{T+g})]$. The reference statistic $\mathcal{M}_{T,g}(\eta)$ is calculated exactly as in the previous section, with η_{T+i} drawn from $N(0, I_q)$. Similarly, univariate modesty statistics are obtained like in (12.53), but with $\Phi_{T,g}^{(i)}(\bar{\eta})$ replaced by $\Phi_{T,g}^{(i)}(N_{T+g})$.

To compute modesty statistic in the Leeper-Zha fashion, we use $\bar{\mu}_{T+i}$ from $i = 1, \ldots, g$ for the shock values. Specifically, the conditional forecast is calculated with these shock values, i.e., based on zero measurement errors and the state vector at T given by $\xi_{T|T}$. The covariance matrix that is used is now:

$$\Omega_{T+g} = H'P_{T+g|T}H,$$
$$P_{T+i|T} = FP_{T+i-1|T}F' + B_0\bar{\alpha}\alpha'B_0',$$

where $\alpha = B_0'HK_1$ and $\bar{\alpha} = \alpha(\alpha'\alpha)^{-1}$. The $q \times q$ matrix $\bar{\alpha}\alpha'$ is idempotent and has rank q_m and serves a similar purpose as the matrix $\bar{M}M'$ in the fixed shock values case. That is, it can be regarded as the covariance matrix of the non-zero structural shocks under the Leeper-Zha approach and the null hypothesis that the shocks are modest.[104]

12.3.3. *Modesty Analysis: Control of the Distribution of a Subset of the Shocks*

For the conditioning approach discussed in Section 12.2.3, the modesty analysis is nearly identical to the analysis discussed for the Waggoner-Zha distribution shocks above. The only difference concerns the computation of the covariances used for the univariate modesty statistics under the Leeper-Zha case. Specifically, since the $q - q_r$ free shocks are set equal to zero, the state covariance matrix $P_{T+i|T}$ used for the tests is now given by

$$P_{T+i|T} = FP_{T+i-1|T}F' + B_0\bar{M}M'B_0'HK_1 \left(K_1'H'B_0\bar{M}M'B_0'HK_1\right)^{-1} K_1'H'B_0\bar{M}M'B_0'.$$

That is, we let $\alpha = \bar{M}M'B_0'HK_1$. In all other respects, the formulas from Section 12.3.2 remain unaltered.

It is worth pointing out that for $q_r = q$ the state covariance matrix for the Leeper-Zha case is identical to the one used under the Waggoner and Zha approach since $M = I_q$. Similarly, if $q_r = q_m$ it is equal to the state covariance matrix under the direct control of the shocks method, i.e., the second term on the right hand side simplifies to $B_0\bar{M}M'B_0'$.

12.4. *Conditional Forecasting with State Variable Assumptions*

As an extension to forecasting conditional on assumptions for the observed variables only we may also consider assumptions for the state variables. Such restrictions can be expressed as:

$$\zeta_{T+i} = K_3'\xi_{T+i}, \quad i = 1, \ldots, g. \tag{12.56}$$

---

[104] The α matrix is obtained from the block diagonal of the matrix $D'K$ and α may therefore be interpreted as representing the constrained structural shocks. See also the expressions for $\mu_{N,T+g}$ under the two approaches.

The dimension of the assumptions in ζ_{T+i} is q_z where K_3 is a $q_z \times r$ matrix with $\text{rank}(K_3) = q_z$. The state variable assumptions not only allows for having a fixed path for a serially correlated state variable, but also for the state shocks in η.

Combining the state variable assumptions in (12.56) with the assumptions for the observed variables in equation (12.6) we have

$$\begin{bmatrix} z_{T+i} \\ \zeta_{T+i} \end{bmatrix} = \begin{bmatrix} K_1' & 0 \\ 0 & K_3' \end{bmatrix} \begin{bmatrix} y_{T+i} \\ \xi_{T+i} \end{bmatrix} + \sum_{j=1}^{i-1} \begin{bmatrix} K_{2j}' & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} y_{T+i-j} \\ \xi_{T+i-j} \end{bmatrix} + \begin{bmatrix} u_T \\ 0 \end{bmatrix}, \quad i = 1, \ldots, g,$$

or more compactly

$$z_{T+i}^* = K_1^{*\prime} y_{T+i}^* + \sum_{j=1}^{i-1} K_{2j}^* y_{T+i-j}^* + u_T^*. \tag{12.57}$$

The vector z_{T+i}^* is $q_m + q_z$ dimensional and a necessary condition for forecasting with these joint conditioning assumptions is that $q_m + q_z \leq \min\{q, n + r\}$, $q_m \leq n$, and $q_z \leq r$. These assumptions constitute the most general hard conditioning assumptions that we may consider for the state-space model.

To use the tools we have discussed in Sections 12.2 and 12.3 we need to express the state-space representation for y_t^* instead of y_t. This can be achieved through a simple stacking of the measurement equation as:

$$\begin{bmatrix} y_t \\ \xi_t \end{bmatrix} = \begin{bmatrix} A' \\ 0 \end{bmatrix} x_t + \begin{bmatrix} H' \\ I_r \end{bmatrix} \xi_t + \begin{bmatrix} w_t \\ 0 \end{bmatrix},$$

or

$$y_t^* = A^{*\prime} x_t + H^{*\prime} \xi_t + w_t^*.$$

There is no need to rewrite the state equation. With these redefinitions of the matrices with parameters on exogenous and state variables and of the measurement errors, we may directly apply the methods for forecasting based on conditioning assumptions for observed variables and the modesty analysis to conditioning assumptions that also involve state variables.

It may be noted that equation (12.57) allows for $q_m = 0$ so that only conditioning assumptions for the state variables are considered. This makes it possible to simplify some of the expressions since $K_{2j}^* = 0$ and $u_T^* = 0$.

12.5. *Prediction Events and Risk Analysis*

Once we have draws from a predictive distribution it may be interesting to compute the probability that a certain event occurs. Moreover, given that the event occurs it may also be relevant to consider how large the event is on average and, more generally, which properties the conditional distribution has. Once we start to combine information about the size of the event as well as its probability we enter the grounds of risk analysis; see, e.g., Machina and Rothschild (1987) as well as Kilian and Manganelli (2007, 2008) and references therein.

The papers by Kilian and Manganelli focus on deflation and excess inflation risks, but we may of course apply their approach to any variable. The exercise begins with setting up an upper and a lower bound of an event. The probability of the event occurring can therefore be computed from, e.g., the paths of the predictive distribution that were obtained for the state-space model in Section 12.1, by counting how many times it is satisfied by a path and comparing this number to the total number of times it could have happened.

Let $y_U \geq y_L$ be the upper an lower bounds for the observed variables y_t. Following Kilian and Manganelli (2007) we define the *downside risk* as follows

$$DR_\alpha(y_L) = -\int_{-\infty}^{y_L} (y_L - y)^\alpha \, dF(y), \quad \alpha \geq 0. \tag{12.58}$$

Notice that if $\alpha = 0$, then $DR(y_L) = -\Pr[y \leq y_L]$. For larger values of α the downside risk measure weighs the probability of the event with the expected value of $(y_L - y)^\alpha$ given that the

event occurs. That is,

$$DR_\alpha(y_L) = -E\left[(y_L - y)^\alpha | y < y_L\right] \Pr\left[y < y_L\right].$$

The *upside risk* can similarly be defined as

$$UR_\beta(y_U) = \int_{y_U}^{\infty} \left(y - y_U\right)^\beta dF(y), \quad \beta \geq 0. \tag{12.59}$$

It therefore follows that the upside risk can be computed as

$$UR_\beta(y_U) = E\left[(y - y_U)^\beta | y > y_U\right] \Pr\left[y > y_U\right].$$

Like Kilian and Manganelli I have here adopted the convention of defining downside risk as a negative number and upside risk as a positive number.[105]

These risk measures are related to the loss function

$$L(y) = \begin{cases} a(y_L - y)^\alpha & \text{if } y < y_L, \\ 0 & \text{if } y_L \leq y \leq y_U, \\ (1 - a)(y - y_U)^\beta & \text{if } y > y_U. \end{cases} \tag{12.60}$$

This mean that the expected loss is given by

$$E\left[L(y)\right] = -aDR_\alpha(y_L) + (1 - a)UR_\beta(y_U), \tag{12.61}$$

where $0 \leq a \leq 1$ thus gives the weight on downside risk relative to upside risk in the loss function.

As pointed out by Kilian and Manganelli it is common in discussions of risk to stress the need to balance the upside and downside risks. Risk balancing in this sense may be considered as taking a weighted average of the upside and downside risks. As shown by Kilian and Manganelli (2007) such a *balance of risk* measure may be derived under optimality arguments and is here given by

$$BR_{\alpha-1,\beta-1}(y_L, y_U) = a\alpha DR_{\alpha-1}(y_L) + (1 - a)\beta UR_{\beta-1}(y_U). \tag{12.62}$$

For the case with a quadratic loss function with equal weights given to upside and downside risks, the balance of risk measure in (12.62) is

$$BR_{1,1}(y_L, y_U) = E\left[(y - y_L)|y < y_L\right] \Pr\left[y < y_L\right] + E\left[(y - y_U)|y > y_U\right] \Pr\left[y > y_U\right].$$

That is, the balance of risks is a weighted average of the expected value of y given that it is below the lower bound and the expected value of y given that it is above the upper bound with weights given by the probabilities that the events occur. Such a measure is, for instance, used by Smets and Wouters (2004) in their study on the forecasting properties of a DSGE model. For more elaborate analyses of such risk balance measures in connection with the zero lower bound on nominal interest rates, see Coenen and Warne (2014).

12.6. *The Predictive Likelihood and Log Predictive Score*

The calculation of the height of the joint or the marginal predictive density is often needed by methods for comparing or evaluating density forecasts; see, e.g., Geweke and Amisano (2010). As Gneiting, Balabdaoui, and Raftery (2007) point out, the assessment of a predictive distribution on the basis of its density and the observed data only—the predictive likelihood—is consistent with the *prequential* approach of Dawid (1984), according to which forecasts are both probabilistic and sequential in nature, taking the form of probability distributions over a sequence of future values; see also Geweke (2010) and Geweke and Amisano (2011, 2012).

The use of the predictive likelihood as a valid Bayesian approach to model selection has long been recognized. Box (1980), for example, has emphasized the complementary roles in the model building process of the posterior and predictive distributions, where the former provides a basis for robust estimation, while the latter is used for diagnostic checking and modifications

---

[105] Risk measures of this type were first proposed in the portfolio allocation literature by Fishburn (1977); see also, e.g., Holthausen (1981).

of the model.[106] Moreover, for models with improper priors the predictive likelihood can still be used for model selection provided that the sample being conditioned on is large enough to train the prior to a proper one; see, e.g., Gelfand and Dey (1994), Eklund and Karlsson (2007), and Strachan and van Dijk (2011).

A forecast comparison exercise is naturally cast as a decision problem within a Bayesian setting and therefore needs to be based on a particular preference ordering. Scoring rules can be used to compare the quality of probabilistic forecasts by giving a numerical value using the predictive distribution and an event or value that materializes. A scoring rule is said to be *proper* if a forecaster who maximizes the expected score provides its true subjective distribution; see Winkler and Murphy (1968). If the maximum is unique then the rule is said to be strictly proper. Proper scoring rules are important since they encourage the forecaster to be honest.

A widely used scoring rule that was suggested by, e.g., Good (1952) is the log predictive score. Based on the predictive density function of $y_{t+1}, \ldots, y_{T+h}$, it can be expressed as

$$S_J(h, m) = \sum_{t=T}^{T+N_h-1} \ln p(y_{t+1}, \ldots, y_{t+h} | \mathcal{Y}_t, m), \quad h = 1, \ldots, H, \quad (12.63)$$

where N_h is the number of time periods the h-step-ahead predictive density is evaluated, $\mathcal{Y}_t$ is the observed data of y_t until period t, and m is an index for the model. If the scoring rule depends on the predictive density only through the realization of y over the prediction sample, then the scoring rule is said to be *local*. Under the assumption that only local scoring rules are considered, Bernardo (1979) showed that every proper scoring rule is equivalent to a positive constant times the log predictive score plus a real valued function that only depends on the realized data; see Bernardo and Smith (2000) for general discussions on related issues and Gneiting and Raftery (2007) for a recent survey on scoring rules.

When evaluating the log score with the realized value of y over the prediction sample, the difference between the log predictive score of model m and model k is equal to the average log predictive Bayes factor of these two models, where a positive value indicates that, on average, model m is better at predicting the variables over the given sample than model k. It is furthermore straightforward to show that the log predictive likelihood of model m is equal to the difference between the log marginal likelihood value when the historical data, $\mathcal{Y}_t$, and the realisations $y_{t+1}, \ldots, y_{t+h}$ are used and the log marginal likelihood value obtained when only the historical data are employed; see, e.g., Geweke (2005, Chapter 2.6.2). In fact, based on the observations $\mathcal{Y}_{T+N_1}$ and with $N_h = N_{h-1} - 1$ we can rewrite the log predictive score in (12.63) as

$$S_J(h, m) = \frac{1}{N_h} \sum_{i=0}^{h-1} \left[\ln p(\mathcal{Y}_{T+N_1-i}, m) - \ln p(\mathcal{Y}_{T+i}, m) \right], \quad h = 1, \ldots, H. \quad (12.64)$$

This means that the log predictive score of model m for one-step-ahead forecasts is proportional to the difference between the log-marginal likelihood for the full sample $\mathcal{Y}_{T+N_1}$ and the historical sample $\mathcal{Y}_T$. Moreover, the calculation of the score for h-step-ahead forecasts based on the joint predictive likelihood requires exactly $2h$ marginal likelihood values, where the first h are based on the samples $\mathcal{Y}_{T+N_1-i}$ and the last h on $\mathcal{Y}_{T+i}$ for $i = 0, \ldots, h-1$.

A scale for determining how much better (or worse) model m is than model k has been suggested by, e.g., Kass and Raftery (1995) for Bayes factors. In the case of one-step-ahead predictions, this scale may also be applied to the log predictive score where values above 5 may be seen as very strong evidence in favor of model m. Provided that one of these models is assumed to be "true", the translation into posterior probabilities is straightforward, where the case of equal prior probabilities for models m and k and a value of the log score of 5 or more corresponds to a posterior probability of above 99 percent for model m.

It can also be seen that the log predictive likelihood in (12.63) can be rewritten as a sum of one-step-ahead log predictive likelihoods. Hence, in essence the log score $S_J(h, m)$ covers

---

[106] The predictive distribution in Box (1980) is described by the density of the observed variables given the model (assumptions). The value of this density at the observed data is equal to the marginal likelihood.

one-step-ahead forecasts only and is therefore not well suited for a comparison of h-step-ahead forecasts when $h > 1$. When comparing the density forecasts of the NAWM and alternative forecast models, Christoffel et al. (2011) therefore focus on the marginal predictive likelihood of the h-step-ahead forecasts rather than the joint predictive likelihood in (12.63). The log predictive score can now be expressed as

$$S_M(h, m) = \sum_{t=T}^{T+N_h-1} \ln p(y_{t+h}|\mathcal{Y}_t, m), \quad h = 1, \ldots, H. \tag{12.65}$$

The relationship between the marginal likelihood and the log predictive score in (12.65) holds when $h = 1$. For other forecast horizons it is claimed by both Christoffel et al. (2011, p. 114) (p. 114) and Adolfson et al. (2007d, p. 324) that this connection breaks down and, hence, that the marginal likelihood cannot detect if some models perform well on certain forecast horizons while other models do better on other horizons. Furthermore, Adolfson et al. (2007d, p. 325) remark that computing $S_M(h, m)$ for $h > 1$ is not an easy task since $p(y_{t+h}|\mathcal{Y}_t, m)$ does not have a closed form solution and that kernel density estimation from the predictive draws is not practical unless the dimension of y_{t+h} is small. They therefore suggest using a normal approximation of the predictive likelihood based on the mean and the covariance of the marginal predictive distribution.

However, going back one step one realizes that Christoffel et al. (2011) and Adolfson et al. are incorrect since

$$p(y_{t+h}|\mathcal{Y}_t, m) = \frac{p(y_{t+h}, \mathcal{Y}_t, m)}{p(\mathcal{Y}_t, m)}, \quad h = 1, \ldots, H. \tag{12.66}$$

The denominator is the marginal likelihood of model m when using the data $\mathcal{Y}_t$ and the numerator is likewise the marginal likelihood for this model when using the data $(y_{t+h}, \mathcal{Y}_t)$. Hence, the connection between the predictive likelihood and the marginal likelihood remains also for $h > 1$. The problem for calculating the log predictive score in (12.65) for $h > 1$ therefore concerns the question: it is possible to compute the marginal likelihood for the sample $(y_{t+h}, \mathcal{Y}_t)$?

A solution to this problem is suggested by Warne, Coenen, and Christoffel (2013); see also Warne, Coenen, and Christoffel (2017). Suppose we replace the realizations of y_{t+i}, $i = 1, \ldots, h - 1$, in $\mathcal{Y}_{t+h}$ with missing observations and apply a valid method for dealing with incomplete-data when evaluating the likelihood function for fixed parameters of model m. This effectively means that we treat missing observations as a method for integrating out variables at certain points in time from the likelihood, and that the marginal likelihood of the model for $(y_{t+h}, \mathcal{Y}_t)$ can thereafter be computed via standard tools.[107] Such an approach can also be used to estimate the marginal likelihood for the data $(y_{t+h}^*, \mathcal{Y}_t)$, where y_{t+h}^* is a subset of the elements of y_{t+h}, as well as for, e.g., the data $(y_{t+1}^*, \ldots, y_{t+h}^*, \mathcal{Y}_t)$. In fact, we may replace data points with missing observations anywhere in the predictive sample $y_{t+1}, \ldots y_{t+h}$ when calculating the likelihood function.

In the case of linear state-space models with Gaussian shocks and measurement errors, the likelihood function can be calculated using a Kalman filter which allows for missing observations; see, e.g., Durbin and Koopman (2012, Chapter 4.10) or Harvey (1989, Chapter 3.4.7). Once we turn to non-linear, non-normal state-space models a missing observations consistent filter, such as the particle filter (sequential Monte Carlo), may instead be applied when computing the likelihood; see Giordani, Pitt, and Kohn (2011) for a survey on filtering in state-space models, or Durbin and Koopman (2012, Chapter 12) for an introduction to particle filtering.

---

[107] This idea is related to but also different from data augmentation and other such EM algorithm extensions. For these algorithms, the model is used to replace missing observations with model-based draws of the latent variables and then use complete-data methods to address the incomplete-data problem; see, e.g., Tanner and Wong (1987) and Rubin (1991).

When the joint predictive density for fixed parameters is Gaussian, marginalization can also be conducted directly via the predictive mean and the covariance matrix for given parameters (provided these moments can be determined analytically) by utilizing well-known properties of the normal distribution.[108] In the case of the linear Gaussian models this approach to marginalization is equivalent to the Kalman filter approach, where the Kalman filter approach to marginalization provides a unifying framework and is as parsimonious as possible when dealing with potentially large matrices.

A generic approach to consistent estimation of the predictive likelihood is importance sampling (IS); see, e.g., Geweke (2005). With $\theta^{(i)}$ being draws from the importance density $g(\theta)$, a general expression of the IS estimator is

$$\hat{p}_{IS}(y_{t+h}^* | \mathcal{Y}_t) = \frac{1}{N} \sum_{i=1}^{N} \frac{L(y_{t+h}^* | \mathcal{Y}_t, \theta^{(i)}) p(\theta^{(i)} | \mathcal{Y}_t)}{g(\theta^{(i)})}. \tag{12.67}$$

Letting $g(\theta) = p(\theta | \mathcal{Y}_t)$ such that $\theta^{(i)} = \theta^{(s)}$ with $N = S$, the estimator of the predictive likelihood in (12.67) is simply the average over the S posterior draws $\theta^{(s)}$ of the conditional likelihood, i.e. standard Monte Carlo (MC) integration based on the conditional likelihood. Under certain conditions, the right hand side of (12.67) converges almost surely to the expected value of $p(y_{t+h}^* | \mathcal{Y}_t, \theta)$ with respect to $p(\theta | \mathcal{Y}_t)$, i.e., to the predictive likelihood $p(y_{t+h}^* | \mathcal{Y}_t)$.[109] Hence, equipped with the posterior draws $\theta^{(s)}$ and the conditional likelihood, $L(y_{t+h}^* | \mathcal{Y}_t, \theta)$, the predictive likelihood can be consistently estimated directly, without having to compute it from two marginal likelihoods, and without having to sample from the distribution of the parameters conditional $(y_{t+h}^*, \mathcal{Y}_t)$ for $h = 1, \ldots, H$.

A further important property of the IS estimator is that it is unbiased (see Chan and Eisenstat, 2015, Proposition 1). The IS (MC) estimator works well in practise when the draws from the importance density (posterior density) covers well enough the parameter region where the conditional likelihood is large. When computing the marginal predictive likelihood with $g(\theta) = p(\theta | \mathcal{Y}_t)$ this is typically the case, but is questionable when dealing with the joint predictive likelihood as h becomes large.[110] For such situations it may be useful to consider cross-entropy methods for selecting the importance density optimally, as in Chan and Eisenstat (2015). For an application to euro area data also covering the period of the Great Recession, see Warne et al. (2017).

We may also compute the numerical standard error of the IS (MC) estimator in (12.67). Assuming that $g(\theta) = p(\theta | \mathcal{Y}_t)$, the numerical standard error of $\hat{p}_{IS}(y_{t+h}^* | \mathcal{Y}_t)$ can be calculated with the Newey and West (1987) estimator, as in equation (8.2) but with $\phi^{(n)}$ replaced with $L(y_{t+h}^* | \mathcal{Y}_t, \theta^{(i)})$ and $\bar{\phi}$ with $\hat{p}_{IS}(y_{t+h}^* | \mathcal{Y}_t)$. The corresponding numerical standard error of the log predictive likelihood can now be calculated as the square root of of that value divided by $\hat{p}_{IS}(y_{t+h}^* | \mathcal{Y}_t)$. That is, we multiply the standard error of the predictive likelihood with the derivative of the log predictive likelihood with respect to the predictive likelihood as prescribed by the delta method.

---

[108] In fact, any distribution where the marginal density can be determined analytically from the joint for given parameters, such as the Student t, can be marginalized in this way.

[109] Importance sampling is based on iid draws from the importance density; see, for instance, Geweke (2005, Chapter 4.2.2) for further details. In the case of DSGE and DSGE-VAR models, the posterior draws are typically obtained via Markov chain Monte Carlo, such as the random walk Metropolis sampler, and are therefore not independent. However, under certain conditions (Tierney, 1994) the estimator in (12.67) is consistent also when the draws from $g(\theta) = p(\theta | \mathcal{Y}_t)$ are not independent. In strict terms, the estimator in (12.67) is not an IS estimator when the iid assumption is violated, but we shall nevertheless use this term also when the draws from the posterior are dependent.

[110] For sufficiently large h the situation resembles the case when the marginal likelihood is computed by averaging the likelihood over the prior draws. Such an estimator typically gives a poor estimate of the marginal likelihood.

For models with a normal likelihood, the log of the conditional likelihood for y^*_{t+h} given the history and the parameters is given by

$$
\ln L\big(y^*_{t+h}|\mathcal{Y}_t;\theta\big) = -\frac{n^*}{2}\ln(2\pi) - \frac{1}{2}\ln\big|\Sigma_{y^*,t+h|t}\big|
$$
$$
-\frac{1}{2}\left(y^*_{t+h} - y^*_{t+h|t}\right)' \Sigma^{-1}_{y^*,t+h|t}\left(y^*_{t+h} - y^*_{t+h|t}\right),
$$
(12.68)

where n^* is the dimension of y^*_{t+h}, $y^*_t = K'y_t$ and $\Sigma_{y^*,t+h|t} = K'\Sigma_{y,t+h|t}K$, with K being an $n \times n^*$ known selection matrix. We here assume that the K matrix is constant across time, but we can also add a time subscript to it and to n^*. For DSGE models the Kalman filter provides us with

$$
y_{t+h|t} = \mu + H'F^h\xi_{t|t},
$$
$$
\Sigma_{y,t+h|t} = H'P_{t+h|t}H + R,
$$
$$
P_{t+h|t} = FP_{t+h-1|t}F' + BB', \quad h = 1,\ldots,H,
$$

where $\xi_{t|t}$ is the filter estimate of the state variables, and $P_{t|t}$ the corresponding filter estimate of the state variable covariance matrix based on the data $\mathcal{Y}_t$. The matrices (μ, H, R, F, B) are all evaluated for a given value of θ.

The Kalman filter for missing observations can also be used when we are interested in the joint predictive likelihood for subsets of variables across a sequence of future dates. To this end, the conditional likelihood is now

$$
\ln L\big(y^*_{t+1},\ldots,y^*_{t+h}|\mathcal{Y}_t;\theta\big) = \sum_{i=1}^{h}\ln L\big(y^*_{t+i}|\mathcal{Y}^*_{t+i-1},\mathcal{Y}_t;\theta\big),
$$
(12.69)

where $\mathcal{Y}^*_{t+i-1} = \{y^*_{t+j}\}_{j=1}^{i-1}$ and

$$
\ln L\big(y^*_{t+i}|\mathcal{Y}^*_{t+i-1},\mathcal{Y}_t;\theta\big) = -\frac{n^*}{2}\ln(2\pi) - \frac{1}{2}\ln\big|\Sigma_{y^*,t+i|t+i-1}\big|
$$
$$
-\frac{1}{2}\left(y^*_{t+i} - y^*_{t+i|t+i-1}\right)' \Sigma^{-1}_{y^*,t+i|t+i-1}\left(y^*_{t+i} - y^*_{t+i|t+i-1}\right).
$$
(12.70)

We assume for notational simplicity but without loss of generality that n^* and K are constant across the h periods, i.e., that the same variables are conditioned on and that the realizations are available. We now have have that $\Sigma_{y^*,t+i|t+i-1} = K'\Sigma_{y,t+i|t+i-1}K$, $H^* = HK$, $A^* = AK$, $R^* = K'RK$, while

$$
y^*_{t+i|t+i-1} = A^{*'}x_{t+i} + H^{*'}\xi_{t+i|t+i-1},
$$
$$
\Sigma^*_{y,t+i|t+i-1} = H^{*'}P_{t+i|t+i-1}H^* + R^*.
$$

The 1-step-ahead state variable forecasts are given by

$$
\xi_{t+i|t+i-1} = F\xi_{t+i-1|t+i-2} + G_{t+i-1}\left(y^*_{t+i-1} - y^*_{t+i-1|t+i-2}\right),
$$

where the Kalman gain matrix is

$$
G_{t+i-1} = FP_{t+i-1|t+i-2}H^*\Sigma^{*-1}_{y^*,t+i-1|t+i-2}.
$$

The 1-step-ahead state variable forecast error covariance matrix is

$$
P_{t+i|t+i-1} = \left(F - G_{t+i-1}H^{*'}\right)P_{t+i-1|t+i-2}\left(F - G_{t+i-1}H^{*'}\right)' + G_{t+i-1}R^*G'_{t+i-1} + Q.
$$

While the outlined solution to the problem of how to calculate the log predictive score based on the marginal predictive likelihood for a subset of the observed variables is straightforward, the calculation of marginal likelihoods for large systems is computationally expensive when based on posterior draws. An approximate but computationally inexpensive estimator of the marginal likelihood is the Laplace approximation, discussed above in Section 10.1; see also Tierney and Kadane (1986), Gelfand and Dey (1994), and Raftery (1996). It requires that the

mode of the log posterior, given by the sum of the log likelihood and the log prior, can be computed and that its Hessian is available.

Letting $\tilde{\theta}$ be the posterior mode of θ and $\tilde{\Sigma}$ be minus the Hessian, the Laplace approximation based on the sample $\mathcal{Y}_t$ is given by

$$\ln \hat{p}_L(\mathcal{Y}_t) = \ln L(\mathcal{Y}_t; \tilde{\theta}) + \ln p(\tilde{\theta}) + \frac{d \ln(2\pi) + \ln|\tilde{\Sigma}^{-1}|}{2}, \qquad (12.71)$$

where d is the dimension of θ. The third term on the right hand side approximates $-\ln p(\tilde{\theta}|\mathcal{Y}_t)$ with $O(t^{-1})$ accuracy and, hence, the expression in (12.71) is a reflection of Bayes theorem through what Chib (1995) calls the basic marginal likelihood identity.

Similarly, let θ^* be the posterior mode when the sample $(y_{t+h}^*, \mathcal{Y}_t)$ is used, with minus the Hessian being denoted by Σ^*. The Laplace approximation of the marginal predictive likelihood is therefore given by:

$$\ln \hat{p}_L(y_{t+h}^*|\mathcal{Y}_t) = \ln L(y_{t+h}^*|\mathcal{Y}_t; \theta^*) + \ln L(\mathcal{Y}_t; \theta^*) - \ln L(\mathcal{Y}_t; \tilde{\theta})$$
$$+ \ln p(\theta^*) - \ln p(\tilde{\theta}) + \frac{\ln|\tilde{\Sigma}| - \ln|\Sigma^*|}{2}, \qquad (12.72)$$

where the expression takes into account that $\ln|A^{-1}| = -\ln|A|$ when A has full rank. Gelfand and Dey (1994) refer to (12.72) as their case (ii) and they note that the approximation has $O(t^{-2})$ accuracy. In other words, the Laplace approximation of the marginal predictive likelihood in (12.72) is "more accurate" than the Laplace approximation of the marginal likelihood.

Alternatively, the log posterior for the sample $(y_{t+h}^*, \mathcal{Y}_t)$ and its Hessian can be evaluated at the parameter value $\tilde{\theta}$ instead of θ^*. This has the advantage that only one posterior mode estimation is required, rather than one *plus* one for each y_{t+h}^* ($h = 1, \ldots, H$) that we are interested in. However, the use of θ^* in (12.72) ensures that the first derivatives of the log posterior for the sample $(y_{t+h}^*, \mathcal{Y}_t)$ are equal to zero, while the use of $\tilde{\theta}$ only ensures that they are, at best, approximately zero. This fact implies an additional error source for the approximation, with the effect that its accuracy is reduced to $O(t^{-1})$, i.e., the same order of accuracy as the one the marginal likelihood approximation in (12.71) has.

Replacing θ^* with $\tilde{\theta}$ in (12.72) yields the following simplified expression of the log of the marginal predictive likelihood:

$$\ln \tilde{p}_L(y_{t+h}^*|\mathcal{Y}_t) = \ln L(y_{t+h}^*|\mathcal{Y}_t; \tilde{\theta}) + \frac{1}{2}\left(\ln|\tilde{\Sigma}_t| - \ln|\tilde{\Sigma}_{t+h}|\right), \qquad (12.73)$$

where $\tilde{\Sigma}_{t+h}$ and $\tilde{\Sigma}_t$ are minus the Hessians of the log posteriors based on the samples $(y_{t+h}^*, \mathcal{Y}_t)$ and $\mathcal{Y}_t$, respectively, evaluated at $\tilde{\theta}$. The first term on the right hand side of (12.73) is the log of the conditional likelihood, evaluated at the posterior mode using the $\mathcal{Y}_t$ data, and we often expect the marginal predictive likelihood to be dominated by this term. Concerning the second term it may be noted that if the two log determinants are equal, then the Laplace approximation is equal to the $\tilde{\theta}$ plug-in estimator of the predictive likelihood. Moreover, it is straightforward to show that

$$\tilde{\Sigma}_{t+h} = \tilde{\Sigma}_t - \frac{\partial^2 \ln L(y_{t+h}^*|\mathcal{Y}_t; \tilde{\theta})}{\partial\theta\partial\theta'} = \tilde{\Sigma}_t + \tilde{\Omega}_{t+h|t}. \qquad (12.74)$$

Hence, we may expect that the second term in (12.73) is often close to zero. Furthermore, the overall computational cost when using (12.73) is *not* reduced by taking the expression on the right hand side of (12.74) into account *unless* analytical derivatives are available.

12.6.1. *The Predictive Likelihood under Annualizations*

In some situations we are also interested in annualizations of variables in first differences when computing the predictive likelihood. For quarterly data of variables in first differences we simply add the current and the previous three quarters to obtain the annual difference, while monthly data require the current and previous 11 months to obtain the annual difference. The joint predictive likelihood is invariant to such variables transformations for any subset of the variables,

but the marginal predictive likelihood is not. For example,

$$
p\left(\sum_{j=0}^{3} \Delta q_{t+i-j}, \ldots, \sum_{j=0}^{3} \Delta q_{t+1-j} \mid \mathcal{Y}_t\right) = \prod_{j=0}^{i-1} p\left(\Delta q_{t+i-j} \mid \Delta q_{t+i-j-1}, \ldots, \Delta q_{t+1}, \mathcal{Y}_t\right)
$$
$$
= p\left(\Delta q_{t+i}, \ldots, \Delta q_{t+1} \mid \mathcal{Y}_t\right), \tag{12.75}
$$

for any horizon i and variable q which is in the information set $\mathcal{Y}_t$. It is here assumed that q_{t-j}, for $j = 0, 1, 2, 3$ are not missing observations. This assumption guarantees that Δq_{t+i-j} can all be uniquely determined from $\sum_{k=0}^{3} \Delta q_{t+i-j-k}$ and q_{t-j}.

From equation (12.75) it also follows that the predictive likelihood of $\sum_{j=0}^{3} \Delta q_{t+i-j}$ for $i > 2$ is not same as the predictive likelihood of Δq_{t+i} since the first equality is not satisfied. Below we shall examine which additions to the Kalman filter that are needed to compute the predictive likelihood under annualizations that involve adding current and past values. Moreover, for models where some variables are given as, say, quarterly first differences while others are given in levels we also need to take the covariance structure between these entities into account.

To provide the covariance matrices for sums of future values of the observed variables, let s be an integer which determines how many current and past values of the variables that will be added. It would be 4 for quarterly data and 12 for monthly. Next, for $i = 1, \ldots, s - 1$

$$
F_i = F_{i-1} + F^i,
$$
$$
Q_i = Q_{i-1} + F_i Q F_i',
$$
$$
P_{t|t}^{(i)} = F_i P_{t|t} F_i',
$$

with initialization values $F_0 = I_r$ and $Q_0 = Q$, resulting in $P_{t|t}^{(0)} = P_{t|t}$ Define the covariance matrices:

$$
C_t^{(i)} = C\left(\sum_{j=i_s}^{i} \xi_{t+j} \mid \mathcal{Y}_t; \theta\right), \tag{12.76}
$$

$$
\Sigma_t^{(i)} = C\left(\sum_{j=i_s}^{i} y_{t+j} \mid \mathcal{Y}_t; \theta\right), \tag{12.77}
$$

where $i_s = \max\{1, i - s + 1\}$. For example, when $s = 4$ this means that for $i = 1, \ldots, 4$ we add ξ_{t+1} until ξ_{t+i} in the $C_t^{(i)}$ covariances, and for $i \geq 5$ we add ξ_{t+i-3} until ξ_{t+i}.

By throwing oneself into a multitude of tedious algebra it can be shown that

$$
C_t^{(i)} = \begin{cases} F P_{t|t}^{(i-1)} F' + Q_{i-1}, & \text{if } i = 1, \ldots, s, \\ F C_t^{(i-1)} F' + Q_{s-1}, & \text{if } i \geq s + 1. \end{cases} \tag{12.78}
$$

Furthermore, it can also be shown that

$$
\Sigma_t^{(i)} = \begin{cases} H' C_t^{(i)} H + iR, & \text{if } i = 1, \ldots, s \\ H' C_t^{(i)} H + sR, & \text{if } i \geq s + 1. \end{cases} \tag{12.79}
$$

Provided that all variables in the y vector are in first differences, the conditional likelihood for $z_{t+i} = \sum_{j=i_s}^{i} y_{t+j}^* = K' \sum_{j=i_s}^{i} y_{t+j}$, $i = 1, \ldots, h$, is now given by:

$$
\ln L\left(z_{t+i} \mid \mathcal{Y}_t; \theta\right) = -\frac{n^*}{2} \ln(2\pi) - \frac{1}{2} \ln \left|\Sigma_{z,t}^{(i)}\right|
$$
$$
- \frac{1}{2}\left(z_{t+i} - z_{t+i|t}\right)' \Sigma_{z,t}^{(i)^{-1}}\left(z_{t+i} - z_{t+i|t}\right), \tag{12.80}
$$

where $\Sigma_{z,t}^{(i)} = K' \Sigma_t^{(i)} K$ and $z_{t+i|t} = \sum_{j=i_s}^{i} y_{t+j|t}^*$.

When some variables in the y vector are in first differences and some in levels, the actuals (y_{t+i}), the forecasts $(y_{t+i|t})$, and the covariance matrices $(\Sigma_t^{(i)})$ need to take this into account. For the actuals and the forecast we simply avoid summing current and past values for the variables in levels, while the diagonal elements of the covariances $\Sigma_t^{(i)}$ corresponding to levels variables are replaced with the diagonal elements of $\Sigma_{y,t+i|t}$ from these positions. It remains to replace the off-diagonal elements of the $\Sigma_t^{(i)}$ matrices which should represent covariances between first differenced variables and levels variables with the correct covariances.

To determine the covariances between sums of first differenced variables and levels variables, define

$$\Xi_t^{(i)} = C\left(\xi_{t+i}, \sum_{j=i_s}^{i} \xi_{t+j} \,\middle|\, y_t; \theta\right), \tag{12.81}$$

$$\Upsilon_t^{(i)} = C\left(y_{t+i}, \sum_{j=i_s}^{i} y_{t+j} \,\middle|\, y_t; \theta\right). \tag{12.82}$$

Some further algebra later it can be established that for $\Xi_t^{(0)} = 0$

$$\Xi_t^{(i)} = \begin{cases} P_{t+i|t} + F\Xi_t^{(i-1)}, & \text{if } i = 1, \ldots, s, \\ F\Xi_t^{(i-1)}F' + F_{s-1}Q, & \text{if } i \geq s+1. \end{cases} \tag{12.83}$$

Finally, it can be shown that

$$\Upsilon_t^{(i)} = H'\Xi_t^{(i)}H + R, \quad i = 1, \ldots, h. \tag{12.84}$$

12.7. *Testing the Normal Approximation of the Predictive Distribution*

The distributional form of the predictive distribution is unknown unless we condition on the parameters, in which case the predictive distribution is normal. For this reason it may be interresting to check if the predictive distribution itself can be approximated by a normal distribution. To begin with, we know how the mean and the covariance matrix can be estimated; see, e.g., equations (12.3) and (12.5) for the unconditional forecasts. With these estimates at hand we check how well the normal approximation works for the marginal predictive distribution of y_{T+i}.

The Kolmogorov-Smirnov test is a well-known nonparametric test for the equality of continuous, one-dimensional probability distributions that can be used to compare the predictive distribution to a normal; in Section 11.14 we considered the test statistic for the case when draws from two empirical distributions are considered. The dimensionality issue means that we can only check for one variable and one forecast horizon at a time.[111] Hence, even if the null hypothesis that the distributions are equal is true, it need not be the case that the full predictive distribution is well approximated by a normal distribution. Nevertheless, such tests may be informative about the appropriateness of using a normal distribution.

Let $F_N(y_j)$ be the cumulated empirical distribution function based on $N = PS$ draws from the predictive distribution of element j of y_{T+i}, while $F(y_j)$ is the normal cdf. The Kolmogorov-Smirnov statistic is now given by

$$D_N = \sup_{y_j} \left| F_N(y_j) - F(y_j) \right|. \tag{12.85}$$

The asymptotic behavior of the statistic is given by

$$\sqrt{N}D_N \Longrightarrow \sup_{t \in [0,1]} |B(t)| = K, \tag{12.86}$$

where K is the Kolmogorov distribution whose cdf is shown in equation (11.68).

---

[111] There are some attempts to determine test statistics for the multivariate case; see, e.g., Fasano and Franceschini (1987) who consider two and three dimensions for the test. For some more recent work on the problem of comparing multivariate distributions, see Loudin and Miettinen (2003).

12.8. Continuous Ranked Probability Score and Energy Score

As noted in Section 12.6, the log predictive score is the only local scoring rule which is proper. However, it may be of interest to go beyond the predictive likelihood and also consider outcomes close but different from the actual values, as well as values further away. A natural candidate for this is to take paths drawn from the predictive distribution into the picture, such as the P paths obtained in Section 12.1.

The *continuous ranked probability score* (CRPS) is an alternative score function when dealing with univariate density forecasts; see, e.g., Gneiting and Raftery (2007). Let $y_{i,T+h}^{(j)}$ be the h-step-ahead prediction of $y_{i,T+h}$ for $i = 1, \ldots, n$ and $j = 1, \ldots, P$, while the actual or observed value is given by $y_{i,T+h}^{(o)}$. As noted by Gneiting and Raftery (2007), the CRPS for the predictive distribution, PD, is given by

$$\text{CRPS}(h, i) = \frac{1}{2} E_{PD} \left[\left| y_{i,T+h} - y_{i,T+h}^{(c)} \right| \right] - E_{PD} \left[\left| y_{i,T+h} - y_{i,T+h}^{(o)} \right| \right], \quad i = 1, \ldots, n, \quad (12.87)$$

where $y_{i,T+h}$ and $y_{i,T+h}^{(c)}$ are independent copies (draws) of a random variable with predictive distribution, PD, while the expectation is taken with respect to this distribution. This score can be consistently estimated using the P forecasts paths and the observed values by

$$\text{CRPS}(h, i) = \frac{1}{2P^2} \sum_{j_1=1}^{P} \sum_{j_2=1}^{P} \left| y_{i,T+h}^{(j_1)} - y_{i,T+h}^{(j_2)} \right| - \frac{1}{P} \sum_{j=1}^{P} \left| y_{i,T+h}^{(j)} - y_{i,T+h}^{(o)} \right|, \quad i = 1, \ldots, n. \quad (12.88)$$

The CRPS is a proper score function relative to the class to which PD belongs, and is also strictly proper under the condition mentioned by Gneiting and Raftery (2007).

If the predictive distribution is normal, the CRPS has, as noted by Gneiting, Raftery, Westveld III, and Goldman (2005), a closed-form expression and can be computed directly from the first two moments of the distribution; see also Jordan, Krüger, and Lerch (2019) for closed-form expressions of the CRPS for a wide range of distributions. With $\varphi(x)$ being the standard normal pdf and $\Phi(x)$ the cdf, the CRPS is now given by:

$$
\begin{aligned}
\text{CRPS}(h, i) = -\sigma_{y_{i,T+h}} \Bigg(& \frac{y_{i,T+h}^{(o)} - \mu_{y_{i,T+h}}}{\sigma_{y_{i,T+h}}} \left[2\Phi \left(\frac{y_{i,T+h}^{(o)} - \mu_{y_{i,T+h}}}{\sigma_{y_{i,T+h}}} \right) - 1 \right] \\
& + 2\varphi \left(\frac{y_{i,T+h}^{(o)} - \mu_{y_{i,T+h}}}{\sigma_{y_{i,T+h}}} \right) - \frac{1}{\sqrt{\pi}} \Bigg),
\end{aligned}
\quad (12.89)
$$

where $\mu_{y_{i,T+h}}$ and $\sigma_{y_{i,T+h}}$ are the predictive mean and standard deviation, respectively, of $y_{i,T+h}$ for $i = 1, \ldots, n$.

A multivariate extension of the CRPS, called the *energy score*, was introduced by Gneiting and Raftery (2007). Based on the specification in Gneiting, Stanberry, Grimit, Held, and Johnson (2008), it can be expressed as

$$\text{ES}(h, s) = \frac{1}{2} E_{PD} \left[\left\| y_{s,T+h} - y_{s,T+h}^{(c)} \right\| \right] - E_{PD} \left[\left\| y_{s,T+h} - y_{s,T+h}^{(o)} \right\| \right], \quad (12.90)$$

where, as before, $\|x\| = \sqrt{x'x}$ is the Euclidean norm, and $y_{s,T+h}$ is a selection of s variables among the n variables in y_{T+h}, with $s \leq n$. This expression can be estimated using the P paths from the predictive distribution as

$$\text{ES}(h, s) = \frac{1}{2P^2} \sum_{j_1=1}^{P} \sum_{j_2=1}^{P} \left\| y_{s,T+h}^{(j_1)} - y_{s,T+h}^{(j_2)} \right\| - \frac{1}{P} \sum_{j=1}^{P} \left\| y_{s,T+h}^{(j)} - y_{s,T+h}^{(o)} \right\|. \quad (12.91)$$

12.9. Probability Integral Transform

The *probability integral transform* (PIT) has long been used to assess if a model is correctly specified. An early paper which considered this idea for density forecasting purposes in econometrics is Diebold, Gunther, and Tay (1998), but it has earlier been emphasized by Dawid

(1984). Rosenblatt (1952) shows that for a correctly specified model
$$\pi_{jt} = F_j\big(y_{j,T+1}|\mathcal{Y}_t\big), \quad j = 1, \ldots, n,$$
is independent and uniformly distributed on the unit interval, where $F_j(\cdot)$ is the cumulative distribution function (cdf). Smith (1985) further noted that $z_{jt} = \Phi^{-1}(\pi_{jt})$, where $\Phi(\cdot)$ is the cdf of the normal distribution, is i.i.d. $N(0,1)$; also also Berkowitz (2001).

Amisano and Geweke (2017) construct a test statistic based on the normality property of the inverse cdf; details are available in their Online Appendix. Specifically, let
$$\pi_{j,T+1|T}^{(i)} = \Phi\left(y_{j,T+1}^{(o)} \Big| \mu_{j,T+1|T}^{(i)}, \sigma_{jj,T+1|T}^{(i)} \right), \quad i = 1, \ldots, N$$
where $\mu_{j,T+1|T}^{(i)}$ is the one-step-ahead point forecast of observed variable $j = 1, \ldots, n$ using the i:th posterior draw of θ, while $\sigma_{jj,T+1|T}^{(i)}$ is the one-step-ahead forecast standard deviation of variable j using $\theta^{(i)}$. A similar expression can also be determined for a subset of the observed variables, such as $y_{s,T+1}^{(o)}$ as well as for h-step-ahead forecasts. A generic expression is then
$$\pi_{s,T+h|T}^{(i)} = \Phi\left(y_{s,T+h}^{(o)} \Big| \mu_{s,T+h|T}^{(i)}, \Sigma_{s,T+h|T}^{(i)} \right).$$

Next, the Monte Carlo average of the N values of the uniform variable is taken such that
$$\pi_{j,T+1|T} = \frac{1}{N} \sum_{i=1}^{N} \pi_{j,T+1|T}^{(i)}, \tag{12.92}$$
while
$$z_{j,T+1|T} = \Phi^{-1}\big(\pi_{j,T+1|T}\big). \tag{12.93}$$
Under the assumption that the model is correctly specified, or similarly that the density forecasts are well calibrated, the variable $z_{j,T+1|T}$ is normally distributed with zero mean and unit variance. This assumption is tested in Amisano and Geweke (2017) using the first q moments and p lags of the $z_{j,T+1|T}$ process. They now consider the test statistic
$$\text{AG} = T_1\big(\bar{m}_{T_1} - m_{q+p}\big)' \Omega^{-1}\big(\bar{m}_{T_1} - m_{q+p}\big) \xrightarrow{d} \chi_{q+p}^2, \tag{12.94}$$
where T_1 is the number of one-step-ahead forecasts,
$$\bar{m}_{T_1} = \begin{bmatrix} T_1^{-1} \sum_{T=1}^{T_1} z_{j,T+1|T} \\ \vdots \\ T_1^{-1} \sum_{T=1}^{T_1} z_{j,T+1|T}^q \\ \big(T_1 - 1\big)^{-1} \sum_{T=2}^{T_1} z_{j,T+1|T} z_{j,T|T-1} \\ \vdots \\ \big(T_1 - p\big)^{-1} \sum_{T=p+1}^{T_1} z_{j,T+1|T} z_{j,T+1-p|T-p} \end{bmatrix}, \quad m_{q+p} = \begin{bmatrix} \mu_1 \\ \vdots \\ \mu_q \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$
The scalar μ_r is the r:the population moment of a normal distribution. The matrix Ω is given by
$$\Omega = \begin{bmatrix} \mu_2 & \mu_3 & \cdots & \mu_{q+1} & 0_{1\times p} \\ \mu_3 & \mu_4 & \cdots & \mu_{q+2} & 0_{1\times p} \\ \vdots & \vdots & & \vdots & \vdots \\ \mu_{q+1} & \mu_{q+2} & \cdots & \mu_{2q} & 0_{1\times p} \\ 0_{p\times 1} & 0_{p\times 1} & \cdots & 0_{p\times 1} & I_p \end{bmatrix},$$

while

$$
\mu_r = \begin{cases} \prod_{i=1}^{r/2}(2i - 1) & \text{if } r \text{ is odd,} \\ 0 & \text{otherwise.} \end{cases}
$$

In practise, Amisano and Geweke (2017) let $q = p = 4$, such that $\mu_1 = \mu_3 == \mu_5 = \mu_7 = 0$, while $\mu_2 = 1$, $\mu_4 = 3$, $\mu_6 = 15$ and $\mu_8 = 105$.

YADA does not compute any tests for the PIT, but provides output on $\pi_{j,T+h|T}$ and $\pi_{s,T+h|T}$ based on the estimaed predictive moments. This output can then be employed by the user to construct their own PIT tests once the `norminv` function in matlab has been applied to the uniforms.

An alternative approach to PITs is discussed by, among others, Geweke and Amisano (2010) and where to object is to examine the uniform property. This approach considers the projected paths of y_{T+h}, as in the case of the CRPS in Section 12.8. For each $\theta^{(i)}$, $i = 1, \ldots, N$, simulate one path per posterior draw, $y_{T+1}^{(i)}, \ldots, y_{T+h^*}^{(i)}$, using the state space form of the DSGE model by drawing shocks and measurement errors for $t = T + 1, \ldots, T + h^*$ and an initial state for $t = T$. For each posterior draw and forecast horizon, we next apply the indicator function to individual observed variables $\mathbb{I}(y_{j,T+h}^{(i)} \leq y_{j,T+h}^{(o)})$, $j = 1, \ldots, n$, which is unity if the event is true and zero otherwise. These zeros and ones may next be used to estimate the probability that the forecast for $T + h$ of variable j is below its observed value with

$$
\hat{F}_h\big(y_{j,T+h} \leq y_{j,T+h}^{(o)}\big) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\big(y_{j,T+h}^{(i)} \leq y_{j,T+h}^{(o)}\big) \overset{\text{a.s.}}{\to} F_h\big(y_{j,T+h} \leq y_{j,T+h}^{(o)}\big), \quad h = 1, \ldots, h^*,
$$

for the model in question, and where $\overset{\text{a.s.}}{\to}$ denotes almost sure convergence and where F is the cdf. Now, the PIT is given by $\hat{F}_h(y_{j,T+h} \leq y_{j,T+h}^{(o)})$, which gives the probability of observing a value less than or equal to $y_{j,T+h}^{(o)}$ in period $T + h$ when computing h-step-ahead forecasts at T. Under the assumption that the model is correctly specified, this random variable is uniform and is independent of the h-step-ahead forecast of $T + s_h h$ for $s_h = 2, 3, \ldots, T_h$, where T_h is the last period for which an h-step-ahead forecast can be computed using the sample at hand. Notice that the, apart from one-step-ahead forecasts, two consecutive h-step-ahead forecasts are not independent and should therefore not be used when analysing the PITs.

To study the deviation from the PIT idea for a model, Geweke and Amisano (2010) determine the average values of

$$
\mathbb{I}\big(((k - 1)/10) < \hat{F}_h\big(y_{j,t} \leq y_{j,t}^{(o)}\big) < (k/10)\big), \quad k = 1, \ldots, 10, \quad t = T + h, T + 2h, \ldots, T_h,
$$

for $h = 1, \ldots, h^*$ and $j = 1, \ldots, n$. The sequence of cdf's is i.i.d. Bernoulli with $p = 0.1$ when the model is correctly specified. The estimated average values for each decile can then be compared with 0.1; see Geweke and Amisano (2010) for an application using such a diagnostic tool graphically.

YADA does not compute the average values for the deciles as this requires multiple forecast origins, but it does estimate $F_h\big(y_{j,T+h} \leq y_{j,T+h}^{(o)}\big)$ for each individual variable once it has located paths from the predictive distribution on disk.

12.10. *Observation Weight Decomposition of the Unconditional Point Forecasts*

The observation weight decompositions of state variables and state shocks for a fixed parameter vector are discussed in Section 5.9, while the observation weights for the structural shocks is covered in Section 11.1. These weights can be applied to any linear function of the state variables and the shocks. For example, he unconditional forecasts of the observed variables for a fixed parameter vector depends on the three matrices A, H and F, the deterministic variables over the forecast horizon and the update estimate of the state variables at the end of the data sample T. This means that once we have an observation weight decomposition of the state variables $\xi_{T|T}$, it can be linearly translated into a decomposition of the point forecasts of $y_{T+i|T}$

for $i = 1, \ldots, h$ based on each individual observed variable and any selection of groups of these variables.

12.11. *YADA Code*

The main functions for computing unconditional prediction paths for the observed variables of the DSGE model are called `DSGEPredictionPathsTheta` and `DSGEPredictionPaths`. The former works with a fixed value for the model parameters θ, while the latter works for a set of draws from the posterior distribution of θ. The calculation of the conditional prediction paths for the observed variables of the DSGE model are handled by `DSGECondPredictionPathsTheta` and `DSGECondPredictionPaths` for the case when the shock values are directly controlled. For the Waggoner-Zha approach with controlled shock distributions, the conditional prediction paths are dealt with by `DSGECondPredictionPathsThetaWZ` and `DSGECondPredictionPathsWZ`. The smooth estimate of the state vector and its covariance matrix are handled through the function `CondPredictionKalmanSmoother` (`CondPredictionKalmanSmootherHt`). Furthermore, there is a total of 16 functions which deal with forecasts subject to the zero lower bound and based on the anticipated shocks method detailed in Section 3.4. These functions are based on a fixed parameter vector or draws from the prior or posterior distributions, they deal with unconditional and conditional forecasts, as well as centered or uncentered forecasts.

Unconditional prediction paths can also be calculated for the Bayesian VAR models. The main functions for this objective are `BVARPredictionPathsPostMode` and `BVARPredictionPaths`. The former function uses a fixed value for the model parameters, while the latter uses draws from the posterior distribution of (Ψ, Π, Ω). Hence, the latter function makes it possible to estimate a predictive distribution of the Bayesian VAR that does not depend on the particular values of the model parameters.

12.11.1. DSGEPredictionPathsTheta

The function `DSGEPredictionPathsTheta` needs 11 inputs. First of all, a set of values for the parameters θ is supplied through the variable `theta`. To use the values properly the vector structure `thetaPositions` and the structure `ModelParameter` that were discussed in Section 7.4 are also needed. Furthermore, the DSGE model information structure `DSGEModel` and the generic initialization structure `CurrINI` must be supplied to the function. The following input is given by the $k \times h$ matrix `X` with the values of the exogenous variables over the h period long prediction sample. Next, the value of h is accepted since `X` is empty if $k = 0$. The 8th input is called `NumPaths` and specifies how many prediction paths to compute, while the boolean variable `AnnualizeData` indicates if the prediction paths should be annualized or not. Similarly, the boolean variable `TransData` indicates if the data should be transformed or not. The final input is given by `NameStr` which indicates the type of values that are used for θ, e.g., the posterior mode estimate.

The main output from the function is the 3-dimensional matrix `PredPaths`, the structure `PredData` and the matrices `PredEventData` and `YObsEventData`. The `PredPaths` matrix dimensions are given by the number of observed variables, the length of the prediction sample, and the number of prediction paths. The matrix `PredEventData` has as many rows as number of observed variables and 7 colummns; the function `CalculatePredictionEvents` computes this matrix; see Section 12.11.13. A prediction event can, for instance, be defined as non-negative inflation for h^* consecutive periods over the prediction period. The h^* integer of always less than or equal to the length of the prediction period. Similarly, the matrix `YObsEventData` has the same dimension as `PredEventData` and holds prediction event data when the mean of the predictive distribution is equal to the realized values for the observed data.

12.11.2. DSGEPredictionPaths

The function `DSGEPredictionPaths` requires 14 inputs. Relative to `DSGEPredictionPathsTheta`, there are two additional inputs (the first and the last) and the second last input variable is different from the last of `DSGEPredictionPathsTheta`. Before the `thetaMode` input, the current

function accepts the matrix `thetaPostSample` with `NumDraws` rows and `NumParam` columns. Despite the name, this matrix can either hold draws from the posterior *or* from the prior distribution. Similarly, `thetaMode` can be the posterior mode estimates as well as the initial values of the parameters θ.

The number of draws from the posterior that are used can vary irrespective of how many draws from the posterior that are available, while the number of draws from the prior are, in principle, arbitrary. Typically, the number of draws of θ that are sent to this function is a small number, such as 500 or 1,000. The last three input variables used by the function are `CurrChain`, which is an integer that indicates the MCMC chain number, `IsPosterior`, a boolean variable which is one if the parameter draws are taken from the posterior distribution and zero if they are taken from the prior, and `Weights` a vector with weights given to the posterior draws. For the MCMC posterior samplers this variable is empty which the function translates into a vector with ones.

As output the function provides 6 variables. The first is the boolean `DoneCalc` that indicates if all calculations were performed or not. The next is the matrix `PredEventData` with prediction event results. The final 4 variables provides the data on the prediction variance decompositions for the observed variables over the whole prediction horizon. These variables are called `StateCov`, `ShockCov`, `MeasureCov`, and `ParameterCov`, respectively. Apart from `MeasureCov` these are all 3D matrices with dimensions $n \times n \times h$, where h is the length of the prediction horizon, while `MeasureCov` is $n \times n$.

The prediction paths are not directly sent as output from the function. These are instead written to disk in `mat`-files, one for each parameter draw. In each file the 3D matrix `PredPaths` is stored. Its dimensions are given by the number of observed variables, the length of the prediction sample, and the number of prediction paths.

12.11.3. DSGECondPredictionPathsTheta(WZ/Mixed)

The function `DSGECondPredictionPathsTheta` (`DSGECondPredictionPathsThetaWZ` for the distribution of shocks (Waggoner and Zha, 1999), or `DSGECondPredictionPathsThetaMixed` for the distribution of a subset of the shocks cases) needs 13 inputs. The first 6 and the last 5 are the same inputs as the function `DSGEPredictionPathsTheta` takes. The 2 additional inputs naturally refer to the conditioning information. Specifically, the 7th input is given by `Z`, an $q_m \times g$ matrix with the conditioning data $[z_{T+1} \cdots z_{T+g}]$, while the 8th input variable is called `U`, an $q_m \times g$ matrix with the initial values $[u_{T-g+1} \cdots u_T]$ for the conditioning; cf. equations (12.6) and (12.43).

The main output from the function is the 3-dimensional matrix `PredPaths`, the matrices with prediction event test results, `PredEventData` and `YObsEventData`, and the modesty results, `MultiModestyStat`, `UniModestyStat`, and `UniModestyStatLZ`. The dimension of `PredPaths` is given by the number of observed variables, the length of the prediction sample, and the number of prediction paths. The matrix `PredEventData` (and `YObsEventData`) has as many rows as number of observed variables and 7 columns. These matrices are computed by the function `CalculatePredictionEvents`. A prediction event can, for instance, be defined as non-negative inflation for h^* consecutive periods over the prediction period. The h^* integer of always less than or equal to the length of the prediction period. The difference between `PredEventData` and `YObsEventData` is that the latter matrix holds prediction event results when the mean of the predictive distribution has been set equal to the realized values of the observed variables.

The modesty statistics are only calculated when `AnnualizeData` is zero. When this condition is met, `MultiModestyStat` is a matrix of dimension `NumPaths` times 2, where the first columns holds the values of $\mathcal{M}_{T,g}(\bar{\eta})$, while the second column gives $\mathcal{M}_{T,g}(\eta)$. The matrix `UniModestyStat` has dimension `NumPaths` times n and gives the univariate modesty statistics in equation (12.53), while `UniModestyStatLZ` is a vector with the n values of the univariate Leeper-Zha related modesty statistic.

12.11.4. DSGECondPredictionPaths(WZ/Mixed)

The function DSGECondPredictionPaths (DSGECondPredictionPathsWZ for the distribution of shocks (Waggoner and Zha, 1999), or DSGECondPredictionPathsMixed for the distribution of a subset of the shocks cases) for computing the conditional predictive distribution requires 16 inputs. The first 7 and the last 7 input variables are the same as those that the function DSGEPredictionPaths accepts. The two additional inputs refer to the same data that the function DSGECondPredictionPathsTheta requires, i.e., to Z and U.

Moreover, as in the case of DSGEPredictionPaths for the unconditional predictive distribution, the majority of the output from this function is not sent through its output arguments, but are written to disk. For instance, the prediction paths are written to disk in mat-files, one for each parameter draw. In each file the 3D matrix PredPaths is stored. Its dimensions are given by the number of observed variables, the length of the prediction sample, and the number of prediction paths. Moreover, the multivariate and univariate modesty statistics are calculated and saved to disk provided that the AnnualizeData variable is zero (no annualization).

The function provides the same 6 output arguments as the DSGEPredictionPaths function. Moreover, the prediction paths data is also written to disk in mat-files, one for each parameter draw. In each file the 3D matrix PredPaths is stored. Its dimensions are given by the number of observed variables, the length of the prediction sample, and the number of prediction paths.

12.11.5. ZLBDSGEPredictionPathsTheta & ZLBDSGECenteredPredictionPathsTheta

The function ZLBDSGEPredictionPathsTheta takes the same input variables as the function DSGEPredictionPathsTheta in order to compute prediction paths for the observed variables and the state variables which satisfy the zero lower bound. The data for this bound is stored in the DSGEModel structure.

The function provides six output variables, given by: Status, ZLBYPaths, ZLBKsiPaths, KsitTh, PredData and PredEvenData. The last two variables include similar information as for the DSGEPredictionPathsTheta function. The boolean variable Status is unity if the calculations were successfully run, and zero otherwise. The 3D matrix ZLBYPaths and ZLBKsiPaths hold the predicted paths of the observed and state variables, respectively.

The function ZLBDSGECenteredPredictionPathsTheta takes centering of the paths into account, where the centering is based on the zero lower bound not being binding. Centering allows for alternative data being used, reflecting the various actuals that the data construction file provides. The additional input variables are YData, an $n \times h$ matrix with the data for centering over the forecast horizon, and YDataStr, a string vector determining which alternative data is sent to the function. The output variables are identical to those given by the un-centered function.

12.11.6. ZLBDSGEPredictionPaths & ZLBDSGECenteredPredictionPaths

The zero lower bound compatible function ZLBDSGEPredictionPaths takes the same 14 input variables as the function DSGEPredictionPaths. The output variables are given by DoneCalc, PredEventData and ObsVarMean, where the last reflects the population mean of the observed variables, while the first two variables have the same functionality as the same named variables in DSGEPredictionPaths.

The function ZLBDSGECenteredPredictionPaths takes centering of the paths into account and therefore requires the same two additional variables as the fixed model parameters based function ZLBDSGECenteredPredictionPathsTheta. The output variables are identical to those for ZLBDSGEPredictionPaths.

12.11.7. ZLBDSGECondPredictionPathsTheta(WZ/Mixed) & ZLBDSGECenteredCondPredictionPathsTheta(WZ/Mixed)

The function ZLBDSGECondPredictionPathsTheta (ZLBDSGECondPredictionPathsThetaWZ for the distribution of shocks and ZLBDSGECondPredictionPathsThetaMixed for the distribution of a subset of the shocks cases) takes the zero lower bound as well as conditioning restrictions

into account and makes use of the same 13 input variables as `DSGECondPredictionPathsTheta`. Furthermore, the output variables are the same as for the unconditional forecasts subject to the zero lower bound in `ZLBDSGEPredictionPathsTheta`.

The function `ZLBDSGECenteredCondPredictionPathsTheta` (with appended part `WZ` for the distribution of shocks and `Mixed` for the distribution of a subset of the shocks cases) covers the case of centering the conditional forecasts. This means that it requires the same two additional input variables as the unconditional forecasts in `ZLBDSGECenteredPredictionPathsTheta`. The output variables are the same as for the uncentered case.

12.11.8. ZLBDSGECondPredictionPaths(WZ/Mixed) & ZLBDSGECenteredCondPredictionPaths(WZ/Mixed)

The function `ZLBDSGECondPredictionPaths` (`ZLBDSGECondPredictionPathsWZ` for the distribution of shocks and `ZLBDSGECondPredictionPathsMixed` for the distribution of a subset of the shocks cases) takes the zero lower bound as well as conditioning restrictions into account and makes use of the same 16 input variables as `DSGECondPredictionPaths`. The output variables are identical to the unconditional forecasts computed in `ZLBDSGEPredictionPaths`.

The function `ZLBDSGECenteredCondPredictionPaths` (with appended part `WZ` for the distribution of shocks and `Mixed` for the distribution of a subset of the shocks cases) covers the case of centering the conditional forecasts. This means that it requires the same two additional input variables as the unconditional forecasts in `ZLBDSGECenteredCondPredictionPathsTheta`. The output variables are unchanged relative to the uncentered case.

12.11.9. CondPredictionSmoother(Ht)

The function `CondPredictionSmoother` (`CondPredictionSmootherHt`) computes smooth estimate of the state variables and its covariance matrix in period T using the conditioning assumptions when $K_2 = 0$ and $u_T = 0$, i.e., when $z_t = K_1' y_t$ in equation (12.6). To fulfill its mission the function needs 15 input variables to compute the smooth estimate of the state variables for the last historical time period (T) using the conditioning assumptions. The first 4 variables are given by `KsiTT1`, `PTT1`, `KsiTT`, and `PTT` which provide the forecast and smooth estimates of the state variables and the corresponding covariance matrices at T. Next, the function needs `YhatT` and `YT`, the forecast and the realization of the observed variables at T. In order to run the Kalman filter over the conditioning sample, the function thereafter needs `XPred`, a matrix with the values for the exogenous variables over this sample, `Z`, a matrix with the conditioning assumptions, and `K1`, which maps the conditioning assumptions into the observed variables. Finally, the function takes the matrices `A`, `H`, `R`, `F`, and `B0` from the state-space represenation for the observed variables, and the integer variable `KalmanAlgorithm`, determining which Kalman filtering approach to apply.

As output the function gives two variables: `KsiTTg` and `PTTg`. The first is a vector with the smooth estimates of the state variables at T conditional on the historical sample and the conditioning assumptions. The second variable is a matrix with the covariance matrix for the state variables based on this estimator. In other words, the first output variable is equal to $\xi^{(z)}_{T|T+g}$, while the second is equal to $P^{(z)}_{T|T+g}$; see Section 12.2.4 for mathematical details.

12.11.10. CondPredictionKalmanSmoother(Ht)

The function `CondPredictionKalmanSmoother` (`CondPredictionKalmanSmootherHt`) computes smooth estimate of the state variables and its covariance matrix in period T using the conditioning assumptions when either $K_2 \neq 0$ or $u_T \neq 0$ in equation (12.6). To complete its task the function needs 13 input variables. To setup initial conditions for running the Kalman smoother that makes use of the conditioning assumptions, smooth estimates of the state vector, its covariance matrix, and the measurement errors are needed. These are accepted as the input variables `KsitT`, `PtT`, and `wtT`, being of dimensions $r \times g$, $r \times r \times g$, and $n \times g$, respectively. The time dimension of this data should correspond to periods $T - g + 1, \ldots, T$, where g is the length of the conditioning sample and T is the last period of the historical sample, i.e., the conditioning

sample covers periods $T + 1, \ldots, T + g$. Next, the conditioning assumptions are given by the $q_m \times g$ matrix Z, while the $q_m \times g$ matrix U contains initial values; see the discussion above for the function DSGECondPredictionPathsTheta. The sixth input is X, a $k \times g$ matrix with data on the exogenous variables over the conditioning sample. The following two inputs are K1 and K2, $n \times q_m$ matrices linking the observed variables to the conditioning assumptions; see equation (12.6) or (12.43). The final 5 input variables are the parameter matrices: A, H, and R for the measurement equation, as well as F and B0 for the state equation. The measurement matrix H has dimension $r \times n$ in CondPredictionKalmanSmoother, while it has dimension $r \times n \times (2g - 1)$ in CondPredictionKalmanSmootherHt.

As output the function gives the r dimensional vector KsiTTz of smooth state variable estimates that take the conditioning assumptions into account and the $r \times r$ covariance matrix PTTz. The former is equal to $\xi_{T|T+g}^{(z)}$, while the latter is $P_{T|T+g}^{(z)}$; see Section 12.2.4 for mathematical details.

12.11.11. StateCondPredictionPathsTheta(WZ/Mixed)

The function StateCondPredictionPathsTheta (StateCondPredictionPathsThetaWZ for the distribution of shocks (Waggoner and Zha, 1999), or StateCondPredictionPathsThetaMixed for the distribution of a subset of the shocks cases) needs 14 input variables. The first 6 and last 7 are exactly the same input variables as the functions in Section 12.11.3 accept. The additional seventh input variable is called Zeta, a $q_z \times g$ matrix with conditioning data for the linear combinations of the state variables $[\zeta_{T+1} \cdots \zeta_{T+g}]$; cf. equation (12.56).

The output variables are identical to those delivered by the conditional forecasting function DSGECondPredictionPathsTheta (DSGECondPredictionPathsThetaWZ).

12.11.12. StateCondPredictionPaths(WZ/Mixed)

The function StateCondPredictionPaths (StateCondPredictionPathsWZ for the distribution of shocks (Waggoner and Zha, 1999), or StateCondPredictionPathsMixed for the distribution of a subset of the shocks cases) requires 17 input variables to perform its task. In addition to the 15 input variables that the conditional forecasting functions in Section 12.11.4 take, the 8th input variable is given by Zeta, described above, while the 17th input variable is CondTypeStr. This variable is a string vector that takes the value State- or State-Obs-. The former value indicates that only assumptions for the state variables are used, while the latter indicates that conditioning assumptions for the observed variables are added.

The output variables are identical to those delivered by the conditional forecasting function DSGECondPredictionPaths (DSGECondPredictionPathsWZ).

12.11.13. CalculatePredictionEvents

The function CalculatePredictionEvents computes prediction event and risk analysis data from the simulated prediction paths. The function requires 2 input variables: PredPaths and PredictionEvent. The first variable is the familiar matrix with all prediction paths for a given parameter value, while the second is a matrix that holds the prediction event information. The number of rows of PredictionEvent is equal to the number of variables, and the number of columns is three; the upper bound of the event, the lower bound, and the number of consecutive periods for the event.

As output the function provides the matrix PredEventData. The number of rows is equal to the number of variables, while the number of columns is 8. The first column gives the number of times the event is true, i.e., the number of times that the paths contain values that fall within the upper and lower bound for the number of periods of the event. The second column holds the number of times that the paths are below the lower bound of the event for the length of the event, the third provides the number of times that the paths are above the upper bound of the event for the length of the event, while the 4th column has the total number of times that the event could be true. The 5th and 6th column store the sum of and the sum of squared deviations from the lower bound of the event when the paths are below the lower bound for

the required length of the event. Similarly, the 7th and 8th column hold the sum of and sum of squared deviations from the upper bound of the event when the paths are above the upper bound for the required length of the event.

12.11.14. DSGEPredictiveLikelihoodTheta

The function `DSGEPredictiveLikelihoodTheta` computes the joint and the marginal predictive likelihood for fixed parameter values using the Laplace approximation. The function requires 17 input variables: `theta`, `thetaPositions`, `thetaIndex`, `thetaDist`, `PriorDist`, `LowerBound`, `UniformBounds`, `ModelParameters`, `YData`, `X`, `IsOriginal`, `IsPlugin`, and also `FirstPeriod`, `LastPeriod`, `StepLength`, `DSGEModel`, and `CurrINI`. Most of these input variables have been described above. The matrix `YData` contains the realizations of the observed variables over the forecast sample, `X` is a matrix with data on the deterministic variables over the same sample, `IsOriginal` is a boolean variable that takes the value 1 if the original data should be forecasted and 0 if annualized data should be predicted, while `IsPlugin` is a boolean variable that takes the value 1 if only the plugin estimator should be computed and 0 if also the Laplace approximation should be calculated. Finally, `StepLength` is the step length used by the function that estimates the Hessian matrix using finite differences; see Abramowitz and Stegun (1964, p. 884) formulas 25.3.24 and 25.3.27, and Gill and King (2004).

The function provides six required output variables and two optional. The required variables are `JointPDH` and `MargPDH`, vectors with the joint and marginal predictive likelihood values, respectively. Furthermore, the variable `LaplaceMargLike` is a scalar with the value of the marginal likelihood for the historical sample using the Laplace approximation, while the cell array `PredVars` contains vectors with positions of the predicted variables among all observed variables. The vectors `MargPlugin` and `JointPlugin` give the plugin estimates of the marginal and joint predictivce likelihood, i.e., when the second term involving the Hessian matrices in equation (12.73) is dropped. The optional output variables are `status` and `kalmanstatus`, which have been discussed above.

13. Frequency Domain Analysis

13.1. *Population Spectrum*

Let $\Sigma_y(h)$ denote the covariance between y_t and y_{t-h}. Under the assumption that y_t is covariance stationary and that the autocovariances are absolutely summable, the *population spectrum* of y is a Fourier transform of the autocovariance function and is given by

$$s_y(\omega) = \frac{1}{2\pi} \sum_{h=-\infty}^{\infty} \Sigma_y(h) \exp(-i\omega h), \tag{13.1}$$

where $i = \sqrt{-1}$ and ω is the angular frequency measured in radians, while $\exp(-i\omega h)$ is a point on the unit circle for all $\omega \in \mathbb{R}$ and integers h; see, e.g., Fuller (1976, Chapter 4) or Hamilton (1994, Chapters 6 and 10). De Moivre's theorem allows us to write

$$\exp(-i\omega h) = \cos(\omega h) - i \sin(\omega h).$$

Making use of the some well known results from trigonometry,[112] it can be shown that the population spectrum can be rewritten as

$$\begin{aligned}
s_y(\omega) = \frac{1}{2\pi} \Bigg[&\Sigma_y(0) + \sum_{h=1}^{\infty} \left(\Sigma_y(h) + \Sigma_y(h)' \right) \cos(\omega h) \\
&- i \sum_{h=1}^{\infty} \left(\Sigma_y(h) - \Sigma_y(h)' \right) \sin(\omega h) \Bigg].
\end{aligned} \tag{13.2}$$

From (13.2) it can be seen that the diagonal elements of the population spectrum are real and symmetric around zero.[113] The off-diagonal elements (cross-spectrum) are complex while the modulus of each such element is symmetric around zero. This follows by noticing that the real part of the spectrum is symmetric, while the imaginary part is skew-symmetric.[114] Hence, the population spectrum is a Hermitian matrix so that $s_y(\omega) = s_y(-\omega)'$, i.e., the complex equivalent of a symmetric matrix. The real part of the cross-spectrum is called the *co-spectrum* while the imaginary part is known as the *quadrature spectrum*.

Moreover, since the sine and cosine functions are periodic such that $\cos(a) = \cos(a + 2\pi k)$ and $\sin(a) = \sin(a + 2\pi k)$ for any integer k the population spectrum is also a periodic function of ω with period 2π. Hence, if the know the value of a diagonal element of the spectrum for all ω between 0 and π, we can infer the value of this element for any ω. Similarly, if we know the value of the modulus of an off-diagonal element of the spectrum for all ω between 0 and π, we can infer the value of the modulus for any ω.

Translating frequencies into time periods we let $\omega_j = 2\pi j / T$, where the frequency ω_j has a period of T / j time units (months, quarters, or years). This means that the frequency is equal to 2π divided by the period in the selected time units. As an example, suppose that low frequencies are regarded as those with a period of 8 years or more, business cycle frequencies those with a period of, say, 1 to 8 years, while high frequencies are those with a period less than 1 year. For quarterly time units, these periods imply that low frequencies are given by $\omega \in [0, \pi / 16]$, busines cycle frequencies by $\omega \in [\pi / 16, \pi / 2]$, while high frequencies are $\omega \in [\pi / 2, \pi]$.

13.2. *Spectral Decompositions*

There is an inverse transformation of (13.1) that allows us to retrieve the autocovariance matrices from the spectral density. Specifically,

$$\Sigma_y(h) = \int_{-\pi}^{\pi} s_y(\omega) \exp(i\omega h) d\omega. \tag{13.3}$$

---

[112] Specifically, recall that $\cos(0) = 1$, $\cos(-a) = \cos(a)$, $\sin(0) = 0$, and $\sin(-a) = -\sin(a)$; see, e.g., Hamilton (1994, Appendix A) for some details.

[113] In addition, the diagonal elements of the population spectrum are non-negative for all ω; see, e.g., Fuller (1976, Theorem 3.1.9).

[114] A matrix B is said to be skew-symmetric if $B' = -B$.

For $h = 0$ it thus follows that the area under the population spectrum is equal to the contemporaneous covariance matrix of y_t. That is,

$$\Sigma_y(0) = \int_{-\pi}^{\pi} s_y(\omega)d\omega. \tag{13.4}$$

For a diagonal element of the population spectrum of the observed variables we can therefore consider 2 times the area between 0 and ω to represent the share of the variance of the corresponding element of y that can be attributed to periodic random components with frequency less than or equal to ω.

The conditional covariance of the state variables in the state-space model are given in equation (11.48), while the conditional covariance of the observed variables are shown in equation (11.49). The covariance matrix of the state variables, Σ_ξ, is related to the conditional covariance of these variables through

$$\Sigma_\xi = \sum_{j=1}^{q} \Sigma_\xi^{(j)}. \tag{13.5}$$

This means that the contemporaneous covariance matrix of the observed variables, $\Sigma_y(0)$, can be expressed as

$$\Sigma_y(0) = H'\left(\sum_{j=1}^{q} \Sigma_\xi^{(j)}\right)H + R. \tag{13.6}$$

Let $s_\xi(\omega)$ be the population spectrum of the state variables, while $s_\xi^{(j)}(\omega)$ is the spectrum of the state variables when all shocks are zero excepts η_j for $j = 1, \ldots, q$. Since the state-space model is linear and the shocks and measurement errors are uncorrelated, the spectrum of the state variables is equal to the sum of these spectra; see, e.g., Fuller (1976, Theorem 4.4.1) for a generalization of this property. That is,

$$s_\xi(\omega) = \sum_{j=1}^{q} s_\xi^{(j)}(\omega). \tag{13.7}$$

Moreover, let $s_w(\omega)$ denotes the spectrum of the measurement errors. Since these errors are just white noise, it follows that $s_w(\omega) = (1/2\pi)R$ is constant.

The population spectrum of the observed variables is equal to the weighted sum of the spectrum of the state variables plus the spectrum of the measurement errors. Specifically,

$$s_y(\omega) = H's_\xi(\omega)H + \frac{1}{2\pi}R = H'\sum_{j=1}^{q} s_\xi^{(j)}(\omega)H + \frac{1}{2\pi}R. \tag{13.8}$$

Combining this with the results on the conditional covariances above it follows that

$$H'\Sigma_\xi^{(j)}H = \int_{-\pi}^{\pi} H's_\xi^{(j)}(\omega)Hd\omega.$$

The matrix $H's_\xi^{(j)}(\omega)H$ is the spectrum of the observed variables when there are no measurement error and when all shocks are zero expect for η_j. From equation (13.4) it follows for a diagonal element of $H's_\xi^{(j)}(\omega)H$ that 2 times the area under this spectrum is equal to the share of the contemporaneous variance of the corresponding element of the observed variables that are due to state shock j.

13.3. *Estimating the Population Spectrum for a State-Space Model*

From equation (5.42) we know that if F has all eigenvalues inside the unit circle, then the autocovariances of the state-space model exist and can be expressed as

$$\Sigma_y(h) = \begin{cases} H'\Sigma_\xi H + R, & \text{if } h = 0, \\ H'F^h\Sigma_\xi H, & \text{for } h = 1, 2, \ldots \end{cases} \tag{13.9}$$

In addition, it is straightforward to show that $\Sigma_y(h) = \Sigma_y(-h)'$ for $h = -1, -2, \ldots$. Since all the eigenvalues of F lie inside the unit circle, the autocovariances of the state-space model are also absolutely summable.

To estimate the population spectrum using the state-space model we may therefore use the values of $\Sigma_y(h)$ from (13.9) for a given θ and substitute them into, e.g., equation (13.2) for $h = 0, 1, \ldots, \bar{h}$. The integer $\bar{h}$ may be selected such that $\Sigma_y(\bar{h})$ is sufficiently close to zero. This yields a parametric estimate of the population spectrum.

However, there is a more efficient use of the state-space model which allows us to not only compute the spectrum with greater numerical accuracy, but also at a higher computational speed. To this end, we express the state equation using the lag operator as

$$F(L)\xi_t = B_0\eta_t,$$

where $F(z) = I_r - Fz$ is invertible for all $|z| \geq 1$ since all the eigenvalues of F lie inside the unit circle. The population spectrum for a VAR model is well known (see, e.g., Hamilton, 1994, Chapter 10) and can for the state equation with $z = \exp(-i\omega)$ be expressed as[115]

$$s_\xi(\omega) = \frac{1}{2\pi}F\big(\exp(-i\omega)\big)^{-1}B_0B_0'\left(F\big(\exp(i\omega)\big)'\right)^{-1}. \tag{13.10}$$

From this expression we can also infer that the spectrum of the state variables when all shocks are zero except η_j is given by

$$s_\xi^{(j)}(\omega) = \frac{1}{2\pi}F\big(\exp(-i\omega)\big)^{-1}B_{0j}B_{0j}'\left(F\big(\exp(i\omega)\big)'\right)^{-1}, \quad j = 1, \ldots, q. \tag{13.11}$$

Notice that the property in equation (13.7) can be seen to hold from equations (13.10) and (13.11) since $B_0B_0' = \sum_{j=1}^{q} B_{0j}B_{0j}'$.

13.4. *Estimating the Population Spectrum from the Observed Data*

Instead of using the structure of the state-space model directly, we can simulate data from the model and estimate the population spectrum with non-parametric methods. This has the advantage that we can directly compare the spectrum for the simulated data to the spectrum for the actual observed data since the same estimation method may be used.

Suppose we have simulated T observations from the state-space model and estimated the autocovariances from

$$\hat{\Sigma}_y(h) = \frac{1}{T}\sum_{t=h+1}^{T}\big(y_t - \bar{y}_t\big)\big(y_{t-h} - \bar{y}_{t-h}\big)', \quad h = 0, 1, \ldots, \bar{h},$$

where $\bar{h} < T$ and

$$\bar{y}_t = \sum_{t=1}^{T} y_t x_t'\left(\sum_{t=1}^{T} x_t x_t'\right)^{-1} x_t.$$

Let κ_h be a symmetric function of h ($\kappa_h = \kappa_{-h}$) that is used to weight the autocovariances of the spectrum. A family of non-parametric estimators of the population spectrum may therefore be expressed as:

$$
\hat{s}_y(\omega) = \frac{1}{2\pi}\Bigg[\hat{\Sigma}_y(0) + \sum_{h=1}^{\bar{h}}\kappa_h\big(\hat{\Sigma}_y(h) + \hat{\Sigma}_y(h)'\big)\cos(\omega h)
$$
$$
\tag{13.12}
$$
$$
- i\sum_{h=1}^{\bar{h}}\kappa_h\big(\hat{\Sigma}_y(h) - \hat{\Sigma}_y(h)'\big)\sin(\omega h)\Bigg].
$$

---

[115] Notice that $\exp(-i\omega)$ takes us along the unit circle from 1 to -1 for the real part and from 0 into negative values and back to 0 for the imaginary part as ω goes from 0 to π. Similarly, $\exp(i\omega)$ traces out the positive region of the unit circle for the imaginary part as ω goes from 0 to π, while the real part again begins at 1 and ends at -1. That is, $\exp(i\omega)$ and $\exp(-i\omega)$ are mirror images along the unit circle.

One popular estimate of the spectrum uses the modified Bartlett kernel, which is given by

$$\kappa_h = \frac{\bar{h} + 1 - h}{\bar{h} + 1}, \quad h = 1, 2, \ldots, \bar{h}.$$

From a practical perspective, the main problem with the non-parametric estimator in (13.12) is how to choose $\bar{h}$; see, e.g, Hamilton (1994) for additional discussions. For the selected value of $\bar{h}$ we can compare the estimated population spectrum for the simulated data to the estimated population spectrum for the observed data. With simulated data we may consider, say, S number of simulated paths per parameter value for θ and P different parameter values from the posterior distributions, thus yielding an estimate of the posterior distribution of the non-parametric estimate of the population spectrum.

13.5. *Filters*

The population spectra for the state-space model and the VAR model are special cases of the filter $v_t = A(L)w_t$, where w_t is a covariance stationary vector time series of dimension k and the $m \times k$ polynomial

$$A(z) = \sum_{j=-\infty}^{\infty} A_j z^j,$$

is absolutely summable. Assuming that the population spectrum of w_t is given by $s_w(\omega)$, the population spectrum of the m dimensional time series v_t is

$$s_v(\omega) = A\big(\exp(-i\omega)\big)s_w(\omega)A\big(\exp(i\omega)\big)', \quad \omega \in [-\pi, \pi], \tag{13.13}$$

see, e.g., Fuller (1976, Theorem 4.4.1) and Hamilton (1994, Chapter 10). A very simple example of such a filter is when we wish to transform a vector w_t from quarterly differences, $(1 - z)$, to annual differences, $(1 - z^4)$. This means that $A(z) = (1 + z + z^2 + z^3)I_k$ so that v_t is the sum of the current and previous 3 quarterly changes.

13.6. *Coherence*

Let $s_{y_k, y_l}(\omega)$ denote the element in row k and column l of $s_y(\omega)$. The *coherence* between y_k and y_l is given by

$$R^2_{y_k, y_l}(\omega) = \frac{s_{y_k, y_l}(\omega)s_{y_k, y_l}(-\omega)}{s_{y_k, y_k}(\omega)s_{y_l, y_l}(\omega)} = \frac{|s_{y_k, y_l}(\omega)|^2}{s_{y_k, y_k}(\omega)s_{y_l, y_l}(\omega)}, \quad \omega \in [0, \pi], \tag{13.14}$$

where $|a|$ denotes the modulus of a and the terms in the denominator are assumed to be non-zero. In the event that one of them is zero the statistic is zero. It can be shown that $0 \leq R^2_{y_k, y_l}(\omega) \leq 1$ for all ω as long as y is covariance stationary with absolutely summable autocovariances; see, e.g., Fuller (1976, p. 156). The coherence statistic thus measures the squared correlation between y_k and y_l at frequency ω, i.e., $R^2_{y_k, y_l}(\omega) = R^2_{y_l, y_k}(\omega)$ for all pairs (y_k, y_l).

We can similarly define a coherence statistic for pairs of state variables in the state-space model. Letting $s_{\xi_k, \xi_l}(\omega)$ denote the element in row k and column l of the population spectrum of the state variables, $s_\xi(\omega)$, it follows that

$$R^2_{\xi_k, \xi_l}(\omega) = \frac{|s_{\xi_k, \xi_l}(\omega)|^2}{s_{\xi_k, \xi_k}(\omega)s_{\xi_l, \xi_l}(\omega)}, \quad \omega \in [0, \pi],$$

is the coherence between ξ_k and ξ_l. Again we note that the statistic is zero whenever one of the terms in the denominator is zero.

Finally, it may be noted that if $q = 1$ and $R = 0$, then the coherence between two observed variables is either zero or one. Hence, there is not any point in computing conherence statistics conditional on all shocks being zero except one.

13.7. *Gain and Phase*

It was mentioned in Section 13.1 that the real part of the cross-spectrum is called the co-spectrum, while the imaginary part is called the quadrature spectrum. Letting $c_{y_k,y_l}(\omega)$ denote the former and $q_{y_k,y_l}(\omega)$ the latter, we have that

$$s_{y_k,y_l}(\omega) = c_{y_k,y_l}(\omega) + i q_{y_k,y_l}(\omega), \quad k,l = 1,\ldots,n, \ k \neq l.$$

The cross-spectrum may be rewritten as

$$s_{y_k,y_l}(\omega) = a_{y_k,y_l}(\omega) \exp\left(i\varphi_{y_k,y_l}(\omega)\right), \tag{13.15}$$

where $a_{y_k,y_l}(\omega) = |s_{y_k,y_l}(\omega)|$ is called the *cross amplitude spectrum* and $\varphi_{y_k,y_l}(\omega)$ the *phase spectrum*; see, e.g., Fuller (1976, p. 159) or Sargent (1987, p. 269). The latter object can be computed from

$$\varphi_{y_k,y_l}(\omega) = \arctan\left[\frac{q_{y_k,y_l}(\omega)}{c_{y_k,y_l}(\omega)}\right], \tag{13.16}$$

where arctan denotes the arctangent, i.e., the inverse tangent.[116] The *gain* of y_l over y_k can be defined as

$$\psi_{y_k,y_l}(\omega) = \frac{a_{y_k,y_l}(\omega)}{s_{y_k,y_k}(\omega)}, \tag{13.17}$$

for those ω where $s_{y_k,y_k}(\omega) > 0$. The gain is sometimes also defined as the numerator in (13.17); see, e.g., Sargent (1987) or Christiano and Vigfusson (2003). The cross amplitude tells how the amplitude in y_l is multiplied in contributing to the amplitude of y_k at frequency ω. Similarly, the phase statistic gives the lead of y_k over y_l at frequency ω. Specifically, if $\varphi_{y_k,y_l}(\omega) > 0$, then y_k leads y_l by $\varphi_{y_k,y_l}(\omega)/\omega$ periods, while $\varphi_{y_k,y_l}(\omega) < 0$ means that y_l leads y_k by $-\varphi_{y_k,y_l}(\omega)/\omega$ periods.

However, the phase is not unique since there are multiple values of $\varphi_{y_k,y_l}(\omega)$ that satisfy equation (13.15) for each $\omega \in [-\pi,\pi]$. For example, if $\varphi_{y_k,y_l}(\omega)$ solves (13.15), then so does $\varphi_{y_k,y_l}(\omega) + 2\pi h$ for $h = 0, \pm 1, \pm 2, \ldots$. In other words, the lead and the lag between two sinusoidal functions with the same period (T/j) is ill-defined.

As a method for resolving the ambiguity in characterizing the lead-lag relationship between variables, Christiano and Vigfusson (2003) proposes an alternative approach. From equation (13.3) we find that

$$\Sigma_{y_k,y_l}(h) = \int_{-\pi}^{\pi} s_{y_k,y_l}(\omega) \exp\left(i\omega h\right) d\omega. \tag{13.18}$$

Using the fact that the population spectrum is Hermitian,[117] equation (13.15) and De Moivre's theorem, Christiano and Vigfusson (2003) show that

$$
\begin{aligned}
\Sigma_{y_k,y_l}(h) &= \int_{0}^{\pi} \left[s_{y_k,y_l}(\omega) \exp\left(i\omega h\right) + s_{y_k,y_l}(-\omega) \exp\left(-i\omega h\right) \right] d\omega \\
&= \int_{0}^{\pi} 2 a_{y_k,y_l}(\omega) \cos\left[\varphi_{y_k,y_l}(\omega) + \omega h\right] d\omega \\
&= \int_{0}^{\pi} \Sigma_{y_k,y_l}(h,\omega) d\omega.
\end{aligned}
\tag{13.19}
$$

---

[116] Recall that the tangent is given by $\tan(x) = \sin(x)/\cos(x)$. The inverse tangent can be defined as

$$\arctan(x) = \tan^{-1}(x) = \frac{i}{2} \ln\left(\frac{i+x}{i-x}\right),$$

where $\arctan(x) \in [-\pi/2, \pi/2]$ for real valued x. The inverses of the trigonometric functions do not meet the usual requirements of inverse functions since their ranges are subsets of the domains of the original functions. The latter is a consequence of, e.g., the cosine function being periodic such that $\cos(0) = \cos(\pi)$, and so on. This means that an inverse of the cosine function will deliver multiple values, unless its range is restricted to a certain subset. For values of x within this subset, the usual requirement of inverse functions apply.

[117] For the cross amplitude spectrum and the phase spectrum it follows from this property that $a_{y_k,y_l}(\omega) = a_{y_k,y_l}(-\omega)$ and $\varphi_{y_k,y_l}(-\omega) = -\varphi_{y_k,y_l}(\omega)$, respectively.

where $\Sigma_{y_k,y_l}(h,\omega)$ is the covariance between the component of $y_{k,t}$ at frequency ω and the component of $y_{l,t-h}$ at frequency ω.

Christiano and Vigfusson note that it is common to characterize the lead-lag reation between two variables by the value of h for which $\Sigma_{y_k,y_l}(h)$ is the largest (in absolute terms). From (13.19) it can be seen that $\Sigma_{y_k,y_l}(h,\omega)$ is maximized for $h = -\varphi_{y_k,y_l}(\omega)/\omega$, since $\cos(0)$ is the unique maximum of the cosine function.

To resolve the ambiguity of $\varphi_{y_k,y_l}(\omega)$, Christiano and Vigfusson suggest that phase is chosen such that

$$h_{y_k,y_l}(\omega,\Delta)^* = \arg\max_h \int_{\omega-\Delta}^{\omega+\Delta} \Sigma_{y_k,y_l}(h,\omega)d\omega, \quad \pi - \Delta \geq \omega \geq \Delta > 0, \qquad (13.20)$$

where $\varphi_{y_k,y_l}(\omega,\Delta)^* = -\omega h_{y_k,y_l}(\omega,\Delta)^*$. The measure of phase suggested by Christiano and Vigfusson is now given by $\varphi_{y_k,y_l}(\omega)^* = \lim_{\Delta\to 0} \varphi_{y_k,y_l}(\omega,\Delta)^*$, or, equivalently, $h_{y_k,y_l}(\omega)^* = \lim_{\Delta\to 0} h_{y_k,y_l}(\omega,\Delta)^*$ when measured in time units. In practise, this seems to imply that Δ is a small positive number (since $\Delta = 0$ means that $h_{y_k,y_l}(\omega)^* = -\varphi_{y_k,y_l}(\omega)/\omega$) while h is taken from a suitable interval, say, (L, U).

13.8. *Fisher's Information Matrix in the Frequency Domain*

Whittle (1953) was the first to present a frequency domain expression of Fisher's information matrix. His results are based on a stationary vector time series with zero mean and they have since been given a solid mathematical treatment by, e.g., Walker (1964), Dunsmuir and Hannan (1976), Deistler, Dunsmuir, and Hannan (1978), and Dunsmuir (1979); see also Harvey (1989). More recent investigations of the matrix are provided by, e.g., Klein and Spreij (2006, 2009).

Let ϑ be the p-dimensional vector of parameters that can affect the population spectrum. This means that we here exclude parameters in θ that *only* affect the mean. For example, in the An and Schorfheide model the parameters $(\gamma^{(Q)}, \pi^{(A)})$ are excluded from ϑ, while the remaining 11 parameters in θ are included in ϑ.

Whittle's estimator of the information matrix can be expressed as

$$\mathcal{I}_\vartheta = \frac{T}{4\pi} \int_{-\pi}^{\pi} K(\vartheta,\omega)d\omega, \qquad (13.21)$$

where

$$K_{k,l}(\vartheta,\omega) = \mathrm{tr}\left[s_y(\omega)^{-1} \frac{\partial s_y(\omega)}{\partial \vartheta_k} s_y(\omega)^{-1} \frac{\partial s_y(\omega)}{\partial \vartheta_l} \right], \quad k, l = 1, \ldots, p. \qquad (13.22)$$

Using the relationship between the trace and the vec operator (see Magnus and Neudecker, 1988, Theorem 2.3) it can be shown that

$$K(\vartheta,\omega) = \left(\frac{\partial \mathrm{vec}\left(s_y(\omega)'\right)}{\partial \vartheta'} \right)' \left[\left(s_y(\omega)'\right)^{-1} \otimes s_y(\omega)^{-1} \right] \frac{\partial \mathrm{vec}\left(s_y(\omega)\right)}{\partial \vartheta'}. \qquad (13.23)$$

Recall that $s_y(\omega)$ is not symmetric, but Hermitian. Note also that the transpose used in (13.23) is *not* the complex conjugate but the standard transpose.[118] It is now straightforward to show that $K(\vartheta,\omega)$ is Hermitian, i.e., $K(\vartheta,\omega) = K(\vartheta,-\omega)'$. It follows that $\mathcal{I}_\vartheta$ in (13.21) is real and symmetric since $K(\vartheta,\omega) + K(\vartheta,-\omega)$ is real and symmetric for each $\omega \in (0,\pi]$, while $K(\vartheta,0)$ is real and symmetric since $s_y(0)$ is; see equation (13.2).

It is important to note that the population spectrum need not be invertible at all frequencies. In particular, it may be the case that it is singular at frequency zero (the long-run covariance matrix is singular). YADA checks such cases and uses the rule that $K(\vartheta,\omega)$ is *not* computed for the frequencies where the population spectral density is singular.

The population spectrum of the state-space model is given by equations (13.8) and (13.10). Since we exclude parameters that concern the mean of the observed variables, we let all partial

---

[118] Recalling footnote 18, the complex conjugate or conjugate transpose for a complex matrix $A = B + iC$ is equal to $A^* = B' - iC'$. This means that a matrix A is Hermitian if $A = A^*$. The standard transpose of A is instead given by $A' = B' + iC'$.

derivatives with respect to the elements of A be equal to zero. Next, using the matrix differential rules in Magnus and Neudecker (1988) it can be shown that

$$\frac{\partial \text{vec}\big(s_y(\omega)\big)}{\partial \text{vec}\big(R\big)'} = \frac{1}{2\pi} I_{n^2}. \tag{13.24}$$

With K_{nn} being defined as the $n^2 \times n^2$ dimensional commutation matrix,[119] it can be shown that

$$\frac{\partial \text{vec}\big(s_y(\omega)\big)}{\partial \text{vec}\big(H\big)'} = K_{nn}\left[I_n \otimes H' s_\xi(\omega)'\right] + \left[I_n \otimes H' s_\xi(\omega)\right]. \tag{13.25}$$

Letting $N_r = (1/2)(I_{r^2} + K_{rr})$, as in Magnus and Neudecker (1988, Theorem 3.11), it can be shown that

$$\frac{\partial \text{vec}\big(s_y(\omega)\big)}{\partial \text{vec}\big(B_0\big)'} = \frac{1}{\pi}\left[H'F\big(\exp(i\omega)\big)^{-1} \otimes H'F\big(\exp(-i\omega)\big)^{-1}\right] N_r\left(B_0 \otimes I_r\right). \tag{13.26}$$

Furthermore, the partial derivatives with respect to the state transition matrix is given by

$$\frac{\partial \text{vec}\big(s_y(\omega)\big)}{\partial \text{vec}\big(F\big)'} = \left[H's_\xi(\omega)' \otimes H'F\big(\exp(-i\omega)\big)^{-1}\right]\exp\big(-i\omega\big)$$
$$+ K_{nn}\left[H's_\xi(\omega) \otimes H'F\big(\exp(i\omega)\big)^{-1}\right]\exp\big(i\omega\big). \tag{13.27}$$

The next step is to determine the partial derivatives of (R, H, F, B_0) with respect to ϑ and, using the chain rule, postmultiply the matrices on the right hand side of equations (13.24)–(13.27) by $\partial \text{vec}(R)/\partial \vartheta'$, $\partial \text{vec}(H)/\partial \vartheta'$, and so on. The partial derivatives of the reduced form parameters of the state-space model with respect to the structural parameters can either be achieved numerically or analytically; see Iskrev (2008). To determine which elements of θ that are included in ϑ, we may simply check which elements of θ that have a non-zero column in at least one of the matrices with partial derivatives $\partial \text{vec}(M)/\partial \theta'$, with $M = R, H, F, B_0$. To obtain $\partial \text{vec}(M)/\partial \vartheta'$ we remove the columns of $\partial \text{vec}(M)/\partial \theta'$ that correspond to the elements of θ that do *not* meet this condition. Finally, the partial derivative of $s_y(\omega)'$ with respect to ϑ is obtained by premultiplying the above matrices of partial derivatives by the commutation matrix, i.e., we use $\text{vec}(s_y(\omega)') = K_{nn}\text{vec}(s_y(\omega))$.

A frequency domain expression of Fisher's information matrix may also be derived as in Harvey (1989, Chapters 4 and 8). When following his approach, it may be expressed as

$$\mathcal{I}_\vartheta = \frac{1}{2}\sum_{j=0}^{T-1} K(\vartheta, \omega_j), \tag{13.28}$$

where $\omega_j = 2\pi j/T$ as above. In practise we may use either (13.21) or (13.28) as the information matrix, but the numerical integration needed for the Whittle's expression is likely to give a better approximation of $\mathcal{I}_\vartheta$ than the right hand side of (13.28). The reason is that a smaller numerical representation of $d\omega$ may be used in (13.21) than the value $\Delta \omega_j = 2\pi/T$ used in (13.28).[120]

13.9. *YADA Code*

The decomposition of the population spectrum for the state-space model is undertaken by the function `DSGESpectralDecompTheta`. This function works for a fixed value of the parameter vector θ and computes the decomposition for either the observed variables or for the state variables.

---

[119] For any $m \times n$ matrix A, the $mn \times mn$ dimensional commutation matrix is defined from $\text{vec}(A) = K_{nm}\text{vec}(A')$, with $K_{mn}K_{nm} = I_{mn}$; see Magnus and Neudecker (1988, Chapter 3) for some properties of this matrix.

[120] It may be noted that the information matrix in (13.28) is based on an asymptotic approximation of the log-likelihood function when transforming it from the time domain to the frequency domain; see Harvey (1989, Chapter 4.3) for details.

When computing spectral densities for the observed variables, the function can also deal with annualized variables. For variables in first differences of quarterly observations the annualization filter is $A(z) = 1 + \sum_{j=1}^{3} z^j$, while for variables in first differences of monthly observations the filter is $A(z) = 1 + \sum_{j=1}^{11} z^j$. The filter computation is taken care of by the function `AnnualFilter`.

The function `DSGEPopulationSpectrum` computes the population spectrum for a fixed parameter value, while `DSGECoherenceTheta` computes the coherence statistic either for the observed variables or for the state variables. Whittle's estimator of the information matrix in (13.21) is calculated by `DSGEFrequencyDomainInfMat` for $T = 1$.

13.9.1. DSGESpectralDecompTheta

The function `DSGESpectralDecompTheta` requires 6 input variables: `theta`, `thetaPositions`, `ModelParameters`, `VarStr`, `DSGEModel`, and `CurrINI`. The first three and last two variables have often appeared above; see, e.g., `CalculateDSGEStateVariables` in Section 11.18.3. The 4th input variable `VarStr` is a string that supports the values 'Observed Variables' and 'State Variables', thus indicating if spectral density decompositions should be performed for observed variables or state variables.

The function provides the output variable `SpecDec`, a structure whose fields depend on the value of `VarStr`. In addition, the function can provide output on `status`, the `mcode` output variable from the DSGE model solving function that has been used, and `kalmanstatus`, the `status` output from the Kalman filter.

For spectral decompositions of observed variables, the `SpecDec` structure has at least 13 fields and at most 18. The additional 5 fields are all related to annualization of the data. The latter operation can only be performed if the vector stored in the field `annual` to the `DSGEModel` input variable has at least one value which is greater than unity. Specifically, a value of 4 means that the corresponding observed variable is annualized by adding the current and previous 3 values while a value of 12 means that the variables is annualized by adding the current and previous 11 values.

The 5 fields with annualized spectral densities are called `AnnualY`, `AnnualMeasureError`, `AnnualYStates`, `AnnualYShocks`, and `AnnualVariance`. The first three hold f dimensional cell arrays with $n \times n$ matrices, where f is the number of used frequencies between 0 and π. The spectral density for the annualized observed variables are located in the `AnnualY` cell array, the term related to the measurement errors in `AnnualMeasureError`, and the total influence of the states variables in `AnnualYStates`. The field `AnnualYShocks` is a $q \times f$ cell array with $n \times n$ matrices with the influence of the individual economic shocks in the rows of the cell array. Finally, the field `AnnualVariance` is a vector with the population variances of the observed variables.

Among the always present fields are `Frequencies`, `Y`, `States`, `Shocks`, `MeasureError`, and `OriginalVariance`. The first is an f dimensional vector with the frequencies that have been used for the spectral densities. The value of f is 300 with $\omega = 0, \pi/299, 2\pi/299, \ldots, \pi$. The following three fields are cell arrays of dimension f, f and $q \times f$, respectively, with the spectral densities for the observed variables, the share of all the state variables, and the shares due to the individual shocks. The latter two are given by $H's_\xi(\omega)H$ and $H's_\xi^{(j)}(\omega)H$; cf. equations (13.10)–(13.11). The field `MeasureError` is an $n \times n$ matrix with the frequency independent influence of the measurement errors on the spectral density, i.e., $(1/2\pi)R$, while the field `OriginalVariance` is a vector with the population variances of the observed variables.

Three fields are further related to the population covariance of the observed variables. They are called `SigmaY`, `SigmaXi`, and `R`. The first is the population covariance matrix itself, while the second field is the share due to the state variables, and the third the share due to the measurement errors.

Furthermore, the structure `SpecDec` has 4 fields that hold data for using shocks groups: `ShockNames`, `ShockGroups`, `ShockGroupNames`, and `ShockGroupColors`. The first and the third

are string matrices where the rows hold the names of the shocks and the shock groups, respectively, where the number of rows is q for the shocks and g for the shock groups, with $q \geq g$. The second field is a vector of dimension q with integers that map each shock to a certain shock group, while the last field is a $g \times 3$ matrix, where each row gives the color as an RGB triple for a shock group. The RGB triple holds values between 0 and 1, representing the combination of red, green and blue, and this scale can be translated into the more common 8-bit scale that is used to represent colors with integer values between 0 and 255.

For spectral decompositions of the state variables, the number of fields in the `SpecDec` structure is smaller. Among the 13 fields that are always present for the observed variables, 4 are no longer available for the spectral decomposition of the state variables. The missing fields are: `Y`, `MeasureError`, `SigmaY`, and `R`. Regarding the cell arrays, the dimension of the matrices stored in each cell is now $r \times r$. This means that, for example, the `States` field is an f dimensional cell array of the $r \times r$ matrices $s_\xi(\omega)$. Moreover, the field `OriginalVariance` is a vector with the population variances of the state variables, while `SigmaXi` is the corresponding $r \times r$ population covariance matrix.

13.9.2. DSGEPopulationSpectrum

The function `DSGEPopulationSpectrum` computes the population spectrum for a set of frequencies $\omega \in [0, \pi]$ using 4 input variables: `H`, `R`, `F`, `B0`. These variables correspond to the H, R, F, and B_0 matrices of the state-space model.

The function provides 7 output variables. The first is given by the cell array `SyOmega` that has 300 entries. Each element in the array is equal to the $n \times n$ matrix $s_y(\omega_j)$ for a given frequency $\omega_j = \pi(j - 1)/299$, $j = 1, \ldots, 300$. The second variable, `Omega`, is a vector of length 300 with the different frequencies ω_j, while the following, `SxiOmega`, is a cell array with 300 entries. This time the array holds the $r \times r$ matrices $s_\xi(\omega_j)$.

All the remaining output variables concerns input for the partial derivatives in (13.26) and (13.27). The variables `Fi` and `Fmi` are cell arrays with 300 elements that hold the inverse of $F(\exp(i\omega_j))^{-1}$ and $F(\exp(-i\omega_j))^{-1}$, respectively, while `ei` and `emi` are vectors with 300 elements, where each entry is $\exp(i\omega_j)$ and $\exp(-i\omega_j)$.

13.9.3. DSGECoherenceTheta

The function `DSGECoherenceTheta` requires 6 input variables to compute coherence for a fixed value of the DSGE model parameters: `theta`, `thetaPositions`, `ModelParameters`, `VarType`, `DSGEModel`, and `CurrINI`. The only unfamiliar variable is `VarType`, a boolean variable which is unity if coherence for the observed variables should be computed and zero if state variables have been selected.

As output, the function provides `SOmega` and `Omega`. The former is a matrix of dimension $m(m - 1)/2 \times f$, where m is the number of variables, i.e., equal to n for the observed variables and to r for the state variables. The dimension f is equal to the number of frequencies between 0 and π. The `Omega` variable is a vector of dimension f with the frequency values ω_j. Following the convention used by the `DSGEPopulationSpectrum` function above, $f = 300$ so that $\omega_j = \pi(j - 1)/299$ for $j = 1, \ldots, 300$.

13.9.4. DSGEFrequencyDomainInfMat

The function `DSGEFrequencyDomainInfMat` takes the same input variables as the time domain function `DSGEInformationMatrix` (see Section 11.18.10). That is, it accepts the 6 variables: `theta`, `thetaPositions`, `ModelParameters`, `ParameterNames`, `DSGEModel`, and `CurrINI`.

As output the function returns `informationMatrix`, the $p \times p$ information matrix in (13.21) with $T = 1$. The dimension p is equal to the number of entries in θ that affect the population spectrum, i.e., the number of entries that affect at least one of the state-space matrices H, R, F, B_0. The positions of these parameters are provided in the output variable `ParamIncluded`, while the third output variable, `ParamExcluded`, holds the positions in θ of the parameters that do not have an effect on the above state-space matrices (and, hence, only concern the A matrix).

14. Bayesian VAR Analysis

Bayesian VAR models can be analysed with YADA. Their main purpose is to provide a competitor when performing multistep out-of-sample forecasts with a DSGE model. Let x_t be a p dimensional covariance stationary vector stochastic process which satisfies the dynamic relation:

$$x_t = \Psi d_t + \sum_{l=1}^{k} \Pi_l (x_{t-l} - \Psi d_{t-l}) + \varepsilon_t, \quad t = 1, \dots, T. \tag{14.1}$$

The vector d_t is deterministic and assumed to be of dimension q. The residuals ε_t are assumed to be iid Gaussian with zero mean and positive definite covariance matrix Ω. The Π_l matrix is $p \times p$ for all lags, while Ψ is $p \times q$ and measures the expected value of x_t conditional on the parameters and other information available at $t = 0$. All Bayesian VAR models that are supported by YADA have an informative prior on the Ψ parameters, the steady state of x_t. Moreover, the elements of the vector x_t (d_t) are all elements of the vector y_t (x_t) in the measurement equation of the DSGE model. It is hoped that this notational overlap will not be confusing for you.

14.1. *The Prior*

The setup of the VAR model in (14.1) is identical to the stationary VAR process in mean-adjusted form that Villani (2009) examines. The prior on the steady state Ψ is also the same as that considered in his paper and used by, e.g., Adolfson, Anderson, Lindé, Villani, and Vredin (2007a). That is, with $\psi = \text{vec}(\Psi)$ I assume that the marginal prior is given by

$$\psi \sim N_{pq}(\theta_\psi, \Sigma_\psi), \tag{14.2}$$

where Σ_ψ is positive definite. YADA allows the user to select any values for θ_ψ and for the diagonal of Σ_ψ. The off-diagonal elements of the prior covariance matrix are assumed to be zero.

Let $\Pi = [\Pi_1 \cdots \Pi_k]$ be the $p \times pk$ matrix with parameters on lagged x. The prior distributions for these parameters that YADA supports are as follows:

(i) a Minnesota-style prior similar to the one considered by Villani (2009);

(ii) a normal conditional on the covariance matrix of the residuals (see, e.g., Kadiyala and Karlsson, 1997); and

(iii) a diffuse prior.

I will address the details about each prior distribution below.

First, for the Minnesota-style prior the marginal prior distribution of Π is given by:

$$\text{vec}(\Pi) \sim N_{p^2 k}(\theta_\pi, \Sigma_\pi), \tag{14.3}$$

where the prior mean θ_π need not be unity for the first own lagged parameters and zero for the remaining. In fact, the general setup considers x_t to be stationary with steady state determined by the prior mean of Ψ and d_t.

Let $\theta_\pi = \text{vec}(\mu_\pi)$, where $\mu_\pi = [\mu_{\Pi_1} \cdots \mu_{\Pi_k}]$ is a $p \times pk$ matrix with the prior mean of Π. The assumption in YADA is that $\mu_{\Pi_l} = 0$ for $l \geq 2$, while μ_{Π_1} is diagonal. The diagonal entries are determined by two hyperparameters, γ_d and γ_l. With $\Pi_{ii,1}$ being the i:th diagonal element of Π_1, the prior mean of this parameter, denoted by μ_{ii,Π_1}, is equal to γ_d if variable i in the x_t vector is regarded as being first differenced (e.g., output growth), and γ_l if variable i is in levels (e.g., the nominal interest rate).

The Minnesota feature of this prior refers to the covariance matrix Σ_π. Let $\Pi_{ij,l}$ denote the element in row (equation) i and column (on variable) j for lag l. The matrix Σ_π is here assumed to be diagonal with

$$\text{Var}(\Pi_{ij,l}) = \begin{cases} \dfrac{\lambda_o}{l^{\lambda_h}}, & \text{if } i = j, \\ \dfrac{\lambda_o \lambda_c \Omega_{ii}}{l^{\lambda_h} \Omega_{jj}}, & \text{otherwise.} \end{cases} \tag{14.4}$$

The parameter Ω_{ii} is simply the variance of the residual in equation i and, hence, the ratio Ω_{ii}/Ω_{jj} takes into account that variable i and variable j may have different scales.

Formally, this parameterization is inconsistent with the prior being a marginal distribution since it depends on Ω. YADA tackles this in the standard way by replacing the Ω_{ii} parameters with the maximum likelihood estimate. The hyperparameter $\lambda_o > 0$ gives the overall tightness of the prior around the mean, while $0 < \lambda_c < 1$ is the cross-equation tightness hyperparameter. Finally, the hyperparameter $\lambda_h > 0$ measures the harmonic lag decay.

Second, when the prior distribution of Π is no longer marginal but conditional on the covariance matrix of the residuals we use the following:

$$\text{vec}(\Pi)|\Omega \sim N_{p^2k}\left(\theta_\pi, \Omega_\pi \otimes \Omega\right), \tag{14.5}$$

where Ω_π is a positive definite $pk \times pk$ matrix, while θ_π is determined in exactly the same way as for the Minnesota-style prior above. A prior of this generic form is, for instance, examined by Kadiyala and Karlsson (1997), where it is also discussed relative to, e.g., the Minnesota prior. The matrix Ω_π is assumed to be block diagonal in YADA, where block $l = 1, \ldots, k$ (corresponding to Π_l) is given by

$$\Omega_{\pi_l} = \frac{\lambda_o}{l^{\lambda_h}} I_p. \tag{14.6}$$

Hence, the overall tightness as well as the harmonic lag decay hyperparameter enter this prior, while the cross-equation hyperparameter cannot be included. This is the price for using the Kronecker structure of the prior covariance matrix. At the same time, different scales of the variables are now handled by conditioning on Ω instead of using sample information.

Finally, in the case of the diffuse prior we simply assume that the prior density $p(\Pi) = 1$.

The marginal prior distribution for Ω is either assumed to be diffuse or inverted Wishart. Let the marginal prior density of Ω be denoted by $p(\Omega)$. In the former case, we simply make use of the standard formula (see, e.g., Zellner, 1971)

$$p(\Omega) \propto |\Omega|^{-(p+1)/2}. \tag{14.7}$$

In the latter case, the density function of Ω is proper and given in equation (4.12) in Section 4.2.4. Recall that the mode of the inverted Wishart is equal to $(1/(p+v+1))A$, while the mean exists if $v \geq p + 2$ and is then given by $E[\Omega] = (1/(v-p-1))A$; see, e.g., Zellner (1971, Appendix B.4) and Bauwens et al. (1999, Appendix A) for details.

The hyperparameter A can be selected in two ways in YADA. The first route is to let A equal the maximum likelihood estimate of Ω. This was suggested by, e.g., Villani (2005). The alternative is to let $A = \lambda_A I_p$, where the hyperparameter λ_A gives the joint marginal prior residual variance; see, e.g., Warne (2006). By selecting the degrees of freedom as small as possible (given finite first moments) the impact of the parameterization for A is minimized, i.e., by letting $v = p + 2$.[121]

Finally, it should be pointed out that the joint prior distribution of (Ψ, Π, Ω) satisfies certain independence conditions. Specifically, Ψ is assumed to be independent of Π and Ω. Under the Minnesota-style prior for Π it is also assumed that Π is a prior independent of Ω.

14.2. Posterior Mode

Before we turn to the estimation of the posterior mode we need to introduce some additional notation. Let x be a $p \times T$ matrix with x_t in column t, while ε is constructed in the same way. Similarly, let d be a $q \times T$ matrix with d_t in column t. Furthermore, let D be a $q(k+1) \times T$ matrix where column t is given by $[d_t' \; -d_{t-1}' \; \cdots \; -d_{t-k}']'$. Also, let $y = x - \Psi d$, while Y is a $pk \times T$ matrix where column t is given by $[y_{t-1}' \; \cdots \; y_{t-k}']'$ with $y_t = x_t - \Psi d_t$. Next, let X be a $pk \times T$ matrix where column t is $[x_{t-1}' \; \cdots \; x_{t-k}']'$. From this we can define $z = x - \Pi X$.

Hence, the stacked VAR may be expressed as either:

$$y = \Pi Y + \varepsilon, \tag{14.8}$$

or

$$z = \Theta D + \varepsilon, \tag{14.9}$$

---

[121] Given that the inverted Wishart prior has been selected for Ω and the normal conditional on Ω prior for Π, it follows by standard distribution theory that the marginal prior of Π is matricvariate t.

where $\Theta = [\Psi \ \Pi_1\Psi \ \cdots \ \Pi_k\Psi]$. Applying the vec operator on Θ gives us:

$$\text{vec}(\Theta) = \begin{bmatrix} I_{pq} \\ (I_q \otimes \Pi_1) \\ \vdots \\ (I_q \otimes \Pi_k) \end{bmatrix} \text{vec}(\Psi), \tag{14.10}$$

$$= U\text{vec}(\Psi).$$

The nonlinearity of the VAR model means that an analytical solution for the mode of the joint posterior distribution of (Ψ, Π, Ω) is not available. However, from the first order conditions we can express three systems of equations that the mode must satisfy, and by iterating on these equations it is possible to quickly solve for the posterior mode. Naturally, the choice of prior influences the three systems of equation.

First, the choice of prior for Π and Ω does not have any effect on the equations that Ψ has to satisfy at the mode conditional on Π and Ω. Here we find that

$$\psi = \left[U'(DD' \otimes \Omega^{-1})U + \Sigma_\psi^{-1}\right]^{-1}\left(U'\text{vec}(\Omega^{-1}zD') + \Sigma_\psi^{-1}\theta_\psi\right). \tag{14.11}$$

Second, in case the Minnesota-style prior is applied to Π, the posterior mode estimate must satisfy the system of equations

$$\text{vec}(\Pi) = \left[(YY' \otimes \Omega^{-1}) + \Sigma_\pi^{-1}\right]^{-1}\left(\text{vec}(\Omega^{-1}yY') + \Sigma_\pi^{-1}\theta_\pi\right). \tag{14.12}$$

Similarly, when a normal conditional on the residual covariance matrix prior is used for Π, then the posterior mode must satisfy:

$$\Pi = \left(yY' + \mu_\pi\Omega_\pi^{-1}\right)\left[YY' + \Omega_\pi^{-1}\right]^{-1}. \tag{14.13}$$

The system of equations that Π needs to satisfy when a diffuse prior is used on these parameters is, for instance, obtained by letting $\Omega_\pi^{-1} = 0$ in (14.13), i.e., $\Pi = yY'(YY')^{-1}$.

Third, in case a Minnesota-style prior is used on Π, then the posterior mode of Ω must satisfy:

$$\Omega = \frac{1}{T + p + \nu + 1}\left(\varepsilon\varepsilon' + A\right). \tag{14.14}$$

If the prior on Ω is diffuse, i.e., given by (14.7), we simply set $\nu = 0$ and $A = 0$ in (14.14).

Similarly, when the prior on Π is given by (14.5), then the posterior mode of Ω satisfies

$$\Omega = \frac{1}{T + p(k+1) + \nu + 1}\left(\varepsilon\varepsilon' + A + \left(\Pi - \mu_\pi\right)\Omega_\pi^{-1}\left(\Pi - \mu_\pi\right)'\right). \tag{14.15}$$

If the prior on Ω is diffuse we again let $\nu = 0$ and $A = 0$. Similarly, if the prior on Π is diffuse, we set $k = 0$ and $\Omega_\pi^{-1} = 0$ in (14.15).

14.3. Gibbs Samplers for a Bayesian VAR

The posterior samplers used by YADA for drawing from the posterior distribution of the parameters of the Bayesian VAR models that it supports are simple Gibbs samplers; see, e.g., Geman and Geman (1984), Casella and George (1992), Tierney (1994), or Geweke (1999, 2005). This means that the full conditional posterior distributions are needed for (Ψ, Π, Ω).[122]

The full conditional posterior distribution of Ψ is given by Villani (2009, Proposition A.1). Let $\mathfrak{D}_T = \{x_{1-k}, \ldots, x_0, x_1 \ldots, x_T, d_{1-k}, \ldots, d_0, d_1 \ldots, d_T\}$. We can now express this distribution as:

$$\psi|\Pi, \Omega, \mathfrak{D}_T \sim N_{pq}(\bar{\theta}_\psi, \bar{\Sigma}_\psi), \tag{14.16}$$

---

[122] The Gibbs sampler is a special case of the Metropolis-Hastings algorithm where the proposal density is equal to the full conditional posterior, with the effect that the accetance rate is always unity. The sampler was given its name by Geman and Geman (1984), who used it for analysing Gibbs distributions on lattices.

where $\bar{\Sigma}_\psi^{-1} = U'(DD' \otimes \Omega^{-1})U + \Sigma_\psi^{-1}$ and $\bar{\theta}_\psi = \bar{\Sigma}_\psi(U'\text{vec}(\Omega^{-1}zD') + \Sigma_\psi^{-1}\theta_\psi)$. Notice that the mean of this conditional distribution has the same general construction as the first order condition expression for Ψ in (14.11).

The full conditional posterior distribution of Π when a Minnesota-style prior is used is also given by Villani (2009, Proposition A.1). Given our notation, this distribution can be expressed as

$$\text{vec}(\Pi)|\Psi, \Omega, \mathfrak{D}_T \sim N_{p^2k}(\bar{\theta}_\pi, \bar{\Sigma}_\pi), \tag{14.17}$$

where $\bar{\Sigma}_\pi^{-1} = (YY' \otimes \Omega^{-1}) + \Sigma_\pi^{-1}$, while $\bar{\theta}_\pi = \bar{\Sigma}_\pi(\text{vec}(\Omega^{-1}yY') + \Sigma_\pi^{-1}\theta_\pi)$. In this case the full conditional distribution of Ω is:

$$\Omega|\Psi, \Pi, \mathfrak{D}_T \sim IW_p(\varepsilon\varepsilon' + A, T + \nu). \tag{14.18}$$

If the prior on Ω is assumed to be diffuse, then we simply let $\nu = 0$ and $A = 0$ in (14.17). This results in the full conditional posterior of Ω in Villani (2009).

The case when the prior distribution of Π is normal conditional on Ω instead implies that the full conditional posterior of the autoregressive parameters is given by:

$$\text{vec}(\Pi)|\Psi, \Omega, \mathfrak{D}_T \sim N_{p^2k}(\text{vec}(\bar{\mu}_\pi), [\bar{\Omega}_\pi \otimes \Omega]), \tag{14.19}$$

where $\bar{\Omega}_\pi^{-1} = YY' + \Omega_\pi^{-1}$ and $\bar{\mu}_\pi = (yY' + \mu_\pi\Omega_\pi^{-1})\bar{\Omega}_\pi$. For this case we find that the full conditional posterior of Ω is:

$$\Omega|\Psi, \Pi, \mathfrak{D}_T \sim IW_p(\varepsilon\varepsilon' + A + (\Pi - \mu_\pi)\Omega_\pi^{-1}(\Pi - \mu_\pi)', T + pk + \nu). \tag{14.20}$$

If the prior on Ω is assumed to be diffuse, then we simply let $\nu = 0$ and $A = 0$ in (14.20).

Finally, in case a diffuse prior is assumed for Π, then the full conditional distribution of Π is given by (14.19), with $\Omega_\pi^{-1} = 0$. Similarly, the full conditional distribution of Ω is now given by (14.20), with $k = 0$ and $\Omega_\pi^{-1} = 0$. Again, if the prior on Ω is assumed to be diffuse, then we simply let $\nu = 0$ and $A = 0$ in (14.20).

14.4. *Marginal Likelihood*

The marginal likelihood for the Bayesian VAR model can be computed using the ideas in Chib (1995); see, also, Geweke (1999, 2005). That is, we first note that

$$p(\mathfrak{D}_T) = \frac{p(\mathfrak{D}_T|\Psi, \Pi, \Omega)p(\Psi, \Pi, \Omega)}{p(\Psi, \Pi, \Omega|\mathfrak{D}_T)}, \tag{14.21}$$

by applying Bayes rule. Chib (1995) refers to this expression as the *basic marginal likelihood identity* and it holds for any parameter point (Ψ, Π, Ω) in the support.

The numerator in (14.21) is simply the likelihood times the joint prior density, while the denominator is the joint posterior density. The numerator can be directly evaluated at any valid parameter point, such as the posterior mode. The density of the joint posterior is, however, unknown, but can be estimated.

The obtain such an estimate we first factorize the joint posterior density as follows:

$$p(\Psi, \Pi, \Omega|\mathfrak{D}_T) = p(\Omega|\Psi, \Pi, \mathfrak{D}_T)p(\Psi|\Pi, \mathfrak{D}_T)p(\Pi|\mathfrak{D}_T). \tag{14.22}$$

The first term on the right hand side of (14.22) is the density of the inverted Wishart and can therefore be evaluated directly at the selected parameter point $(\Psi, \Pi, \Omega) = (\tilde{\Psi}, \tilde{\Pi}, \tilde{\Omega})$; cf. equation (14.18) or (14.20).

Let $(\Psi^{(i)}, \Pi^{(i)}, \Omega^{(i)})$ be N posterior draws using the relevant Gibbs sampler in Section 14.3 for the full conditional posterior. The third term on the right hand side of (14.22) can now be estimated as

$$\hat{p}(\tilde{\Pi}|\mathfrak{D}_T) = N^{-1}\sum_{i=1}^N p(\tilde{\Pi}|\Psi^{(i)}, \Omega^{(i)}, \mathfrak{D}_T). \tag{14.23}$$

The density on the right hand side of (14.23) is normal and parameterized as shown in equation (14.17) or (14.19).

There remains to estimate the conditional posterior density $p(\Psi|\Pi, \mathfrak{D}_T)$ at the selected parameter point. In this case we cannot, as Chib (1995) explains, use the posterior draws from the Gibbs sampler from the full conditional posteriors above. Instead, we can apply Gibbs samplers for (Ψ, Ω) for the fixed value of $\Pi = \tilde{\Pi}$. That is, we draw $\Psi^{(j)}$ from (14.16) with $\Pi = \tilde{\Pi}$. Similarly, we draw $\Omega^{(j)}$ from either (14.18) or from (14.20) with $\Pi = \tilde{\Pi}$. The gives us N posterior draws $(\Psi^{(j)}, \Omega^{(j)})$ that are all based on a fixed value of Π. We can now estimate $p(\Psi|\Pi, \mathfrak{D}_T)$ at $(\tilde{\Psi}, \tilde{\Pi})$ using

$$\hat{p}(\tilde{\Psi}|\tilde{\Pi}, \mathfrak{D}_T) = N^{-1} \sum_{j=1}^{N} p(\tilde{\Psi}|\Omega^{(j)}, \tilde{\Pi}, \mathfrak{D}_T). \tag{14.24}$$

The density on the right hand side is normal with parameters given by equation (14.16).

There are, of course, alternative ways of estimating the marginal likelihood $p(\mathfrak{D}_T)$ for the Bayesian VAR model; see, e.g., Geweke (1999). The approach advocated by Chib (1995) may be regarded as reliable when a parameter point with a high posterior density is used. The posterior mode, discussed in Section 14.2, is one such point, but one may also consider an estimate of the joint posterior mean. YADA always makes use of the posterior mode, thus explaining why the posterior mode must be estimated prior to running the posterior sampler for the Bayesian VAR. Moreover, when the Gibbs sampler based on the full conditional posteriors is applied, then the point of initialization is given by the posterior mode.

The chosen order of factorization in equation (14.22) influences how the Chib estimator of the marginal likelihood is carried out. Since Π generally has (a lot) more parameters than Ψ or Ω it is useful to fix Π first. The additional Gibbs steps for Ψ and Ω can be carried out much more quickly than the more time consuming Π step. The choice between using the full condition posterior for Ψ or Ω is not so important. From a computational perspective it should generally not matter much if we estimate $p(\Psi|\Pi, \mathfrak{D}_T)$ or $p(\Omega|\Pi, \mathfrak{D}_T)$ since the dimensions of Ψ and Ω are generally fairly low.

14.5. Unconditional Forecasting with a VAR Model

The Thompson and Miller (1986) procedure may also be applied to the Bayesian VAR in Section 14. In this case we let $\theta = (\Psi, \Pi, \Omega)$ where the draws are obtained using the Gibbs samplers discussed in Section 14.3. For a given draw θ from its posterior distribution we may first draw residuals $\varepsilon_{T+1}, \ldots, \varepsilon_{T+h}$ from a normal distribution with mean zero and covariance matrix Ω. Next, we simulate the $x_{T+1}, \ldots, x_{T+h}$ by feeding the residual draws into the VAR system in (14.1). Repeating this P times for the given θ gives us P sample paths conditional on θ. By using S draws of θ from its posterior we end up with PS paths of $x_{T+1}, \ldots, x_{T+h}$ from its predictive density.

For the Bayesian VAR we may decompose the prediction uncertainty into two components, residual uncertainty and parameter uncertainty. That is,

$$C(x_{T+i}|\mathfrak{D}_T) = E_T[C(x_{T+i}|\mathfrak{D}_T; \theta)] + C_T[E(x_{T+i}|\mathfrak{D}_T; \theta)], \tag{14.25}$$

where the deterministic process $d_{T+1}, \ldots, d_{T+h}$ has been suppressed from the expressions to simplify notation. The first term on the right hand side measures the residual uncertainty, while the second measures parameter uncertainty. To parameterize these two terms, we first rewrite the VAR model in (14.1) in first order form:

$$Y_{T+i} = BY_{T+i-1} + J_k \varepsilon_{T+i}, \quad i = 1, \ldots, h, \tag{14.26}$$

where J_k is a $pk \times p$ matrix with I_p on top and a zero matrix below. This means that $y_{T+i} = J_k' Y_{T+i}$. Furthermore, the $pk \times pk$ matrix B is given by

$$B = \begin{bmatrix} \Pi_1 & \cdots & \Pi_{k-1} & \Pi_k \\ I_p & & 0 & 0 \\ & \ddots & & \\ 0 & & I_p & 0 \end{bmatrix}. \tag{14.27}$$

Using these well known expressions we first find that the residual uncertainty term is:

$$E_T\left[C\left(x_{T+i}|\mathfrak{D}_T;\theta\right)\right] = E_T\left[\sum_{j=0}^{i-1} J_k' B^j J_k \Omega J_k' \left(B^j\right)' J_k\right] \quad i = 1,\ldots,h, \tag{14.28}$$

while the parameter uncertainty term is given by:

$$C_T\left[E\left(x_{T+i}|\mathfrak{D}_T;\theta\right)\right] = C_T\left[\Psi d_{T+i} + J_k' B^i Y_T\right] \quad i = 1,\ldots,h. \tag{14.29}$$

It may be noted that the parametric expression for the residual uncertainty term can be simplified such that the summation over j is avoided for all i. Define the $pk \times pk$ matrix $\bar{\Sigma}_X^{(i)}$ from the difference equation

$$\bar{\Sigma}_X^{(i)} = J_k \Omega J_k' + B\bar{\Sigma}_X^{(i-1)} B', \quad i = 1,\ldots,h,$$

where $\bar{\Sigma}_X^{(0)} = 0$. It can now be shown that:

$$J_k'\bar{\Sigma}_X^{(i)} J_k = \sum_{j=0}^{i-1} J_k' B^j J_k \Omega J_k' \left(B^j\right)' J_k, \quad i = 1,\ldots,h.$$

14.6. Conditional Forecasting with the VAR Model

Conditional forecasting with a reduced form Bayesian VAR can be conducted in YADA using the ideas from Waggoner and Zha (1999). That is, reduced form shocks ε_{T+i} are drawn over the conditioning sample from a normal distribution with a mean and covariance matrix which guarantees that the assumptions are satisfied. The approach of fixing certain shocks is not used for the VAR since the individual reduced form shocks do not have any particular interpretation other than being 1-step ahead forecast errors when the parameters are given.

The conditioning assumptions for the VAR model are expressed as

$$\tilde{z}_{T+i} = \tilde{K}_1' x_{T+i} + \sum_{j=1}^{i-1} \tilde{K}_{2j}' x_{T+i-j} + \tilde{u}_T, \quad i = 1,\ldots,g, \tag{14.30}$$

where $\tilde{z}_{T+i}$ is m dimensional with $m < p$, and $\tilde{u}_T$ is an m dimensional vector of known constants. The matrices $\tilde{K}_1$ and $\tilde{K}_{2j}$ are $p \times m$ with $\text{rank}(\tilde{K}_1) = m$ and $j = 1,\ldots,g-1$.

To derive the moments of the distribution for the shocks over the conditioning sample it is convenient to write the VAR model on state-space form. Hence, let the measurement equation be given by

$$x_t = \Psi d_t + J_k' Y_t, \tag{14.31}$$

while the state equation is

$$Y_t = BY_{t-1} + J_k \varepsilon_t, \tag{14.32}$$

where J_k, Y_t, and B are defined in Section 14.5.

For convenience and, as we shall soon see, without loss of generality, let

$$\varepsilon_t = \Omega^{1/2} \tilde{\eta}_t, \tag{14.33}$$

where $\Omega^{1/2}$ is any matrix such that $\Omega = \Omega^{1/2}\Omega^{1/2\prime}$. The shocks $\tilde{\eta}_t$ are iid $N(0, I_p)$. The first step shall be to derive the mean and the covariance matrix for these normalized shocks which guarantee that the conditioning assumptions in (14.30) are satisfied. We shall then translate these moments into the ones that apply for the reduced form shocks ε_t. Given that the moments do not involve a specific choice for $\Omega^{1/2}$, the above claim is implied.

The state equation for period $T+i$ can be expressed relative to Y_T and the normalized shocks from periods $T+1,\ldots,T+i$ by

$$Y_{T+i} = B^i Y_T + \sum_{j=0}^{i-1} B^j J_k \Omega^{1/2} \tilde{\eta}_{T+i-j}, \quad i = 1,\ldots,g.$$

Substituting this into the measurement equation for x_{T+i} we obtain

$$x_{T+i} = \Psi d_{T+i} + J_k' B^i Y_T + \sum_{j=0}^{i-1} J_k' B^j J_k \Omega^{1/2} \tilde{\eta}_{T+i-j}, \quad i = 1, \ldots, g.$$

For periods $T + 1, \ldots, T + g$ we can stack these equations as

$$
\begin{bmatrix} x_{T+g} \\ x_{T+g-1} \\ \vdots \\ x_{T+1} \end{bmatrix} = \begin{bmatrix} \Psi d_{T+g} \\ \Psi d_{T+g-1} \\ \vdots \\ \Psi d_{T+1} \end{bmatrix} + \begin{bmatrix} J_k' B^g \\ J_k' B^{g-1} \\ \vdots \\ J_k' B \end{bmatrix} Y_T +
$$

$$
+ \begin{bmatrix} \Omega^{1/2} & J_k' B J_k \Omega^{1/2} & \cdots & J_k' B^{g-1} J_k \Omega^{1/2} \\ 0 & \Omega^{1/2} & & J_k' B^{g-2} J_k \Omega^{1/2} \\ \vdots & & \ddots & \vdots \\ 0 & 0 & & \Omega^{1/2} \end{bmatrix} \begin{bmatrix} \tilde{\eta}_{T+g} \\ \tilde{\eta}_{T+g-1} \\ \vdots \\ \tilde{\eta}_{T+1} \end{bmatrix},
$$

or

$$\tilde{X}_{T+g} = \Psi_{T+g} + G Y_T + D \tilde{N}_{T+g}. \tag{14.34}$$

The conditioning assumptions in (14.30) can also be stacked in the same manner as the conditioning assumptions for the DSGE model in equation (12.15). This means that:

$$
\begin{bmatrix} \tilde{z}_{T+g} \\ \tilde{z}_{T+g-1} \\ \vdots \\ \tilde{z}_{T+1} \end{bmatrix} = \begin{bmatrix} \tilde{K}_1' & \tilde{K}_{21}' & \cdots & \tilde{K}_{2g-1}' \\ 0 & \tilde{K}_1' & & \tilde{K}_{2g-2}' \\ \vdots & & \ddots & \\ 0 & 0 & & \tilde{K}_1' \end{bmatrix} \begin{bmatrix} x_{T+g} \\ x_{T+g-1} \\ \vdots \\ x_{T+1} \end{bmatrix} + \begin{bmatrix} \tilde{u}_T \\ \tilde{u}_T \\ \vdots \\ \tilde{u}_T \end{bmatrix},
$$

or

$$\tilde{Z}_{T+g} = \tilde{K}' \tilde{X}_{T+g} + \tilde{U}_T. \tag{14.35}$$

Substituting for $\tilde{X}_{T+g}$ from (14.34) in (14.35) and rearranging terms gives us the following linear restrictions that the shocks $\tilde{N}_{T+g}$ must satisfy in order to meet the conditioning assumptions

$$\tilde{K}' D \tilde{N}_{T+g} = \tilde{k}_{T+g}, \tag{14.36}$$

where $\tilde{k}_{T+g} = \tilde{Z}_{T+g} - \tilde{U}_T - \tilde{K}'(\Psi_{T+g} + G Y_T)$.

Like in Waggoner and Zha (1999), the distribution of the shocks $\tilde{N}_{T+g}$ conditional on the restriction (14.36) is normal with mean $\mu_{\tilde{N},T+g}$ and idempotent covariance matrix $\Sigma_{\tilde{N},T+g}$. We here find that

$$
\begin{aligned}
\mu_{\tilde{N},T+g} &= D'\tilde{K} \left(\tilde{K}' D D' \tilde{K} \right)^{-1} \tilde{k}_{T+g}, \\
\Sigma_{\tilde{N},T+g} &= I_{pg} - D'\tilde{K} \left(\tilde{K}' D D' \tilde{K} \right)^{-1} \tilde{K}' D.
\end{aligned}
\tag{14.37}
$$

This concludes the first step of deriving the mean and the covariance matrix that the standardized shocks should be drawn from to ensure that the conditioning assumptions are satisfied.

For the second step, where we will show that the choice of $\Omega^{1/2}$ does not have any effect on the reduced form shocks subject to the conditioning assumptions, let $\mathcal{E}_{T+g}$ be the stacking of $\varepsilon_{T+g}, \ldots, \varepsilon_{T+1}$. This means that

$$\mathcal{E}_{T+g} = \left(I_g \otimes \Omega^{1/2} \right) \tilde{N}_{T+g}. \tag{14.38}$$

The restriction (14.36) can now be expressed in terms of $\mathcal{E}_{T+g}$ as

$$\tilde{K}' \tilde{D} \mathcal{E}_{T+g} = \tilde{k}_{T+g}. \tag{14.39}$$

The matrix $\tilde{D} = D(I_g \otimes \Omega^{-1/2})$ and does not depend on $\Omega^{1/2}$. In fact

$$
\tilde{D} = \begin{bmatrix}
I_p & J_k' B J_k & \cdots & J_k' B^{g-1} J_k \\
0 & I_p & & J_k' B^{g-2} J_k \\
\vdots & & \ddots & \vdots \\
0 & 0 & & I_p
\end{bmatrix}.
$$

Moreover, the definition in (14.38) also means that the distribution of the reduced form shocks $\mathcal{E}_{T+g}$ conditional on the restriction (14.39) is normal with mean $\mu_{\mathcal{E},T+g}$ and covariance matrix $\Sigma_{\mathcal{E},T+g}$. These moments are equal to

$$
\begin{aligned}
\mu_{\mathcal{E},T+g} &= \left(I_g \otimes \Omega\right) \tilde{D}' \tilde{K} \left[\tilde{K}' \tilde{D}\left(I_g \otimes \Omega\right) \tilde{D}' \tilde{K}\right]^{-1} \tilde{k}_{T+g}, \\
\Sigma_{\mathcal{E},T+g} &= \left(I_g \otimes \Omega\right) - \left(I_g \otimes \Omega\right) \tilde{D}' \tilde{K} \left[\tilde{K}' \tilde{D}\left(I_g \otimes \Omega\right) \tilde{D}' \tilde{K}\right]^{-1} \tilde{K}' \tilde{D}\left(I_g \otimes \Omega\right).
\end{aligned}
\tag{14.40}
$$

From these expressions we find that the moments do not depend on a particular choice of $\Omega^{1/2}$ and the claim has therefore been established.

The computation of the conditional predictive distribution can now proceed as follow. For a draw $\theta = (\Psi, \Pi, \Omega)$ from the joint posterior distribution, we may first draw $\mathcal{E}_{T+g}$ from $N(\mu_{\mathcal{E},T+g}, \Sigma_{\mathcal{E},T+g})$ thus yielding a sequence of shocks, $\varepsilon_{T+1}, \ldots, \varepsilon_{T+g}$, which guarantees that the conditioning assumptions (14.30) are met. Next, we draw ε_{T+i} for $i = g+1, \ldots, h$ from $N(0, \Omega)$. With these shocks we can simulate the path $x_{T+1}, \ldots, x_{T+h}$ by feeding the residuals into the VAR system (14.1). Repeating this P times for the given θ gives us P sample paths from the predictive distribution conditional on the historical data, the conditioning assumptions, and θ. Repeating the above procedure for S draws of θ from its joint posterior distribution means that we end up with PS paths of $x_{T+1}, \ldots, x_{T+h}$ from the conditional predictive distribution.

For each draw θ we can also estimate the population mean of $x_{T+1}, \ldots, x_{T+h}$ by letting $\varepsilon_{T+1}, \ldots, \varepsilon_{T+g}$ be equal to $\mu_{\mathcal{E},T+g}$. The shocks ε_{T+i} are next set to zero for $i = g+1, \ldots, h$. By feeding these shock values into the VAR system we obtain a path for $E[x_{T+i}|\mathcal{D}_T, \tilde{Z}_{T+g}; \theta]$, $i = 1, \ldots, h$. Repeating this S times for the different θ draws we may estimate the population mean of the conditional predictive distribution by taking the average of these S paths.

The prediction uncertainty of the conditional forecasts can be decomposed into error (or residual) uncertainty and parameter uncertainty. The equivalent to equation (14.25) is now

$$
C\left(x_{T+i}|\mathcal{D}_T, \tilde{Z}_{T+g}\right) = E_T\left[C\left(x_{T+i}|\mathcal{D}_T, \tilde{Z}_{T+g}; \theta\right)\right] + C_T\left[E\left(x_{T+i}|\mathcal{D}_T, \tilde{Z}_{T+g}; \theta\right)\right],
\tag{14.41}
$$

for $i = 1, \ldots, h$, where the E_T and C_T as in Section 12.1 denotes the expectation and covariance with respect to the posterior of θ at time T. Once again, the deterministic variables over the prediction horizon have been suppressed from the expression.

To parameterize these terms we first note that

$$
E\left[x_{T+i}|\mathcal{D}_T, \tilde{Z}_{T+g}; \theta\right] = \Psi d_{T+i} + J_k' B^i Y_T + \sum_{j=0}^{i-1} J_k' B^j J_k \bar{\mu}_{\mathcal{E},T+i-j},
$$

where $\bar{\mu}_{\mathcal{E},T+i-j} = 0$ if $i - j > g$. These expected values satisfy the conditioning assumptions for $i = 1, \ldots, g$. Moreover, the forecast error for a given θ is

$$
x_{T+i} - E\left[x_{T+i}|\mathcal{D}_T, \tilde{Z}_{T+g}; \theta\right] = \sum_{j=0}^{i-1} J_k' B^j J_k \left(\varepsilon_{T+i-j} - \bar{\mu}_{\mathcal{E},T+i-j}\right).
$$

Next, the covariance matrix $\Sigma_{\mathcal{E},T+g}$ is invariant to T since $\tilde{D}$ and $\tilde{K}$ are both invariant to T. Partitioning this $gp \times gp$ matrix as follows

$$\Sigma_{\mathcal{E},T+g} = \begin{bmatrix} \bar{\Sigma}^{(g,g)} & \cdots & \bar{\Sigma}^{(g,1)} \\ \vdots & \ddots & \vdots \\ \bar{\Sigma}^{(1,g)} & \cdots & \bar{\Sigma}^{(1,1)} \end{bmatrix},$$

where $\bar{\Sigma}^{(i,j)} = \bar{\Sigma}^{(j,i)\prime}$ is the $p \times p$ covariance matrix for the vector pair $(\varepsilon_{T+i}, \varepsilon_{T+j})$ for all $i,j = 1,\ldots,g$. The forecast error covariance matrix of x_{T+i} conditional on $\mathfrak{D}_T, \tilde{Z}_{T+g}$ and θ is now equal to

$$C\left[x_{T+i}|\mathfrak{D}_T, \tilde{Z}_{T+g}; \theta\right] = \sum_{j=0}^{i-1} \sum_{l=0}^{i-1} J_k' B^j J_k \bar{\Omega}^{(i,j)} J_k' \left(B'\right)^l J_k,$$

where

$$\bar{\Omega}^{(i,j)} = \begin{cases} \bar{\Sigma}^{(i,j)}, & \text{if } i,j = 1,\ldots,g, \\ \Omega, & \text{if } i = j, i = g+1,\ldots,h \\ 0 & \text{otherwise.} \end{cases}$$

These covariance matrices also satisfy the conditioning assumptions, meaning that, for instance, $\tilde{K}_1' \bar{\Sigma}^{(1,1)} = 0$.

The moment expression for fixed parameters may also be expressed more compactly. For forecast horizons until g we have that

$$E\left[\tilde{X}_{T+g}|\mathfrak{D}_T, \tilde{Z}_{T+g}; \theta\right] = \Psi_{T+g} + GY_T + \tilde{D}\mu_{\mathcal{E},T+g}, \tag{14.42}$$

gives us the mean predictions from $T+1$ until $T+g$. The forecast error covariance matrix is here given by

$$C\left(\tilde{X}_{T+g}|\mathfrak{D}_T, \tilde{Z}_{T+g}; \theta\right) = \tilde{D}\Sigma_{\mathcal{E},T+g}\tilde{D}'. \tag{14.43}$$

Premultiplying the mean predictions in (14.42) by $\tilde{K}'$ gives us $\tilde{Z}_{T+g} - \tilde{U}_T$ and, hence, they satisfy the conditioning assumptions. Similarly, premultiplying the covariance matrix by $\tilde{K}'$ or postmultiplying it by $\tilde{K}$ gives us a zero matrix. It follows that the conditioning assumptions are always satisfied by any draw from the predictive distribution for fixed θ.

For forecast horizons i greater than the conditioning horizon g it can be shown that

$$E\left[x_{T+i}|\mathfrak{D}_T, \tilde{Z}_{T+g}; \theta\right] = \Psi d_{T+i} + J_k' B^i Y_T + J_k' B^{i-g}\tilde{B}\mu_{\mathcal{E},T+g}, \quad i = g+1,\ldots,h, \tag{14.44}$$

where the $pk \times pg$ matrix

$$\tilde{B} = \begin{bmatrix} J_k & BJ_k & \cdots & B^{g-1}J_k \end{bmatrix}.$$

The forecast error covariance matrix for fixed θ is now given by

$$C\left(x_{T+i}|\mathfrak{D}_T, \tilde{Z}_{T+g}; \theta\right) = \sum_{j=0}^{i-g-1} J_k' B^j J_k \Omega J_k' \left(B'\right)^j J_k + J_k' B^{i-g}\tilde{B}\Sigma_{\mathcal{E},T+g}\tilde{B}'\left(B'\right)^{i-g} J_k, \tag{14.45}$$

for $i = g+1,\ldots,h$. Notice that for a stationary VAR model the mean prediction converges to the mean of x as $i \to \infty$ and the prediction covariance matrix to the covariance matrix of x.

The modesty analysis can also be performed in the VAR setting. Like in the case of the state-space model we can consider one multivariate and two univariate statistics. These are again based on the ideas of Adolfson et al. (2005) and Leeper and Zha (2003). For a given draw $\tilde{\mathcal{E}}_{T+g}$ from $N(\mu_{\mathcal{E},T+g}, \Sigma_{\mathcal{E},T+g})$ the difference between the period $T+g$ simulated conditional forecast value of the endogenous variables and the unconditional forecast (given θ) is

$$\Phi_{T,g}\left(\tilde{\mathcal{E}}_{T+g}\right) = x_{T+g}\left(\tilde{\mathcal{E}}_{T+g}; \theta\right) - E\left[x_{T+g}|\mathfrak{D}_T; \theta\right] = \sum_{j=0}^{g-1} J_k' B^j J_k \tilde{\varepsilon}_{T+g-j} = J_k' \tilde{B}\tilde{\mathcal{E}}_{T+g}, \tag{14.46}$$

where $\bar{\mathcal{E}}_{T+g} = [\bar{\varepsilon}'_{T+g} \cdots \bar{\varepsilon}'_{T+1}]'$. The forecast error covariance matrix for the unconditional forecast of x_{T+g} is

$$\Omega_{T+g} = \sum_{j=0}^{g-1} J_k' B^j J_k \Omega \left(J_k' B^j J_k \right)'. \tag{14.47}$$

From these expressions we can define a multivariate modesty statistic as

$$\mathcal{M}_{T,g}\left(\bar{\mathcal{E}}_{T+g}\right) = \Phi_{T,g}\left(\bar{\mathcal{E}}_{T+g}\right)' \Omega_{T+g}^{-1} \Phi_{T,g}\left(\bar{\mathcal{E}}_{T+g}\right). \tag{14.48}$$

Under the hypothesis that the conditioning shocks are modest this statistic is $\chi^2(p)$. An alternative reference distribution can be generated by computing the same statistic with $\bar{\mathcal{E}}_{T+g}$ replaced with ε_{T+i} drawn from $N(0, \Omega)$ for $i = 1, \ldots, g$ and defining this reference statistic as $\mathcal{M}_{T,g}(\mathcal{E}_{T+g})$. The event $\{\mathcal{M}_{T,g}(\mathcal{E}_{T+g}) \geq \mathcal{M}_{T,g}(\bar{\mathcal{E}}_{T+g})\}$ can then be tested for each one of the PS conditional forecast paths that is computed, making it possible to estimate the probability of this event. If the probability is sufficiently small we may say the hypothesis of modest conditioning assumptions is rejected.

Univariate modesty statistics can now be specified by selecting elements from the vector in (14.46) and the matrix in (14.47). Specifically, we let

$$\mathcal{M}_{T,g}^{(i)}\left(\bar{\mathcal{E}}_{T+g}\right) = \frac{\Phi_{T,g}^{(i)}\left(\bar{\mathcal{E}}_{T+g}\right)}{\sqrt{\Omega_{T+g}^{(i,i)}}}, \quad i = 1, \ldots, p. \tag{14.49}$$

This statistic has a standard normal distribution under the assumption that the conditioning information is modest and, like the multivariate statistic, it takes into account that there is uncertainty about all shocks.

For a Leeper-Zha type of univariate modesty statistic we set the reduced form shocks equal to the mean $\mu_{\mathcal{E},T+g} = [\bar{\mu}'_{\mathcal{E},T+g} \cdots \bar{\mu}'_{\mathcal{E},T+1}]'$ value. The covariance matrix for the forecast errors thus becomes singular and is given by

$$\bar{\Omega}_{T+g} = \sum_{j=0}^{g-1} J_k' B^j J_k \bar{\Sigma}_{\mathcal{E},T+g-j} J_k' \left(B'\right)^j J_k. \tag{14.50}$$

where the "covariance matrix" of the non-zero shocks is given by

$$\bar{\Sigma}_{\mathcal{E},T+j} = \Omega \tilde{K}_1 \left(\tilde{K}_1' \Omega \tilde{K}_1 \right)^{-1} \tilde{K}_1' \Omega,$$

for $j = 1, \ldots, g$. The univariate Leeper-Zha type of modesty statistic is now given by

$$\mathcal{M}_{T,g}^{(i)}\left(\mu_{\mathcal{E},T+g}\right) = \frac{\Phi_{T,g}^{(i)}\left(\mu_{\mathcal{E},T+g}\right)}{\sqrt{\bar{\Omega}_{T+g}^{(i,i)}}}, \quad i = 1, \ldots, p. \tag{14.51}$$

This statistic can now be compared with a standard normal distribution.

14.7. Estimating the Population Spectrum for a VAR Model

As already mentioned in Section 13.3, the population spectrum for a VAR model is provided in, e.g., Hamilton (1994, Chapter 10). The model in equation (14.1) may be rewritten as

$$\Pi(L)\left(x_t - \Psi d_t\right) = \varepsilon_t, \tag{14.52}$$

where

$$\Pi(z) = I_p - \sum_{l=1}^{k} \Pi_l z^l.$$

If all roots of $\Pi(z)$ lie outside the unit circle[123] it follows that x_t is covariance stationary and that the autocovariances are absolutely summable.

---

[123] Or equivalently, all eigenvalues of B in (14.27) lie inside the unit circle.

The population spectrum of x, denoted by $s_x(\omega)$, exists for all ω and is now equal to

$$s_x(\omega) = \frac{1}{2\pi}\Pi\big(\exp(-i\omega)\big)^{-1}\Omega\left(\Pi\big(\exp(i\omega)\big)'\right)^{-1}, \quad \omega \in [-\pi, \pi]. \tag{14.53}$$

Rather than comparing the population spectrum of the state-space model to a non-parametric estimate it may be useful to compare it to a VAR based estimate. The latter model often gives a reasonably good approximation of the covariance properties of the data; see, for instance, King and Watson (1996) and Christiano and Vigfusson (2003) for papers using unrestricted VARs when comparing frequency domain properties of estimated models to the data.

14.8. *YADA Code*

YADA contains a wide range of functions for the Bayesian VAR analysis. In this section I will limit the discussion to the four main topics above, i.e., the prior, the posterior mode estimation, the Gibbs sampler, and the marginal likelihood calculation.

14.8.1. *Functions for Computing the Prior*

This section concerns two functions, `MinnesotaPrior` and `NormalConditionPrior`. These functions both compute elements needed for the prior covariance matrix of the Π parameters. The first functions is used when the Minnesota-style prior is assumed for these parameters, i.e., when Σ_π in (14.3) is determined as in equation (14.4). Similarly, the second function is applied when the normal condition on Ω prior is assumed. In this case, the matrix Ω_π in (14.5) is determined as in equation (14.6).

14.8.1.1. `MinnesotaPrior`

The function `MinnesotaPrior` requires 5 inputs: `OverallTightness` (λ_o), `CrossEqTightness` (λ_c), `HarmonicLagDecay` (λ_h), `OmegaVec`, and `k`. While the first three inputs are hyperparameters, the fourth is a vector with the diagonal elements of Ω, i.e., with the residual variances. YADA always uses the maximum likelihood estimate of Ω to generate these residual variances. Finally, the fifth input is the lag order of the Bayesian VAR. The function provides the $p^2 k \times p^2 k$ matrix `SigmaPi` (Σ_π) as output.

14.8.1.2. `NormalConditionPrior`

The function `NormalConditionPrior` takes 4 inputs: `OverallTightness`, `HarmonicLagDecay`, `p`, and `k`. The first two are the same hyperparameters as the `MinnesotaPrior` function uses, while the third input is the number of endogenous variables, and the fourth the lag order. As output the function provides the $pk \times pk$ matrix `OmegaPi` (Ω_π).

14.8.2. *Functions for Estimating the Mode of the Joint Posterior*

The function `BVARPosteriorModeEstimation` is used to estimate the posterior mode of the Bayesian VAR parameters. It handles all the types of priors discussed in Section 14.1. The main inputs for this function are the structures `DSGEModel` and `CurrINI`; see Section 7.4.

From the perspective of analysing a Bayesian VAR model, the `DSGEModel` structure contains information about the type of prior to use for VAR parameters, their hyperparameters, the lag order, as well as which endogenous and exogenous variables to use, for which sample, the data, the maximum number of iterations to consider, and the tolerance value of the convergence criterion. This information allows the function to compute the maximum likelihood estimates of Ψ, Π, and Ω and fully set up the prior. The maximum likelihood estimates are used as initial values for the posterior mode estimation algorithm. The maximum likelihood estimate of Ω is adjusted to take the prior into account. For example, if a diffuse prior is used for Ω and for Π, then the maximum likelihood estimate of Ω is multiplied by $T/(T + p + 1)$. Similarly, if the inverted Wishart prior in (4.12) is assumed with $\nu \geq p + 2$, then the maximum likelihood estimate is multiplied by T, A is added to this, and everything is divided by $T + p + \nu + 1$.

As discussed in Section 14.2, it is not possible to solve for the posterior mode analytically. Instead, it is possible to iterate on the first order conditions until a set of values that satisfy

these conditions can be found. The posterior mode estimation routine in YADA first evaluates the log posterior at the initial values. For each iteration i YADA computes:

- $\Pi^{(i)}$ given $\Psi^{(i-1)}$ and $\Omega^{(i-1)}$ as shown in equation (14.12) or (14.13);
- $\Psi^{(i)}$ given $\Pi^{(i)}$ and $\Omega^{(i-1)}$ as shown in equation (14.11); and
- $\Omega^{(i)}$ given $\Psi^{(i)}$ and $\Pi^{(i)}$ as shown in equation (14.14) or (14.15).

With a new set of parameter values, the log posterior is recalculated and if then absolute change is not sufficiently small, the algorithm computes iteration $i + 1$. Otherwise it exits.

YADA has three functions for calculating the log posterior. First, if a diffuse prior on Π is assumed, then BVARLogPosteriorDiffuse is called. Second, if the prior on Π is of the Minnesota-style, then the function BVARLogPosteriorMinnesota is used. Finally, the function BVARLogPosteriorNormalCond is considered when a normal conditional on the residual covariance matrix prior on Π is assumed.

Furthermore, YADA has 5 functions for dealing with the computations in (14.11)–(14.15). These functions are: BVARPsiMean (equation 14.11) for Ψ, BVARPiMeanMinnesota (equation 14.12) or BVARPiMeanNormalCond (equation 14.13) for Π, and BVAROmegaMinnesota (equation 14.14) or BVAROmegaNormal (equation 14.15) for Ω. The functions are also used by the Gibbs sampling routine for the Bayesian VAR; cf. Section 14.8.3.

14.8.2.1. BVARLogPosteriorDiffuse

The function BVARLogPosteriorDiffuse needs 11 inputs. First of all it requires the parameter values Omega (Ω), Pi (Π), and Psi (Ψ). Next, the hyperparameters of the prior of the residual covariance matrix are needed as Amatrix (A) and qDF (ν), and the hyperparameters of the steady state prior thetaPsi (θ_ψ) and SigmaPsi (Σ_ψ). Finally, the function needs information about the endogenous and exogenous variables in terms of the matrices x (x), X (X), d (d), and dLag. The latter matrix is given by d_{-1} in $D = [d'\ d'_{-1}]'$, i.e., the final qk rows of D.

The function provides the output scalar logPost which is the sum of the log-likelihood, the log-prior of Ψ and the log-prior of Ω. All proper log-densities include their integration constants. In case the Ω prior is diffuse (improper), it simply uses the log of the right hand side of equation (14.7).

14.8.2.2. BVARLogPosteriorMinnesota

The function BVARLogPosteriorMinnesota requires 13 inputs. All the inputs that the function BVARLogPosteriorDiffuse accepts are included as well as two additional inputs. These extra inputs appear as argument eight and nine and are called thetaPi (θ_π) and SigmaPi (Σ_π).

As output the function provides the scalar logPost which is the sum of the log-likelihood, the log-prior of Ψ, the log-prior of Π, and the log-prior of Ω.

14.8.2.3. BVARLogPosteriorNormalCond

The function BVARLogPosteriorNormalCond requires 13 inputs. All the inputs that the function BVARLogPosteriorDiffuse accepts are included as well as two additional inputs. These extra inputs appear as argument eight and nine and are called muPi (μ_π) and OmegaPi (Ω_π).

As output the function provides the scalar logPost which is the sum of the log-likelihood, the log-prior of Ψ, the log-prior of Π, and the log-prior of Ω.

14.8.2.4. BVARPsiMean

The function BVARPsiMean is used to compute the value of $\bar{\theta}_\psi$ is equation (14.16). Optionally, it can also calculate the $pq \times pq$ matrix $\bar{\Sigma}_\psi$ as well. A total of 8 inputs are needed by the function. These are given by: Omega, Pi, thetaPsi, invSigmaPsi, x, X, d, and dLag. Apart from invSigmaPsi all these inputs are discussed for the BVARLogPosteriorDiffuse function. The matrix invSigmaPsi is simply the inverse of Σ_ψ.

The required output from the function is ThetaBarPsi, a $p \times q$ matrix. If we apply the vec operator on this matrix we obtain $\bar{\theta}_\psi$. The optional output is called SigmaBarPsi. While only

the required output is needed by the posterior mode estimation routine, the optional output is needed by the Gibbs sampler for drawing from the posterior distribution.

14.8.2.5. BVARPiMeanMinnesota

The function BVARPiMeanMinnesota accepts 8 inputs: Omega, Psi, thetaPi, invOmegaPi, x, X, d, and dLag. Apart from invOmegaPi all these inputs are described above for the log posterior function BVARLogPosteriorMinnesota. The input invOmegaPi is simply the inverse of Σ_π.

The output ThetaBarPi is required and is a $p \times pk$ matrix. The vector $\bar\theta_\pi$ in (14.17) is obtained by applying the vec operator to this output. Optionally, the matrix SigmaBarPi is provided. This output is equal to $\bar\Sigma_\pi$ in the same equation.

14.8.2.6. BVARPiMeanNormalCond

The function BVARPiMeanNormalCond needs 7 inputs: Psi, muPi, invOmegaPi, x, X, d, and dLag. All these inputs apart from invOmegaPi are discussed above for the log posterior function BVARLogPosteriorNormalCond. The input invOmegaPi is, of course, the inverse of Ω_π.

The output muBarPi is required and is the $p \times pk$ matrix $\bar\mu_\pi$ in (14.19). Optionally, the matrix OmegaBarPi is provided. This output is equal to $\bar\Omega_\pi$ in the same equation.

14.8.2.7. BVAROmegaMinnesota

The function BVAROmegaMinnesota needs 8 inputs: Pi, Psi, A, qDF, x, X, d, and dLag. These inputs are also required by BVARLogPosteriorMinnesota. As output the function provides Omega, the $p \times p$ matrix in equation (14.14).

14.8.2.8. BVAROmegaNormal

The function BVAROmegaNormal accepts 10 inputs: Pi, Psi, A, qDF, muPi, invOmegaPi, x, X, d, and dLag. These inputs are discussed above; see the BVARLogPosteriorNormalCond and the BVARPiMeanNormalCond functions. As output the function provides Omega, the $p \times p$ matrix in equation (14.15).

14.8.3. *Gibbs Sampling*

The function BVARPosteriorSampling controls the events regarding the posterior sampling algorithm. The function takes exactly the same inputs as the posterior mode estimation function BVARPosteriorModeEstimation. The Bayesian VAR estimation routines follow the same type of logic as the DSGE model estimation routines. This means that you have to run the posterior mode estimation before the posterior sampling function can be run. The reason is simply that the posterior sampling routine for the Bayesian VAR uses the posterior mode estimates of the parameters to initialize the Gibbs sampler.

As in the case of posterior sampling for the DSGE model, the sampling function for the Bayesian VAR model reads a number of variable entries from the posterior mode estimation output file. These data are read from file to ensure that exactly the same data is used for posterior sampling as was used for posterior mode estimation. Hence, if you have changed some hyperparameter after running the posterior mode estimation part, the new value will not be used by YADA. Similarly, changes to the sample will be ignored as well as any other changes to the data.

The posterior sampling function can look for sampling data that you have already generated and can load this data. These features operate in exactly the same way for the posterior sampling function for the DSGE and the Bayesian VAR model.

The precise Gibbs sampler for the Bayesian VAR depends on the prior you are using for the Ψ, Π, and Ω parameters. The posterior mode estimates are always used as initial values for the sampler. To generate draw number i of the parameters YADA first draws $\Omega^{(i)}$ conditional on $\Psi^{(i-1)}$ and $\Pi^{(i-1)}$ with the function InvWishartRndFcn. Since the marginal likelihood is also estimated as described in Section 14.4 YADA also draws a value for Ω conditional on Π fixed at the posterior mode and a theoretically consistent previous value of Ψ, i.e., one that is also conditioned on Π at the posterior mode. Next, YADA draws $\Pi^{(i)}$ conditional on

$\Psi^{(i-1)}$ and $\Omega^{(i)}$. Here it utilizes the function `MultiNormalRndFcn`. To finish the i:th draw, a value of $\Psi^{(i)}$ conditional on $\Pi^{(i)}$ and $\Omega^{(i)}$ is obtained. Again, YADA makes use of the function `MultiNormalRndFcn`.

As in the case of Ω YADA also draws a value of Ψ conditional on Π fixed at the posterior mode and the draw obtained for Ω when Π is fixed at the mode. This value of Ψ is used for the next draw of Ω conditional on Π fixed at the posterior mode. In this fashion YADA generates two sets of Gibbs sampler draws. The first full set $(\Psi^{(i)}, \Pi^{(i)}, \Omega^{(i)})$ are draws from the joint posterior and are also used to estimate the density $p(\tilde{\Pi}|\mathfrak{D}_T)$ in equation (14.23). The second partial set $(\Psi^{(j)}, \Omega^{(j)})$ is only used to estimate the conditional density $p(\tilde{\Psi}|\tilde{\Pi}, \mathfrak{D}_T)$ in equation (14.24).

14.8.3.1. `InvWishartRndFcn`

The function `InvWishartRndFcn` requires two inputs to generate a draw from the inverted Wishart distribution. These inputs are `A` and `df`, representing the location parameter and the degrees of freedom parameter respectively. As output the function provides `Omega`.

14.8.3.2. `MultiNormalRndFcn`

The function `MultiNormalRndFcn` generates a desired number of draws from the multivariate normal distribution. As input the function needs `mu`, `Sigma`, and `NumDraws`. These inputs provide the mean, the covariance matrix and the number of desired draws, respectively. The last input is optional and defaults to 1.

As output the function gives `z`, a matrix with as the same number of rows as the dimension of the mean and number of columns gives by `NumDraws`.

14.8.4. *Marginal Likelihood of the Bayesian VAR*

Estimation of the marginal likelihood is handled by the function `MargLikeChib` in YADA. Before discussing this function in more detail it is worthwhile to keep Lindley's (also known as Bartlett's) paradox in mind; see Bartlett (1957) and Lindley (1957). That is, as a rule of thumb we should only compare the marginal likelihood value across two models if the prior is proper in the dimensions where they differ. By proper we mean that the prior density should integrate to unity (a finite constant) over these parameters. For instance, if the Minnesota-style prior is assumed for Π, then we can compare the marginal likelihood for models that differ in terms of the lag order given that the same sample dates and variables are covered by x_t. If, instead, the diffuse prior $p(\Pi) \propto 1$ is used for these parameters, then the marginal likelihoods should not be compared in this dimension. The paradox here states that the model with fewer lags will have a greater marginal likelihood value regardless of the information in the data.

14.8.4.1. `MargLikeChib`

The function `MargLikeChib` computes the log marginal likelihood using Chib's marginal likelihood identity. As input the function requires 9 inputs (and accepts a 10th). First of all it takes two vectors with `NumIter` elements of values of the log densities $p(\tilde{\Pi}|\Psi^{(i)}, \Omega^{(i)}, \mathfrak{D}_T)$ (`LogPiDensity`) and $p(\tilde{\Psi}|\tilde{\Pi}, \Omega^{(j)}, \mathfrak{D}_T)$ (`LogPsiDensity`). Next, the scalars `LogPosterior` and `LogOmegaDensity` are accepted, where the first measures the sum of the log-likelihood and the log-prior, while the second measures the log of $p(\tilde{\Omega}|\tilde{\Psi}, \tilde{\Pi}, \mathfrak{D}_T)$. The fifth input is `q`, giving the number of exogenous variables.

The following and sixth input is the integer `NumBurnin` which provides the number of burn-in draws to be removed from the top of the log density vector. Next, `MargLikeChib` accepts the boolean input `ComputeSequential`. The function will compute sequential estimates of the log marginal likelihood is this input is unity, and only the final estimate if it is zero. The first of the remaining inputs is `SequentialStartIterationValue`. This integer determines after how many draws the first sequential estimate shall be performed. Similarly, the last required input `SequentialStepLengthValue` determines how many draws to use as increment.

The function provides the matrix `LogMargs` as output. The number of rows of this matrix is equal to the number of sequential estimates. The first column contains the number of draws

used for a particular estimate, the second column the estimated log marginal likelihood, and the third column the numerical standard error of the estimate based on the Newey and West (1987) correction for autocorrelation.

14.8.5. *Forecasting with a Bayesian VAR*

14.8.5.1. BVARPredictionPathsPostMode

The function `BVARPredictionPathsPostMode` requires 9 inputs. The first group is `Psi`, `Pi`, and `Omega` with fixed values for Ψ, Π, and Ω, respectively. Next, the function takes the structures `DSGEModel` and `CurrINI`. Furthermore, the $p \times h$ matrix `DPred` with data on the exogenous variables over the h periods in the prediction sample as well as `h`, the prediction sample length are needed. Finally, the function requires the integer `NumPaths` and the boolean `AnnualizaData`. The former determines the number of prediction paths to compute at the fixed parameter value, while the latter indicates if the prediction paths should be annualized or not.

The number of output variables is equal to 6. The first is the 3-dimensional matrix `PredPaths`, whose dimensions are given by the number of observed variables, the length of the prediction sample, and the number of prediction paths. The second output variable is `PredMean`, a matrix with the population mean predictions of the observed variables. The following output variables are the matrices `PredEventData`, which stores the prediction event results, and `YObsEventData`, which stores the observed event paths, i.e., those when the mean of the paths is equal to the observed ath. These two matrices are obtained from the function `CalculatePredictionEvents` and have as many rows as variables and 7 columns; see Section 12.11.13. The last two output variables are called `KernelX` and `KernelY`, 3-dimensional matrices with kernel density estimates of the marginal predictive densities. The dimensions of both matrices are equal to the number of observed variables, the number of grid points, and the prediction sample length.

14.8.5.2. BVARPredictionPaths

The function `BVARPredictionPaths` also requires 9 inputs. The final 6 input variables are identical to the final 6 input variables for the `BVARPredictionPathsPostMode` function. The first 3, however, are now given by the matrices `PsiPostSample`, `PiPostSample`, and `OmegaPostSample`. The number of rows of these matrices is `NumDraws`, while the number of columns is equal to the number of parameters of Ψ, Π, and Ω, respectively.

The function gives 4 variables as output. First, the boolean variable `DoneCalc` indicates if the calculations were finished or not. The second output is `PredEventData`, a $p \times 2$ matrix with the prediction event results. Furthermore, the prediction uncertainty decomposition into the residual uncertainty and the parameter uncertainty is provided through the 3D matrices `ShockCov` and `ParameterCov`. The dimensions of these matrices are $p \times p \times h$, where h is the length of the prediction sample. This decomposition is only calculated when the boolean input variable `AnnualizeData` is zero.

14.8.5.3. BVARCondPredictionPathsPostMode

The function `BVARCondPredictionPathsPostMode` for computing conditional forecasts with the BVAR at posterior mode values of the parameters requires 11 input variable. Nine of these variables are shared with `BVARPredictionPathsPostMode`, the function for unconditional forecasts at posterior mode. The additional two input variables are `Z` and `U`. The matrix `Z` is an $m \times g$ matrix with the conditioning data used by the Bayesian VAR, while the vector `U` holds the initial values $\tilde{u}_T$; see equation (14.30).

The number of output variables supplied by the function is equal to 10 and in addition to the variables given by `BVARPredictionPathsPostMode` it also provides `MeanShocks`, a $p \times h$ matrix with the mean value of the shocks at the population mean prediction, and three variables related to modesty analysis: `MultiModestyStat`, `UniModestyStat`, and `UniModestyStatLZ`. The modesty statistics are only calculated when `AnnualizeData` is 0. In that case, `MultiModestyStat` is a `NumPaths` times 2 matrix, with the multivariate modesty statistic and the reference statistic in the two columns. The univariate Adolfson et al. (2005) statistics are stored in the `NumPaths`

times p matrix `UniModestyStat`, while the univariate Leeper-Zha related statistics are given by the p-dimensional vector `UniModestyStatLZ`.

14.8.5.4. BVARCondPredictionPaths

The function `BVARCondPredictionPaths` calculates conditional forecasts with the BVAR using draws from the posterior distribution of the model parameters. It uses 11 input variables, where nine are shared with `BVARPredictionPaths`, and the additional two inputs are given by `Z` and `U`, discussed above for `BVARCondPredictionPathsPostMode`.

The function supplies 5 output variables and 4 of these are shared with the unconditional forecasting function `BVARPredictionPaths`. The remaining variable is `ShockMean`, a $p \times h$ matrix with the estimated population mean of the residuals over the posterior draws.

15. Misspecification Analysis of DSGE Models with DSGE-VARs

An important aspect of Bayesian analysis is that it does not rely on the assumption that the model is correctly specified. The so called DSGE-VAR approach, advocated in a series of articles by Del Negro and Schorfheide (2004, 2006, 2009) and Del Negro, Schorfheide, Smets, and Wouters (2007), has been suggested as a tool for measuring the degree of misspecification of a DSGE model by approximating it by a VAR; see also An and Schorfheide (2007). The idea of using VARs as an alternative data generating process to a DSGE model can be traced back to the literature on indirect inference; see Smith (1993) in particular, but also Gouriéroux, Monfort, and Renault (1993).

An early attempt of combining DSGE models with Bayesian VARs is Ingram and Whiteman (1994), where the VAR parameters were expressed as a function of the DSGE model parameters by solving the latter model. A prior for the DSGE model parameters then implied a prior for the VAR parameters through a first-order Taylor expansion of the mapping. This idea was considerably enriched by Del Negro and Schorfheide (2004) where the prior distribution of the VAR model parameters was determined from the DSGE model by parameterizing the distribution through the implied first and second moments of the DSGE model.

DSGE-VARs may be indexed by a single parameter that has the DSGE model approximation at one end and an unrestricted VAR at the other end. In between these extreme values, a large number of models exist. Apart from providing a measure of the degree to which the DSGE model may be misspecified, the approach also allows for posterior analysis of the DSGE model parameters as well as any real valued functions of these parameters (and the data) via the VAR model; see Smith (1993) for similar ideas. For the DSGE-VAR, this is achieved by integrating out the VAR parameters from the joint posterior of the VAR and DSGE model parameters. The resulting likelihood function may be combined with the prior of the DSGE model parameters and used for posterior sampling or posterior mode estimation. While the framework in Del Negro and Schorfheide (2004) was designed to improve forecasting and monetary policy analysis with VARs, the extension to a model evaluation tool was carried out by Del Negro and Schorfheide (2006), while Del Negro et al. (2007) used it to assess the fit of a DSGE model. The DSGE-VAR approach, as it has been implemented in YADA, is discussed here. First, however, we turn to two preliminary issues that needs attention. Namely, how to deal with the deterministic variables and a time-varying H matrix in the measurement equation.

15.1. *Preliminary Considerations*

A DSGE-VAR model is typically assumed to have a constant term, but not any other deterministic components. Provided that all the roots of the polynomial function of the VAR representation lie outside the unit circle, the endogenous variables will have a constant mean and autocovariance function; see, e.g., equation (14.52). As a consequence, the non-central second moments are also constant over time. When the deterministic variables are time-varying, the mean of the observed variables is time-varying and thereby also non-central second moments even for constant central second moments.

One solution to this problem is to require that x_t in the DSGE model is a constant so that $E[y_t; \theta] = A$ or that $A = 0$. In addition, the autocovariances are constant over time if a time-varying H matrix in the measurement equation is ruled out. Should the DSGE model not satisfy these two conditions, then the DSGE-VAR would simply not be available for estimation in the program. Alternatively, these restrictions may be weakened by taking sample averages of the time-varying population mean and autocovariances. Should x_t be constant and the H matrix time-invariant, then the model based population moments are constant over time and equal to the sample averages. YADA uses this more general framework for its DSGE-VAR implementation.

Under the latter approach, we first note that the DSGE model based population mean of y_t conditional on θ is:
$$E[y_t; \theta] = A' x_t, \quad t = 1, \ldots, T. \tag{15.1}$$
Moreover,
$$E[y_t x'_{t-j}; \theta] = A' x_t x'_{t-j}, \quad t = 1, \ldots, T, \text{ and } j = 0, 1, \ldots. \tag{15.2}$$

Similarly, the central DSGE model based population autocovariances of y_t conditional on θ may be expressed as:

$$\Sigma_{y,t}^{(j)} = E\left[\left(y_t - A'x_t\right)\left(y_{t-j} - A'x_{t-j}\right)';\theta\right] = \begin{cases} H_t'\Sigma_\xi H_t + R, & \text{if } j = 0, \\ H_t'F^j\Sigma_\xi H_{t-j}, & \text{for } j = 1, 2, \ldots. \end{cases} \quad (15.3)$$

Recall that Σ_ξ is the central population covariance matrix of the state variables conditional on θ, i.e., it satisfies the Lyapunov equation (5.15). It now follows from equations (15.2) and (15.3) that the non-central population autocovariances are

$$E\left[y_t y_{t-j}';\theta\right] = \Sigma_{y,t}^{(j)} + A'x_t x_{t-j}'A, \quad t = 1, \ldots, T, \text{ and } j = 0, 1, \ldots. \quad (15.4)$$

We may next define average DSGE model based population moments from these expressions. Specifically, the sample average of the products for the deterministic variables is

$$\bar{\Sigma}_x^{(j)} = \frac{1}{T}\sum_{t=1}^{T} x_t x_{t-j}', \quad j = 0, 1, \ldots, \quad (15.5)$$

while

$$\bar{\Sigma}_y^{(j)} = \frac{1}{T}\sum_{t=1}^{T} \Sigma_{y,t}^{(j)}, \quad j = 0, 1, \ldots, \quad (15.6)$$

gives the sample average of the central population autocovariances conditional on θ. If $H_t = H$ for all t, then $\Sigma_{y,t}^{(j)}$ is equal to $\bar{\Sigma}_y^{(j)}$. Moreover, if $x_t = 1$, then $\bar{\Sigma}_x^{(j)}$ is equal to unity.

In terms of non-central moments, these averages imply that

$$\frac{1}{T}\sum_{t=1}^{T} E\left[y_t y_{t-j}';\theta\right] = \bar{\Sigma}_y^{(j)} + A'\bar{\Sigma}_x^{(j)}A, \quad j = 0, 1, \ldots, \quad (15.7)$$

while

$$\frac{1}{T}\sum_{t=1}^{T} E\left[y_t x_{t-j}';\theta\right] = A'\bar{\Sigma}_x^{(j)}, \quad j = 0, 1, \ldots. \quad (15.8)$$

We shall use these average moments below when parameterizing the prior distribution of the DSGE-VAR.

15.2. *The DSGE-VAR Model*

To setup the DSGE-VAR, we proceed as follows. The VAR representation of y_t can be written as:

$$y_t = \Phi_0 x_t + \sum_{j=1}^{p} \Phi_j y_{t-j} + \epsilon_t, \quad t = 1, \ldots, T, \quad (15.9)$$

where $\epsilon_t \sim N_n(0, \Sigma_\epsilon)$. The matrix Φ_0 is $n \times k$, while Φ_j is $n \times n$ for $j = 1, \ldots, p$. We assume that initial values for y_t and x_t exists for $t = 0, \ldots, 1 - p$.

Let $Y_t = [x_t'\ y_{t-1}'\ \cdots\ y_{t-p}']'$ be an $(np + k)$-dimensional vector, while the matrix $\Phi = [\Phi_0\ \Phi_1\ \cdots\ \Phi_p]$ has dimension $n \times (np + k)$. This means that the VAR can be expressed as

$$y_t = \Phi Y_t + \epsilon_t.$$

Next, stack the variables as $y = [y_1\ \cdots\ y_T]$, $Y = [Y_1\ \cdots\ Y_T]$, while $\epsilon = [\epsilon_1\ \cdots\ \epsilon_T]$, yieding the system

$$y = \Phi Y + \epsilon. \quad (15.10)$$

Del Negro and Schorfheide (2004) suggests that a VAR approximation of the DSGE model can be obtained by replacing the VAR parameters by the implied "estimates" using the population moments conditional on θ. That is, let $\Gamma_{YY}(\theta)$ be an $(np + k) \times (np + k)$ matrix with average

non-central population moments based on the Y_t vector:

$$\Gamma_{YY}(\theta) = \begin{bmatrix} \bar{\Sigma}_x^{(0)} & \bar{\Sigma}_x^{(1)}A & \cdots & \bar{\Sigma}_x^{(p)}A \\ A'\bar{\Sigma}_x^{(1)\prime} & \bar{\Sigma}_y^{(0)} + A'\bar{\Sigma}_x^{(0)}A & \cdots & \bar{\Sigma}_y^{(p-1)} + A'\bar{\Sigma}_x^{(p-1)}A \\ \vdots & \vdots & \ddots & \vdots \\ A'\bar{\Sigma}_x^{(p)\prime} & \bar{\Sigma}_y^{(p-1)\prime} + A'\bar{\Sigma}_x^{(p-1)\prime}A & \cdots & \bar{\Sigma}_y^{(0)} + A'\bar{\Sigma}_x^{(0)}A \end{bmatrix}. \tag{15.11}$$

Similarly, let $\Gamma_{yY}(\theta)$ be an $n \times (np+k)$ matrix with the average non-central population moments based on the y_t and Y_t vectors

$$\Gamma_{yY}(\theta) = \begin{bmatrix} A'\bar{\Sigma}_x^{(0)} & \bar{\Sigma}_y^{(1)} + A'\bar{\Sigma}_x^{(1)}A & \cdots & \bar{\Sigma}_y^{(p)} + A'\bar{\Sigma}_x^{(p)}A \end{bmatrix}. \tag{15.12}$$

If x_t is a constant and the H matrix in the measurement equation is not time-varying, these average population moments are the same as those given Del Negro and Schorfheide (2004, Section A.2).

A population based regression can now determine the mapping from the DSGE model parameters to the VAR parameters. Specifically, suppose that $\Gamma_{YY}(\theta)$ is invertible, then

$$\Phi(\theta) = \Gamma_{yY}(\theta)\Gamma_{YY}^{-1}(\theta), \tag{15.13}$$

$$\Sigma_\epsilon(\theta) = \Gamma_{yy}(\theta) - \Gamma_{yY}(\theta)\Gamma_{YY}^{-1}(\theta)\Gamma_{yY}'(\theta), \tag{15.14}$$

where $\Gamma_{yy}(\theta) = \bar{\Sigma}_y^{(0)} + A'\bar{\Sigma}_x^{(0)}A$ is an $n \times n$ matrix with average non-central population moments based on the y_t vector. The matrices $\Phi(\theta)$ and $\Sigma_\epsilon(\theta)$ are restriction functions that will be used to center the prior distribution of (Φ, Σ_ϵ) conditional on θ and a hyperparameter $\lambda \geq 0$ that measures the deviation of the DSGE-VAR from the VAR approximation of the DSGE model.

15.3. *Prior Distribution of the DSGE-VAR*

The joint prior distribution of the VAR and DSGE model parameters has the following hierarchical structure:

$$p(\Phi, \Sigma_\epsilon, \theta|\lambda) = p(\Phi, \Sigma_\epsilon|\theta, \lambda)p(\theta). \tag{15.15}$$

The conditional prior distribution of the VAR parameters will be centered at the VAR approximation of the DSGE model $(\Phi(\theta), \Sigma_\epsilon(\theta))$, but will allow for deviations from the restrictions to account for possible misspecification. The precision of the prior is determined by the hyperparameter λ, which generates a continuum of models that have an unrestricted VAR at one extreme (λ close to zero) and the DSGE model approximation at the other ($\lambda = \infty$). In practise, a grid is used for λ such that the values $\Lambda = \{\lambda_1, \ldots, \lambda_q\}$ are considered.

One interpretation of the hyperparameter λ is related to dummy observations through $T_\lambda = \lambda T$. That is, we may augment the VAR model with T_λ observations of the endogenous variables that are generated from the DSGE model. Specifically, the prior of the VAR parameters takes the form

$$\Sigma_\epsilon|\theta, \lambda \sim IW_n\big(T_\lambda \Sigma_\epsilon(\theta), T_\lambda - (np+k)\big), \tag{15.16}$$

$$\text{vec}(\Phi)|\Sigma_\epsilon, \theta, \lambda \sim N_{n(np+k)}\big(\text{vec}(\Phi(\theta)), T_\lambda^{-1}\big[\Gamma_{YY}^{-1}(\theta) \otimes \Sigma_\epsilon\big]\big). \tag{15.17}$$

The conjugate normal-inverted Wishart prior assumed here is proper (integrates to unity) when $T_\lambda \geq n(p+1) + k$. Hence, the domain of λ is restricted to the interval $[(n(p+1)+k)/T, \infty]$ for the DSGE-VAR.[124]

15.4. *Conditional Posterior Distribution of the VAR parameters*

The posterior density is proportional to the product of the prior density and the likelihood function. Conditional on (θ, λ) the DSGE-VAR prior and likelihood are conjugate. It follows that the posterior distribution of Φ and Σ_ϵ are also of the normal-inverted Wishart form; see

---

[124] The density function of the inverted Wishart is provided in equation (4.12); see Section 4.2.4.

Zellner (1971). Let the non-central sample product moment matrices be given by:

$$\hat{\Gamma}_{yy} = \frac{1}{T}\sum_{t=1}^{T} y_t y_t', \quad \hat{\Gamma}_{YY} = \frac{1}{T}\sum_{t=1}^{T} Y_t Y_t', \quad \hat{\Gamma}_{yY} = \frac{1}{T}\sum_{t=1}^{T} y_t Y_t'.$$

From, e.g., Del Negro and Schorfheide (2004), the conditional posterior distributions of the DSGE-VAR parameters can now be expressed as:

$$\Sigma_\epsilon | y, Y_1, \theta, \lambda \sim IW_n\big((1+\lambda)T\hat{\Sigma}_\epsilon(\theta), (1+\lambda)T - (np+k)\big), \tag{15.18}$$

$$\text{vec}(\Phi) | y, Y_1, \Sigma_\epsilon, \theta, \lambda \sim N_{n(np+k)}\big(\text{vec}(\hat{\Phi}(\theta)), \big[(1/T)[\lambda\Gamma_{YY}(\theta) + \hat{\Gamma}_{YY}]^{-1} \otimes \Sigma_\epsilon\big]\big), \tag{15.19}$$

where

$$\hat{\Phi}(\theta) = \left(\frac{\lambda}{1+\lambda}\Gamma_{yY}(\theta) + \frac{1}{1+\lambda}\hat{\Gamma}_{yY}\right)\left(\frac{\lambda}{1+\lambda}\Gamma_{YY}(\theta) + \frac{1}{1+\lambda}\hat{\Gamma}_{YY}\right)^{-1}, \tag{15.20}$$

$$\hat{\Sigma}_\epsilon(\theta) = \left(\frac{\lambda}{1+\lambda}\Gamma_{yy}(\theta) + \frac{1}{1+\lambda}\hat{\Gamma}_{yy}\right) - \left(\frac{\lambda}{1+\lambda}\Gamma_{yY}(\theta) + \frac{1}{1+\lambda}\hat{\Gamma}_{yY}\right) \times$$
$$\left(\frac{\lambda}{1+\lambda}\Gamma_{YY}(\theta) + \frac{1}{1+\lambda}\hat{\Gamma}_{YY}\right)^{-1}\left(\frac{\lambda}{1+\lambda}\Gamma_{yY}(\theta) + \frac{1}{1+\lambda}\hat{\Gamma}_{yY}\right)'. \tag{15.21}$$

Notice that the posterior distributions in (15.18) and (15.19) depend on the initial values Y_1. When we wish to compare marginal likelihoods for the DSGE-VAR to the DSGE model, the information sets need to be the same. This may be handled in the DSGE model by using the sample $y_{1-p}, \ldots, y_0$ as a training sample for the Kalman filter.

From the expressions in (15.20) and (15.21) it can also be seen that the larger λ is, the closer the posterior mean of the VAR parameters is to $\Phi(\theta)$ and $\Sigma_\epsilon(\theta)$, the values that respect the cross equation restrictions of the DSGE model. At the same time, the smaller λ becomes, the closer the posterior mean is to the classical maximum likelihood estimates of Φ and Σ_ϵ.

15.5. *Posterior Sampling of the DSGE Model Parameters*

The joint posterior density of the DSGE and VAR model parameters can be factorized as:

$$p\big(\Phi, \Sigma_\epsilon, \theta | y, Y_1, \lambda\big) = p\big(\Phi, \Sigma_\epsilon | y, Y_1, \theta, \lambda\big)p\big(\theta | y, Y_1, \lambda\big). \tag{15.22}$$

The first term on the right hand side is given by the product of the densities in (15.18) and (15.19). The second term is the marginal posterior density of θ for a given λ and it can be determined via the marginal likelihood function $p(y|Y_1, \theta, \lambda)$, the prior of the DSGE model parameters, and a suitable posterior sampler. It is straightforward to show that:

$$p\left(y | Y_1, \theta, \lambda\right) = \frac{p\big(y | Y_1, \Phi, \Sigma_\epsilon, \theta, \lambda\big)p\big(\Phi, \Sigma_\epsilon | \theta, \lambda\big)}{p\big(\Phi, \Sigma_\epsilon | y, Y_1, \theta, \lambda\big)}. \tag{15.23}$$

The first term in the numerator is the likelihood function of the VAR in (15.9), while the second term is the prior density of the VAR parameters conditional on θ and λ; cf. equations (15.16) and (15.17). The numerator is the conditional posterior density of the VAR parameters. Del Negro and Schorfheide (2004) shows that the ratio on the right hand side is equal to

$$p\left(y | Y_1, \theta, \lambda\right) = \frac{\left|\lambda T\Gamma_{YY}(\theta) + T\hat{\Gamma}_{YY}\right|^{-n/2}\left|(1+\lambda)T\hat{\Sigma}_\epsilon(\theta)\right|^{-((1+\lambda)T-np-k)/2}}{\left|\lambda T\Gamma_{YY}(\theta)\right|^{-n/2}\left|\lambda T\Sigma_\epsilon(\theta)\right|^{-(\lambda T-np-k)/2}}$$
$$\times \frac{\big(\pi\big)^{-nT/2}\Gamma_n\big((1+\lambda)T - np - k\big)}{\Gamma_n\big(\lambda T - np - k\big)}, \tag{15.24}$$

where $\Gamma_b(a) = \prod_{i=1}^{b}\Gamma([a+1-i]/2)$ for positive integers a, b with $a \geq b$, and $\Gamma(\cdot)$ is the gamma function; see equation (4.4).[125]

---

[125] Relative to equation (A.2) in Del Negro and Schorfheide (2004), the expression in (15.24) takes into account that all terms involving powers of 2 cancel out in the numerator and denominator. The expression in (15.24) can

These results are valid when λ is finite. The case of $\lambda = \infty$ implies that the VAR parameters (Φ, Σ_ϵ) are equal to $(\Phi(\theta), \Sigma_\epsilon(\theta))$. The two densities for the VAR parameters in (15.23) are therefore unity, so that

$$p\left(y \,|Y_1, \theta, \lambda = \infty\right) = p\left(y \,|Y_1, \Phi(\theta), \Sigma_\epsilon(\theta), \lambda = \infty\right). \tag{15.25}$$

The right hand side of (15.25) is the likelihood function of the VAR and, hence, the multivariate normal density provides us with

$$p\left(y \,|Y_1, \Phi(\theta), \Sigma_\epsilon(\theta), \lambda = \infty\right) = \left(2\pi\right)^{-nT/2} |\Sigma_\epsilon(\theta)|^{-T/2} \exp\left(-\frac{T}{2} \text{tr}\left[\Sigma_\epsilon^{-1}(\theta)\tilde{\Sigma}_\epsilon(\theta)\right]\right), \tag{15.26}$$

where

$$\tilde{\Sigma}_\epsilon(\theta) = \hat{\Gamma}_{yy} - \hat{\Gamma}_{yY}\hat{\Gamma}_{YY}^{-1}\hat{\Gamma}_{yY}' + \left(\Phi(\theta) - \hat{\Phi}\right)\hat{\Gamma}_{YY}\left(\Phi(\theta) - \hat{\Phi}\right)',$$

and $\hat{\Phi} = \hat{\Gamma}_{yY}\hat{\Gamma}_{YY}^{-1}$ is the maximum likelihood estimator of Φ.

The posterior density of the original DSGE model parameters for a given λ is proportional to the marginal likelihood in (15.24) times the prior of θ. That is

$$p\left(\theta|y, Y_1, \lambda\right) \propto p\left(y \,|Y_1, \theta, \lambda\right) p\left(\theta\right). \tag{15.27}$$

Since the marginal likelihood in (15.24) is equal to the marginal likelihood for the transformed parameters ϕ, the posterior density of the transformed DSGE model parameters is proportional to the marginal likelihood times the prior of ϕ. The latter prior is, as noted in Section 6, equal to the product of the Jacobian in the transformation from ϕ into θ and the prior of θ; see, e.g., Section 4.2.1. With $\theta = g^{-1}(\phi)$, this means that

$$p\left(\phi|y, Y_1, \lambda\right) \propto p\left(y \,|Y_1, g(\phi), \lambda\right) p\left(\phi\right), \tag{15.28}$$

where $p(\phi) = J(\phi)p(g^{-1}(\phi))$.

It is now possible to sample from the posterior distribution of ϕ for each $\lambda \in \Lambda$ by relying on one of the MCMC algorithms discussed in Section 8, on the SMC with likelihood or data tempering algorithms, or on IS based on the MitISEM algorithm. The RWM algorithm with a normal proposal density is considered in Del Negro and Schorfheide (2004, 2006) and Del Negro et al. (2007). If one of the RWM algorithms is made use of, then the proposal density from the DSGE model may be used. Alternatively, the posterior mode and inverse Hessian at the mode of ϕ can be computed from the above expressions via a numerical optimization routine, as in Section 7, and then used for the posterior sampler. If the DSGE model is severely misspecified this latter approach may result in a better proposal density. YADA allows for both possibilities and when the posterior mode exists for all λ as well as for the DSGE model, the user will be asked which approach to take. For the SMC with likelihood or data tempering algorithms, the posterior mode results are not directly used, but are nevertheless required by YADA before the sampler can be executed.

15.6. *Marginal Likelihood for a DSGE-VAR*

Once a posterior sample has been calculated for the DSGE-VAR(λ) model, the marginal likelihood is evaluated. Letting this function be denoted by $p(y|Y_1, \lambda)$, Del Negro and Schorfheide (2004) suggests to pick λ such that:

$$\hat{\lambda} = \arg\max_{\lambda \in \Lambda} p\left(y|Y_1, \lambda\right). \tag{15.29}$$

---

furthermore be simplified by noting that

$$\frac{\left|\lambda T \Gamma_{YY}(\theta) + T\hat{\Gamma}_{YY}\right|}{\left|\lambda T \Gamma_{YY}(\theta)\right|} = \frac{\left|\Gamma_{YY}(\theta) + (1/\lambda)\hat{\Gamma}_{YY}\right|}{\left|\Gamma_{YY}(\theta)\right|}.$$

Moreover,

$$\frac{\left|(1+\lambda)T\hat{\Sigma}_\epsilon(\theta)\right|^{-((1+\lambda)T-np-k)/2}}{\left|\lambda T \Sigma_\epsilon(\theta)\right|^{-(\lambda T-np-k)/2}} = \frac{\left(\lambda T\right)^{-nT/2}\left|(1+(1/\lambda))\hat{\Sigma}_\epsilon(\theta)\right|^{-((1+\lambda)T-np-k)/2}}{\left|\Sigma_\epsilon(\theta)\right|^{-(\lambda T-np-k)/2}}.$$

The expressions on the right hand side of these two relations are preferable from a numerical perspective since they are less likely to involve matrices with large numerical values.

If we assign equal probabilities to the elements of Λ, then the posterior probabilities for this hyperparameter are proportional to the marginal likelihood. This means that $\hat{\lambda}$ in (15.29) is the posterior mode of λ.[126]

The function $p(y|Y_1, \lambda)$ summarizes the time series evidence on model misspecification and documents by how much the DSGE model must be relaxed to balance within-sample fit and model complexity. To estimate the marginal likelihood we may use either the modified harmonic mean due to Geweke (1999, 2005) that was discussed in Section 10.2, or the marginal likelihood identity based estimator due to Chib and Jeliazkov (2001) that was presented in Section 10.3. The latter approach should only be used if the RWM algorithm has been used for posterior sampling.

The posterior distribution of the DSGE-VAR parameters can be computed by generating a pair $(\Phi^{(s)}, \Sigma_\epsilon^{(s)})$ from the normal-inverted Wishart distribution in (15.18)–(15.19) for each $\theta^{(s)}$ that was obtained under $\hat{\lambda}$ for $s = 1, \ldots, N$. Once these parameters have been sampled, the DSGE-VAR model can be applied to any exercise that is valid for a reduced form VAR model, such as forecasting or estimating the implied population or sample moments of the model.

As noted by Adolfson et al. (2008b), an appealing feature of the comparison of the DSGE model to DSGE-VARs is that the same prior distribution is used to weight the likelihood functions across models when forming the marginal likelihood. Bayesian model probabilities have great appeal, but they are sensitive to the choice of prior. This sensitivity may not be so large when the models are similar and the prior is elicited in a similar way. A comparison between a DSGE model and a Bayesian VAR using a statistically motivated prior, as in Section 14.1, is more likely to be sensitive to the selected priors.[127]

Even if the DSGE model does not have a finite order VAR representation, the VAR model mainly functions as a tool to relax the cross-equation restrictions and to obtain a specification with superior empirical fit. The VAR model does not have to nest the DSGE model for this analysis to remain sensible since the moments of the DSGE model that are used to form the prior on the VAR are exact regardless of how good the approximation is. This means that a large $\hat{\lambda}$ indicates that the cross product moments of the DSGE model that are used to form the prior agree well with the likelihood function.

15.7. *Posterior Mode of the DSGE-VAR*

The mode of the marginal posterior of ϕ can be determined by maximizing $p(\phi|y, Y_1, \lambda)$ in (15.28) numerically with respect to ϕ. The mode of the marginal posterior of θ is thereafter obtained by using the transformation function $\theta = g^{-1}(\phi)$ when the posterior density for the transformed parameters has been utilized; cf. Section 6. Alternatively, the marginal posterior mode of θ can be computed by maximizing (15.27) numerically with respect to the original parameters. Given that we have located the posterior mode for each $\lambda \in \Lambda$, the Laplace approximation of the marginal likelihood for $p(y|Y_1, \lambda)$ may be computed using the expression

---

[126] To elicit a proper prior for a continuous $\lambda \geq T(n(p+1)+k) = \lambda_l$ which we are willing to regard as "fair" is not a trivial problem. To see why, notice that a natural transformation of λ is $\tau = \lambda/(1+\lambda)$, where $\tau \in [\tau_l, 1]$ and $0 < \tau_l = \lambda_l/(1+\lambda_l) < 1$. The transformation is natural in the sense that it delivers a parameter which is defined over a finite interval, thus making it feasible to use a uniform distribution. Although such a prior for τ may seem fair since it gives different values an equal weight, it implies that λ has the density

$$p(\lambda|\lambda_l) = \frac{1+\lambda_l}{(1+\lambda)^2}.$$

Hence, λ is Pareto distributed, with cdf $F(\lambda|\lambda_l) = 1 - (1+\lambda_l)/(1+\lambda)$; see Section 4.2.12. Since the shape parameter (a in equation 4.34) is unity, the moments do not exists. The location parameter, λ_l, is both equal to the lower bound and to the mode of the distribution (while the origin parameter, c, is -1). Hence, this prior puts an extreme weight of values of λ close to the lower bound, λ_l, and therefore on models which are as far away as possible from the DSGE model. Moreover, the density height decreases exponentially as λ increases. While such a prior may seem appropriate among economists who think DSGE model are of little or no value, the "penalties" on models with larger λ values are extreme. Although this prior for λ is proper, it is not even in the neightborhood of being fair. In fact, it is not even in the fringes of the fringes of the fringes of satisfying such a (loose) concept.

[127] The Bayesian VAR prior is radically different from the economic prior of the DSGE model.

in equation (10.2). Although this may be a rather crude approximation unless sufficient care is taken when calculating the Hessian matrix numerically, it can nevertheless yield some information about which values of λ the data tend to favor. In fact, this is the approach taken in Adolfson et al. (2008b, Table 2) when comparing their DSGE models to DSGE-VARs with and without cointegration relations.

When $\lambda = \infty$ the VAR parameters are fully determined by θ and, hence, the joint posterior mode of all parameters is determined directly from the mode of the marginal posterior of θ. When λ is finite, however, we only approximate the mode of the joint posterior of $(\Phi, \Sigma_\epsilon, \theta)$ through the mode of the joint conditional posterior of (Φ, Σ_ϵ) if we plug in the marginal mode of θ into the relations that determine the joint conditional mode. The approximation result follows when we note that the joint conditional posterior of (Φ, Σ_ϵ) depends on θ and this dependence is not taken into account when we compute the mode of the marginal posterior of θ. Given that the marginal mode of θ is close to the joint mode of θ, the approximation of the joint posterior mode of (Φ, Σ_ϵ) can be expected to be very good.

Still, once we have determined the mode of the joint conditional posterior of the VAR parameters we can compute the concentrated likelihood for θ. From this likelihood of θ the joint mode may be computed through numerical optimization of the concentrated posterior of θ. The resulting value may then be used in the expressions determining the joint mode of the VAR parameters and, thus, provide us with the mode of the joint posterior of $(\Phi, \Sigma_\epsilon, \theta)$.

The joint posterior of $(\Phi, \Sigma_\epsilon, \theta)$ can be factorized as in (15.22), where the first term on the right hand side can be rewritten as

$$p\big(\Phi, \Sigma_\epsilon | y, Y_1, \theta, \lambda\big) = p\big(\Phi | y, Y_1, \Sigma_\epsilon, \theta, \lambda\big) p\big(\Sigma_\epsilon | y, Y_1, \theta, \lambda\big). \tag{15.30}$$

Since the full conditional posterior of Φ is given by the normal distribution, it follows that at the mode of the joint distribution Φ is equal to its mean

$$\bar{\Phi}(\theta) = \hat{\Phi}(\theta), \tag{15.31}$$

where the term on the right hand side is given in equation (15.20). Substituting this value into the first density function on the right hand side of (15.30) we obtain an expression from which the joint mode of Σ_ϵ can be determined as a function of θ. Maximizing this function with respect to the covariance matrix of the VAR residuals one arrives at

$$\bar{\Sigma}_\epsilon(\theta) = \frac{(1 + \lambda)T}{(1 + \lambda)T + n + 1} \hat{\Sigma}_\epsilon(\theta), \tag{15.32}$$

where the second term on the right hand side is given in equation (15.21). It is now straightforward to show that the mode of Σ_ϵ for the joint posterior is less than the mode of the conditional posterior density in (15.18).[128] As noted above, if we plug in the marginal mode of θ into equations (15.31) and (15.32) we may use these values as an approximation of the joint mode of the DSGE-VAR. However, to actually determine the joint mode we need to continue a few more steps.

First, substituting the mode expressions in (15.31) and (15.32) for Φ and Σ_ϵ, respectively, into (15.30) and rearranging terms we find that

$$p\big(\bar{\Phi}(\theta), \bar{\Sigma}_\epsilon(\theta) | y, Y_1, \theta, \lambda\big) = c\big(\lambda, T\big) \big| \lambda T \mathsf{T}_{YY}(\theta) + T \hat{\mathsf{T}}_{YY} \big|^{n/2}$$
$$\big| (1 + \lambda)T \hat{\Sigma}_\epsilon(\theta) \big|^{-(n(p+1)+k+1)/2}, \tag{15.33}$$

---

[128] It was noted in Section 14.1 that if $\Omega \sim IW_p(A, \nu)$, then its mode is equal to $\bar{\Omega} = (1 / (p + \nu + 1))A$. This means that the mode of the conditional posterior of Σ_ϵ in (15.18) is equal to

$$\bar{\Sigma}_\epsilon^{(c)}(\theta) = \frac{(1 + \lambda)T}{(1 + \lambda)T + n + 1 - (np + k)} \hat{\Sigma}_\epsilon(\theta) = \left[1 + \frac{np + k}{(1 + \lambda)T + n + 1 - (np + k)} \right] \bar{\Sigma}_\epsilon(\theta).$$

Accordingly, $\bar{\Sigma}_\epsilon(\theta) < \bar{\Sigma}_\epsilon^{(c)}(\theta)$.

where

$$c(\lambda, T) = \exp\left(-n\left((1+\lambda)T + n + 1\right)/2\right)\left[(1+\lambda)T + n + 1\right]^{n((1+\lambda)T+n+1)/2} \times$$
$$\pi^{-n(np+k)/2}2^{-n(1+\lambda)T/2}\pi^{-n(n-1)/4}\Gamma_n\left((1+\lambda)T - np - k\right)^{-1}.$$

When evaluating the conditional posterior of (Φ, Σ_ϵ) at the joint mode, we therefore have two terms that depend on θ and that will influence the determination of the joint posterior mode for all parameters.

Second, to estimate the mode of θ from the joint posterior of $(\Phi, \Sigma_\epsilon, \theta)$ we need to multiply the marginal likelihood of θ in (15.24) by the right hand side of (15.33). The corresponding concentrated likelihood of θ is given by:

$$p_c\left(y|Y_1, \theta, \lambda\right) = \tilde{c}\left(\lambda, T\right)\frac{\left|(1+\lambda)T\hat{\Sigma}_\epsilon(\theta)\right|^{-((1+\lambda)T+n+1)/2}}{\left|\lambda T\Gamma_{YY}(\theta)\right|^{-n/2}\left|\lambda T\Sigma_\epsilon(\theta)\right|^{-(\lambda T - np - k)/2}}, \tag{15.34}$$

where

$$\tilde{c}(\lambda, T) = \exp\left(-n\left((1+\lambda)T + n + 1\right)/2\right)\left[(1+\lambda)T + n + 1\right]^{n((1+\lambda)T+n+1)/2} \times$$
$$\pi^{-n(T+np+k)/2}2^{-n(1+\lambda)T/2}\pi^{-n(n-1)/4}\Gamma_n\left(\lambda T - np - k\right)^{-1}.$$

Finally, with m being the dimension of ϕ the last step is to solve the following numerical problem

$$\bar{\phi} = \arg\max_{\phi \in \mathbb{R}^m}\left(p_c\left(y\left|Y_1, g^{-1}(\phi), \lambda\right)p(\phi)\right), \tag{15.35}$$

while $\hat{\theta} = g^{-1}(\bar{\phi})$ gives us the joint posterior estimate of θ. It may be noted that when we compare the marginal likelihood function of θ in (15.24) to the concentrated likelihood function in (15.34) it can be seen that when multiplied by the prior, the resulting marginal posterior and concentrated posterior will have approximately the same mode once T is large enough.

Alternatively, the joint posterior mode of of the original parameters may be obtained from

$$\bar{\theta} = \arg\max_{\theta \in \Theta}\left(p_c\left(y|Y_1, \theta, \lambda\right)p(\theta)\right), \tag{15.36}$$

while the posterior estimate of the transformed parameters is given by $\hat{\phi} = g(\bar{\theta})$.

The Hessian matrix of the joint log posterior, evaluated at the mode, can either be determined numerically, or by using a combination of numerical and analytical results. Since the dimension of this matrix can be great, the latter approach is in practise recommended, especially since the matrix may need to be inverted. The joint log posterior can be expressed as:

$$\ln p\left(\Phi, \Sigma_\epsilon, \phi|y, Y_1, \lambda\right) = \ln c^*(\lambda, T) + \frac{n}{2}\ln\left(|\Gamma_{YY}(\theta)|\right) + \frac{\lambda T - np - k}{2}\ln\left(|\Sigma_\epsilon(\theta)|\right)$$
$$\frac{(1+\lambda)T + n + 1}{2}\ln\left(|\Sigma_\epsilon|\right) + \ln p(\phi) - \ln p(y|Y_1, \lambda) - \frac{(1+\lambda)T}{2}\text{tr}\left[\Sigma_\epsilon^{-1}\hat{\Sigma}_\epsilon(\theta)\right] \tag{15.37}$$
$$-\frac{T}{2}\text{tr}\left[\Sigma_\epsilon^{-1}\left(\Phi - \hat{\Phi}(\theta)\right)\left(\lambda\Gamma_{YY}(\theta) + \hat{\Gamma}_{YY}\right)\left(\Phi - \hat{\Phi}(\theta)\right)'\right],$$

where $\theta = g^{-1}(\phi)$, while $c^*(\lambda, T)$ is a function that does not affect the Hessian matrix. Using the tools for matrix differential calculus that are described in Magnus and Neudecker (1988), it can be shown that:

$$\frac{\partial^2 \ln p\left(\Phi, \Sigma_\epsilon, \phi|y, Y_1, \lambda\right)}{\partial\text{vec}(\Phi)\partial\text{vec}(\Phi)'} = -T\left[\left(\lambda\Gamma_{YY}(\theta) + \hat{\Gamma}_{YY}\right) \otimes \bar{\Sigma}_\epsilon(\theta)^{-1}\right], \tag{15.38}$$

$$\frac{\partial^2 \ln p\left(\Phi, \Sigma_\epsilon, \phi|y, Y_1, \lambda\right)}{\partial\text{vec}(\Phi)\partial\text{vech}(\Sigma_\epsilon)'} = 0, \tag{15.39}$$

$$\frac{\partial^2 \ln p\left(\Phi, \Sigma_\epsilon, \phi|y, Y_1, \lambda\right)}{\partial\text{vec}(\Phi)\partial\phi'} = T\left[\left(\lambda\Gamma_{YY}(\theta) + \hat{\Gamma}_{YY}\right) \otimes \bar{\Sigma}_\epsilon(\theta)^{-1}\right]F_\Phi(\theta), \tag{15.40}$$

where vech is the column stacking operator that only takes the elements on and below the diagonal, and the $n(np + k) \times m$ matrix $F_\Phi(\theta) = (\partial \text{vec}(\hat{\Phi}(\theta)) / \partial \phi')$ is given in equation (15.43) below. Furthermore,

$$\frac{\partial^2 \ln p\big(\Phi, \Sigma_\epsilon, \phi | y, Y_1, \lambda\big)}{\partial \text{vech}(\Sigma_\epsilon) \partial \text{vech}(\Sigma_\epsilon)'} = -\frac{(1 + \lambda)T + n + 1}{2} D_n' \left[\bar{\Sigma}_\epsilon(\theta)^{-1} \otimes \bar{\Sigma}_\epsilon(\theta)^{-1} \right] D_n, \qquad (15.41)$$

$$\frac{\partial^2 \ln p\big(\Phi, \Sigma_\epsilon, \phi | y, Y_1, \lambda\big)}{\partial \text{vech}(\Sigma_\epsilon) \partial \phi'} = \frac{(1 + \lambda)T}{2} D_n' \left[\bar{\Sigma}_\epsilon(\theta)^{-1} \otimes \bar{\Sigma}_\epsilon(\theta)^{-1} \right] D_n F_{\Sigma_\epsilon}(\theta), \qquad (15.42)$$

where D_n is the $n^2 \times n(n + 1)/2$ dimensional duplication matrix,[129] while the $n(n + 1)/2 \times m$ matrix $F_{\Sigma_\epsilon}(\theta) = (\partial \text{vech}(\hat{\Sigma}_\epsilon(\theta)) / \partial \phi')$ is provided in equation (15.44) below. Define the matrices with partial derivatives of the non-central population moments with respect to ϕ as:

$$G_{yy}(\theta) = \frac{\partial \text{vech}\big(\Gamma_{yy}(\theta)\big)}{\partial \phi'}, \quad G_{yY}(\theta) = \frac{\partial \text{vec}\big(\Gamma_{yY}(\theta)\big)}{\partial \phi'}, \text{ and } \quad G_{YY}(\theta) = \frac{\partial \text{vech}\big(\Gamma_{YY}(\theta)\big)}{\partial \phi'}.$$

These matrices have dimensions $(n(n+1)/2) \times m$, $n(np+k) \times m$, and $((np+k)(np+k+1)/2) \times m$ and may be calculated numerically. Let $D_n^+ = (D_n' D_n)^{-1} D_n'$ be the Moore-Penrose inverse of D_n. It can now be shown that the $n(np + k) \times m$ dimensional matrix $F_\Phi(\theta)$ is given by

$$\frac{\partial \text{vec}\big(\hat{\Phi}(\theta)\big)}{\partial \phi'} = \lambda \left[\big(\lambda \Gamma_{YY}(\theta) + \hat{\Gamma}_{YY}\big)^{-1} \otimes I_n \right]$$
$$\times \left[G_{yY}(\theta) - \big(I_{np+k} \otimes \hat{\Phi}(\theta)\big) D_{np+k} G_{YY}(\theta) \right]. \qquad (15.43)$$

Furthermore, the $n(n + 1)/2 \times m$ matrix $F_{\Sigma_\epsilon}(\theta)$ is given by:

$$\frac{\partial \text{vech}\big(\hat{\Sigma}_\epsilon(\theta)\big)}{\partial \phi'} = \frac{\lambda}{1 + \lambda} \left[G_{yy}(\theta) + D_n^+ \left[\hat{\Phi}(\theta) \otimes \hat{\Phi}(\theta) \right] D_{np+k} G_{YY}(\theta) \right.$$
$$\left. - 2 D_n^+ \left[\hat{\Phi}(\theta) \otimes I_n \right] G_{yY}(\theta) \right]. \qquad (15.44)$$

For the third term within large brackets on the right hand side, the result $D_n D_n^+ = (1/2)(I_{n^2} + K_{nn}) = N_n$ has been used (see Magnus and Neudecker, 1988, Theorem 3.12) so that $D_n^+ N_n = D_n^+$.

The last matrix in the partitioned Hessian at the mode is given by

$$\frac{\partial^2 \ln p\big(\Phi, \Sigma_\epsilon, \phi | y, Y_1, \lambda\big)}{\partial \phi \partial \phi'} = - T F_\Phi(\theta)' \left[\big(\lambda \Gamma_{YY}(\theta) + \hat{\Gamma}_{YY}\big) \otimes \bar{\Sigma}_\epsilon(\theta)^{-1} \right] F_\Phi(\theta)$$
$$- \frac{(1 + \lambda)T}{2} \frac{\partial^2 \text{tr}\left[\bar{\Sigma}_\epsilon(\bar{\theta})^{-1} \hat{\Sigma}_\epsilon(\theta) \right]}{\partial \phi \partial \phi'} + \frac{n}{2} \frac{\partial^2 \ln |\Gamma_{YY}(\theta)|}{\partial \phi \partial \phi'} \qquad (15.45)$$
$$+ \frac{\lambda T - np - k}{2} \frac{\partial^2 \ln |\Sigma_\epsilon(\theta)|}{\partial \phi \partial \phi'} + \frac{\partial^2 \ln p(\phi)}{\partial \phi \partial \phi'}.$$

The last four terms on the right hand side can be computed numerically. Since they are all $m \times m$ matrices, the dimensions are kept down substantially relative to a numerical Hessian for the joint log posterior. It may be noted that for the second term on the right hand side, the matrix $\bar{\Sigma}_\epsilon(\bar{\theta})$ is kept fixed at $\theta = \bar{\theta}$, while the matrix $\hat{\Sigma}_\epsilon(\theta)$ varies with θ.

For the inverse Hessian at the mode it is recommended to make use of results for partitioned matrices and inverses of Kronecker products. Let the Hessian be described by the matrix

$$H = \begin{bmatrix} H_{\Phi,\Phi} & 0 & H_{\Phi,\phi} \\ 0 & H_{\Sigma,\Sigma} & H_{\Sigma,\phi} \\ H_{\Phi,\phi}' & H_{\Sigma,\phi}' & H_{\phi,\phi} \end{bmatrix},$$

[129] The duplication matrix is defined from the relationship $\text{vec}(A) = D_n \text{vech}(A)$ for a symmetric $n \times n$ matrix A, where vech is the column stacking operator that only takes the elements on and below the diagonal.

where
$$H_{\Phi,\Phi} = \left[\left(\Gamma_{YY}(\theta) + (1/\lambda)\hat{\Gamma}_{YY} \right) \otimes \bar{\Sigma}_\epsilon(\theta)^{-1} \right],$$
i.e., we have multiplied equation (15.38) with $-1/\lambda T$. The inverse of this matrix is simply
$$H_{\Phi,\Phi}^{-1} = \left[\left(\Gamma_{YY}(\theta) + (1/\lambda)\hat{\Gamma}_{YY} \right)^{-1} \otimes \bar{\Sigma}_\epsilon(\theta) \right].$$
Similarly,
$$H_{\Sigma,\Sigma} = \frac{(1+\lambda)T + n + 1}{2\lambda T} D_n' \left[\bar{\Sigma}_\epsilon(\theta)^{-1} \otimes \bar{\Sigma}_\epsilon(\theta)^{-1} \right] D_n,$$
where the inverse is given by
$$H_{\Sigma,\Sigma}^{-1} = \frac{2\lambda T}{(1+\lambda)T + n + 1} D_n^+ \left[\bar{\Sigma}_\epsilon(\theta) \otimes \bar{\Sigma}_\epsilon(\theta) \right] D_n^{+\prime}.$$
The remaining partions of H follow from equations (15.40), (15.42), and (15.45), remembering that they should also be multiplied by $-1/\lambda T$. Now, define the matrix:
$$H_{\phi,\phi}^+ = \left(H_{\phi,\phi} - H_{\Sigma,\phi}' H_{\Sigma,\Sigma}^{-1} H_{\Sigma,\phi} - H_{\Phi,\phi}' H_{\Phi,\Phi}^{-1} H_{\Phi,\phi} \right)^{-1}.$$
The inverse matrix on the right hand side exists and is positive definite if H is positive definite. The inverse of H can now be expressed as:
$$H^{-1} = \begin{bmatrix} H_{\Phi,\Phi}^+ & H_{\Phi,\Sigma}^+ & H_{\Phi,\phi}^+ \\ H_{\Phi,\Sigma}^{+\prime} & H_{\Sigma,\Sigma}^+ & H_{\Sigma,\phi}^+ \\ H_{\Phi,\phi}^{+\prime} & H_{\Sigma,\phi}^{+\prime} & H_{\phi,\phi}^+ \end{bmatrix},$$
where the $m \times m$ $H_{\phi,\phi}^+$ matrix has already been determined. The remaining 5 partitions of the inverse of H are given by:
$$H_{\Phi,\Phi}^+ = H_{\Phi,\Phi}^{-1} + H_{\Phi,\Phi}^{-1} H_{\Phi,\phi} H_{\phi,\phi}^+ H_{\Phi,\phi}' H_{\Phi,\Phi}^{-1},$$
$$H_{\Sigma,\Sigma}^+ = H_{\Sigma,\Sigma}^{-1} + H_{\Sigma,\Sigma}^{-1} H_{\Sigma,\phi} H_{\phi,\phi}^+ H_{\Sigma,\phi}' H_{\Sigma,\Sigma}^{-1},$$
$$H_{\Phi,\Sigma}^+ = H_{\Phi,\Phi}^{-1} H_{\Phi,\phi} H_{\phi,\phi}^+ H_{\Sigma,\phi}' H_{\Sigma,\Sigma}^{-1},$$
$$H_{\Phi,\phi}^+ = -H_{\Phi,\Phi}^{-1} H_{\Phi,\phi} H_{\phi,\phi}^+,$$
$$H_{\Sigma,\phi}^+ = -H_{\Sigma,\Sigma}^{-1} H_{\Sigma,\phi} H_{\phi,\phi}^+.$$
Finally, the inverse Hessian at the mode is obtained by multiplying H^{-1} by $(1/\lambda T)$; see, e.g., Magnus and Neudecker (1988, Theorem 1.3) for the inverse of the partioned matrix H.

It is noteworthy that if the inverse Hessian is computed by combining these analytical results with numerical derivatives we find that the dimensions of the matrices that need to be inverted are $np + k$, n, and m, while the dimension of the inverse Hessian itself is $n(np + k) + n(n + 1)/2 + m$. For medium size DSGE models we can therefore expect that numerical precision can be greatly improved by using the above procedure and that the computation time itself is also shortened.

15.8. *Identifying Structural Shocks of the DSGE-VAR*

To undertake counterfactual exercises that require the identification of structural shocks, we need to be able to identify these shocks to the DSGE-VAR. Suppose that $v_t \sim N_n(0, I_n)$ are these shocks and that they are related to the VAR residuals ϵ_t through
$$\epsilon_t = A_0 v_t. \tag{15.46}$$
The non-singular A_0 matrix is of dimension $n \times n$ and each column is equal to the contemporaneous response in y_t from a unit impulse to the corresponding element of v_t. To be exactly identified we need to impose $n(n-1)/2$ identifying restrictions in addition to the $n(n+1)/2$ that have already been implicitly imposed through the assumed covariance matrix of v_t. The

matrix A_0 can always be decomposed such that:

$$A_0 = \Sigma_\epsilon^{1/2}\Upsilon, \tag{15.47}$$

where $\Sigma_\epsilon^{1/2}$ is the lower triangular Choleski decomposition of Σ_ϵ, while Υ is an orthogonal matrix ($\Upsilon\Upsilon' = I_n$) that cannot be identified from the likelihood function of ϵ_t. The identification problem therefore boils down to selecting an Υ matrix.

Del Negro and Schorfheide (2004) suggests that Υ can be determined from the DSGE model. Specifically, from equation (11.18) we know that the contemporaneous response in y_t from unit impulses to η_t are

$$A_0(\theta) = H'B_0. \tag{15.48}$$

It is possible to determine an $n \times n$ matrix $\Upsilon(\theta)$ from $A_0(\theta)$ if $q = n$, i.e., the number of economic shocks in the DSGE model is equal to the number of observed variables. Del Negro and Schorfheide suggest using a Q-R factorization of $A_0(\theta)$; see, e.g., Golub and van Loan (1983) for details on the Q-R factorization.[130] That is, they let

$$A_0(\theta) = \Sigma_\epsilon^{1/2}(\theta)\Upsilon(\theta), \tag{15.49}$$

where $\Sigma^{1/2}(\theta)$ is lower triangular and $\Upsilon(\theta)$ is orthogonal.[131] To determine Υ Del Negro and Schorfheide (2004) suggests to let it be equal to $\Upsilon(\theta)$. This means that the rotation matrix Υ is chosen such that, in the absense of misspecification, the contemporaneous impulse responses of the DSGE and the DSGE-VAR are approximately equal for all shocks. When sampling from the posterior this means that $\Upsilon^{(s)} = \Upsilon(\theta^{(s)})$ for $s = 1, \ldots, N$.

15.9. *YADA Code*

This section contains information about the YADA functions that compute the marginal and the joint posterior mode, as well as the functions that handle posterior sampling. The functions are grouped into those related to computing parameters, such as the prior mean and covariance of the VAR parameters, those that deal with posterior densities, functions that handle estimation, and finally functions dealing with sampling.

15.9.1. *Parameter Functions*

15.9.1.1. DSGEVARPrior

The function DSGEVARPrior takes 11 input variables: A, H, R, F, B0, DetProductMoments, p, initP, MaxIter, Tolerance, and HSample. The first five variables are simply the matrices in the state-space representation of the DSGE model. The next variable is a 3D matrix ($k \times k \times p + 1$) with product moments for the deterministic variables; see equation (15.5) in Section 15.1.[132] The lag order is given by p, while the next three input variables determine how the unconditional state covariance matrix Σ_ξ is calculated and are discussed in Section 5.17.1 for the Kalman filter implementations in YADA. Finally, the vector HSample determines the sample to use for a time-varying H matrix.

The output variables are given by Gammayy, GammaYY, GammayY, and PriorStatus. The three gamma variables concern the $\Gamma_{yy}(\theta)$, $\Gamma_{YY}(\theta)$, and $\Gamma_{yY}(\theta)$ matrices that are needed to parameterize the prior of the DSGE-VAR. The last variable, PriorStatus, is a boolean variable that takes the value 1 if GammaYY is invertible and 0 otherwise.

---

[130] If the H matrix is time-varying we may replace it with its sample average.

[131] The Q-R factorization of an $n \times n$ matrix A of rank n is given by $A = QR$. In practise, the $n \times n$ matrix Q is orthogonal, while the $n \times n$ matrix R is upper triangular; see, e.g., Golub and van Loan (1983, p. 147) or the qr function in Matlab. Hence, to obtain the matrices in (15.49) we instead compute the Q-R factorization for $A_0(\theta)'$. Moreover, since some diagonal elements of R may be negative it is necessary to premultiply this matrix with a diagonal $n \times n$ matrix S, whose diagonal entries are 1 (-1) when the corresponding diagonal elements of R are positive (negative). The resulting matrix SR is upper triangular with only positive diagonal elements and is therefore a suitable candidate for $\Sigma^{1/2}(\theta)'$. Furthermore, we need to postmultiply Q with S. Since S is orthogonal it follows that QS is also orthogonal and may be used as $\Upsilon(\theta)'$.

[132] These moments are calculated by the function CalcDeterministicProductMoments.

15.9.1.2. GetDSGEVARPriorParameters

The function `GetDSGEVARPriorParameters` needs 8 input variables: `theta`, `thetaPositions`, `ModelParameters`, `PriorDist`, `thetaDist`, `LowerBound`, `DSGEModel`, and `CurrINI`. These are needed to solve the DSGE model and provide the state-space form at the value θ of the original DSGE model parameters; see Section 11.18.

As output the function returns 11 variables. First of all, the matrices `Phi` and `SigmaEpsilon` with the prior parameters in equations (15.13) and (15.14). The following 5 output variables are the matrices from the state-space representation of the DSGE model, `A`, `H`, `R`, `F`, and `B0`. Next, the variable `AIMData` is given, which is followed by `DetProductMoments` and `HSample` that are input variables for `DSGEVARPrior` above. Finally, the function provides a boolean `Status` variables which is unity if the prior parameters could be computed and 0 otherwise.

15.9.1.3. DSGEVARParameters

The function `DSGEVARParameters` requires 15 input variables: `A`, `H`, `R`, `F`, `B0`, `lambda`, `GammaHatyy`, `GammaHatyY`, `GammaHatYY`, `DetProductMoments`, `p`, `initP`, `MaxIter`, `Tolerance`, and `HSample`. Most of these variables are discussed above for the `DSGEVARPrior` function. The variable `lambda` is equal to the λ hyperparameter that determines how closely the DSGE-VAR model is to the VAR approximation of the DSGE model. The variables `GammaHatyy`, `GammaHatyY` and `GammaHatYY` are simply the non-central sample product moment matrices $\hat{\Gamma}_{yy}$, $\hat{\Gamma}_{yY}$, and $\hat{\Gamma}_{YY}$.

The function provides 6 output variables: `Phi`, `SigmaEpsilon`, `Gammayy`, `GammaYY`, `GammayY`, and `Status`. The first two matrices are equal to $\hat{\Phi}(\theta)$ and $\hat{\Sigma}_\epsilon(\theta)$ in equations (15.20) and (15.21), while the next three output variables are given by the `DSGEVARPrior` function. The final output, `Status`, is optional and takes the value 1 if all the calculations could be performed, and 0 otherwise.

15.9.1.4. DSGEVARIdentifyShocks

The function `DSGEVARIdentifyShocks` need 5 input variables to perform its task. They are: `SigmaEpsilon`, `H`, `B0`, `DSGEVARShocks`, and `HSample`. The first variable is a value for the residual covariance matrix Σ_ϵ, while the `H` and `B0` matrices determine the contemporaneous response of the observed variables to the shocks in the DSGE model, i.e., the H and B_0 matrices discussed in Section 15.8. The variable `DSGEVARShocks` is a vector determining which of the DSGE model shocks that should be used in the DSGE-VAR. The number of shocks to the VAR is equal to the number of endogenous variables. Since the number of DSGE model shocks can be greater than the number of observed variables, the vector `DSGEVARShocks` ensures that only a subset is used. Finally, the vector `HSample` determines the sample to use for a time-varying H matrix.

The function provides 2 output variables. The first is `A0`, the $n \times n$ matrix with contemporaneous responses in the DSGE-VAR to the n identified shocks. This means that the matrix is equal to $\Sigma_\epsilon^{1/2} \Upsilon(\theta)$, where the latter matrix is given by the factorization in (15.49). The second output variables is `IdentifyStatus`, a boolean that takes the value 1 if `A0` could be determined and 0 otherwise.

15.9.2. *Density Functions*

15.9.2.1. logPosteriorPhiDSGEVAR

The function `logPosteriorPhiDSGEVAR` computes the log marginal posterior for the transformed parameters. The function needs 22 variables to achieve this. First, it takes 10 variables needed to solve DSGE model and compute the value of the prior for the transformed parameters. These variables are: `phi`, `thetaIndex`, `UniformBounds`, `LowerBound`, `thetaPositions`, `thetaDist`, `PriorDist`, `ModelParameters`, `DSGEModel`, and `AIMData`. In addition, the function needs DSGE-VAR related input variables, i.e, `lambda`, `T`, `n`, `p`, `npk`, `GammaHatyy`, `GammaHatyY`, `GammaHatYY`, `DetProductMoments`, `HSample`, `logGPR`, and `OrderQZ`. The variable `T` is the sample size, `n` the number of endogenous variables, `npk` the number of explanatory variables per equation of the

VAR ($np + k$). Finally, the variable `logGPR` is the log of the gamma product ratio in equation (15.24), i.e., $\ln\Gamma_n((1+\lambda)T - np - k) - \ln\Gamma_n(\lambda T - np - k)$.

The function provides one required output variable, `logPost`, i.e., minus the height of the log posterior density up to the constant determined by the value of the log marginal likelihood. In addition, the `logPosteriorPhiDSGEVAR` function also supports the optional `logLike` output variable. It is equal to `logLikeValue` when all computations could be carried out successfully, and to `NaN` otherwise.

15.9.2.2. `logPosteriorThetaDSGEVAR`

The function `logPosteriorThetaDSGEVAR` computes the log marginal posterior for the original parameters. Like its counterpart `logPosteriorPhiDSGEVAR` it needs 22 input variables and, with the exception of `phi` and `UniformBounds`, they are identical. Specifically, these two variables are replaced with `theta` and `ParameterBounds`. The first is the vector of original parameters, while the second is a matrix with the lower and upper bounds for the parameters in the columns.

The output variables are identical to those provided by `logPosteriorPhiDSGEVAR`.

15.9.2.3. `logLikelihoodDSGEVAR`

The function `logLikelihoodDSGEVAR` computes the value of the log of the marginal likelihood in equation (15.24) for finite values of λ. To accomplish this it needs 15 input variables: `ModelParameters`, `DSGEModel`, `AIMData`, `lambda`, `T`, `n`, `p`, `npk`, `GammaHatyy`, `GammaHatyY`, `GammaHatYY`, `DetProductMoments`, `HSample`, `logGPR`, and `OrderQZ`. These variables have already been discussed above as well as in Section 15.9.1 and in Section 7.4.

The function provides 4 output variables. The first is `logLikeValue`, the value of the log marginal likelihood at the given of θ, the DSGE model parameter. Next, it provides `Solution`, a structure with fields giving the different matrices from the solution to the DSGE model, `mcode` to indicate if the DSGE model has a unique convergent solutions at θ. Finally, it gives `PriorStatus`, a boolean variable that is 1 if the log-likelihood could be calculated and 0 otherwise.

15.9.2.4. `logLikelihoodDSGEVARInf`

The function `logLikelihoodDSGEVARInf` computes the value of the log marginal likelihood in equation (15.26) for $\lambda = \infty$. The function needs 12 input variables to achieve this. Namely, the same variables as `logLikelihoodDSGEVAR` except `lambda`, `npk`, and `logGPR`. The output variables are the same as `logLikelihoodDSGEVAR`.

15.9.2.5. `logConcPosteriorPhiDSGEVAR`

The function `logConcPosteriorPhiDSGEVAR` computes the value of the concentrated log posterior for the transformed parameters using the concentrated likelihood in (15.34). The function takes the same 22 input variables as `logPosteriorPhiDSGEVAR`, except for `logGPR` being replaced with `logCLC`, the log of the constant term for the concentrated likelihood. The latter variable is determined by `DSGEVARLogConcLikelihoodConstant` as requires the input variables `lambda`, `T`, `n`, and `npk`.

The function provides one required output variable, `logPost`, i.e., minus the height of the log of the concentrated posterior density of the transformed parameters up to the constant determined by the value of the log marginal likelihood. At the posterior mode of the DSGE model parameters, this value is equal to minus the height of the log of the joint posterior density up to the constant determined by the value of the log marginal likelihood. An optional output variable can also be extracted from the function called `logLike`, which gives the logged value of the concentrated likelihood.

15.9.2.6. `logConcPosteriorThetaDSGEVAR`

The function `logConcPosteriorThetaDSGEVAR` calculates the value of the concentrated log posterior for the original parameters based on the expression on the right hand side of equation (15.36). The function takes the same 22 input variables as `logPosteriorThetaDSGEVAR`, except for `logGPR` being replaced with `logCLC`.

The function only provides one erquired output variable, `logPost`, i.e., minus the height of the log of the concentrated posterior density of the original parameters up to the constant determined by the value of the log marginal likelihood. An optional output variable, `logLike`, is available and is identical to the optional output variable of `logConcPosteriorPhiDSGEVAR`.

15.9.2.7. `logConcLikelihoodDSGEVAR`

The function `logConcLikelihoodDSGEVAR` calculates the value of the centrated log-likelihood in (15.34) for finite λ. It makes use of the same 15 input variables as `logLikelihoodDSGEVAR`, except for `logGPR` being replaced with `logCLC`. The function also gives the same 4 output variables as `logLikelihoodDSGEVAR`.

15.9.2.8. *Additional Density Function*

When computing the marginal or joint posterior mode with Marco Ratto's `newrat`, the YADA implementation needs log posteriors and log-likelihoods that have names ending with `4Time` and which provide as a second output variable the time t values of the respective function. For the log posteriors, these functions have exactly the same input variables as their counterparts, e.g., `logPosteriorPhiDSGEVAR4Time` has the same input variables as `logPosteriorPhiDSGEVAR`. For the log-likelihood functions, they also require the input variables `y` and `Y` with the actual data on the endogenous and the deterministic and lagged endogenous variables, respectively.

To compute the time t log-likelihood based on the marginal likelihood in (15.24) the code utilizes a very simple approach. Namely, it takes into account that this likelihood is equal to the ratio of the likelihood for a sample t observations and $t-1$ observation. The log-likelihood for periods t can thus be computed recursively once the full sample value has been determined. It should be noted that when $\lambda t <= n(p+1)+k-1$ then the time $t+1$ value of the log-likelihood can no longer be computed since the gamma function is only defined for positive values; see Section 4.2.2. Furthermore, the same approach is used when time t values for the concertrated likelihood function are computed based on equation (15.34).

15.9.3. *Estimation Functions*

15.9.3.1. `DSGEVARMargPosteriorModeEstimation`

The function `DSGEVARMargPosteriorModeEstimation` can estimate the marginal posterior mode of either the transformed DSGE model parameters or the original parameters through the lens of the DSGE-VAR for each λ value that is determined in the data construction file; see Section 18.5.5. The function therefore works in much the same way as `PosteriorModeEstimation` which deals with posterior mode estimation of these parameters for the DSGE model; see Section 7.4. The input variables of these two functions are identical.

In contrast to `PosteriorModeEstimation`, the DSGE-VAR marginal posterior mode estimation routine provides one output variable. Namely, the vector `MarginalLambda` which holds the positions in the vector `Lambda` of the λ values the user selected to use. This makes it possible to consider different λ values for the marginal and the joint posterior mode estimation. Moreover, it allows the user to estimate the posterior mode in suitable batches.

Apart from computing the posterior mode of ϕ following the procedure laid out in posterior mode estimation function for the DSGE model, the function plugs these values into the posterior mode expressions for the VAR parameters in equations (15.31) and (15.32) and compares the log marginal likelihood across the λ values to determine which λ gives the DSGE-VAR has the largest posterior probability. The marginal likelihood is here computed using the Laplace approximation in (10.2), where $\ln L(Y; g^{-1}(\tilde{\phi}))$ is replaced with $\ln L(y|Y_1, g^{-1}(\tilde{\phi}), \lambda)$ for each $\lambda \in \Lambda$. These log marginal likelihood values are thereafter plotted and, if available, compared with the log marginal likelihood value for the DSGE model. Again, the Laplace approximation is used and this value is available if the posterior mode estimation has been completed for the DSGE model.

15.9.3.2. `DSGEVARPosteriorModeEstimation`

The function `DSGEVARPosteriorModeEstimation` estimates the joint posterior mode of the VAR parameters (Φ and Σ_ϵ) and either the transformed DSGE model parameters (ϕ) or the original parameters (θ). It operates is the same manner as the function that computes the marginal posterior mode in `DSGEVARMargPosteriorModeEstimation`. The input variables are the same and the type of output that is produced is similar, except that it refers to the joint posterior mode. In particular, the output variable is given by `JointLambda`, a vector with the positions in the vector `Lambda` that the user selected to use for joint posterior mode estimation.

15.9.3.3. `DSGEVARJointPosteriorInvHessian`

The function `DSGEVARJointPosteriorInvHessian` attempts to compute the inverse Hessian at the posterior mode of the joint posterior distribution of all the DSGE-VAR parameters using a combination of analytical results and numerical approximations; see Section 15.7. The function takes the same 21 input variables as `logConcPosteriorPhiDSGEVAR`, except for `logCLC` being replaced with `StepLength`. The latter variable determines the step length when computing numerical approximations of the matrices with second partial derivatives in equation (15.45).[133]

15.9.4. *Sampling Functions*

15.9.4.1. `DSGEVARPriorSampling`

The function `DSGEVARPriorSampling` computes a sample of draws from the prior distribution of the DSGE (θ) and VAR (Φ, Σ_ϵ) parameters. To achieve its objective it requires 9 input variables: `theta`, `thetaPositions`, `ModelParameters`, `PriorDist`, `thetaDist`, `LowerBound`, `NumPriorDraws`, `DSGEModel`, and `CurrINI`. All these variables are familiar with the exception of `NumPriorDraws`, which is an integer determining to total number of draws from the joint prior distribution $p(\Phi, \Sigma_\epsilon, \theta | \lambda)$. The function computes the prior draws for all $\lambda \in \Lambda$ and saves them to disk.

15.9.4.2. `DSGEVARRWMPosteriorSampling` & `DSGEVARFixedBlockingRWMPosteriorSampling` & `DSGEVARRandomBlockingRWMPosteriorSampling`

The functions `DSGEVARRWMPosteriorSampling` and `DSGEVARRWMStudentPosteriorSampling` for the full parameter vector, the functions `DSGEVARFixedBlockingRWMPosteriorSampling` and `DSGEVARFixedBlockingRWMStudentPosteriorSampling` for a fixed number of blocks of the parameters, and the functions `DSGEVARRandomBlockingRWMPosteriorSampling` (for a normal proposal) and `DSGEVARRandomBlockingRWMStudentPosteriorSampling` (for a Student-t proposal) for randomly drawn number of parameter blocks operate in much the same way as the corresponding RWM sampling functions for the DSGE model; see Section 8.7.2. In addition to the input variables accepted by the function `DSGERWMPosteriorSampling` and its siblings, the DSGE-VAR versions require the `maingui` and `controls` input variables. The first variable is the handle to the YADA dialog, typically taking the value 1, while the second is the structure with handles to all the controls on the YADA dialog. Optionally, these functions also accept the `SelectedMode` and `lambdai` input variables. The former takes the values 1, 2, or 3, representing the DSGE model posterior mode, the DSGE-VAR marginal posterior mode, and the DSGE-VAR joint posterior mode, respectively. In addition, the second optional input variable is simply the value of the λ parameter. The optional input variables are only used when sequential estimation is performed.

---

[133] The matrices with second partial derivatives are computed as in Abramowitz and Stegun (1964, equations 25.3.24 and 25.3.27, p. 884). This classic book is also hosted online: see, for instance, the homepage of Colin Macdonald at the University of British Columbia. The Hessian is thereafter examined and, if needed, corrected with the approach in Gill and King (2004).

Corresponding to the two optional input variables, the DSGE-VAR posterior samplers also accept two optional output variables, called `SelMode` and `Sellambdai`. Although the names differ slightly, these variables typically match the two optional input variables.

Posterior sampling of the DSGE model parameters (θ) through the lens of the VAR model is always performed for a certain value of the λ hyperparameter. When determining the location and scale parameters of the proposal density for the RWM algorithm, the function checks which posterior mode results exist on disk. The function can make use of the posterior mode results for the DSGE model, the marginal or the joint posterior mode results for the DSGE-VAR.

The functions can compute the marginal likelihood for the DSGE-VAR using either one of the modified harmonic mean approaches (Section 10.2) or the Chib and Jeliazkov marginal likelihood identity based estimator (Section 10.3). In addition, the functions can compute the conditional marginal likelihood of the DSGE model when draws from the posterior distribution of the DSGE model exist on disk by using a training sampling with minimum length p.

The calculation of the conditional marginal likelihood for the DSGE model is performed by the function `CondMargLikeModifiedHarmonic` for the modified harmonic mean estimator, and by `CondMargLikeChibJeliazkov` for the Chib and Jeliazkov estimator. In addition, when the latter estimator is used to compute the marginal likelihood for the DSGE-VAR, the function `MargLikeChibJeliazkovDSGEVAR` is employed. This function is identical to the function employed by the DSGE model (`MargLikeChibJeliazkov`) except that it calls the log posterior of the DSGE-VAR (`logPosteriorPhiDSGEVAR`) instead of the log posterior of the DSGE model (`logPosteriorPhiDSGE`) to evaluate the α function; see the denominator in equation (10.20).

15.9.4.3. `DSGEVARSlicePosteriorSampling`

The function `DSGEVARSlicePosteriorSampling` performs posterior sampling with the slice sampling algorithm. It uses the same input variables as `DSGEVARRWMPosteriorSampling` and behaves in essentially the same way as the RWM function except for the actual sampling part. Moreover, while the RWM keeps track of the acceptance rate, the slice sampler counts the number of times the log posterior is evaluated. Moreover, when the posterior draws are obtained through the slice sampler, YADA will only compute the log marginal likelihood of the DSGE-VAR as well as of the DSGE model with the modified harmonic mean estimator.

15.9.4.4. `DSGEVARSMCLikelihoodTemperingPosteriorSampler` & `DSGEVARSMCLikelihoodTemperingPosteriorSampler`

The function `DSGEVARSMCLikelihoodTemperingPosteriorSampler` runs the sequential Monte Carlo with likelihood tempering posterior sampler for a DSGE-VAR model, while the data tempering sampler is handled by `DSGEVARSMCDataTemperingPosteriorSampler`. These function behave similarly as the DSGE model equivalents. The functions use the same input variables as the above DSGE-VAR posterior samplers, and provide the same optional output variables.

15.9.4.5. `DSGEVARISMitISEMPosteriorSampler`

The function `DSGEVARISMitISEMPosteriorSampler` handles importance sampling based on the MitISEM algorithm for estimating the candidate density when using a DSGE-VAR model. The function behaves much the same as the DSGE model version. It uses the same input variables as the other DSGE-VAR model-based posterior samplers of the DSGE model parameters, and, of course, gives the same optional output variables.

15.9.4.6. `DSGEVARPosteriorSampling`

The function `DSGEVARPosteriorSampling` performs posterior sampling of the VAR parameters. It takes the same input variables as the posterior samplers for the DSGE model parameters and computes draws from the conditional posteriors of Σ_ϵ and Φ; cf. equations (15.18) and (15.19) in Section 15.4. Since the dimensions of the VAR parameter matrices can be huge, posterior sampling of the VAR parameters can use a fraction of the θ parameters. The size of the subset is determined by the "percentage use of posterior draws for impulse responses, etc" option on the *Posterior Sampling* frame on the *Options* tab; see Figure 4.

16. Analysing the Properties of a DSGE-VAR

This section turns to analyses of the properties of the observed variables from the DSGE model when studied through the DSGE-VAR. We shall first turn to matters such as estimation of the structural shocks, impulse response functions, forecast error variance decompositions, and observed variable decompositions. All these analytical tools require that the structural shocks in the model can be identified, i.e., that the number of economic shocks in the DSGE model, q, is equal to or greater than the number of observed variables, n.

Finally, we shall discuss forecasting within the DSGE-VAR framework. The discussion largely relies on the results already presented in Section 12 concerning the Bayesian VAR and we shall cover both unconditional and conditional forecasts.

16.1. *Estimation of the Economic Shocks in the DSGE-VAR*

Identification of the economic shocks in the DSGE-VAR was discussed above in Section 15.8. Provided that the number of economic shocks in the DSGE model, q, is at least equal to the number of observed variables, it is possible to select a subset of these shocks and estimate them through the DSGE-VAR. Let v_t be the selected subset from η_t, while we stick to denoting B_0 as the matrix of parameters on v_t in the DSGE model.

By substituting for ϵ_t in equation (15.9) using (15.46), inverting A_0 and rearranging terms, the economic shocks are determined by:

$$v_t = A_0^{-1} y_t - \sum_{j=1}^{p} A_0^{-1} \Phi_j y_{t-j} - A_0^{-1} \Phi_0 x_t, \quad t = 1, \ldots, T, \tag{16.1}$$

where $A_0^{-1} = \Upsilon' \Sigma_\epsilon^{-1/2}$. These estimates may be compared to either the update estimates of shocks in η_t that are included in v_t or to the smooth estimates.

16.2. *Impulse Response Functions*

Provided that the structural shocks, v_t, are uniquely identified, it is straightforward to compute the responses in the endogenous variables from impulses to these shocks. Let us first rewrite the VAR in (15.9), taking (15.46) into account, in first order form:

$$Y_t = J_p \Phi_0 x_t + \Psi Y_{t-1} + J_p A_0 v_t, \tag{16.2}$$

where $Y_t = [y_t' \cdots y_{t-p+1}']'$ is now an $np \times 1$ dimensional vector. The matrix J_p has dimension $np \times n$ with I_n on top and zeros below such that $y_t = J_p' Y_t$. The $np \times np$ matrix Ψ is given by

$$\Psi = \begin{bmatrix} \Phi_1 & \cdots & \Phi_{p-1} & \Phi_p \\ I_n & & 0 & 0 \\ & \ddots & & \\ 0 & & I_n & 0 \end{bmatrix}. \tag{16.3}$$

Suppose that $v_t = e_j$ and zero thereafter, with e_j being the j:th column of I_n. That is, we shall consider a one standard deviation impulse for the j:th structural shock. From equation (16.2) it now follows that the responses of the endogenous variables are:

$$\text{resp}\left(y_{t+h} \,\middle|\, v_t = e_j \right) = J_p' \Psi^h J_p A_0 e_j, \quad h \geq 0. \tag{16.4}$$

Provided that y_t is covariance stationary, the responses of the endogenous variables tend to zero as the response horizon h increases.

When some of the endogenous variables are expressed in first differences, we can also compute the levels responses of all observed variables. As in Section 11.18.2, let C be an $n \times n$ diagonal matrix with 0 (1) in diagonal element i if endogenous variable i is measured in levels (first differences). This means that the levels responses to the shock $v_t = e_j$ are given by:

$$\text{resp}\left(y_{t+h}^L \,\middle|\, v_t = e_j \right) = C \cdot \text{resp}\left(y_{t+h-1}^L \,\middle|\, v_t = e_j \right) + J_p' \Psi^h J_p A_0 e_j, \quad h \geq 1, \tag{16.5}$$

with $\text{resp}(y_t^L | v_t = e_j) = A_0 e_j$. The matrix C acts as an indicator for the variables where an accumulation of the responses in (16.4) gives the responses in the levels.

16.3. *Forecast Error Variance Decompositions*

Forecast error variance decomposition in DSGE-VAR models can be performed just like in any other VAR model. From Section 11.4 we know that the conditional variance decompositions of the DSGE model function in a similar way. Let

$$R_i = J_p' \Psi^i J_p A_0,$$

be the $n \times n$ matrix with all impulse responses in y for period i, the h-step ahead forecast error variance decomposition can be expressed as the $n \times n$ matrix v_h given in equation (11.24) with $q = n$.

The variance decompositions for the levels (or the accumulation of the observed variables) can also be computed as in Section 11.4. Moreover, the long-run forecast error variance decomposition for the levels is given by equation (11.30) with

$$R_{lr} = \sum_{i=0}^{\infty} J_p' \Psi^i J_p A_0 = J_p' \left(I_{np} - \Psi \right)^{-1} J_p A_0.$$

16.4. *Historical Decomposition of the Endogenous Variables*

Given the estimates of the structural shocks in equation (16.1) we can compute a historical decomposition of the endogenous variables. By substituting for Y_{t-i} recursively in (16.2) it can be shown that

$$y_t = \sum_{i=0}^{t-1} J_p' \Psi^i J_p \Phi_0 x_{t-i} + J_p' \Psi^t Y_0 + \sum_{i=0}^{t-1} J_p' \Psi^i J_p A_0 v_{t-i}, \quad t = 1, \dots, T. \tag{16.6}$$

The endogenous variables can thus be decomposed into three terms given by (i) the deterministic variables, (ii) the impact of the initial value (Y_0), and (iii) the structural shocks. This is analogous to the historical observed variable decomposition in (11.57); cf. Section 11.8.

When $x_t = 1$ for all t, the above can be developed further. Specifically, we then know that the VAR based population mean of y_t conditional on the parameters is given by

$$\mu_y = J_p' \left(I_{np} - \Psi \right)^{-1} J_p \Phi_0. \tag{16.7}$$

The decomposition in (16.6) can now be expressed as

$$y_t = \mu_y + J_p' \Psi^t \left(Y_0 - \left[\iota_p \otimes I_n \right] \mu_y \right) + \sum_{i=0}^{t-1} J_p' \Psi^i J_p A_0 v_{t-i}, \quad t = 1, \dots, T, \tag{16.8}$$

where ι_p is a $p \times 1$ vector with ones. The historical decomposition of the endogenous variables is now given by (i) its population mean, (ii) the initial value in deviation from its population mean, and (iii) the structural shocks.

The decompositions can be generalized into decompositions for all possible subsamples $\{t_0 + 1, \dots, T\}$, where $t_0 = 0, 1, \dots, T-1$. For arbitrary point of initialization, t_0, the decomposition in (16.6) gives us

$$y_t = J_p' \Psi^{t-t_0} Y_{t_0} + \sum_{i=0}^{t-t_0-1} J_p' \Psi^i J_p \Phi_0 x_{t-i} + \sum_{i=0}^{t-t_0-1} J_p' \Psi^i J_p A_0 v_{t-i}, \quad t = t_0 + 1, \dots, T, \tag{16.9}$$

while (16.8) can be generalized as

$$y_t = \mu_y + J_p' \Psi^{t-t0} \left(Y_{t_0} - \left[\iota_p \otimes I_n \right] \mu_y \right) + \sum_{i=0}^{t-t_0-1} J_p' \Psi^i J_p A_0 v_{t-i}, \quad t = t_0 + 1, \dots, T. \tag{16.10}$$

We now find that the endogenous variables are decomposed into (i) deterministic variables, (ii) the history of the endogenous variables until period t_0, and (iii) the structural shocks from period t_0 until period t.

16.5. Observed Variable Correlations

Let $\mu_{y_t} = E[y_t | \Phi, \Sigma_\epsilon]$ be the population mean of the endogenous variables conditional on the parameters (Φ, Σ_ϵ) and the exogenous variables. In case $x_t = 1$ this vector is equal to μ_y in equation (16.7). Defining $z_t = y_t - \mu_{y_t}$, the stacked DSGE-VAR in equation (16.2) can be rewritten as

$$Z_t = \Psi Z_{t-1} + J_p \epsilon_t, \tag{16.11}$$

where $\epsilon_t = A_0 v_t$ has been utilized. Let $\Sigma_y(h) = E[z_t z'_{t-h} | \Phi, \Sigma_\epsilon]$ be the population covariance matrix of the endogenous variables, while the $np \times np$ contemporaneous covariance matrix of Z_t is given by

$$\Sigma_Z = \begin{bmatrix} \Sigma_y(0) & \Sigma_y(1) & \cdots & \Sigma_y(p-1) \\ \Sigma_y(1)' & \Sigma_y(0) & & \Sigma_y(p-2) \\ \vdots & & \ddots & \vdots \\ \Sigma_y(p-1)' & \Sigma_y(p-2)' & \cdots & \Sigma_y(0) \end{bmatrix}. \tag{16.12}$$

It follows from (16.11) that this covariance matrix satisfies the Lyapunov equation

$$\Sigma_Z = \Psi \Sigma_Z \Psi' + J_p \Sigma_\epsilon J'_p. \tag{16.13}$$

From this expression we can determine $\Sigma_y(j)$ for $j = 0, \ldots, p-1$. For all other autocovariances we have that

$$\Sigma_y(j) = \sum_{i=1}^{p} \Phi_i \Sigma_y(j-i), \quad j = p, p+1, \ldots.$$

Rather than calculating population based autocovariances, we can instead simulate data from the DSGE-VAR and compute sample based estimates of the autocovariances from this data. Since $\epsilon_t \sim N(0, \Sigma_\epsilon)$ we simulate a path for the endogenous variables by drawing T values for $\epsilon_t^{(s)}$ from its distribution and letting

$$Z_t^{(s)} = \Psi Z_{t-1}^{(s)} + J_p \epsilon_t^{(s)}, \quad t = 1, \ldots, T.$$

In case $x_t = 1$ we may draw $Z_0^{(s)}$ from $N(0, \Sigma_Z)$, while a time varying mean can be handled by simply letting $Z_0^{(s)} = Z_0$. Finally, we use the relationship between y_t and Z_t such that

$$y_t^{(s)} = \mu_{y_t} + J'_p Z_t^{(s)}, \quad t = 1, \ldots, T.$$

The autocovariances can now be estimated directly from the simulated data, taking only the deterministic variables x_t into account. By repeating this S times, we obtain a sample based distribution of the autocovariances for a given (Φ, Σ_ϵ).

16.6. Conditional Correlations and Correlation Decompositions

The calculation of conditional correlations from a (log-linearized) DSGE model via its state-space representation were discussed in Section 11.7. Below we shall address how such correlations can be computed from a DSGE-VAR model.

The conditional correlations are based on letting all structural shocks be zero except for shock j. With A_{0j} being the j:th column of A_0, the conditional covariance matrix $\Sigma_y^{(j)}(0)$ can be computed from $\Sigma_Z^{(j)}$ in (16.12) using (16.13) where Σ_ϵ has been replaced with $A_{0j} A'_{0j}$.

From these population-based conditional covariances we may compute the correlation decompositions for the DSGE-VAR using the same tools as in Section 11.7. Relative to equation (11.54), the only change is that the (conditional) covariance matrices of the DSGE-VAR are used instead of the (conditional) covariance matrices of the DSGE model.

Instead of computing such population-based conditional central second moments, we can make use of simulation methods to obtain estimates of the sample moments. Since $v_t \sim N(0, I_n)$ we can simulate a path for Z conditional on only shock j being non-zero by drawing T values

for $v_{j,t}^{(s)}$ from a standard normal and letting

$$Z_t^{(s)} = \Psi Z_{t-1}^{(s)} + J_p A_{0j} v_{j,t}^{(s)}, \quad t = 1, \ldots, T. \tag{16.14}$$

In case $x_t = 1$ we may draw $Z_0^{(s)}$ from $N(0, \Sigma_Z^{(j)})$, while a time varying mean can be handled by simply letting $Z_0^{(s)} = Z_0$. Like for the autocovariances, we may take the simulation one step further. Namely, we let $y_t^{(s)} = \mu_{y_t} + J_p' Z_t^{(s)}$ and then estimate $Z_t^{(s)}$ by regressing $y_t^{(s)}$ on x_t and stacking it. With $\hat{Z}_t^{(s)}$ denoting the estimated stacked vector, the sample estimate of the conditional covariance matrix for simulation s is given by

$$\hat{\Sigma}_Z^{(j,s)} = \frac{1}{T} \sum_{t=1}^{T} \hat{Z}_t^{(s)} \hat{Z}_t^{(s)\prime}, \quad s = 1, \ldots, S. \tag{16.15}$$

By repeating the simulations S times we can estimate the distribution of the conditional sample correlations for a given (Φ, A_0) for a DSGE-VAR.

16.7. *Spectral Decomposition*

The spectral decomposition of the DSGE model was discussed in Section 13.2. A similar decomposition can also be determined for DSGE-VARs. The population spectrum of the latter is given by

$$s_y(\omega) = \frac{1}{2\pi} \Phi\big(\exp(-i\omega)\big)^{-1} \Sigma_\epsilon \left(\Phi\big(\exp(i\omega)\big)' \right)^{-1}, \quad \omega \in [-\pi, \pi], \tag{16.16}$$

where $\Phi(z) = I_n - \sum_{l=1}^{p} \Phi_l z^l$. When the structural shocks, v_t, can be identified we know that $\Sigma_\epsilon = A_0 A_0'$. Letting A_{0j} be the j:th column of A_0 it follows that

$$s_y(\omega) = \sum_{j=1}^{n} s_y^{(j)}(\omega) = \sum_{j=1}^{n} \frac{1}{2\pi} \Phi\big(\exp(-i\omega)\big)^{-1} A_{0j} A_{0j}' \left(\Phi\big(\exp(i\omega)\big)' \right)^{-1}. \tag{16.17}$$

The contemporaneous population covariance matrix of the endogenous variables conditional on the parameters satisfies equation (13.4). Let $\Sigma_y^{(j)}(0)$ be the covariance matrix of the endogenous variables conditional on the parameters and on all shocks of the DSGE-VAR being zero except for v_j; cf. Section 16.6. We then find that

$$\Sigma_y^{(j)}(0) = \int_{-\pi}^{\pi} s_y^{(j)}(\omega) d\omega. \tag{16.18}$$

That is, the conditional contemporanoues covariance matrix of the endogenous variables based on only v_j being non-zero is equal to the integral of the spectral decomposition based on this shock.

16.8. *Unconditional Forecasting*

It is straightforward to apply the *sampling the future* procedure of Thompson and Miller (1986) to a DSGE-VAR model; cf. Section 14.5. For a given draw (Φ, Σ_ϵ) from the posterior distribution of a DSGE-VAR we first simulate residuals $\epsilon_{T+1}, \ldots, \epsilon_{T+h}$ from a normal distribution with mean zero and covariance matrix Σ_ϵ. Next, we simulate a path for $y_{T+1}, \ldots, y_{T+h}$ by feeding the residuals into the VAR system in equation (15.9). Repeating this P times for the given (Φ, Σ_ϵ) yields P sample paths conditional on the parameters. By taking S draws of (Φ, Σ_ϵ) from its posterior we end up with PS paths of $y_{T+1}, \ldots, y_{T+h}$ from its predictive density.

The VAR system can be conveniently rewritten for a forecasting exercise. Starting from equation (16.9) we set $t = T + i$ and $t_0 = T$ such that

$$y_{T+i} = \sum_{j=0}^{i-1} J_p' \Psi^j J_p \Phi_0 x_{T+i-j} + J_p' \Psi^i Y_T + \sum_{j=0}^{i-1} J_p' \Psi^j J_p \epsilon_{T+i-j}, \quad i = 1, \ldots, h. \tag{16.19}$$

Compared with calculating forecast paths from equation (16.2) premultiplied by J_p', the expression in (16.19) has the advantage that the lagged endogenous variables are fixed at the same

value for all i. At the same time the terms capturing the influence of the exogenous variables and the innovations involve sums of weighted current and past values and therefore appear to be more complex than those in (16.2). To simplify these two terms, let

$$\bar{x}_{T+i} = J_p \Phi_0 x_{T+i} + \Psi \bar{x}_{T+i-1}, \tag{16.20}$$

$$\bar{\epsilon}_{T+i} = J_p \epsilon_{T+i} + \Psi \bar{\epsilon}_{T+i-1}, \quad i = 1, \dots, h, \tag{16.21}$$

where these np-dimensional vectors are initialized through $\bar{x}_T = \bar{\epsilon}_T = 0$. We can then express the value of the endogenous variables at $T + i$ as

$$y_{T+i} = J_p' \bar{x}_{T+i} + J_p' \Psi^i Y_T + J_p' \bar{\epsilon}_{T+i}, \quad i = 1, \dots, h. \tag{16.22}$$

This equation makes it straightforward to compute a path for the endogenous variables over the forecast sample since it is not necessary to loop over $j = 0, 1, \dots, i - 1$.

We can decompose the prediction uncertainty for the DSGE-VAR into two components, residual or shock uncertainty and parameter uncertainty. That is,

$$C(y_{T+i}|\mathcal{Y}_T) = E_T\left[C(y_{T+i}|\mathcal{Y}_T; \Phi, \Sigma_\epsilon)\right] + C_T\left[E(y_{T+i}|\mathcal{Y}_T; \Phi, \Sigma_\epsilon)\right], \tag{16.23}$$

where E_T and C_T denotes the expectation and covariance with respect to the posterior of (Φ, Σ_ϵ) at time T and where, for notational simplicity, the sequence of exogenous variables $x_{T+1}, \dots, x_{T+h}$ has been suppressed from the expressions.

To develop a simple expression for the first term on the right hand side of (16.23), let $\bar{\Sigma}_Y^{(i)}$ be defined from the difference equation

$$\bar{\Sigma}_Y^{(i)} = J_p \Sigma_\epsilon J_p' + \Psi \bar{\Sigma}_Y^{(i-1)} \Psi', \quad i = 1, \dots, h, \tag{16.24}$$

with the $np \times np$ matrix $\bar{\Sigma}_Y^{(0)} = 0$. This means that

$$J_p' \bar{\Sigma}_Y^{(i)} J_p = \sum_{j=0}^{i-1} J_p' \Psi^j J_p \Sigma_\epsilon J_p' (\Psi^j)' J_p.$$

It now follows that the residual uncertainty term is:

$$E_T\left[C(y_{T+i}|\mathcal{Y}_T; \Phi, \Sigma_\epsilon)\right] = E_T\left[J_p' \bar{\Sigma}_Y^{(i)} J_p\right], \quad i = 1, \dots, h, \tag{16.25}$$

while from (16.22) we find that the parameter uncertainty term is given by:

$$C_T\left[E(y_{T+i}|\mathcal{Y}_T; \Phi, \Sigma_\epsilon)\right] = C_T\left[J_p' \bar{x}_{T+i} + J_p' \Psi^i Y_T\right] \quad i = 1, \dots, h. \tag{16.26}$$

16.9. *Conditional Forecasting*

Conditional forecasting with the DSGE-VAR can be performed in the same spirit as for the state-space model in Section 12.2 and the BVAR in Section 14.6. The conditioning assumptions are, as in the case of the state-space model, given by equation (12.6) and we shall consider both direct control of the structural shocks and control of the distribution of the shocks. The former method requires an identified DSGE-VAR since we need to be able to control individual shocks, while the latter method can be directly applied to the reduced form of the model; cf. Section 14.6.

16.9.1. *Direct Control of the Shocks*

As noted above, the direct control method requires that we have identified the structural shocks, i.e., that the number of shocks in the DSGE model is at least as great as the number of observed variables ($q \geq n$). To ensure that the q_m conditioning assumptions z_{T+i} are satisfied for $i = 1, \dots, g$ we choose a subset of the structural shocks which takes numerical values such that the conditioning assumptions in (12.6) are satisfied.

To this end, let M be an $n \times q_m$ full column rank matrix ($n \geq q_m$) which is used to select the q_m shocks that will be manipulated, while $M_\perp$ is an $n \times (n - q_m)$ full column rank orthogonal matrix ($M_\perp' M = 0$) that selects the free or non-manipulated shocks. The $n \times n$ matrix $N = [M_\perp \ M]$ is therefore $n \times n$ and has full rank n. With $\bar{M} = M(M'M)^{-1}$ and $\bar{M}_\perp = M_\perp(M_\perp' M_\perp)^{-1}$ this

means that

$$v_t = \bar{M}_\perp v_t^{(n-q_m)} + \bar{M} v_t^{(q_m)}, \tag{16.27}$$

where $v_t^{(n-q_m)} = M_\perp' v_t$ is the $(n-q_m)$ dimensional vector with the free shocks, while $v_t^{(q_m)} = M' v_t$ is the q_m dimensional vector with the manipulated (or controlled) shocks. The VAR model in (16.19) can therefore be rewritten as

$$y_{T+i} = \Phi_0 x_{T+i} + J_p' \Psi \tilde{Y}_{T+i-1} + A_0 \bar{M}_\perp v_{T+i}^{(n-q_m)} + A_0 \bar{M} v_{T+i}^{(q_m)}, \quad i = 1,\dots,g, \tag{16.28}$$

where $\tilde{Y}_{T+i-1}$ is the np-dimensional vector $[y_{T+i-1}' \cdots y_{T+i-p}']'$. It here replaces Y_{T+i-1} from the previous Section to allow us to use the Y-vector in the same format as for the DSGE model.

As in Section 12.2 for the state-space model, it is straightforward to show that a necessary and sufficient condition for $v_{T+i}^{(q_m)}$ to be uniquely determined from the conditioning assumptions, the history of the observed variables, and the shocks $v_{T+i}^{(n-q_m)}$ is that the $q_m \times q_m$ matrix $K_1' A_0 \bar{M}$ has full rank. With this in mind, we find that

$$v_{T+i}^{(q_m)} = \left(K_1' A_0 \bar{M}\right)^{-1} \left[z_{T+i} - K_1' \Phi_0 x_{T+i} - K_1' J_p' \Psi \tilde{Y}_{T+i-1}^{(q_m)} + \right.$$
$$\left. - K_1' A_0 \bar{M}_\perp v_{T+i}^{(n-q_m)} - \sum_{j=1}^{i-1} K_{2j}' y_{T+i-j}^{(q_m)} - u_T \right], \quad i = 1,\dots,g \tag{16.29}$$

where the $np \times 1$ vector $\tilde{Y}_{T+i-1}^{(q_m)} = [y_{T+i-1}^{(q_m)\prime} \cdots y_{T+1}^{(q_m)\prime} \, y_T' \cdots y_{T+i-p}']'$, while

$$y_{T+i}^{(q_m)} = \Phi_0 x_{T+i} + J_p' \Psi \tilde{Y}_{T+i-1}^{(q_m)} + A_0 v_{T+i}, \quad i = 1,\dots,g, \tag{16.30}$$

where v_{T+i} satisfies equation (16.27) with $v_{T+i}^{(q_m)}$ determined by (16.29), and where the shocks $v_{T+i}^{(n-q_m)} \sim N(0, M_\perp' M_\perp)$.

For $i > g$ there are not any conditioning assumptions to take into account and $v_{T+i} \sim N(0, I_n)$ when applying the *sampling the future* procedure.

In order to derive moments from the predictive distribution of the conditional forecasts we begin by stacking the system in (16.19) for $T+1,\dots,T+g$, and using equation (16.20) along with the relationship $\epsilon_t = A_0 v_t$. This gives us

$$
\begin{bmatrix} y_{T+g} \\ y_{T+g-1} \\ \vdots \\ y_{T+1} \end{bmatrix}
=
\begin{bmatrix} J_p' \bar{x}_{T+g} \\ J_p' \bar{x}_{T+g-1} \\ \vdots \\ J_p' \bar{x}_{T+1} \end{bmatrix}
+
\begin{bmatrix} J_p' \Psi^g \\ J_p' \Psi^{g-1} \\ \vdots \\ J_p' \Psi \end{bmatrix} \tilde{Y}_T +
$$
$$
+
\begin{bmatrix} I_n & J_p' \Psi J_p & \cdots & J_p' \Psi^{g-1} J_p \\ 0 & I_n & & J_p' \Psi^{g-2} J_p \\ \vdots & & \ddots & \vdots \\ 0 & 0 & & I_n \end{bmatrix}
\left(I_g \otimes A_0\right)
\begin{bmatrix} v_{T+g} \\ v_{T+g-1} \\ \vdots \\ v_{T+1} \end{bmatrix}
$$

or

$$Y_{T+g} = X_{T+g} + G\tilde{Y}_T + D\left(I_g \otimes A_0\right) \Upsilon_{T+g}. \tag{16.31}$$

Furthermore, based on equation (16.27) we can decompose the shocks such that

$$
\begin{bmatrix} v_{T+g} \\ \vdots \\ v_{T+1} \end{bmatrix}
=
\begin{bmatrix} \bar{M}_\perp & & 0 \\ & \ddots & \\ 0 & & \bar{M}_\perp \end{bmatrix}
\begin{bmatrix} v_{T+g}^{(n-q_m)} \\ \vdots \\ v_{T+1}^{(n-q_m)} \end{bmatrix}
+
\begin{bmatrix} \bar{M} & & 0 \\ & \ddots & \\ 0 & & \bar{M} \end{bmatrix}
\begin{bmatrix} v_{T+g}^{(q_m)} \\ \vdots \\ v_{T+1}^{(q_m)} \end{bmatrix},
$$

or

$$\Upsilon_{T+g} = \left(I_g \otimes \bar{M}_\perp\right) \Upsilon_{T+g}^{(n-q_m)} + \left(I_g \otimes \bar{M}\right) \Upsilon_{T+g}^{(q_m)}. \tag{16.32}$$

The stacked conditioning assumptions are given by equation (12.15). Subsituting for Y_{T+g} from equation (16.31), using (16.32) and rearranging we find that

$$K'D\left(I_g \otimes A_0\bar{M}\right)\Upsilon_{T+g}^{(q_m)} = Z_{T+g} - U_T - K'\left(X_{T+g} + G\tilde{Y}_T\right) - K'D\left(I_g \otimes A_0\bar{M}_\perp\right)\Upsilon_{T+g}^{(n-q_m)}.$$

The matrix expression on the left hand side is invertible when the $q_m \times q_m$ matrix $K_1'A_0\bar{M}$ has full rank. In that case we obtain a stacked version of equation (16.29) where the manipulated shocks are given as a function of the conditioning assumptions, the exogenous variables, the historical data on the observed variables, and the freely determined structural shocks $\Upsilon_{T+g}^{(n-q_m)} \sim N(0, [I_g \otimes M_\perp' M_\perp])$.

With these results in mind it can be shown that

$$\Upsilon_{T+g}^{(q_m)} = \mu_{\Upsilon,T+g}^{(q_m)} - \left[K'D\left(I_g \otimes A_0\bar{M}\right)\right]^{-1}\left[K'D\left(I_g \otimes A_0\bar{M}_\perp\right)\right]\Upsilon_{T+g}^{(n-q_m)}, \qquad (16.33)$$

where the population mean of the manipulated shocks is

$$\mu_{\Upsilon,T+g}^{(q_m)} = \left[K'D\left(I_g \otimes A_0\bar{M}\right)\right]^{-1}\left[Z_{T+g} - U_T - K'\left(X_{T+g} + G\tilde{Y}_T\right)\right].$$

The conditional population mean of the conditionally predicted observed variables for fixed parameters is therefore given by

$$E\left[Y_{T+g}|\mathcal{Y}_T, Z_{T+g}; \Phi, A_0\right] = X_{T+g} + G\tilde{Y}_T + D\left(I_g \otimes A_0\bar{M}\right)\mu_{\Upsilon,T+g}^{(q_m)}. \qquad (16.34)$$

Notice that the expectation is here also taken with respect to M, i.e., the selection of shocks used the ensure that the conditioning assumptions are satisfied. For notational convenience, however, it has been left out of the conditioning information. Premultiplication of both sides by K' we find that the right hand side is equal to $Z_{T+g} - U_T$ and, hence, the conditional mean predictions satisfy the conditioning assumptions.

Moreover, the population covariance matrix of the conditional predictions is given by

$$C\left(Y_{T+g}|\mathcal{Y}_T, Z_{T+g}; \Phi, A_0\right) = D\left(I_g \otimes A_0\right)\tilde{D}\left(I_g \otimes \bar{M}_\perp M_\perp'\right)\tilde{D}'\left(I_g \otimes A_0'\right)D', \qquad (16.35)$$

where

$$\tilde{D} = I_{ng} - \left(I_g \otimes \bar{M}\right)\left[K'D\left(I_g \otimes A_0\bar{M}\right)\right]^{-1}K'D\left(I_g \otimes A_0\right).$$

Premultiplication of the covariance matrix in (16.35) by K' or postmultiplication by K yields a zero matrix. Hence, the conditional predictive distribution of the observed variables satisfies the conditioning assumptions.[134]

For forecast horizons i beyond the conditioning horizon g it can be shown that

$$y_{T+i} = J_p'\bar{x}_{T+i} + J_p'\Psi^i\tilde{Y}_T + \sum_{j=0}^{i-g-1} J_p'\Psi^j J_p \epsilon_{T+i-j} + J_p'\Psi^{i-g}\tilde{\Psi}\left(I_g \otimes A_0\right)\Upsilon_{T+g}, \qquad (16.36)$$

for $i = g+1, \ldots, h$, and where the $np \times ng$ matrix

$$\tilde{\Psi} = \begin{bmatrix} J_p & \Psi J_p & \cdots & \Psi^{g-1}J_p \end{bmatrix}.$$

The mean of the predictive distribution is therefore given by

$$E\left[y_{T+i}|\mathcal{Y}_T, Z_{T+g}; \Phi, A_0\right] = J_p'\bar{x}_{T+i} + J_p'\Psi^i\tilde{Y}_T + J_p'\Psi^{i-g}\tilde{\Psi}\left(I_g \otimes A_0\bar{M}\right)\mu_{\Upsilon,T+g}^{(q_m)}. \qquad (16.37)$$

Moreover, the covariance matrix of the forecast error is

$$C\left(y_{T+i}|\mathcal{Y}_T, Z_{T+g}; \Phi, A_0\right) = \sum_{j=0}^{i-g-1} J_p'\Psi^j J_p A_0 A_0' J_p'\left(\Psi'\right)^j J_p + J_p'\Psi^{i-g}\tilde{\Psi}\left(I_g \otimes A_0\right)\times$$
$$\times \tilde{D}\left(I_g \otimes \bar{M}_\perp M_\perp'\right)\tilde{D}'\left(I_g \otimes A_0'\right)\tilde{\Psi}'\left(\Psi'\right)^{i-g}J_p. \qquad (16.38)$$

Provided that all the eigenvalues of Ψ are less than unity in absolute terms, the conditional predictions approaches the mean of the observed variables as $i \to \infty$, while the covariance matrix of the forecast error converges to the unconditional covariance matrix of the variables.

---

[134] This can also be seen by noting that the forecast errors $Y_{T+g} - E\left[Y_{T+g}|\mathcal{Y}_T, Z_{T+g}; \Phi, \Sigma_\epsilon\right]$ are orthogonal to K'.

From the moments of the conditional predictive distribution for fixed parameter values, we may determine the mean and covariance of the predictive distribution once the influence of the model parameters has been integrated out. As in equation (12.24), the corresponding expression for the covariance matrix of the DSGE-VAR is

$$C\big(y_{T+i}|\mathcal{Y}_T, Z_{T+g}\big) = E_T\big[C\big(y_{T+i}|\mathcal{Y}_T, Z_{T+g}; \Phi, A_0\big)\big] + C_T\big[E\big(y_{T+i}|\mathcal{Y}_T, Z_{T+g}; \Phi, A_0\big)\big],$$

for $i = 1, \ldots, h$, and where E_T and C_T denote the expectation and covariance with respect to the posterior of (Φ, A_0) at time T. The first and the second term on the right hand side represent shock and parameters uncertainty, respectively.

16.9.2. *Control of the Distribution of the Shocks*

The Waggoner and Zha (1999) approach can be implemented for the DSGE-VAR in the same basic way as for the BVAR considered in Section 14.6. The conditioning assumptions, however, are expressed as in equation (12.6) or in the stacked version (12.15).

Unlike the case when particular shocks of the DSGE-VAR are manipulated to ensure that the conditioning assumptions are met, the Waggoner and Zha approach does not require identification of the structural shocks. Hence, we may consider a variant of the stacked system (16.31) where ϵ_t is used instead of υ_t. That is,

$$Y_{T+g} = X_{T+g} + G\tilde{Y}_T + D\mathcal{E}_{T+g}, \tag{16.39}$$

where $\mathcal{E}_{T+g} = [\epsilon'_{T+g} \cdots \epsilon'_{T+1}]'$. The restrictions that the ng dimensional vector of innovations needs to satisfy can therefore be expressed as

$$K'D\mathcal{E}_{T+g} = k_{T+g}, \tag{16.40}$$

where the $q_m g \times 1$ vector

$$k_{T+g} = Z_{T+g} - U_T - K'\big(X_{T+g} + G\tilde{Y}_T\big).$$

Like in Section 14.6 it can now be shown that if $\mathcal{E}_{T+g} \sim N(\mu_{\mathcal{E},T+g}, \Sigma_{\mathcal{E},T+g})$ then the restrictions in (16.40) are satisfied for all values of $\mathcal{E}_{T+g}$. The moments of the distribution are here given by

$$\mu_{\mathcal{E},T+g} = \big(I_g \otimes \Sigma_\epsilon\big)D'K\big[K'D\big(I_g \otimes \Sigma_\epsilon\big)D'K\big]^{-1}k_{T+g},$$
$$\Sigma_{\mathcal{E},T+g} = \big(I_g \otimes \Sigma_\epsilon\big) - \big(I_g \otimes \Sigma_\epsilon\big)D'K\big[K'D\big(I_g \otimes \Sigma_\epsilon\big)D'K\big]^{-1}K'D\big(I_g \otimes \Sigma_\epsilon\big). \tag{16.41}$$

The properties of the conditional predictive distribution can now be derived without much ado. First of all, the mean of the distribution for fixed values of the parameters (Φ, Σ_ϵ) is

$$E\big[Y_{T+g}|\mathcal{Y}_T, Z_{T+g}; \Phi, \Sigma_\epsilon\big] = X_{T+g} + G\tilde{Y}_T + D\mu_{\mathcal{E},T+g}, \tag{16.42}$$

while the covariance matrix is

$$C\big(Y_{T+g}|\mathcal{Y}_T, Z_{T+g}; \Phi, \Sigma_\epsilon\big) = D\Sigma_{\mathcal{E},T+g}D'. \tag{16.43}$$

Premultiplication of both sides of equation (16.42) by K' we find that the left hand side is equal to $Z_{T+g} - U_T$ and, hence, that the conditioning assumptions are satisfied by the mean predictions. Moreover, premultiplication of the covariance matrix in (16.43) by K' or postmultiplication by K yields a zero matrix.

For forecast horizons i beyond the conditioning horizon g we may use a minor rewrite of equation (16.36). Specifically, we let

$$y_{T+i} = J'_p \bar{x}_{T+i} + J'_p \Psi^i \tilde{Y}_T + \sum_{j=0}^{i-g-1} J'_p \Psi^j J_p \epsilon_{T+i-j} + J'_p \Psi^{i-g} \tilde{\Psi} \mathcal{E}_{T+g},$$

for $i = g+1, \ldots, h$. From this it is straightforward to infer that

$$E\big[y_{T+i}|\mathcal{Y}_T, Z_{T+g}; \Phi, \Sigma_\epsilon\big] = J'_p \bar{x}_{T+i} + J'_p \Psi^i \tilde{Y}_T + J'_p \Psi^{i-g} \tilde{\Psi} \mu_{\mathcal{E},T+g}, \tag{16.44}$$

while the covariance matrix is

$$C\big(y_{T+i}|\mathcal{Y}_T, Z_{T+g}; \Phi, \Sigma_\epsilon\big) = \sum_{j=0}^{i-g-1} J'_p \Psi^j J_p \Sigma_\epsilon J'_p \big(\Psi'\big)^j J_p + J'_p \Psi^{i-g} \tilde{\Psi} \Sigma_{\mathcal{E},T+g} \tilde{\Psi}' \big(\Psi'\big)^{i-g} J_p. \tag{16.45}$$

Given that the eigenvalues of Ψ are less than unity in absolute term, both these moments converge to the unconditional moments of the observed variables when $i \to \infty$. A decomposition of the overall predictive uncertainty can now be obtained as in the end of Section 16.9.1. This provides us with the overall share of the predictive uncertainty under the conditioning assumptions which is due to uncertainty about the shocks and the share which is due to the uncertainty about the parameters.

16.9.3. *Control of the Distribution of a Subset of the Shocks*

The previous two conditioning approaches may be viewed as special cases of a generalized conditioning method where a subset of the shocks is selected to have a distribution which guarantees that the conditioning assumptions are fulfilled, while the remaining shocks are free. However, this implicitly depends on the assumptions that the shocks are structural and can be identified, i.e., that A_0 is uniquely determined from Σ_ϵ.

With this in mind, let M is an $n \times q_r$ matrix with rank q_r such that $v_t^{(q_r)} = M' v_t$ for $q_m \leq q_r \leq n$, while the $n \times (n - q_r)$ matrix $M_\perp$ is selected such that $M_\perp' M = 0$ and $v_t^{(n-q_r)} = M_\perp' v_t$. As above, we define the matrices $\bar{M}$ and $\bar{M}_\perp$ and let

$$v_t = \bar{M} v_t^{(q_r)} + \bar{M}_\perp v_t^{(n-q_r)}. \tag{16.46}$$

For the stacked system of shocks Υ_{T+g} we then have that

$$\Upsilon_{T+g} = \left(I_g \otimes \bar{M} \right) \Upsilon_{T+g}^{(q_r)} + \left(I_g \otimes \bar{M}_\perp \right) \Upsilon_{T+g}^{(n-q_r)}, \tag{16.47}$$

where

$$\Upsilon_{T+g}^{(n-q_r)} \sim N\left(0, \left(I_g \otimes M_\perp' M_\perp \right) \right),$$

are the free shocks over the conditioning horizon. The conditioning problem is now one of determining the distribution of $\Upsilon_{T+g}^{(q_r)} | \Upsilon_{T+g}^{(n-q_r)}$ such that the conditioning assumptions in (12.15) are satisfied.

The restrictions that the shocks $\Upsilon_{T+g}^{(q_r)}$ have to satisfy can here be expressed as

$$K' D \left(I_g \otimes A_0 \bar{M} \right) \Upsilon_{T+g}^{(q_r)} = k_{T+g}^{(q_r)}, \tag{16.48}$$

where

$$k_{T+g}^{(q_r)} = Z_{T+g} - U_T - K' \left(X_{T+g} + G\tilde{Y}_T + D\left(I_g \otimes A_0 \bar{M}_\perp \right) \Upsilon_{T+g}^{(n-q_r)} \right).$$

The distribution of $\Upsilon_{T+g}^{(q_r)}$ conditional on $\Upsilon_{T+g}^{(n-q_r)}$ can now be shown to be normal with mean and covariance matrix given by

$$
\begin{aligned}
\mu_{\Upsilon,T+g}^{(q_r)} &= \left(I_g \otimes \bar{M}' A_0' \right) D' K \left[K' D \left(I_g \otimes A_0 \bar{M} \bar{M}' A_0' \right) D' K \right]^{-1} k_{T+g}^{(q_r)}, \\
\Sigma_{\Upsilon,T+g}^{(q_r)} &= I_{q_r g} - \left(I_g \otimes \bar{M}' A_0' \right) D' K \left[K' D \left(I_g \otimes A_0 \bar{M} \bar{M}' A_0' \right) D' K \right]^{-1} K' D \left(I_g \otimes A_0 \bar{M} \right).
\end{aligned}
\tag{16.49}
$$

The mean of the conditional predictive distribution for fixed parameters is now given by

$$E\left[Y_{T+g} | \mathcal{Y}_T, Z_{T+g}; \Phi, \Sigma_\epsilon \right] = X_{T+g} + G\tilde{Y}_T + D\left(I_g \otimes A_0 \bar{M} \right) \bar{\mu}_{\Upsilon,T+g}^{(q_r)}, \tag{16.50}$$

where

$$
\begin{aligned}
\bar{\mu}_{\Upsilon,T+g}^{(q_r)} &= \left(I_g \otimes \bar{M}' A_0' \right) D' K \left[K' D \left(I_g \otimes A_0 \bar{M} \bar{M}' A_0' \right) D' K \right]^{-1} \bar{k}_{T+g}^{(q_r)}, \\
\bar{k}_{T+g}^{(q_r)} &= Z_{T+g} - U_T - K' \left(X_{T+g} + G\tilde{Y}_T \right).
\end{aligned}
$$

Furthermore, the covariance matrix of the distribution can be expressed as

$$
\begin{aligned}
C\left(Y_{T+g} | \mathcal{Y}_T, Z_{T+g}; \Phi, \Sigma_\epsilon \right) = D\left(I_g \otimes A_0 \right) \Bigg[\bar{D}\left(I_g \otimes \bar{M}_\perp M_\perp' \right) \bar{D}' + \\
+ \left(I_g \otimes \bar{M} \right) \Sigma_{\Upsilon,T+g}^{(q_r)} \left(I_g \otimes \bar{M}' \right) \Bigg] \left(I_g \otimes A_0' \right) D',
\end{aligned}
\tag{16.51}
$$

where
$$\bar{D} = I_{ng} - \left(I_g \otimes \bar{M}\bar{M}'A_0'\right)D'K\left[K'D\left(I_g \otimes A_0\bar{M}\bar{M}'A_0'\right)D'K\right]^{-1}K'D\left(I_g \otimes A_0\right).$$

It is now straightforward to show that the predictive distribution of Y_{T+g} satisfies the conditioning assumptions. Premultiplication of the conditional mean in (16.50) by K' we find that the right hand side is equal to $Z_{T+g} - U_T$. Furthermore, premultiplication of the covariance in (16.51) by K' and postmultiplication by K gives us a zero matrix.

In the event that $q_r = q_m$ we find that the covariance matrix $\Sigma_{\Upsilon,T+g}^{(q_r)}$ is zero since the $q_m g \times q_m g$ matrix $K'D(I_g \otimes A_0\bar{M})$ is invertible. As a consequence, the subset of shocks $\Upsilon_{T+g}^{(q_r)} = \mu_{\Upsilon,T+g}^{(q_r)}$ and is also equal to $\Upsilon_{T+g}^{(q_m)}$ in equation (16.33). Hence, the distribution of a subset of shocks method is identical to the direct control of shocks method. Furthermore, with $q_r = q$ we may let $M = I_n$. With $\Sigma_\epsilon = A_0A_0'$ we now find that the mean and the covariance matrix of the predictive distribution in (16.50) and (16.51) are identical to those in (16.42) and (16.43). Hence, the distribution of a subset of the shocks conditioning method is identical to the distribution of the shocks method from Section 16.9.2 when $q_r = q$ and the structural shocks of the DSGE-VAR can be identified.

For forecast horizons beyond the conditioning horizon it is straightforward to show that
$$E\left[y_{T+i}|\mathcal{Y}_T, Z_{T+g}; \Phi, \Sigma_\epsilon\right] = J_p'\bar{x}_{T+i} + J_p'\Psi^i\tilde{Y}_T + J_p'\Psi^{i-g}\tilde{\Psi}\left(I_g \otimes A_0\bar{M}\right)\bar{\mu}_{\Upsilon,T+g}^{(q_r)}, \tag{16.52}$$
for $i = g+1, \ldots, h$, while the covariance matrix is given by
$$C\left(y_{T+i}|\mathcal{Y}_T, Z_{T+g}; \Phi, \Sigma_\epsilon\right) = \sum_{j=0}^{i-g-1} J_p'\Psi^j J_p \Sigma_\epsilon J_p'\left(\Psi'\right)^j J_p + J_p'\Psi^{i-g}\tilde{\Psi}\left(I_g \otimes A_0\right)\times$$
$$\left[\bar{D}\left(I_g \otimes \bar{M}_\perp M_\perp'\right)\bar{D}' + \left(I_g \otimes \bar{M}\right)\Sigma_{\Upsilon,T+g}^{(q_r)}\left(I_g \otimes \bar{M}'\right)\right]\left(I_g \otimes A_0'\right)\tilde{\Psi}'\left(\Psi'\right)^{i-g}J_p. \tag{16.53}$$

Provided that the eigenvalues of Ψ are less than unity in aboslute term, the moments in (16.52) and (16.53) converge to the unconditional moments of the observed variables when $i \to \infty$. A decomposition of the overall predictive uncertainty can now be obtained as in the end of Section 16.9.1. This provides us with the overall share of the predictive uncertainty under the conditioning assumptions which is due to uncertainty about the shocks and the share which is due to the uncertainty about the parameters.

16.9.4. *Modesty Statistics for the DSGE-VAR*

Modesty statistics for conditional forecasts from the DSGE model were discussed in Section 12.3 while similar statistics were considered for conditional forecasts from the BVAR in Section 14.6. With these results in mind, the introduction of such tests of the likelihood that the agents of the economy would detect that the conditioning information included paths of future values for some of the endogenous variables in the DSGE-VAR framework is fairly straightforward.

Beginning with the case when we use direct control of certain structural shocks to ensure that the conditioning assumptions are satisfied, the difference between a path for the conditional forecast and the unconditional mean forecast is given by
$$\Phi_{T,g}\left(\bar{\Upsilon}_{T+g}\right) = J_p'\left[J_p \quad \Psi J_p \quad \cdots \quad \Psi^{g-1}J_p\right]\left(I_g \otimes A_0\right)\bar{\Upsilon}_{T+g} = J_p'\tilde{\Psi}\left(I_g \otimes A_0\right)\bar{\Upsilon}_{T+g}, \tag{16.54}$$

where $\bar{\Upsilon}_{T+g}$ is a draw of the identified shocks such that $\bar{\Upsilon}_{T+g}^{(n-q_m)} \sim N(0, [I_g \otimes M_\perp'M_\perp])$, while the manipulated shocks $\bar{\Upsilon}_{T+g}^{(q_m)}$ satisfy equation (16.33). Since the covariance matrix of the unconditional forecasts is given by $J_p'\bar{\Sigma}_Y^{(g)}J_p$, where $\bar{\Sigma}_Y^{(g)}$ is determined by equation (16.24), a multivariate modesty statistic based on the ideas in Adolfson et al. (2005) may be expressed as
$$\mathcal{M}_{T,g}\left(\bar{\Upsilon}_{T+g}\right) = \Phi_{T,g}\left(\bar{\Upsilon}_{T+g}\right)'\left[J_p'\bar{\Sigma}_Y^{(g)}J_p\right]^{-1}\Phi_{T,g}\left(\bar{\Upsilon}_{T+g}\right). \tag{16.55}$$

Provided that the manipulated shocks are modest, i.e., $\{v_t^{(q_m)}\}_{t=T+1}^{T+g}$ can be regarded as being drawn from a multivariate normal distribution with mean zero and covariance matrix $M'M$, the statistic in (16.55) is $\chi^2(n)$. Alternatively, a reference distribution for the statistics may be simulated by feeding a sequence of standard normal shocks Υ_{T+g} into the statistic in (16.55), denoted by $\mathcal{M}_{T,g}(\Upsilon_{T+g})$, and thereafter computing the tail probability $\Pr[\mathcal{M}_{T,g}(\Upsilon_{T+g}) \geq \mathcal{M}_{T,g}(\bar{\Upsilon}_{T+g})]$ to determine if the conditioning assumptions are modest or not.

A set of univariate statistics based on these ideas may also be considered. Following Adolfson et al. (2005) we here consider

$$\mathcal{M}_{T,g}^{(i)}(\bar{\Upsilon}_{T+g}) = \frac{\Phi_{T,g}^{(i)}(\bar{\Upsilon}_{T+g})}{\sqrt{e_i' J_p' \bar{\Sigma}_Y^{(g)} J_p e_i}}, \quad i = 1, \ldots, n, \tag{16.56}$$

where the numerator is element i of the vector in (16.54), and e_i is the i:th column of I_n. This statistic has a standard normal distribution under the assumption that the manipulated shocks are modest.

The third modesty statistic for the case of direct control of the structural shocks is the Leeper-Zha inspired statistic. All shocks are here set to zero except for the manipulated ones. This means that we require the covariance matrix based on using only the manipulated shocks

$$\tilde{\Sigma}_Y^{(i)} = J_p A_0 \bar{M} M' A_0' J_p' + \Psi \tilde{\Sigma}_Y^{(i-1)} \Psi', \quad i = 1, \ldots, g,$$

where $\tilde{\Sigma}_Y^{(0)} = 0$, and replace $\bar{\Sigma}_Y^{(g)}$ in equation (16.56) with $\tilde{\Sigma}_Y^{(g)}$. Moreover, the numerator is evaluated at

$$\bar{\Upsilon}_{T+g} = (I_g \otimes \bar{M}) \mu_{\Upsilon,T+g}^{(q_m)}.$$

Under the Waggoner and Zha approach we replace the forecast difference in equation (16.54) with

$$\Phi_{T,g}(\bar{\mathcal{E}}_{T+g}) = J_p' \tilde{\Psi} \bar{\mathcal{E}}_{T+g},$$

where $\bar{\mathcal{E}}_{T+g}$ is a draw from $N(\mu_{\mathcal{E},T+g}, \Sigma_{\mathcal{E},T+g})$. The multivariate modesty statistic in (16.55) is therefore given by

$$\mathcal{M}_{T,g}(\bar{\mathcal{E}}_{T+g}) = \Phi_{T,g}(\bar{\mathcal{E}}_{T+g})' \left[J_p' \bar{\Sigma}_Y^{(g)} J_p \right]^{-1} \Phi_{T,g}(\bar{\mathcal{E}}_{T+g}).$$

The univariate statistics in (16.56) is likewise computed with element i from $\Phi_{T,g}(\bar{\mathcal{E}}_{T+g})$ rather than this element from $\Phi_{T,g}(\bar{\Upsilon}_{T+g})$. The denominator is unaffected by the choice of conditioning method.

For the Leeper-Zha approach we use element i from $\Phi_{T,g}(\mu_{\mathcal{E},T+g})$ in the numerator of the test statistic, while the covariance matrix $\bar{\Sigma}_Y^{(g)}$ is replaced with $\Sigma_Y^{(g)}$, where

$$\Sigma_Y^{(i)} = J_p \Sigma_\epsilon K_1 \left(K_1' \Sigma_\epsilon K_1 \right)^{-1} K_1' \Sigma_\epsilon J_p' + \Psi \Sigma_Y^{(i-1)} \Psi', \quad i = 1, \ldots, g,$$

and where $\Sigma_Y^{(0)} = 0$.

Finally, under the distribution for a subset of the shocks method discussed in Section 16.9.3, the multivariate and univariate modesty statistics based on the Adolfson, Laséen, Lindé, and Villani approach are calculated as in the case of direct control of the shocks, i.e., based on the structural shocks rather than on the VAR residuals. For the univariate Leeper-Zha based modesty statistic, however, the covariance matrix is calculated as under the Waggoner and Zha approach, except that Σ_ϵ is replaced with $A_0 \bar{M} M' A_0'$. When $q_r = q_m$ we find that $K_1' A_0 \bar{M}$ is invertible and, hence, that the Leeper-Zha covariance matrix is equal to the one determined under direct control of the shocks method. When $q_r = q$ we likewise find that $M = I_n$ while $\Sigma_\epsilon = A_0 A_0'$ so that the Leeper-Zha covariance matrix is identical to the one obtained under the Waggoner and Zha method.

16.10. *The Predictive Likelihood for DSGE-VARs*

The general expressions for estimating the predictive likelihood that were presented in Section 12.6 are also valid for DSGE-VAR models. In this regard it is important to keep in mind

that the DSGE-VAR has parameters beyond θ, except for the case when $\lambda = \infty$, and that the VAR parameters need to be dealt with when computing the marginal likelihood for the sample $(y^*_{t+h}, \mathcal{Y}_t)$. The simplified expressions for the Laplace approximation in equation (12.68) can also be used in the DSGE-VAR framework, where we only need to provide formulas for $y_{t+h|t}$ and $\Sigma_{y,t+h|t}$. This is easily achieved through results in Section 16.8 regarding the population moments of the marginal predictive distribution, but again the caveat about taking all the DSGE-VAR parameters when computing the marginal likelihood for the sample $(y^*_{t+h}, \mathcal{Y}_t)$ needs to be taken into account.

We can directly make use of equations (16.22) and (16.24) to determine $y_{t+h|t}$ and $\Sigma_{y,t+h|t}$. That is, setting $\bar{\epsilon}_{t+h}$ to zero

$$y_{t+h|t} = J'_p \bar{x}_{t+h} + J'_p \Psi^h Y_t,$$

$$\Sigma_{y,t+h|t} = J'_p \bar{\Sigma}^{(h)}_Y J_p, \quad h = 1, \ldots, H,$$

where the $(\Phi, \Sigma_\epsilon, \theta)$ parameters are evaluated at the posterior mode. Based on the joint conditional posterior density of the VAR parameters, the joint posterior mode of the VAR parameters conditional on θ is given by equations (15.31) and (15.32).

The joint predictive likelihood for a subset of variables can be derived from the Kalman filter equations in Section 5.2. The conditional likelihood is given by equation (12.69), where the period $t+i$ term is shown in (12.70). It therefore remains to provide expressions for the forecast of y^*_{t+i} and the corresponding covariance matrix conditional on the information $\mathcal{Y}^*_{t+i-1}, \mathcal{Y}_t$. This means that

$$y^*_{t+i|t+i-1} = K' J'_p Y_{t+i|t+i-1},$$

$$\Sigma^*_{y,t+i|t+i-1} = K' J'_p \Omega_{t+i|t+i-1} J_p K, \quad i = 1, \ldots, h,$$

where K is an $n \times n^*$ known selection matrix such that $y^*_t = K' y_t$. The 1-step-ahead forecasts of the Y_{t+i} vector are

$$Y_{t+i|t+i-1} = \begin{cases} J_p \Phi_0 x_{t+i} + \Psi Y_{t+i-1|t+i-2} + G_{t+i-1} \left(y^*_{t+i-1} - y^*_{t+i-1|t+i-2} \right), & \text{if } i \geq 2, \\ J_p \Phi_0 x_{t+1} + \Psi Y_t, & \text{if } i = 1. \end{cases}$$

The Kalman gain matrix

$$G_{t+i-1} = \Psi \Omega_{t+i-1} J_p K \Sigma^{*-1}_{y,t+i-1|t+i-2}, \quad i \geq 2,$$

while the 1-step-ahead covariance matrix is

$$\Omega_{t+i|t+i-1} = \begin{cases} \left(\Psi - G_{t+i-1} K' J'_p \right) \Omega_{t+i|t+i-1} \left(\Psi - G_{t+i-1} K' J'_p \right)' + J_p \Sigma_\epsilon J'_p, & \text{if } i \geq 2, \\ J_p \Sigma_\epsilon J'_p, & \text{if } i = 1. \end{cases}$$

Posterior mode estimation of θ via a DSGE-VAR model can, as discussed in Section 15.7, be conducted in at least two different ways. That is, we may either use the marginal likelihood in equation (15.24) or the concentrated likelihood in equation (15.34) when λ is finite, and the likelihood in equation (15.26) when $\lambda = \infty$. When combined with the prior of θ (or ϕ) the first gives us a marginal posterior mode estimate of θ, while the second gives us a joint posterior mode estimate. The case when $\lambda = \infty$ means that the marginal and the joint mode are equal. Given any such estimate, the corresponding posterior mode of the VAR parameters is obtained by plugging the value of the estimated DSGE model parameters into equations (15.31) and (15.32). Both these approaches yield legitimate candidate estimates of the posterior mode when computing the height of the predictive density in (12.68).

A peculiar property of the DSGE-VAR model is that its prior in equations (15.16) and (15.17) depends on T. That is, the prior changes as the number of observation in the likelihood increases. Specifically, when going from T to $T+1$ observations on y_t, the change of the prior of

the VAR parameters is given by:

$$\Delta \ln p\big(\Phi, \Sigma_\epsilon | \theta, \lambda\big) = \frac{nT\lambda}{2} \ln(1 + T^{-1}) + \frac{n\lambda}{2} \ln((T+1)\lambda/2) - \ln \Gamma_n((T+1)\lambda - np - k)$$

$$+ \ln \Gamma_n(T\lambda - np - k) + \frac{\lambda}{2} \ln |\Sigma_\epsilon(\theta)| - \frac{\lambda}{2} \ln |\Sigma_\epsilon| - \frac{\lambda}{2} \mathrm{tr}\big[\Sigma_\epsilon^{-1} \Sigma_\epsilon(\theta)\big].$$

If the predictive likelihood for, say, $y_{T+1} | \mathcal{Y}_T$ is calculated using the marginal likelihood values from the Laplace approximations for periods $T+1$ and T it follows that the predictive likelihood is directly influenced by this feature of the prior. In additional, the three terms involving Σ_ϵ and $\Sigma_\epsilon(\theta)$ imply that the link between the Hessian matrices in equation (12.74) is not valid. Both these effects from the DSGE-VAR prior being dependent on T are clearly unfortunate when using such models in a forecast comparison exercise.

If we instead make use of the more reasonable assumption that the prior remains unchanged when evaluating forecasts, i.e., we fix T at T^* in the prior, and compute the Laplace approximation through (12.73), the Hessian matrix $\tilde{\Sigma}_t$ is available through equations (15.38)–(15.45). Furthermore, equation (12.74) is now valid with the effect that the Hessian matrix $\tilde{\Sigma}_{t+h}$ is equal to sum of $\tilde{\Sigma}_t$ and $\tilde{\Omega}_{t+h|t}$. For the DSGE-VAR models with finite λ it follows that

$$\tilde{\Omega}_{t+h|t} = \begin{bmatrix} -\dfrac{\partial^2 \ln L\big(y_{t+h}^* | y, Y_1; \tilde{\Phi}, \tilde{\Sigma}_\epsilon\big)}{\partial \mathrm{vec}(\Phi)\mathrm{vec}(\Phi)'} & -\dfrac{\partial^2 \ln L\big(y_{t+h}^* | y, Y_1; \tilde{\Phi}, \tilde{\Sigma}_\epsilon\big)}{\partial \mathrm{vec}(\Phi)\mathrm{vech}(\Sigma_\epsilon)'} & 0 \\[2ex] -\dfrac{\partial^2 \ln L\big(y_{t+h}^* | y, Y_1; \tilde{\Phi}, \tilde{\Sigma}_\epsilon\big)}{\partial \mathrm{vech}(\Sigma_\epsilon)\mathrm{vec}(\Phi)'} & -\dfrac{\partial^2 \ln L\big(y_{t+h}^* | y, Y_1; \tilde{\Phi}, \tilde{\Sigma}_\epsilon\big)}{\partial \mathrm{vech}(\Sigma_\epsilon)\mathrm{vech}(\Sigma_\epsilon)'} & 0 \\[2ex] 0 & 0 & 0 \end{bmatrix}. \tag{16.57}$$

The zeros are due to the fact that the conditional likelihood is invariant to the DSGE model parameters and we have assumed that the prior of the VAR parameters is fixed, with $t = t^*$ for the forecast horizon $t + h$ with $h = 1, \ldots, H$.

By contrast, for the DSGE-VAR models with $\lambda = \infty$, the VAR parameters are functions of the DSGE model parameters. To determine $\tilde{\Omega}_{t+h|t}$ we may then apply the chain rule to the partial derivatives. Letting the nonzero matrices in (16.57) be denoted by $\Omega_{\Phi,\Phi}$, $\Omega_{\Phi,\Sigma} = \Omega'_{\Sigma,\Phi}$, and $\Omega_{\Sigma,\Sigma}$, respectively, it follows that

$$\tilde{\Omega}_{t+h|t}^{(\lambda=\infty)} = \left(\frac{\partial \mathrm{vec}(\Phi(\theta))}{\partial \phi'}\right)' \Omega_{\Phi,\Phi} \frac{\partial \mathrm{vec}(\Phi(\theta))}{\partial \phi'} + \left(\frac{\partial \mathrm{vec}(\Phi(\theta))}{\partial \phi'}\right)' \Omega_{\Phi,\Sigma} \frac{\partial \mathrm{vech}(\Sigma_\epsilon(\theta))}{\partial \phi'}$$

$$+ \left(\frac{\partial \mathrm{vech}(\Sigma_\epsilon(\theta))}{\partial \phi'}\right)' \Omega_{\Sigma,\Phi} \frac{\partial \mathrm{vec}(\Phi(\theta))}{\partial \phi'} + \left(\frac{\partial \mathrm{vech}(\Sigma_\epsilon(\theta))}{\partial \phi'}\right) \Omega_{\Sigma,\Sigma} \frac{\partial \mathrm{vech}(\Sigma_\epsilon(\theta))}{\partial \phi'}.$$

The partial derivatives of $\Phi(\theta)$ and $\Sigma_\epsilon(\theta)$ with respect to ϕ are:

$$\frac{\partial \mathrm{vec}\big(\Phi(\theta)\big)}{\partial \phi'} = \big[\Gamma_{YY}^{-1}(\theta) \otimes I_n\big] G_{yY}(\theta) - \big[\Gamma_{YY}^{-1}(\theta) \otimes \Phi(\theta)\big] D_{np+k} G_{YY}(\theta),$$

and

$$\frac{\partial \mathrm{vech}\big(\Sigma_\epsilon(\theta)\big)}{\partial \phi'} = G_{yy}(\theta) + D_n^+ \big[\Phi(\theta) \otimes \Phi(\theta)\big] D_{np+k} G_{YY}(\theta) - 2D_n^+ \big[\Phi(\theta) \otimes I_n\big] G_{yY}(\theta),$$

and the G matrices are given in Section 15.7.

It remains the provide analytical expressions of the $\Omega_{i,j}$ matrices. After making considerable use of the matrix differential calculus tricks in Magnus and Neudecker (1988) it can be shown

that the following monster appears for $h = 1, 2, \ldots, H$:

$$
\begin{aligned}
\Omega_{\Phi,\Phi} = \frac{1}{2} &\left(\frac{\partial \mathrm{vec}(\bar{\Sigma}_Y^{(h)})}{\partial \mathrm{vec}(\Phi)'} \right)' \left[J_p K \Sigma_{K,h}^{-1} K' J_p' \otimes J_p K \left(2\Sigma_{K,h}^{-1} \varepsilon_h^* \varepsilon_h^{*\prime} - I_{n^*} \right) \right. \\
&\left. \times \Sigma_{K,h}^{-1} K' J_p' \right] \frac{\partial \mathrm{vec}(\bar{\Sigma}_Y^{(h)})}{\partial \mathrm{vec}(\Phi)'} + \sum_{i=0}^{h-2} \left\{ M' \left[\bar{\Sigma}_Y^{(h-1-i)} \otimes G_{i,h} \right] M \right. \\
&+ \left(\frac{\partial \mathrm{vec}(\bar{\Sigma}_Y^{(h-1-i)})}{\partial \mathrm{vec}(\Phi)'} \right)' \left[I_{np} \otimes \Psi' G_{i,h} \right] M + M' \left[I_{np} \otimes G_{i,h} \Psi \right] \\
&\times \frac{\partial \mathrm{vec}(\bar{\Sigma}_Y^{(h-1-i)})}{\partial \mathrm{vec}(\Phi)'} + \left(\frac{\partial \bar{x}_{t+h-1-i}}{\partial \mathrm{vec}(\Phi)'} \right)' \left[I_{np} \otimes \varepsilon_h^{*\prime} \Sigma_{K,h}^{-1} K' J_p' \Psi^i \right] M \\
&+ M' \left[I_{np} \otimes (\Psi')^i J_p K \Sigma_{K,h}^{-1} \varepsilon_h^* \right] \frac{\partial \bar{x}_{t+h-1-i}}{\partial \mathrm{vec}(\Phi)'} + \left(\frac{\partial \mathrm{vec}(\Psi^{h-1-i})}{\partial \mathrm{vec}(\Phi)'} \right)' \\
&\times \left[I_{np} \otimes J_p K \Sigma_{K,h}^{-1} \varepsilon_h^* Y_t' (\Psi')^i \right] C_{np} M + M' C_{np}' \left[I_{np} \otimes \Psi^i Y_t \varepsilon_h^{*\prime} \right. \\
&\left. \times \Sigma_{K,h}^{-1} K' J_p' \right] \frac{\partial \mathrm{vec}(\Psi^{h-1-i})}{\partial \mathrm{vec}(\Phi)'} \right\} + \left(\frac{\partial \varepsilon_h^*}{\partial \mathrm{vec}(\Phi)'} \right)' \Sigma_{K,h}^{-1} \frac{\partial \varepsilon_h^*}{\partial \mathrm{vec}(\Phi)'} \\
&- \frac{1}{2} \left(\frac{\partial \varepsilon_h^*}{\partial \mathrm{vec}(\Phi)'} \right)' \left[\Sigma_{K,h}^{-1} K' J_p' \otimes \varepsilon_h^{*\prime} \Sigma_{K,h}^{-1} K' J_p' \right] \frac{\partial \mathrm{vec}(\bar{\Sigma}_Y^{(h)})}{\partial \mathrm{vec}(\Phi)'} \\
&- \frac{1}{2} \left(\frac{\partial \mathrm{vec}(\bar{\Sigma}_Y^{(h)})}{\partial \mathrm{vec}(\Phi)'} \right)' \left[J_p K \Sigma_{K,h}^{-1} \otimes J_p K \Sigma_{K,h}^{-1} \varepsilon_h^* \right] \frac{\partial \varepsilon_h^*}{\partial \mathrm{vec}(\Phi)'}.
\end{aligned}
\tag{16.58}
$$

Furthermore,

$$
\begin{aligned}
\Omega_{\Phi,\Sigma} = \frac{1}{2} &\left(\frac{\partial \mathrm{vec}(\bar{\Sigma}_Y^{(h)})}{\partial \mathrm{vec}(\Phi)'} \right)' \left[J_p K \Sigma_{K,h}^{-1} K' J_p' \otimes J_p K \left(2\Sigma_{K,h}^{-1} \varepsilon_h^* \varepsilon_h^{*\prime} - I_{n^*} \right) \right. \\
&\left. \times \Sigma_{K,h}^{-1} K' J_p' \right] \frac{\partial \mathrm{vec}(\bar{\Sigma}_Y^{(h)})}{\partial \mathrm{vech}(\Sigma_\epsilon)'} + \sum_{i=0}^{h-2} M' \left[I_{np} \otimes G_{i,h} \Psi \right] \frac{\partial \mathrm{vec}(\bar{\Sigma}_Y^{(h-1-i)})}{\partial \mathrm{vech}(\Sigma_\epsilon)'} \\
&- \frac{1}{2} \left(\frac{\partial \varepsilon_h^*}{\partial \mathrm{vec}(\Phi)'} \right)' \left[\Sigma_{K,h}^{-1} K' J_p' \otimes \varepsilon_h^{*\prime} \Sigma_{K,h}^{-1} K' J_p' \right] \frac{\partial \mathrm{vec}(\bar{\Sigma}_Y^{(h)})}{\partial \mathrm{vech}(\Sigma_\epsilon)'},
\end{aligned}
\tag{16.59}
$$

and

$$
\begin{aligned}
\Omega_{\Sigma,\Sigma} = \frac{1}{2} &\left(\frac{\partial \mathrm{vec}(\bar{\Sigma}_Y^{(h)})}{\partial \mathrm{vech}(\Sigma_\epsilon)'} \right)' \left[J_p K \Sigma_{K,h}^{-1} K' J_p' \otimes J_p K \left(2\Sigma_{K,h}^{-1} \varepsilon_h^* \varepsilon_h^{*\prime} - I_{n^*} \right) \right. \\
&\left. \times \Sigma_{K,h}^{-1} K' J_p' \right] \frac{\partial \mathrm{vec}(\bar{\Sigma}_Y^{(h)})}{\partial \mathrm{vech}(\Sigma_\epsilon)'}.
\end{aligned}
\tag{16.60}
$$

These expression are evaluated at the posterior mode and rely on the following:

$$
\varepsilon_h^* = K' \left(y_{t+h} - J_p' \bar{x}_{t+h} - J_p \Psi^h Y_t \right),
$$

$$
G_{i,h} = (\Psi')^i J_p K \left(I_{n^*} - \Sigma_{K,h}^{-1} \varepsilon_h^* \varepsilon_h^{*\prime} \right) \Sigma_{K,h}^{-1} K' J_p' \Psi^i, \quad i = 0, 1, \ldots, h-2,
$$

$$
\Sigma_{K,h} = K' J_p' \bar{\Sigma}_Y^{(h)} J_p K,
$$

$$
\frac{\partial \mathrm{vec}(\bar{\Sigma}_Y^{(h)})}{\partial \mathrm{vec}(\Phi)'} = \left(I_{n^2 p^2} + C_{np} \right) \left[\Psi \bar{\Sigma}_Y^{(h-1)} \otimes I_{np} \right] M + [\Psi \otimes \Psi] \frac{\partial \mathrm{vec}(\bar{\Sigma}_Y^{(h-1)})}{\partial \mathrm{vec}(\Phi)'}, \quad h = 1, 2, \ldots, H,
$$

where $\bar{\Sigma}_Y^{(0)} = 0$, $\partial\text{vec}(\bar{\Sigma}_Y^{(0)})/\partial\text{vec}(\Phi)' = 0$, and C_{np} is the $n^2 p^2 \times n^2 p^2$ commutation matrix such that $C_{np}\text{vec}(\Psi) = \text{vec}(\Psi')$. Moreover,

$$\frac{\partial\text{vec}(\bar{\Sigma}_Y^{(h)})}{\partial\text{vech}(\Sigma_\epsilon)'} = \left[J_p \otimes J_p\right] D_n + \left[\Psi \otimes \Psi\right]\frac{\partial\text{vec}(\bar{\Sigma}_Y^{(h-1)})}{\partial\text{vech}(\Sigma_\epsilon)'}, \quad h = 1, 2, \ldots, H,$$

where again $\partial\text{vec}(\bar{\Sigma}_Y^{(0)})/\partial\text{vech}(\Sigma_\epsilon)' = 0$. Next,

$$\frac{\partial\bar{x}_{t+h}}{\partial\text{vec}(\Phi)'} = J_p N + \left[\bar{x}'_{t+h-1} \otimes I_{np}\right] M + \Psi\frac{\partial\bar{x}_{t+h-1}}{\partial\text{vec}(\Phi)'}, \quad h = 1, 2, \ldots, H,$$

where $\bar{x}_t = 0$ and $\partial\bar{x}_t/\partial\text{vec}(\Phi)'$. Furthermore,

$$\frac{\partial\text{vec}(\Psi^h)}{\partial\text{vec}(\Phi)'} = \left[I_{np} \otimes \Psi^{h-1}\right] M + \left[\Psi' \otimes I_{np}\right]\frac{\partial\text{vec}(\Psi^{h-1})}{\partial\text{vec}(\Phi)'}, \quad h = 1, 2, \ldots, H,$$

where $\partial\text{vec}(\Psi^0)/\partial\text{vec}(\Phi)' = 0$. Finally,

$$\frac{\partial\varepsilon_h^*}{\partial\text{vec}(\Phi)'} = -K'J_p'\frac{\partial\bar{x}_{t+h}}{\partial\text{vec}(\Phi)'} - \left[Y_t' \otimes K'J_p'\right]\frac{\partial\text{vec}(\Psi^h)}{\partial\text{vec}(\Phi)'}, \quad h = 1, 2, \ldots, H,$$

and the $n^2 p^2 \times n(np + k)$ and $nk \times n(np + k)$ matrices

$$M = \begin{bmatrix} 0_{n^2 p^2 \times nk} & (I_{np} \otimes J_p) \end{bmatrix}, \qquad N = \begin{bmatrix} I_{nk} & 0_{nk \times n^2 p} \end{bmatrix}.$$

Notice that $d\text{vec}(\Psi) = Md\text{vec}(\Phi)$ and $\text{vec}(\Phi_0) = N\text{vec}(\Phi)$.

It is also possible to derive similar analytical results for the Laplace approximation of the joint predictive likelihood. For large models (large n and/or p), it is noteworthy that the calculation of Hessian can be extremely time consuming and for such models it seems reasonable to instead use a Monte Carlo integration based method, such as the importance sampling estimator in (12.67).

16.11. *YADA Code*

This section contains information about the functions that are used to compute the different tools discussed above.

16.11.1. DSGEVARImpulseResponses

The function DSGEVARImpulseResponses computes DSGE-VAR based impulse responses for original, annualized, and levels data. A total of 8 input variables are required for this purpose: Phi, A0, p, h, IRType, levels, annual, and annualscale. The first is the Φ matrix in equation (15.10), except that the parameters on the exogenous variables, Φ_0, have been removed. The second is the identified A_0 matrix, while p and h are the lag order and response horizon, respectively. The 5th input variable is an indicator such that the value 1 means that original impulse responses should be calculated, 2 that annualized impulse responses are needed, while 3 results in levels responses. The levels variable is a vector of length equal to the number of endogenous variables (n), where 1 indicates that the variable is measured as a levels variable, and 0 that it is measured as a first differenced variable. The final 2 input variables, annual and annualscale, are vector of the length n and are used to annualize the original impulse responses. If an element in the vector annual is unity the variable is already annualized, 4 means that it is measured as quarterly first differences and, hence, that annualization is achieved by summing the current and previous 3 values of the variable. An entry equal to 12 means that it is measured as monthly first differences and that adding the current and previous 11 values gives the annualization. Finally, the vector annualscale contains scale constants that are multiplied by the variables.

The function only returns one output variable: IRF. This matrix is of dimension $n \times n \times (h+1)$, where the responses in the endogenous variables are found in the rows, while the columns

represent the n structural shocks of the DSGE-VAR. The third dimension gives the horizon for the responses, where the first concerns the contemporaneous responses.

16.11.2. DSGEVARVarianceDecompositions

The function `DSGEVARVarianceDecompositions` computes DSGE-VAR based variance decompositions for the original or the levels data. It requires 6 input variables to achieve its task: `Phi`, `A0`, `p`, `h`, `VDType`, and `levels`. All variables except `VDType` are used by `DSGEVARImpulseResponses` discussed above. The `VDType` input is an integer that takes the value 1 if the forecast errors concern the original data and 2 if they refer to the levels data.

The three output variables from the function are called `FEVDs`, `LRVD` and `VarShares`. The first is a matrix of dimension $n \times n \times h$ with the forecast error variance decompositions for the n endogenous variables in the rows, the n structural shocks in the columns, and the forecast horizon in the third dimension, with h being the maximum horizon. The second output variable, `LRVD`, is an $n \times n$ matrix with the long-run variance decompositions. For the original data, the long-run is determined by the maximum of h and 200, and for levels data it is computed via the matrix R_{lr} in Section 16.3 along with the expression in equation (11.30). The final output variable is given by `VarShares`, which measures the convergence to the long-run by dividing the h-step ahead forecast error variances with the long-run variances. Values close to unity suggest convergence, while small values indicates the model has not yet converged to the long run.

16.11.3. DSGEVARObsVarDecomp

The function `DSGEVARObsVarDecomp` calculates the observed variable decompositions from a DSGE-VAR model. It takes 7 input variables: `Phi`, `A0`, `p`, `k`, `y`, `Y`, and `IsConstant`. The first 4 variables have been discussed above, but it may be noted that `Phi` includes the parameters on the exogenous variables. The input variable `y` is an $n \times T$ matrix with the data on the endogenous variables, while `Y` is an $np+k \times T$ matrix with the data on the exogenous and lagged endogenous variables. Finally, the variable `IsConstant` is a boolean variable that is unity if $x_t = 1$ for all t and zero otherwise. The decomposition in (16.6) is used in the latter case, while (16.8) is used in the former.

The three output variables are given by `yMean`, `yInitial`, and `yShocks`. The first is an $n \times T$ matrix with the population mean of the data given the parameters (and the exogenous variables term when `IsConstant` is zero). Similarly, the variable `yInitial` is an $n \times T$ matrix with the initial value term in the decomposition. The last output variable is `yShocks`, an $n \times T \times n$ matrix with the decomposition of the endogenous variables (rows) for each time period (columns) due to the n structural shocks (third dimension).

16.11.4. DSGEVARCorrelationParam

The function `DSGEVARCorrelationParam` requires only three input variables: `Phi`, `SigmaEpsilon` and `DSGEModel`. The matrix `Phi` is of dimension $n \times np + k$ are thus includes the parameters on the exogenous variables as well as those on lagged endougenous. The matrix `SigmaEpsilon` is the residual covariance matrix, while the last input variable is well known.

The output variable is called `SVEs` and is a structure with 7 fields. The DSGE-VAR model population mean matrix of dimension $n \times k$ is stored in the field `Mean`. To get the actual population mean, this matrix needs to be multiplied by x_t for each time period. The population mean computation is based on the approximation

$$\tilde{\mu}_{y_t} = J_p' \left(I_{np} - \Psi \right)^{-1} J_p \Phi_0 x_t,$$

i.e., an expression which is correct for $x_t = 1$. The actual population mean is given by

$$\mu_{y_t} = \sum_{j=0}^{\infty} J_p' \Psi^j J_p \Phi_0 x_{t-j}.$$

To get an idea about the approximation error, YADA computes μ_{y_t} through the recursion

$$\mu_{y_t} = \Phi_0 x_t + \sum_{i=1}^{p} \Phi_i \mu_{y_{t-i}},$$

based on the initial values $\mu_{y_\tau} = \tilde{\mu}_{y_\tau}$ for $\tau = 0, \ldots, 1 - p$. The maximum (absolute) approximation error for $\tilde{\mu}_{y_t} - \mu_{y_t}$ is stored in the field meanerror, a vector of dimension n.

Furthermore, the field AutoCovHorizon is a vector with integer values ranging from $-h$ to h, with h being the largest autocovariance horizon. The field Cov contains the contemporaneous covariance matrix, and Std the vector of standard deviations. The autocorrelations are available in the field Corr, a matrix with dimensions $n(n+1)/2 \times 2h + 1$.

In addition to the population moments, the structure SVEs also has a field called Data with the data moments. This field is itself a structure with fields Mean, Cov, Std, and Corr.

16.11.5. DSGEVARCorrelationSimulationParam

The function DSGEVARCorrelationSimulationParam computes sample-based moments from a DSGE-VAR by simulating data from the model. The function requires the 7 input variables: Phi, SigmaEpsilon, NumPaths, lambda, EstStr, DSGEModel, and CurrINI. The first two variables are identical to those in DSGEVARCorrelationParam, while the last two are well known. The NumPaths variable is an integer with the number of paths to simulate for the observed variables. The lambda variable is the usual λ hyperparameter for the DSGE-VARs, while EstStr is a string vector that reflects the type of parameter estimates that are used by the function.

The output variable is the structure SVEs, which has the same field entries as the same named output variable of DSGEVARCorrelationParam, except that the field meanerror is not used.

16.11.6. DSGEVARCorrDecompParam

The function DSGEVARCorrDecompParam computes the correlation decompositions of a DSGE-VAR for fixed parameter values. This task is achieved through 5 input variables: Phi, A0, EstStr, DSGEModel, and CurrINI. The first input variable is the $n \times np$ matrix obtained from Φ in equation (15.10) by excluding the parameters on exogenous variables (Φ_0). All other input variables have been discussed above.

As output the function provides the structure CorrDec. The fields are nearly identical to the fields of this structure provided by the DSGE model correlation decomposition function DSGECorrelationDecompTheta; cf. Section 11.18.7. The exceptions concern the fields Xi, which is missing, and the field Y, which here has the dimension $n \times (2h + 1) \times n$, reflecting that the number of shocks of the DSGE-VAR is n rather than q and that there is no measurement error.

16.11.7. DSGEVARConditionalCorrsParam

The function DSGEVARConditionalCorrsParam computes either sample-based or population-based conditional correlations for fixed parameter values. To deal with its task it takes 8 input variables: Phi, A0, lambda, ShockNames, NumPaths, EstStr, DSGEModel, and CurrINI. The matrix Phi here includes parameters on the deterministic variables and therefore has dimension $n \times (np + k)$. Apart from ShockNames, a string matrix with the names of the structural shocks in the rows, the remaining variables have been discussed above.

The conditional correlations results are provided in the vector structure CondCorr, whose length is equal to the number of structural shocks, i.e., n. For each element of this vector structure the fields are given by Mean and possibly Quantiles. The latter fields is included when sample-based conditional correlations have been calculated. Furthermore, the first element of CondCorr also contains the field ShockNames.

16.11.8. DSGEVARCoherenceParam

The function DSGEVARCoherenceParam computes the coherence using the population spectrum of a DSGE-VAR for fixed parameter values. To this end the function takes 6 input variables: Phi, SigmaEpsilon, lambda, EstStr, DSGEModel, and CurrINI. The first input variable is the $n \times np$

matrix obtained from Φ in equation (15.10) by excluding the parameters on exogenous variables (Φ_0). All other variables are shared with `DSGEVARCorrelationSimulationParam` above.

Two output variables are provided, `SOmega` and `Omega`. The first is an $n(n-1)/2 \times f$ matrix with the unique coherence values for pairs of observed variables across the f frequencies, obtained from the vector `Omega`. Like for all other spectral density functions in YADA, the integer $f = 300$, while the entries in `Omega` are given by $\omega_j = \pi(j-1)/299$; see, e.g., the `DSGECoherenceTheta` function in Section 13.9.3.

16.11.9. DSGEVARSpectralDecomposition

The function `DSGEVARSpectralDecomposition` requires 6 input variables: `Phi`, `A0`, `lambda`, `EstStr`, `DSGEModel`, and `CurrINI`. The `Phi` matrix does not include parameters on the deterministic variables and is therefore setup in the same way as for the function that computes the forecast error variance decomposition; cf. `DSGEVARVarianceDecompositions` above. The scalar `lambda` gives the λ hyperparameter, while `EstStr` is a string that provides information about which values of the VAR parameters that are used, i.e., the initial values, the marginal or the joint posterior mode values. The structures `DSGEModel` and `CurrINI` are by now well known.

The output variables from the function are first of all given by `SyOmega` and `SyOmegaAnnual`. The first variable is an $n \times f$ cell array of $n \times n$ matrices with the spectral decompositions $s_y^{(j)}(\omega)$ for shock j and frequency ω; see equation (16.17). The second variable is the annualized version of the conditional population spectrum; see Section 13.5. Next, the vectors `OriginalVariance` and `AnnualVariance` are given as output. These hold the model-based population variances of the endogenous variables and of the annualized endogenous variables, respectively. The fifth and last output variable is `Omega`, an f-dimensional vector with the 300 frequencies that are considered, i.e., $\omega = 0, \pi/299, 2\pi/299, \ldots, \pi$.

16.11.10. DSGEVARPredictionPathsParam

The function `DSGEVARPredictionPathsParam` computes unconditional forecasts of the endogenous variables for fixed values of the parameters. To achieve this objective, 12 input variables are needed: `Phi0`, `Phi`, `SigmaEpsilon`, `X`, `LastPeriod`, `h`, `lambda`, `NumPaths`, `EstStr`, `ForecastType`, `DSGEModel`, and `CurrINI`. The matrix `Phi0` contains the parameters on the exogenous variables, while `Phi` are the parameters on lagged endogenous variables and the residual covariance matrix is given by `SigmaEpsilon`. Data over the forecast sample on the exogenous variables are located in the $k \times h$ matrix `X`, while the integer `LastPeriod` gives the position in the full sample of the last period that is viewed as observed. This means that y_T is located in position `LastPeriod` of the matrix `DSGEModel.Y`. The integer `h` is the maximum forecast horizon, while `lambda` is the usual λ hyperparameter that defines the weight on the prior for the DSGE-VAR. The integer `NumPaths` is equal to the number of forecast paths to calculate, `EstStr` is a string that indicates if initial values, marginal or joint posterior mode values are used for the computations. The integer `ForecastType` is equal to 1 if the original data format of the endogenous variables is forecasted, 2 if annualized data is forecasted, and 3 if the function should forecast transformed data. The last two input variables are familiar.

The number of output variables provided by the function is 4. They are given by `PredPaths`, `PredData`, `PredEventData`, and `YObsEventData`. These variables are nearly the same as those supplied by the unconditional forecasting function, `DSGEPredictionPathsTheta`, for the DSGE model. Specifically, the fields of `PredData` differ somewhat. The DSGE-VAR forecasting function has the following fields: `PredMean`, `epsShocks`, `epsMean`, `Shocks`, `KernelX`, and `KernelY`. The `PredMean` field gives the $n \times h$ matrix with population mean forecasts of the endogenous variables. The field `epsShocks` is an $n \times h$ matrix with the population mean of the residuals over the forecast horizon, i.e., a zero matrix, while `epsMean` is the sample mean of the simulated residuals. The paths for all residuals are stored in the matrix `Shocks`, with dimensions $n \times h \times P$ (with P being the number of simulated paths). The last two fields of the `PredData` structure give the horizontal and vertical axes values for kernel density estimates of the marginal predictive densities. These matrices have dimensions $n \times 2^8 \times h$.

16.11.11. `DSGEVARPredictionPaths`

The function `DSGEVARPredictionPaths` needs 16 input variables. The prior or posterior draws of the parameters of the VAR are given by `PhiDraws` and `SigmaDraws`. The matrices have dimensions $d \times n(np + k)$ and $d \times n(n + 1)/2$, respectively, where d is the number of draws from the corresponding parameter distribution. Each row of the `PhiDraws` matrix is given by the transpose of the column vectorization of Φ, while the rows of `SigmaDraws` contains the draws of $\text{vech}(\Sigma_\epsilon)'$. The following two input variables are given by `PhiMode` and `SigmaModeEpsilon`, fixed parameter values of Φ and Σ_ϵ which are used as backup values for the VAR parameters. For the prior distribution these matrices are equal to the mode values given the initial values of the DSGE model parameters, and for the posterior by the mode values given the posterior mode values of the DSGE model parameters. The latter parameter vector depends on how the posterior sampler was parameterized.

The next four input variables are directly related to forecasting. They are: `X`, `LastPeriod`, `h`, and `ForecastType` and are identical to the variables with the same names in the unconditional forecast function for fixed parameter values discussed above (`DSGEVARPredictionPathsParam`). Thereafter we have 5 input variables related to sampling of the distribution for the DSGE-VAR model: `CurrChain`, `IsPosterior`, `PME`, `lambda`, and `NumPaths`. The first is an integer that denotes the number of the selected Markov chain; for the prior distribution this variable can be empty. The following is a boolean variable which is unity if draws from the posterior distribution are provided to the function and 0 if they stem from the prior. Next, `PME` is an integer that is 1 if the posterior mode estimator of the DSGE model parameters used by the posterior sampler of the DSGE-VAR is taken from the DSGE model, 2 if it was given by the marginal posterior mode estimator from the DSGE-VAR model of θ, and 3 if was the joint posterior mode from the DSGE-VAR. The following variable simply gives the λ hyperparameter of the DSGE-VAR, while the number of paths per parameter value is `NumPaths`. The last 3 input variables are `DSGEModel`, `CurrINI` and `controls`, all being documented above.

As output the function provides 4 variables. The first is the boolean `DoneCalc` that indicates if the computations were completed or not. The next is the matrix `PredEventData` with prediction event results. The last two variables give the decomposition of the prediction uncertainty into the residual/shock uncertainty and parameter uncertainty terms; see (16.25) and (16.26). Both these variables are 3D matrices with dimensions $n \times n \times h$. The actual prediction paths are stored in mat-files on disk and are not provided as output variables from the function.

16.11.12. `DSGEVARCondPredictionPathsParam(WZ/Mixed)`

The function `DSGEVARCondPredictionPathsParam` computes conditional forecasts of the observed variables for fixed values of the parameters using the direct control of the shocks method discussed in Section 16.9.1, `DSGEVARCondPredictionPathsParamWZ` makes use of the Waggoner and Zha approach discussed in Section 16.9.2, while `DSGEVARCondPredictionPathsParamMixed` is based on the distribution of a subset of the shocks method in Section 16.9.3. In addition to the 12 variable required by the function `DSGEVARPredictionPathsParam` for unconditional predictions, the conditional forecasting functions both require two further variables: `Z` and `U`. The former holds the conditioning data, while the latter takes care of initial conditions (u_T in equation (12.6)). It should also be noted that since `DSGEVARCondPredictionPathsParam` uses the direct control of shocks method it needs the matrix A_0, which is multiplied by the structural shocks, instead of the residual covariance matrix Σ_ϵ. Hence, the input variable `A0` replaces `SigmaEpsilon` in this case. Moreover, this function also needs the input variable `DSGEVARShocks`, a vector with integer values determining the positions among the shocks in the DSGE model of the shocks used in the DSGE-VAR model. This variable is not needed by the `DSGEVARCondPredictionPathsParamWZ` function since it does not require structural shocks.

Both function provide seven output variables. The first four are identical to the outputs given by `DSGEVARPredictionPathsParam`. The final three variables are the modesty statistics called `MultiModestyStat`, `UniModestyStat`, and `UniModestyStatLZ`, and they are only calculated when `ForecastType` is unity. When this condition is met, `MultiModestyStat` is a matrix

of dimension `NumPaths` times 2, where the first columns holds the values of $\mathcal{M}_{T,g}(\bar{\Upsilon}_{T+g})$, while the second column gives $\mathcal{M}_{T,g}(\Upsilon_{T+g})$ under the direct control of the shocks method; see equation (16.55); the shock processes $\bar{\Upsilon}_{T+g}$ and Υ_{T+g} are replaced with $\bar{\mathcal{E}}_{T+g}$ and $\mathcal{E}_{T+g}$, respectively, when the Waggoner and Zha approach is applied. The matrix `UniModestyStat` has dimension `NumPaths` times n and gives the univariate modesty statistics, while `UniModestyStatLZ` is a vector with the n values of the univariate Leeper-Zha related modesty statistic.

16.11.13. DSGEVARCondPredictionPaths(WZ/Mixed)

The function `DSGEVARCondPredictionPaths` computes conditional forecasts of the observed variables for a sample of parameters from the prior or the posterior distribution using the direct control of shocks method; see Section 16.9.1. Similarly, `DSGEVARCondPredictionPathsWZ` performs the same task using the Waggoner and Zha method that was discussed in Section 16.9.2 and the function `DSGEVARCondPredictionPathsMixed` is based on the distribution of a subset of the shocks method in Section 16.9.3. Both functions need a total of 18 input variables. To begin with, 16 input variables are shared with the function `DSGEVARPredictionPaths`. Since the direct control of shocks method requires the structural form of the DSGE-VAR, the matrix `SigmaDraws` is replaced with `A0Draws` and the covariance matrix `SigmaModeEpsilon` with `A0Mode`. The remaining two input variables are given by `Z` and `U`, which contains the conditioning assumptions; see the last Section above.

Both functions provide 6 output variables, 4 of which are shared with unconditional forecasting function `DSGEVARPredictionPaths`. In addition, the conditional forecasting functions provide the variables `ShockMean` and `ShockNames`. The former variable is a matrix of dimension $n \times h$ with the population mean over the prediction horizon of the residuals needed to ensure that the conditioning assumptions are satisfied. The population mean is estimated as the average over the used VAR parameter draws of the population mean of the resisuals for a fixed value of these parameters. Finally, the `ShockNames` output variable is a string matrix with the names of the structural shocks used by the DSGE-VAR.

16.11.14. DSGEVARPredictiveLikelihoodParam

The function `DSGEVARPredictiveLikelihoodParam` calculates the joint and marginal predictive likelihood of a DSGE-VAR model for fixed parameter values using the Laplace approximation. The function needs 28 input variables to achieve this: `theta`, `thetaPositions`, `thetaIndex`, `thetaDist`, `PriorDist`, `LowerBound`, `UniformBounds`, `ModelParameters`, `AIMData`, `lambda`, `T`, `n`, `p`, `npk`, `GammaHatyy`, `GammaHatyY`, `GammaHatYY`, `DetProductMoments`, `HSample`, `logGPR`, `YData`, `X`, `IsOriginal`, `IsPlugin`, `ModeValue`, `StepLength`, `DSGEModel`, and `CurrINI`. `ModeValue` is an integer that takes the value 1 if posterior mode value from the DSGE model or the initial values are supplied, the value 2 when marginal posterior mode values are given, and 3 if joint posterior mode values are provided. In the latter case, the value of `logGPR` is interpreted as `logCPC`, the constant term in the concentrated likelihood function if the DSGE-VAR.

The six output variables are `JointPDH`, `MargPDH`, `LaplaceMargLike`, `PredVars`, `MargPlugin`, and `JointPlugin`. All these variables have been discussed in Section 12.11.14 in connection with the function `DSGEPredictiveLikelihoodTheta`.

17. Learning in DSGE Models

The structural form of a DSGE model is given in equation (3.2) and is for convenience repeated here:

$$H_{-1} z_{t-1} + H_0 z_t + H_1 E_t \left[z_{t+1} \right] = D \eta_t, \tag{17.1}$$

where z_t is a p-dimensional vector of model variables and η_t is a q-dimensional vector of Gaussian i.i.d. shocks with identify covariance matrix. Under the assumption of *rational expectations* (RE), the solution to this stochastic difference equation is given by equation (3.3), i.e.

$$z_t = B_1 z_{t-1} + B_0 \eta_t, \tag{17.2}$$

where B_1 and B_0 satisfies the conditions provided in Section 3. This corresponds to the state equation (5.2) in Section 5 with $\xi_t \equiv z_t$ and $B_1 = F$, while the measurement equation is given by (5.1).

Over the last decades, alternative approaches to modelling expectations have been suggested in the literature. These include but are not limited to the bounded rationality model of Sargent (1993), rational inattention as in Sims (2003), the sticky information model of Mankiw and Reis (2002), partial information as in Svensson and Woodford (2003) and the learning approach of Evans and Honkapohja (2001). Below we discuss the adaptive learning approach suggested by Slobodyan and Wouters (2012) to DSGE models.

17.1. *Updating Model Expectations through a Kalman Filter*

To relax the strict implications of the RE assumption, Slobodyan and Wouters (2012) assume that agents forecast the forward looking variables of a model as a reduced form of the lagged state variables. A special case of this is given by the expression in equation (17.2), but it is also possible that the reduced form model differs from the RE solution. First, the parameters of the reduced form need not satisfy the cross equation restrictions of the RE solution. Second, the reduced form may involve additional lags of the state variables or include other variables, such as the deterministic variables x_t, which influence the law of motion of the forward looking variables in z_t. In this section we shall first consider a transformation of the structural form where only the forward looking variables appear in the expectations term. After we have established such a rewrite of the model, the so called *perceived law of motion* (PLM) and updating of its belief parameters through a Kalman filter are considered.

17.1.1. *A Transformation of the Structural Form*

The forward looking variables may be determined by inspecting the model equations. For example, in the case of the An and Schorfheide (2007) model in Section 2.1 we have three *candidates* in equation (2.1): consumption ($\hat{c}_t$), inflation $\hat{\pi}_t$ and technology $\hat{z}_t$. By being candidates we only mean that they appear in the expectation term. Other models, such as the Smets and Wouters (2007) model in Section 2.4 has 12 possible forward looking variables, seven in the sticky economy (consumption, hours, investment, real value of the existing capital stock, real rate of capital, inflation and real wages) and five in the flexible (same variables excluding inflation and real wages). In fact, when applying adaptive learning to the Smets and Wouters model Slobodyan and Wouters (2012) shut down the flexible part of the model economy and assume that the output gap is instead determined through the TFP shocks, $\hat{\epsilon}_t^a$, thereby reducing the number of forward looking variables to exactly seven. Note that all forward looking variables are non-predetermined in the sense of Buiter (1982), but at least in principle there may be fewer forward looking variables than non-predetermined variables.[135]

The forward looking variables in the model can be extracted from z_{t+1} in equation (17.1) by constructing a 0-1 selection matrix S of dimension $p \times f$ and having rank $f \leq p$ such that

$$z_t^f = S' z_t.$$

---

[135] This is related to the rank of H_1 and the fact that the number of non-zero columns of this matrix can be greater than its rank.

Note that S is made up of f distinct columns of I_p. The remaining $p - f$ columns are denoted by $S_\perp$ such that the non-forward looking variables are given by

$$z_t^{nf} = S_\perp' z_t.$$

We can now define the matrix $\tilde{S}$, which we will employed to transform (re-order) the structural equations and the state variables, as follows:

$$\tilde{S} = \begin{bmatrix} S & S_\perp \end{bmatrix},$$

where $\tilde{S}^{-1} = \tilde{S}'$.

The order of the equations in the DSGE model is arbitrary and for the transformation approach below it is important that the order at least temporarily follows the order in which the forward looking variables appear in z. Moreover, it is required that

$$\text{rank}\begin{bmatrix} H_1 \end{bmatrix} \leq f,$$

i.e., the rank of H_1 provides a lower bound for the number of forward looking variables that are supported by the model. In YADA practise, this means that the user selects f forward looking variables and YADA thereafter checks if the corresponding f columns of H_1 are all nonzero and have rank equal to the rank of H_1.

Concerning the reordering of equations, each forward looking variable in the expectation term should appear in an equation having the same order number as the variable itself has among z. To this end, the matrix C is defined as a $p \times p$ matrix of rank p containing only unique rows from I_p. This means that C satisfies $C'C = CC' = I_p$.[136]

Premultiplying the system in (17.1) by $\tilde{S}'C$, it can be rewritten as follows

$$\tilde{H}_{-1}\tilde{z}_{t-1} + \tilde{H}_0\tilde{z}_t + \tilde{H}_1 E_t\begin{bmatrix} \tilde{z}_{t+1} \end{bmatrix} = \tilde{D}\eta_t, \tag{17.3}$$

where $\tilde{z}_t = \tilde{S}'z_t$, $\tilde{H}_i = \tilde{S}'CH_i\tilde{S}$ for $i = -1, 0, 1$, and $\tilde{D} = \tilde{S}'CD$. This means that the equations for the forward looking variables are ordered in the top f equations (rows) and the bottom $p - f$ equations (rows) are those for the non-forward looking variables. Furthermore, the former variables are given in the first f rows of $\tilde{z}_t$ and the latter variables in the bottom $p - f$ rows.

The matrices $\tilde{H}_i$ can be expressed in matrix blocks as follows

$$\tilde{H}_i = \begin{bmatrix} S'CH_iS & S'CH_iS_\perp \\ S_\perp'CH_iS & S_\perp'CH_iS_\perp \end{bmatrix}, \quad i = -1, 0, 1.$$

In the case of $\tilde{H}_1$, the $f \times f$ sub-matrix $S'CH_1S$ has full rank f, while the sub-matrix $S_\perp'CH_1S_\perp = 0$. These results follow directly from the assumptions that z_t^f is forward looking and that z_t^{nf} is non-forward looking. For the sub-matrix in the bottom right corner of $\tilde{H}_1$ to be zero, we find that either $CH_1S_\perp = 0$ or $S_\perp'CH_1 = 0$.

For the case when $CH_1S_\perp = 0$, we find that the final $p - f$ columns of $\tilde{H}_1$ are zero and that only the expected next period values of the forward looking variables appear in the p equations. There is therefore no need for any further transformation of the DSGE model as

$$\tilde{H}_1 E_t\begin{bmatrix} \tilde{z}_{t+1} \end{bmatrix} = \tilde{H}_{1,f} E_t\begin{bmatrix} z_{t+1}^f \end{bmatrix}.$$

Based on this we can replace the expectations with the adaptive learning mechanism for the forward looking variables when we solve the model.

The second case with $S_\perp'CH_1 = 0$ is somewhat more complicated as we need to transform the system by substituting for the expectations of the non-forward looking variables in the top f equations. To accomplish this, we note that the bottom $p - f$ equations in (17.3) do not involve

---

[136] Let ι_f denote an f dimensional vector with integers giving the position of each forward looking variable in z. Similarly, let ι_e be an f dimensional vector with positions of the rows in H_1 that are non-zero. This vector need not be unique as more rows than f may contain non-zero elements. For such situations, it is only required that the sub-matrix formed from H_1 using the rows ι_e and the columns ι_f has rank f. Provided that this condition is met, C is constructed by first setting it to I_p. Next, the rows ι_e in C are replaced with the rows ι_f from I_p. Last, the rows ι_f of C are replaced with ι_e of I_p.

any expectations, but only contemporaneous and lagged variables. Under the assumption that the model has a solution, it follows that

$$\text{rank}\left(S'_\perp CH_0 S_\perp\right) = p - f.$$

Accoringly, the solution of the DSGE model includes the following representation for the non-forward looking variables

$$z_t^{nf} = -\left(S'_\perp CH_0 S_\perp\right)^{-1} S'_\perp CH_0 S z_t^f - \left(S'_\perp CH_0 S_\perp\right)^{-1}\left[S'_\perp CH_{-1}S \quad S'_\perp CH_{-1}S_\perp\right]\tilde{z}_{t-1}$$
$$+ \left(S'_\perp CH_0 S_\perp\right)^{-1} S'_\perp D\eta_t.$$

From this equation we see that the unbiased expectation of the non-forward looking variables at $t + 1$ based on information at t is

$$E_t\left[z_{t+1}^{nf}\right] = -\left(S'_\perp CH_0 S_\perp\right)^{-1} S'_\perp CH_0 S E_t\left[z_{t+1}^f\right] - \left(S'_\perp CH_0 S_\perp\right)^{-1}\left[S'_\perp CH_{-1}S \quad S'_\perp CH_{-1}S_\perp\right]\tilde{z}_t.$$

Substituting this into equation (17.3), making use of $S'_\perp CH_1 = 0$ and collecting terms, we obtain

$$\bar{H}_{-1}\tilde{z}_{t-1} + \bar{H}_0\tilde{z}_t + \bar{H}_1 E_t\left[\tilde{z}_{t+1}\right] = \tilde{D}\eta_t, \tag{17.4}$$

where $\bar{H}_{-1} = \tilde{H}_{-1}$,

$$\bar{H}_0 = \tilde{H}_0 - \begin{bmatrix} S'CH_1 S_\perp\left(S'_\perp CH_0 S_\perp\right)^{-1} S'_\perp CH_{-1}S & S'CH_1 S_\perp\left(S'_\perp CH_0 S_\perp\right)^{-1} S'_\perp CH_{-1}S_\perp \\ 0 & 0 \end{bmatrix},$$

while

$$\bar{H}_1 = \begin{bmatrix} S'CH_1 S - S'CH_1 S_\perp\left(S'_\perp CH_0 S_\perp\right)^{-1} S'_\perp CH_0 S & 0 \\ 0 & 0 \end{bmatrix}.$$

For this transformation we find that

$$\bar{H}_1 E_t\left[\tilde{z}_{t+1}\right] = \bar{H}_{1,f} E_t\left[z_{t+1}^f\right].$$

At this stage, it is useful to premultiply the structural form by $C'\tilde{S}$ and replace $\tilde{z}_t$ and $\tilde{z}_{t-1}$ with z_t and z_{t-1}, respectively, while the expectations term is kept. This provides us with

$$H_{-1}^* z_{t-1} + H_0^* z_t + H_{1,f}^* E_t\left[z_{t+1}^f\right] = D\eta_t. \tag{17.5}$$

The structural form matrices are now given by $H_{-1}^* = H_{-1}$,

$$H_0^* = \begin{cases} C'\tilde{S}\bar{H}_0\tilde{S}' = H_0 & \text{if } CH_1 S_\perp = 0, \\ C'\tilde{S}\bar{H}_0\tilde{S}' & \text{if } S'_\perp CH_1 = 0, \end{cases}$$

and

$$H_{1,f}^* = \begin{cases} C'\tilde{S}\bar{H}_{1,f} = H_1 S & \text{if } CH_1 S_\perp = 0, \\ C'\tilde{S}\bar{H}_{1,f} & \text{if } S'_\perp CH_1 = 0. \end{cases}$$

These conditions and transformations are simple to apply once the forward looking variables have been established. The case when all columns of H_1 that are multiplied by a non-forward looking variable are zero is very easy to spot and require hardly any rewrite of the DSGE model. The second case when all rows of H_1 in the equations for the non-forward looking variables are zero, require a little bit more work, but is swiftly dealt with by computer code.

Finally, the requirement that f is at least equal to the rank of H_1 hints at a deeper issue concerning the uniqueness of solutions under adaptive learning. Different but equivalent formulations of a DSGE model from a rational expectations perspective can all have the same unique solution, but this is not necessarily the case when rational expectations are replaced with adaptive learning. Rather, the choice of forward looking variables has a direct implication for the solution. How different the solutions are for the various possibilities is an empirical question, but it needs to be recognized. Furthermore, the matrix H_1 has rank 5 in the Slobodyan and Wouters (2012) model while the number of nonzero columns of this matrix is 7.

This means that we may choose between five and seven forward looking variables. At the same time, not all of the seven candidates can be selected if we set $f = 5$. For any valid selection of five forward looking variables, the matrix $H_1 S$ must have rank f.

17.1.2. *The Perceived Law of Motion and Kalman Filtering*

When applying adaptive learning to a DSGE model, it is important to make sure that all the forward looking variables are endogenous, i.e., that some of them are *not* specified as being backward looking, such as a shock processes. This would clearly lead to an inconsistent system since we cannot have two separate backward looking equations for the same variable. One such variable appears in the An and Schorfheide (2007) model: technology ($\hat{z}_t$). Hence, this model only has two forward looking variables in consumption and inflation. Furthermore, suppose we replace consumption in the consumption Euler equation with the expression in the aggregate resource constraint. The model now has four variables in expectations with consumption being replaced with output and government spending. Since the latter variable is also exogenous and driven by and AR(1) process, it is not consistent with the model structure to include a separate belief equation for it. YADA deals with this issue by allowing the user to specify the forward looking variables among the candidates that it has derived from the H_1 matrix. For this version of the model, the forward looking variables are output and inflation. Once the user selected forward looking variables have been specified, the matrix S can be constructed and hopefully the user has chosen wisely.

Slobodyan and Wouters (2012) assume that each forward looking variable, $z_{t,j}^f$, is determined by a limited number of variables, denoted by $q_{t-1,j}$ and having dimension g_j, through the PLM

$$z_{t,j}^f = q_{t-1,j}' \beta_{t-1,j} + u_{t,j}, \quad j = 1, \ldots, f. \tag{17.6}$$

In their benchmark case, $q_{t-1,j}$ is given by a constant and two lags of $z_{t,j}^f$. Below we consider a general expression for the PLM, with the only restriction that it can only contain z variables and deterministic variables.

Stacking the forecasting model in equation (17.6) in SURE form yields

$$z_t^f = q_{t-1}' \beta_{t-1} + u_t, \tag{17.7}$$

where u_t is i.i.d. Gaussian with zero mean and covariance matrix

$$E[u_t u_t'] = \Sigma_u,$$

an $f \times f$ positive definite matrix. The u_t's are linear combinations of the model innovations. Furthermore, we have that

$$q_{t-1} = \begin{bmatrix} q_{t-1,1} & 0 & \cdots & 0 \\ 0 & q_{t-1,2} & & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & & q_{t-1,f} \end{bmatrix},$$

is a $g \times f$ matrix with $g = \sum_{j=1}^{f} g_j$. Consequently, β_t is a time-varying vector of dimension g, capturing how the PLM changes over time.

The β_t vector is estimated with a Kalman filter which treats (17.7) as a measurement equation. The state equation is assumed to be

$$\beta_t = \beta + \Gamma(\beta_{t-1} - \beta) + \varepsilon_t, \tag{17.8}$$

where β is the initial belief (steady-state) for the unobserved β_t-process. The $g \times g$ matrix Γ is assumed to be a diagonal matrix with

$$\Gamma = \rho I_g,$$

where the scalar parameter $0 \leq \rho \leq 1$. The g-dimensional vector of errors ε_t is assumed to be i.i.d. Gaussian with mean zero, independent of u_t, and with

$$E\left[\varepsilon_t \varepsilon_t'\right] = \Sigma_\varepsilon,$$

a $g \times g$ positive definite matrix.

The Kalman filter forecasting (transition) and updating equations for the belief coefficients and their covariance matrices are given by

$$\beta_{t|t} = \beta_{t|t-1} + R_{t|t-1} q_{t-1} \left(q_{t-1}' R_{t|t-1} q_{t-1} + \Sigma_u \right)^{-1} \left(z_t^f - q_{t-1}' \beta_{t-1|t-1} \right), \qquad (17.9)$$

$$\beta_{t+1|t} = \beta + \Gamma \left(\beta_{t|t} - \beta \right), \qquad (17.10)$$

$$R_{t|t} = R_{t|t-1} - R_{t|t-1} q_{t-1} \left(q_{t-1}' R_{t|t-1} q_{t-1} + \Sigma_u \right)^{-1} q_{t-1}' R_{t|t-1}, \qquad (17.11)$$

$$R_{t+1|t} = \Gamma R_{t|t} \Gamma' + \Sigma_\varepsilon. \qquad (17.12)$$

Notice that $q_{t-1}' \beta_{t-1|t-1}$ is the one-step-ahead forecast of z_t^f, while $q_{t-1}' R_{t|t-1} q_{t-1} + \Sigma_u$ is the corresponding covariance matrix.

The estimation of β_t is based on all variables in the PLM being observable. In practise, however, we observe y_t, a smaller dimensional vector such that z_t^f and q_{t-1} need to be replaced with $z_{t|t}^f$ and $q_{t-1|t-1}$, respectively. We return to the details in Section 17.5 where we introduce the full algorithm. In addition, the parameters β, Σ_u, Σ_ε, as well as the initial values for $\beta_{1|0}$ and $R_{1|0}$ need to be determined. We turn to this problem next.

17.2. Initial Values for the Beliefs

Slobodyan and Wouters (2012) suggest to use population moments to determine the initial values for the belief parametera. Specifically, they let

$$\beta_{1|0} = \beta = E\left[q_{t-1} q_{t-1}'; \theta\right]^{-1} E\left[q_{t-1} z_t^f; \theta\right]$$

where θ is the parameter vector of the DSGE model under RE. Similarly, the covariance matrix of the expectation errors is given by

$$\Sigma_u = E\left[\left(z_t^f - q_{t-1}' \beta \right) \left(z_t^f - q_{t-1}' \beta \right)'; \theta \right].$$

Finally, they let

$$R_{1|0} = \sigma_r \left(E\left[q_{t-1} \Sigma_u^{-1} q_{t-1}'; \theta \right] \right)^{-1},$$

$$\Sigma_\varepsilon = \sigma_\varepsilon \left(E\left[q_{t-1} \Sigma_u^{-1} q_{t-1}'; \theta \right] \right)^{-1},$$

where σ_r and σ_ε are positive scalars, while the $g \times g$ matrix appearing in $R_{1|0}$ and Σ_ε is called the "GLS" matrix below. The determination of parametric expressions for the population moments conditional on θ and based on the RE version of the DSGE model is the target for the discussion below. However, it may first be noted that the learning dynamics involves three unknown parameters in addition to θ. Namely, the scale parameters σ_r and σ_ε as well as the autocorrelation parameter ρ. Slobodyan and Wouters (2012) estimate the latter parameter while the scale parameters are calibrated.[137]

---

[137] In their empirical application, Slobodyan and Wouters (2012) set the scale parameters to $\sigma_r = 0.03$ and $\sigma_\varepsilon = 0.003$, respectively, while ρ is provided with a $(0, 1)$ uniform prior.

17.2.1. *The β Vector*

The population covariances for the model variables z_t based on RE are shown in Section 5.3. That is,

$$E\left[z_t z'_{t-j}; \theta\right] = \begin{cases} \Sigma_\xi, & \text{if } j = 0, \\ F^j \Sigma_\xi, & \text{if } j = 1, 2, \ldots. \end{cases} \tag{17.13}$$

Consequently, we also have that $E[z_t^f z'_{t-j}; \theta] = S' E[z_t z'_{t-j}; \theta]$, $E[z_t x'_t; \theta] = 0$, while $\Sigma_x^{(0)} = (1/T) \sum_{t=1}^T x_t x'_t$. Let Z_{t-1} be a $(k + pm)$-dimensional vector such that

$$Z_{t-1} = \begin{bmatrix} x'_t & z'_{t-1} & \cdots & z'_{t-m} \end{bmatrix}',$$

while G_j is a $(k + pm) \times g_j$ selection matrix such that

$$q_{t-1,j} = G'_j Z_{t-1}, \quad j = 1, \ldots, f. \tag{17.14}$$

In other words, m is the maximum number of lags of the model variables that appear in $q_{t-1,j}$ for all f forward looking variables. The benchmark case in Slobodyan and Wouters (2012) is based on $m = 2$, with G_j selecting the first and the second lag of $z_{t,j}^f$ as well as a constant from x_t for all j.

To obtain an analytical expression for β (and $\beta_{1|0}$) based on the RE version of the DSGE model, we proceed by noting that $E[q_{t-1} q'_{t-1}; \theta]$ is a $g \times g$ block diagonal matrix with typical block element

$$E\left[q_{t-1,j} q'_{t-1,j}; \theta\right] = G'_j E\left[Z_{t-1} Z'_{t-1}; \theta\right] G_j = G'_j \Sigma_Z G_j, \quad j = 1, \ldots, f.$$

The $(k + pm) \times (k + pm)$ matrix Σ_Z is obtained from equation (17.13) as well as from the properties with the deterministic variables such that

$$\Sigma_Z = \begin{bmatrix} \Sigma_x^{(0)} & 0 & 0 & \cdots & 0 \\ 0 & \Sigma_\xi & F\Sigma_\xi & \cdots & F^{m-1}\Sigma_\xi \\ 0 & \Sigma_\xi F' & \Sigma_\xi & \cdots & F^{m-2}\Sigma_\xi \\ \vdots & \vdots & \vdots & \ddots & \\ 0 & \Sigma_\xi (F')^{m-1} & \Sigma_\xi (F')^{m-2} & & \Sigma_\xi \end{bmatrix}.$$

This matrix is positive semidefinite, but need not be positive definite. For example, z_t may include a contemporaneous entry for a variable and the first lag of the same variable. For this case, z_t and z_{t-1} share a variable and the covariance matrix for Z_t is therefore singular when $m > 1$.

Let S_j denote column j of S such that $z_{t,j}^f = S'_j z_t^f$. Furthermore, let J be a $(k + pm) \times p$ matrix such that

$$J' = \begin{bmatrix} 0_{p \times k} & I_p & 0_{p \times p(m-1)} \end{bmatrix}.$$

It can now be shown that

$$E\left[q_{t-1} z_t^f; \theta\right] = \begin{bmatrix} G'_1 \Sigma_Z J F' S_1 \\ \vdots \\ G'_f \Sigma_Z J F' S_f \end{bmatrix}.$$

It therefore follows that

$$\beta = \begin{bmatrix} \left(G'_1 \Sigma_Z G_1\right)^{-1} G'_1 \Sigma_Z J F' S_1 \\ \vdots \\ \left(G'_f \Sigma_Z G_f\right)^{-1} G'_f \Sigma_Z J F' S_f \end{bmatrix}, \tag{17.15}$$

a vector of equation-by-equation estimates of β_j for $j = 1, \ldots, f$.

17.2.2. *The Σ_u Matrix*

The derive an analytical expression of Σ_u, note that

$$q'_{t-1}\beta = (\beta' \otimes I_f)\text{vec}(q'_{t-1}).$$

Since q_{t-1} is a $g \times f$ matrix, we know from Magnus and Neudecker (1988) that

$$\text{vec}(q'_{t-1}) = K_{gf}\text{vec}(q_{t-1}),$$

where K_{gf} is a $gf \times gf$ commutation matrix. Hence,

$$q'_{t-1}\beta = (\beta' \otimes I_f)K_{gf}\text{vec}(q_{t-1}).$$

Next, define the $g \times f(k+pm)$ matrix G such that

$$G = \begin{bmatrix} G'_1 & 0 & \cdots & 0 \\ 0 & G'_2 & & 0 \\ \vdots & & \ddots & \\ 0 & 0 & & G'_f \end{bmatrix} = \begin{bmatrix} \tilde{G}_1 & \cdots & \tilde{G}_f \end{bmatrix},$$

where $\tilde{G}_j$ is $g \times (k+pm)$ containing columns $(k+pm)(j-1)+1$ until $(k+pm)j$ of G for $j = 1,\ldots,f$. Furthemore, let $\tilde{G}$ be a $gf \times (k+pm)$ matrix with

$$\tilde{G} = \begin{bmatrix} \tilde{G}_1 \\ \vdots \\ \tilde{G}_f \end{bmatrix}. \tag{17.16}$$

It then holds that

$$\text{vec}(q_{t-1}) = \tilde{G}Z_{t-1}.$$

Accordingly,

$$q'_{t-1}\beta = (\beta' \otimes I_f)K_{gf}\tilde{G}Z_{t-1} = \tilde{\beta}Z_{t-1}. \tag{17.17}$$

With this result in mind it follows that

$$E\left[z_t^f z_t^{f'};\theta\right] = S'\Sigma_\xi S = S'J'\Sigma_Z JS, \tag{17.18}$$

$$E\left[q'_{t-1}\beta z_t^{f'};\theta\right] = \tilde{\beta}\Sigma_Z JF'S, \tag{17.19}$$

$$E\left[q'_{t-1}\beta\beta' q_{t-1};\theta\right] = \tilde{\beta}\Sigma_Z \tilde{\beta}'. \tag{17.20}$$

Accordingly, an analytical expression of Σ_u is obtained from

$$\Sigma_u = S'J'\Sigma_Z JS - \tilde{\beta}\Sigma_Z JF'S - S'FJ'\Sigma_Z\tilde{\beta}' + \tilde{\beta}\Sigma_Z\tilde{\beta}'. \tag{17.21}$$

17.2.3. *The GLS Matrix*

Let the inverse of the Σ_u matrix be expressed as

$$\Sigma_u^{-1} = \begin{bmatrix} \omega_{11} & \cdots & \omega_{1f} \\ \vdots & & \vdots \\ \omega_{1f} & \cdots & \omega_{ff.} \end{bmatrix}$$

It is then straightforward to show that the $g \times g$ inverse of the GLS matrix is given by:

$$E\left[q_{t-1}\Sigma_u^{-1}q'_{t-1};\theta\right] = \begin{bmatrix} \omega_{11}G'_1\Sigma_Z G_1 & \cdots & \omega_{1f}G'_1\Sigma_Z G_f \\ \vdots & & \vdots \\ \omega_{1f}G'_f\Sigma_Z G_1 & \cdots & \omega_{ff}G'_f\Sigma_Z G_f \end{bmatrix}. \tag{17.22}$$

Notice that if Σ_u is singular or if $E[q_{t-1}\Sigma_u^{-1}q'_{t-1}; \theta]$ is, the matrix will be "inverted" by YADA using an eigenvalue decomposition where all zero eigenvalues and eigenvectors linked to them are removed.

17.3. *The Actual Law Of Motion*

To determine the actual law of motion (ALM) of all variables in z_t we first solve for the expectations in equation (17.1). The derivations employed to obtain equation (17.17) can be used to show that

$$q'_t \beta_t = \tilde{\beta}_t Z_t,$$

where

$$\tilde{\beta}_t = (\beta'_t \otimes I_f) K_{gf} \tilde{G}.$$

This means that

$$E_t \left[z_{t+1}^f \right] = \tilde{\beta}_{t|t} Z_t, \tag{17.23}$$

where $\tilde{\beta}_{t|t}$ is determined in Section 17.1. It is useful to decompose $\tilde{\beta}_t$ such that

$$\tilde{\beta}_t = \begin{bmatrix} \tilde{\beta}_{t,0} & \tilde{\beta}_{t,1} & \cdots & \tilde{\beta}_{t,m} \end{bmatrix}, \tag{17.24}$$

where $\tilde{\beta}_{t,0}$ is an $f \times k$ matrix and $\tilde{\beta}_{t,i}$ is an $f \times p$ matrix for $i = 1, \ldots, m$.

Before we turn to the general solution, let us assume that $m = 2$ and $x_t = 1$ as in Slobodyan and Wouters (2012). This means that

$$E_t \left[z_{t+1}^f \right] = \tilde{\beta}_{t|t,0} + \tilde{\beta}_{t|t,1} z_t + \tilde{\beta}_{t|t,2} z_{t-1}.$$

Making use of (17.5) and the above expression for the projected forward looking variables, provides us with the following structural form of the DSGE model

$$\left[H_0^* + H_{1,f}^* \tilde{\beta}_{t|t,1} \right] z_t + \left[H_{-1}^* + H_{1,f}^* \tilde{\beta}_{t|t,2} \right] z_{t-1} + H_{1,f}^* \tilde{\beta}_{t|t,0} = D\eta_t. \tag{17.25}$$

It now follows that the ALM is given by

$$z_t = \tilde{\mu}_t + \tilde{F}_{t,1} z_{t-1} + \tilde{B}_{t,0} \eta_t, \tag{17.26}$$

where

$$\tilde{\mu}_t = -\left[H_0^* + H_{1,f}^* \tilde{\beta}_{t|t,1} \right]^{-1} H_{1,f}^* \tilde{\beta}_{t|t,0},$$

$$\tilde{F}_{t,1} = -\left[H_0^* + H_{1,f}^* \tilde{\beta}_{t|t,1} \right]^{-1} \left[H_{-1}^* + H_{1,f}^* \tilde{\beta}_{t|t,2} \right],$$

$$\tilde{B}_{t,0} = \left[H_0^* + H_{1,f}^* \tilde{\beta}_{t|t,1} \right]^{-1} D.$$

The condition for a unique solution at t is therefore that the matrix premultiplied by z_t in equation (17.25) is invertible. The ALM in (17.26) with $\xi_t = z_t$ is now the state equation for the DSGE model with adaptive learning and it can be used together with the measurement equation in (5.1) to form the state-space representation. It should also be noted that premultiplying z_t in (17.26) with S' gives the ALM for the forward looking variables z_t^f and the corresponding equation differs from the PLM in equation (17.7). In other words, expectations are *not* model consistent.[138]

For the general case of a finite m and deterministic variables x_t, substitution of the PLM into the structural form and rearranging terms gives us

$$\left[H_0^* + H_{1,f}^* \tilde{\beta}_{t|t,1} \right] z_t + \left[H_{-1}^* + H_{1,f}^* \tilde{\beta}_{t|t,2} \right] z_{t-1}$$
$$+ H_{1,f}^* \sum_{j=2}^{m-1} \tilde{\beta}_{t|t,j+1} z_{t-j} + H_{1,f}^* \tilde{\beta}_{t|t,0} x_t = D\eta_t. \tag{17.27}$$

---

[138] It is possible that expectations are model consistent when $\beta_t = \beta$ for all t and when $q_{t-1,j}$ has been selected to exactly match the variables that appear in the RE solution for all forward looking variables.

It follows that a unique solution at t exists under the same conditions as when $m = 2$ and it is given by

$$z_t = \tilde{\mu}_t x_t + \sum_{j=1}^{m^*} \tilde{F}_{t,j} z_{t-j} + \tilde{B}_{t,0} \eta_t, \tag{17.28}$$

where $m^* = \max\{m - 1, 1\}$, while $\tilde{\mu}_t$ and $\tilde{B}_{t,0}$ are given by the expressions below equation (17.26), while

$$\tilde{F}_{t,j} = \begin{cases} -\left[H_0^* + H_{1,f}^* \tilde{\beta}_{t|t,1} \right]^{-1} \left[H_{-1}^* + H_{1,f}^* \tilde{\beta}_{t|t,2} \right], & \text{if } j = 1, \\ -\left[H_0^* + H_{1,f}^* \tilde{\beta}_{t|t,1} \right]^{-1} H_{1,f}^* \tilde{\beta}_{t|t,j+1}, & \text{if } j = 2, \ldots, m - 1. \end{cases}$$

Notice that $m = 1$ implies that $\tilde{\beta}_{t|t,2} = 0$ for all t since z_{t-1} no longer appears in Z_t, the vector of variables in the PLM.

Finally, it should be noted that when $m \geq 3$, the state equation for the Kalman filter of the DSGE model is given by

$$\begin{bmatrix} z_t \\ z_{t-1} \\ z_{t-2} \\ \vdots \\ z_{t-m+2} \end{bmatrix} = \begin{bmatrix} \tilde{\mu}_t \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} x_t + \begin{bmatrix} \tilde{F}_{t,1} & \tilde{F}_{t,2} & \cdots & \tilde{F}_{t,m-2} & \tilde{F}_{t,m-1} \\ I_p & 0 & \cdots & 0 & 0 \\ 0 & I_p & & 0 & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & & I_p & 0 \end{bmatrix} \begin{bmatrix} z_{t-1} \\ z_{t-2} \\ z_{t-3} \\ \vdots \\ z_{t-m+1} \end{bmatrix} + \begin{bmatrix} \tilde{B}_{t,0} \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \eta_t, \tag{17.29}$$

or more compactly

$$\xi_t = \tilde{M}_t x_t + \tilde{F}_t \xi_{t-1} + \tilde{B}_t \eta_t. \tag{17.30}$$

The vector ξ_t is r-dimensional with $r = pm^*$. In case $m = 1$, a unique solution is given by equation (17.26) with $\tilde{\beta}_{t|t,2} = 0$ in the expression for $\tilde{F}_{t,1}$. The general expression in (17.30) is employed for all m cases below. It may also be kept in mind that YADA restricts m not to be greater than 2. This restriction can be overcome by introducing new state variables in the model, thereby keeping the overall length of ξ_t at bay.[139]

17.4. A Kalman Filter for the DSGE Model with Adaptive Learning

Kalman filtering has been thoroughly discussed in Section 5. The state equation in (17.30) differs slightly from the setup used there. Specifically, the state equation now includes deterministic variables and the parameter matrices are time-varying. We will therefore provide the filtering, updating and smoothing equations below under the more general setup we have found for the adaptive learning model. Although the measurement equation has not changed, it is still useful to allow the H matrix in (5.1) to vary with time and we simply call it H_t. The remaining parameters of the measurement equation, A and R, can be kept constant.

Before we give the filtering and updating equations, it is useful to take a step back and consider which information is available at time t and how that can be used. First, the Kalman filter for the belief coefficients in Section 17.1.2 is based on having observed z_t^f at t or having an update estimate $z_{t|t}^f$ based on y_t. With this information, the filter and update equations can be executed, yielding $\beta_{t|t}$ which can then be used to form the solution matrices $\tilde{M}_t$, $\tilde{F}_t$ and $\tilde{B}_t$, representing the state equation for the DSGE model under adaptive learning in (17.30). But to run the Kalman filter using this state equation requires that the values for these matrices are known at $t - 1$, which is not the case.

---

[139] A simple example can be constructed as follows. In the An and Schorfheide model in Section 2.1, consumption is a forward looking variable. Suppose we wish to introduce a PLM for this variable with 3 own lags and a constant:

$$\hat{c}_t = \beta_{c,0,t} + \beta_{c,1,t} \hat{c}_{t-1} + \beta_{c,2,t} \hat{c}_{t-2} + \beta_{c,3,t} \hat{c}_{t-3} + u_{c,t}.$$

This case can be covered by a maximum value of $m = 2$ by defining $\bar{c}_t = \hat{c}_{t-1}$ and replacing $\hat{c}_{t-3}$ in this PLM with $\bar{c}_{t-2}$.

There are now two options: Since the only unknown component of the state equation matrices is $\beta_{t|t}$, we may either replace it with $\beta_{t|t-1}$ or with $\beta_{t-1|t-1}$. Formally, the first option seems like a natural choice since the belief coefficients in period t should be used when forming the expectations for period $t + 1$ in the structural form. However, $\beta_{t-1|t-1}$ will be used below since the standard treatment in the learning literature is to assume that beliefs formed today are taken as given and that agents do not take into account that they will update their beliefs in the future. This is indeed the approach taken by Slobodyan and Wouters (2012).

Keeping the Kalman filter from Section 5 in mind, we now find that the one-step-ahead forecast of y_t is given by

$$y_{t|t-1} = A'x_t + H_t'\xi_{t|t-1}, \tag{17.31}$$

while the covariance matrix of the observed variable forecast is

$$\Sigma_{y,t|t-1} = H_t'P_{t|t-1}H_t + R. \tag{17.32}$$

The update equation for the state variables is

$$\xi_{t|t} = \xi_{t|t-1} + P_{t|t-1}H_t\Sigma_{y,t|t-1}^{-1}\left(y_t - y_{t|t-1}\right), \tag{17.33}$$

while the update covariance matrix is

$$P_{t|t} = P_{t|t-1} - P_{t|t-1}H_t\Sigma_{y,t|t-1}^{-1}H_t'P_{t|t-1}. \tag{17.34}$$

Finally, the state variables are projected forward

$$\xi_{t+1|t} = \tilde{M}_{t-1}x_{t+1} + \tilde{F}_{t-1}\xi_{t|t}. \tag{17.35}$$

The covariance matrix of the state variable forecast is here given by

$$P_{t+1|t} = \tilde{F}_{t-1}P_{t|t}\tilde{F}_{t-1}' + \tilde{B}_{t-1}\tilde{B}_{t-1}'. \tag{17.36}$$

Notice that the solution matrices from $t - 1$ are used at t when projecting the state variables forward.

17.4.1. *Initial Values for the Filter*

To initialize this filter we first note that the matrices $\tilde{M}_{t-1}$, $\tilde{F}_{t-1}$ and $\tilde{B}_{t-1}$ are time-varing only due to $\beta_{t-1|t-1}$. We may therefore consider replacing the Kalman filter estimate from the belief equation with the population moment based on the RE version of the DSGE model, i.e., β in equation (17.15). It follows that

$$\tilde{\mu} = -\left[H_0^* + H_{1,f}^*\tilde{\beta}_1\right]^{-1}\tilde{\beta}_0,$$

$$\tilde{B}^* = \left[H_0^* + H_{1,f}^*\tilde{\beta}_1\right]^{-1}D,$$

$$\tilde{F}_j = \begin{cases} -\left[H_0^* + H_{1,f}^*\tilde{\beta}_1\right]^{-1}\left[H_{-1}^* + H_{1,f}^*\tilde{\beta}_2\right], & \text{if } j = 1, \\ -\left[H_0^* + H_{1,f}^*\tilde{\beta}_1\right]^{-1}H_{1,f}^*\tilde{\beta}_{j+1}, & \text{if } j = 2, \ldots, m - 1. \end{cases}$$

where $\tilde{\beta}_i$ is obtained from a decomposition of $\tilde{\beta}$ analogous to the one of $\tilde{\beta}_t$ in (17.24). From these matrices we can initialize the vector of state variables $\xi_{1|0} = \mu_\xi$ with

$$\mu_\xi = \left(I_r - \tilde{F}\right)^{-1}\tilde{M}\bar{x},$$

where $\bar{x}$ is given by a unit element for the constant and, for instance, zeros for non-constant deterministic variables. Similarly, we may let $P_{1|0}$ be initialized through the steady-state covariance matrix conditional on the parameters satisfying the Lyapunov equation:

$$\Sigma_\xi = \tilde{F}\Sigma_\xi\tilde{F}' + \tilde{B}\tilde{B}',$$

where $\tilde{B}$ is constructed from $\tilde{B}^*$ as in equation (17.29). The Lyapunov equation can, technically, be solved by vectorization, but in practise it is usually better to rely on the doubling algorithm discussed in Section 5.3.

17.5. *A Joint Kalman Filter Algorithm for Computing the State Variables and Belief Coefficients*

The two Kalman filters for β_t and ξ_t can be combined with the solution method in a straight-forward manner. As initial values the algorithm requires β, Σ_u, Σ_ϵ, $\beta_{1|0}$, $R_{1|0}$, $\tilde{M}$, $\tilde{F}$, $\tilde{B}$, μ_ξ, Σ_ξ, $\xi_{1|0}$ and $P_{1|0}$, whose determination has been discussed above. We also let $\tilde{M}_0 = \tilde{M}$, $\tilde{F}_0 = \tilde{F}$ and $\tilde{B}_0 = \tilde{B}$. The algorithm runs forward over iterations $t = 1, \ldots, T$:

(1) Compute $y_{t|t-1}$ and $\Sigma_{y,t|t-1}$ using equations (17.31)–(17.32). The log-likelihood function for period t can be computed as in Section 5.4, but using the expression for the one-step-ahead forecast of y_t and the covariance matrix in equations (17.31) and (17.32);

(2) Compute $\xi_{t|t}$ and $P_{t|t}$ from (17.33) and (17.34);

(3) Compute $\xi_{t+1|t}$, $P_{t+1|t}$ from (17.35) and (17.36) if $t < T$;

(4) Compute the belief coefficients and the covariance matrices using equations (17.9)–(17.12), with $z_{t|t}^f = S'\xi_{t|t}$ from iteration t and $z_{t-j|t-1}$, $j = 1, \ldots, m$, from iteration $t-1$; if $t \leq m$, then $z_{t-j|t-1} = 0$ for $j \geq t$. The new solution matrices $\tilde{M}_t$, $\tilde{F}_t$ and $\tilde{B}_t$ are thereafter determined using $\beta_{t|t}$.

Notice that the algorithm requires smoothing whenever $m \geq 2$ since $Z_{t-1|t-1}$ is required by the Kalman filter part of the belief equation. Smooth estimates are also more generally of interest when estimating the structural shocks and the state variables using future information, such as for the full sample.

Instead of using the smooth estimate of, say, $z_{t-2|t-1}$ in $Z_{t-1|t-1}$ one may use the update estimate $z_{t-2|t-2}$. From email discussions with Raf Wouters, the latter possibility is used in Slobodyan and Wouters (2012) and for this reason YADA has an option which makes such an alternative estimate of z_{t-2} available to the user.

Finally, the Kalman filter of the belief coefficients when $t \leq m$ is based on setting $z_{0|0} = z_{-1|0} = z_{0|1} = 0$, the steady state value, while deterministic variables are included with their time t values.

17.6. *Smooth Estimates of the State Variables*

The Kalman smoother for the state variables described in Section 5.5 can likewise be employed, with a few smaller adjustments. Specifically,

$$\xi_{t|T} = \xi_{t|t-1} + P_{t|t-1}r_{t|T}, \tag{17.37}$$

where

$$r_{t|T} = H_t\Sigma_{y,t|t-1}^{-1}\left(y_t - y_{t|t-1}\right) + \left(\tilde{F}_{t-1} - \tilde{K}_{t-1}H_t'\right)'r_{t+1|T}, \tag{17.38}$$

with the initial condition $r_{T+1|T} = 0$, and where the Kalman gain matrix is

$$\tilde{K}_{t-1} = \tilde{F}_{t-1}P_{t|t-1}H_t\Sigma_{y,t|t-1}^{-1}. \tag{17.39}$$

Furthermore, the smoothed state covariance matrix is again

$$P_{t|T} = P_{t|t-1} - P_{t|t-1}N_{t|T}P_{t|t-1}, \tag{17.40}$$

where

$$N_{t|T} = H_t\Sigma_{y,t|t-1}^{-1}H_t' + \left(\tilde{F}_{t-1} - \tilde{K}_{t-1}H_t'\right)'N_{t+1|T}\left(\tilde{F}_{t-1} - \tilde{K}_{t-1}H_t'\right), \tag{17.41}$$

and with the initial condition $N_{T+1|T} = 0$. Mote that we have chosen to compute the smoother based on the solution matrices which make use of $\beta_{t-1|t-1}$. In principle, one may also consider a more sophisticated smoothing algorithm which switches between smoothing the state variables and the belief coefficients with a recomputation of the solution matrices from the smoothed $\beta_{t|T}$, but we leave this issue open for future research.

Returning to Step (4) of the algorithm in Section 17.5, equations (17.37)–(17.38) apply to any $t \leq T$ so that

$$\xi_{t-j|t-1} = \xi_{t-j|t-j-1} + P_{t-j|t-j-1}r_{t-j|t-1}, \quad j = 1, \ldots, m,$$

where

$$r_{t-j|t-1} = H_{t-j}\Sigma_{y,t-j|t-j-1}^{-1}\left(y_{t-j} - y_{t-j|t-j-1}\right) + \left(\tilde{F}_{t-j-1} - \tilde{K}_{t-j-1}H_{t-j}\right)'r_{t-j+1|t-1},$$

where $r_{t|t-1} = 0$.

17.6.1. *Update and Smooth Estimates of the Measurement Errors and the Structural Shocks*

The update and the smooth estimates of the measurement errors and the structural shocks can be derived using equations (17.33)–(17.39). To begin with the update estimate of the measurement error is

$$w_{t|t} = R\Sigma_{y,t|t-1}^{-1}(y_t - y_{t|t-1}),$$ (17.42)

with covariance matrix

$$E[w_{t|t}w_{t|t}'] = R\Sigma_{y,t|t-1}^{-1}R.$$ (17.43)

Similarly, the update estimate of the structural shocks is

$$\eta_{t|t} = \tilde{B}_{t-2}'H_t\Sigma_{y,t|t-1}^{-1}(y_t - y_{t|t-1}).$$ (17.44)

The timing of the solution matrix $\tilde{B}_{t-2}$ stems from the observation that $\xi_{t|t-1}$ is computed from the solutions obtained at $t-2$ in the algorithm and employed for updating and filtering at $t-1$. The covariance matrix is therefore given by

$$E[\eta_{t|t}\eta_{t|t}'] = \tilde{B}_{t-2}'H_t\Sigma_{y,t|t-1}^{-1}H_t'\tilde{B}_{t-2}.$$ (17.45)

Next, the smooth estimate of the measurement error is

$$w_{t|T} = y_t - A'x_t - H_t'\xi_{t|T} = R\Sigma_{y,t|t-1}^{-1}(y_t - y_{t|t-1} - H_t'P_{t|t-1}\tilde{F}_{t-1}'r_{t+1|T}),$$ (17.46)

with covariance matrix

$$E[w_{t|T}w_{t|T}'] = R\left(\Sigma_{y,t|t-1}^{-1} + \tilde{K}_{t-1}'N_{t+1|T}\tilde{K}_{t-1}\right)R.$$ (17.47)

The smooth estimate of the structural shocks is here

$$\eta_{t|T} = \tilde{B}_{t-2}'r_{t|T},$$ (17.48)

while its covariance matrix is

$$E[\eta_{t|T}\eta_{t|T}'] = \tilde{B}_{t-2}'N_{t|T}\tilde{B}_{t-2}.$$ (17.49)

17.6.2. *Historical Observed and State Variable Decompositions*

For the smooth estimates of the state variables and the structural shocks, it can be verified by using equations (17.33)–(17.39) that

$$\xi_{t|T} = \tilde{M}_{t-2}x_t + \tilde{F}_{t-2}\xi_{t-1|T} + \tilde{B}_{t-2}\eta_{t|T}.$$ (17.50)

Again, the timing of the solution matrices stems from the observation that $\xi_{t|t-1}$ is computed from the solution matrices obtained at $t-2$ in the algorithm and employed for updating and filtering at $t-1$. By substituting for $\xi_{t-1|T}$, $\xi_{t-2|T}$ until $\xi_{1|T}$, we obtain the following decomposition for the smooth estimate of the state variables at t:

$$\xi_{t|T} = \sum_{j=0}^{t-1}\left(\prod_{i=1}^{j}\tilde{F}_{t-1-i}\right)\tilde{M}_{t-2-j}x_{t-j} + \left(\prod_{i=1}^{t}\tilde{F}_{t-1-i}\right)\xi_{0|T} + \sum_{j=0}^{t-1}\left(\prod_{i=1}^{j}\tilde{F}_{t-1-i}\right)\tilde{B}_{t-2-j}\eta_{t-j|T}.$$ (17.51)

Notice that the values of the solution matrices at $t = 0, -1$ are given by the initial values $\tilde{M}$, $\tilde{F}$ and $\tilde{B}$. In terms of the observed variables we therefore find that

$$\begin{aligned}
y_t = {} & A'x_t + H_t'\sum_{j=0}^{t-1}\left(\prod_{i=1}^{j}\tilde{F}_{t-1-i}\right)\tilde{M}_{t-2-j}x_{t-j} + H_t'\left(\prod_{i=1}^{t}\tilde{F}_{t-1-i}\right)\xi_{0|T} \\
& + H_t'\sum_{j=0}^{t-1}\left(\prod_{i=1}^{j}\tilde{F}_{t-1-i}\right)\tilde{B}_{t-2-j}\eta_{t-j|T} + w_{t|T}, \quad t = 1, \ldots, T.
\end{aligned}$$ (17.52)

This means that the observed data can be decomposed into four terms. The first contains the influence of deterministic variables, the second the initial state, the third the impact of structural shocks, and last the measurement error.

To compute the decompositions for the state and the observed variables, we first note that $\xi_{0|T}$ cannot be estimated directly. However, the term $\tilde{F}_{-1}\xi_{0|T}$ can be obtained as a residual from $\xi_{1|T} - \tilde{M}_{-1}x_1 - \tilde{B}_{-1}\eta_{1|T}$. Second, the most straightforward approach to computing the decomposition is a recursive one using equation (17.50) for $t = 1, \ldots, T$. Once the decomposition for the state variables has been performed, the decomposition for the observed variables can be obtained through the former and the measurement equations.

17.6.3. *Laws of Motion*

The adaptive learning based DSGE model has two types of laws of motion. In Section 17.1.2, the perceived law of motion is presented in equation (17.7), while the actual law of motion is given in Section 17.3. To compare these two "laws" we focus on the forward looking variables and, specifically, the update estimates $z^f_{t|t}$. This can be regarded as the target variable, where the PLM equation provides us with estimate $q'_{t-1|t-1}\beta_{t-1|t-1}$, while the ALM suggests the one-step-ahead forecast of the forward looking variables, $z^f_{t|t-1} = S'\xi_{t|t-1}$.

17.7. *Impulse Response Functions*

Slobodyan and Wouters (2012) compute impulse response functions based on fixed belief coefficients for each date t. Such functions therefore disregards the updating mechanism of beliefs that can otherwise be influenced by the shock. If we make use of the solution in (17.30), it means for time t that the solution matrices dated t should be used, i.e., based on $\beta_{t|t}$.

The solution in (17.30) is based on the assumption that ξ_t is observed at t by the agents of the economy. From the perspective of the econometrician who uses the Kalman filter in Section 17.5, the state variables are represented by the update and, for $m > 1$, smooth estimates, with the consequence that at t they are based on the solution matrices from $t - 2$. When computing the impulse responses, we need to decide which of these two cases should be applied. In terms of the state variables we therefore have that for $\tau = t, t - 2$:

$$\text{resp}(\xi_{t+h}|\eta_t = e_j, \tau) = \tilde{F}^h_\tau \tilde{B}_\tau e_j, \quad h \geq 0. \tag{17.53}$$

As a consequence, the impulse responses vary over time and the initial impact of the shocks depends on the structural form matrices H^*_0, $H^*_{1,f}$, D as well as the belief coefficients $\tilde{\beta}_{\tau|\tau,1}$. The responses of the observed variables can now be computed via the measurement equation, which provides us with

$$\text{resp}(y_{t+h}|\eta_t = e_j, \tau) = H'_t \tilde{F}^h_\tau \tilde{B}_\tau e_j, \quad h \geq 0. \tag{17.54}$$

The impulse responses in YADA are currently computed for $\tau = t$, as in Slobodyan and Wouters (2012), but this may be changed to also allow for $\tau = t - 2$.

17.8. *Fisher's Information Matrix*

When the information matrix is estimated in the time domain for the DSGE model under adaptive learning, the same approach as in the rational expectations case can be used; see Section 11.12. The changes to the Kalman filter in Section 17.5 need to be considered and since YADA uses numerical derivatives of $\tilde{y}_t$ and $\Sigma_{y,t|t-1}$ with respect to θ, this is straightforward.

Concerning the frequency domain expression of the information matrix, YADA uses the steady-state solution of the adaptive learning model. Compared with Section 13.8, the only changes to the estimation approach concern F and B_0 from the solution under rational expectations. They are replaced with the steady-state matrices $\tilde{F}$ and $\tilde{B}$ from Section 17.4.1.

17.9. *Forecasting with Adaptive Learning*

Let T be the last historical time period when forecasting state and observed variables with the DSGE model. The Kalman filter in Section 17.5 includes an update estimate of $\xi_{T|T}$ being given by equation (17.33), with the consequence that the solution matrices are $(\tilde{M}_{T-1}, \tilde{F}_{T-1}, \tilde{B}_{T-1})$. A

logical consequence of the filter is that the one-step-ahead forecast $\xi_{T+1|T}$ is also computed from this solution.

However, at T we also have access to an update solution based on $\beta_{T|T}$ which may be used when forecasting the state and the observed variables. It is tempting to argue in favor of this case as it would make make use of the additional information available at T through the update estimate of the forward looking variables at T. At the same time, one may argue that the Kalman filter routine should then also be constructed such that $\xi_{t+1|t}$ is computed with the solution matrices based on $\beta_{t|t}$ rather than on $\beta_{t-1|t-1}$. In that case, the calculation routines for smoothing would also need to take into account that different information sets are used for the update and the forecast. This is a challenging topic and it is left for future research. Hence, when forecasting the adaptive learning version of DSGE models in YADA, the solution matrices $(\tilde{M}_{T-1}, \tilde{F}_{T-1}, \tilde{B}_{T-1})$ are always used for the $T+1|T$ forecast.

The next issue to consider is the development of the solution matrices over the forecast horizon. The forecasts presented in Slobodyan and Wouters (2012) are, like the impulse responses, based on fixed belief coefficients. However, they note that they have also studied the forecasts when the beliefs are updated based on the Kalman filter equations (17.9)–(17.12). YADA supports both these cases with fixed belief coefficients being the default.

17.9.1. *Unconditional Forecasts with Fixed Beliefs*

If we assume that the belief coefficients are fixed at the $T-1$ values over the whole forecast horizon it follows that the point forecasts of the state variables for fixed parameters θ are:

$$
\begin{aligned}
\xi_{T+i|T} &= \tilde{M}_{T-1}x_{T+i} + \tilde{F}_{T-1}\xi_{T+i-1|T} \\
&= \sum_{j=0}^{i-1} \tilde{F}_{T-1}^{j}\tilde{M}_{T-1}x_{T+i-j} + \tilde{F}_{T-1}^{i}\xi_{T|T},
\end{aligned}
\tag{17.55}
$$

with $i = 1, \ldots, h$. The point forecast of the observed variables for fixed parameters is therefore

$$
y_{T+i|T} = A'x_{T+i} + H_{T+i}'\xi_{T+i|T}.
\tag{17.56}
$$

Furthermore, the forecast error covariance matrix of the state variables is given by

$$
\begin{aligned}
P_{T+i|T} &= \tilde{F}_{T-1}P_{T+i-1|T}\tilde{F}_{T-1}' + \tilde{B}_{T-1}\tilde{B}_{T-1}' \\
&= \sum_{j=0}^{i-1} \tilde{F}_{T-1}^{j}\tilde{B}_{T-1}\tilde{B}_{T-1}'\big(\tilde{F}_{T-1}'\big)^{j} + \tilde{F}_{T-1}^{i}P_{T|T}\big(\tilde{F}_{T-1}'\big)^{i},
\end{aligned}
\tag{17.57}
$$

while the forecast error covariance matrix of the observed variables is

$$
\Sigma_{y,T+i|T} = H_{T+i}'P_{T+i|T}H_{T+i} + R.
\tag{17.58}
$$

If the data set is subject to missing observations for some of the observed variables towards the end of the sameple, a so called *ragged edge* and as discussed by, for instance, Wallis (1986), the need for nowcasts and backcasts appears. In the current setting this means that the nowcast and backcast of the observed variables uses the smooth estimates of the state variables. Without loss of generality, assume that there are no measurement errors or, equivalently, that these errors are placed in the state equations. It follows that the nowcast is

$$
y_{T|T} = A'x_T + H_T'\xi_{T|T},
\tag{17.59}
$$

with covariance matrix

$$
\Sigma_{y,T|T} = H_T'P_{T|T}H_T.
\tag{17.60}
$$

Similar, the backcast for period $T-1$ is

$$
y_{T-1|T} = A'x_{T-1} + H_{T-1}'\xi_{T-1|T},
\tag{17.61}
$$

with covariance matrix

$$
\Sigma_{y,T-1|T} = H_{T-1}'P_{T-1|T}H_{T-1}.
\tag{17.62}
$$

Backcasts that go back to $T-t$, for $t = 2, 3, \ldots$ and so on, can be constructed analogously.

17.9.2. *Unconditional Forecasts with Updated Beliefs*

If we instead assume that the forecasts are made subject to beliefs being updated over the forecast horizon through the Kalman filter equations in (17.9)–(17.12), we first note that $\beta_{T-1|T-1}$ is still used when forecasting variables in $T + 1$ given T. To forecast $T + 2|T$, the belief coefficients $\beta_{T|T}$ are used, and more generally to forecast $T + i|T$ we use $\beta_{T+i-2|T+i-2}$ for $i \geq 1$. Since there is *no* information about $T + 1$, $T + 2$, etc., the state equation for the belief coefficients in (17.8) provides the following forecasts of the belief coefficients

$$\beta_{T+i|T} = \beta + \Gamma\big(\beta_{T+i-1|T} - \beta\big), \quad i = 1, 2, \ldots, h - 2. \tag{17.63}$$

The solution matrices $(\tilde{M}_{T+i}, \tilde{F}_{T+i}, \tilde{B}_{T+i})$ can thereafter be determined as in Section 17.3 using $\beta_{T+i|T}$.

The forecasts of the state variables for fixed parameters can now be computed recursively as

$$\xi_{T+i|T} = \tilde{M}_{T+i-2}x_{T+i} + \tilde{F}_{T+i-2}\xi_{T+i-1|T}, \quad i = 1, \ldots, h, \tag{17.64}$$

while the forecasts of the observed variables are provided by equation (17.56). Finally, the covariance matrix of the state variable forecasts for fixed parameters is recursively determined from

$$P_{T+i|T} = \tilde{F}_{T+i-2}P_{T+i-1|T}\tilde{F}'_{T+i-2} + \tilde{B}_{T+i-2}\tilde{B}'_{T+i-2}, \quad i = 1, \ldots, h, \tag{17.65}$$

while the forecast error covariance matrix of the observed variables is given by equation (17.58).

The nowcasts and backcasts are produced exactly as in Section 17.9.1 as there is no need to update the beliefs.

17.9.3. *Estimating the Predictive Likelihood and the Predictive Moments*

Estimation of the predictive likelihood is discussed above in Section 12.6 and also in Warne et al. (2017). The main inputs for estimating this likelihood with Monte Carlo integration is the predictive likelihood conditional on the parameters, i.e., the conditional likelihood. This function is given by a normal density with mean and covariance matrix given by the moments conditional on the parameters and evaluated at the actual values for the variables of interest.

The variables of interest for forecasting are given by the n^*-dimensional vector y_t^*. This vector is obtained from the observed variables y_t using

$$y_t^* = K'y_t,$$

where K is a known $n \times n^*$ selection matrix, typically containing n^* unique columns of the $n \times n$ identity matrix. The log of the conditional likelihood is given by equation (12.68), where $y_{T+h|T}^* = Ky_{T+h|h}$ and $\Sigma_{y^*, t+h|t} = K'\Sigma_{y, t+h|t}K$. The predictive likelihood can now be estimated with Monte Carlo integration by averaging the conditional likelihood over a suitable sample of MCMC based posterior draws of the parameters θ. In case the posterior draws are obtained from an SMC or IS sampler, the individual conditional likelihoods should be weighted using the relevant SMC or IS weights. Similarly, the predictive moments can be estimated by averaging over the (weighted) moments based on fixed parameters.

For the joint predictive likelihood one may also proceed as in Section 12.6 and build up the point forecasts and their covariances via a Kalman filter and as described below equation (12.69). The matrices F and B_0 need to be replaced with their adaptive learning variant. An interesting issue for concerns the treatment of this filter under fixed version updated beliefs. In the former case the belief coefficients are unaffected by the actual values, while in the latter case they also need to take these values into account. In other words, for the updated beliefs we need to use the kalman filter algorithm in Section ssec:loglikelearning, while under fixed beliefs step (4) of the algorithm is skipped. Furthermore, only the variables of interest are used in the filter.

17.9.4. *Conditional Forecasts under Direct Control of the Shocks*

17.9.4.1. *Fixed Beliefs*

As in Section 12.2, we start from the relationship in equation (12.6) which links the q_m hard conditioning assumtions z^*_{T+i} to current and past observables y_{t+i-j} and initial conditions u_T for a total of $g^* \leq h$ periods.[140] Under the assumption of *direct control of the shocks* to enforce the conditioning assumptions, as in Section 12.2.1, the q shocks are grouped as in equation (12.7) with q_m shocks that are used to ensure that the q_m conditioning assumptions hold for each time period over the conditioning sample, and $q - q_m$ shocks that are free and set to zero for the population mean and drawn from a normal distribution when computing paths. With fixed beliefs over the forecast sample, the state variables are given by

$$\xi_{T+i} = \tilde{M}_{T-1}x_{T+i} + \tilde{F}_{T-1}\xi_{T+i-1} + \tilde{B}_{T-1}\bar{M}_\perp \eta^{(q-q_m)}_{T+i} + \tilde{B}_{T-1}\bar{M}\eta^{(q_m)}_{T+i}, \quad i = 1, \dots, h.$$

Recall that $\eta^{(q-q_m)}_t = M'_\perp \eta_t$ are the $q - q_m$ shocks which are assumed to be $N(0, M'_\perp M_\perp)$ for all t, while $\eta^{(q_m)}_t = M'\eta_t$ are $N(0, M'M)$ except when $t = T + 1, \dots, T + g^*$.

Following the logic in Section 12.2.1 we now find that for

$$
\begin{aligned}
\eta^{(q_m)}_{T+i} = \left(K'_1 H'_{T+i}\tilde{B}_{T-1}\bar{M}\right)^{-1}\Big[& z^*_{T+i} - K'_1\left(A' + H'_{T+i}\tilde{M}_{T-1}\right)x_{T+i} - K'_1 w_{T+i} \\
& - K'_1 H'_{T+i}\tilde{F}_{T-1}\xi^{(q_m)}_{T+i-1} - K'_1 H'_{T+i}\tilde{B}_{T-1}\bar{M}_\perp\eta^{(q-q_m)}_{T+i} \\
& - \sum_{j=1}^{i-1} K'_{2j}y^{(q_m)}_{T+i-j} - u_T\Big], \quad i = 1, \dots, g,
\end{aligned}
\tag{17.66}
$$

while the states and observables evolve according to

$$\xi^{(q_m)}_{T+i} = \tilde{M}_{T-1}x_{T+i} + \tilde{F}_{T-1}\xi^{(q_m)}_{T+i-1} + \tilde{B}_{T-1}\bar{M}_\perp\eta^{(q-q_m)}_{T+i} + \tilde{B}_{T-1}\bar{M}\eta^{(q_m)}_{T+i}, \quad i = 1, \dots, g^*, \tag{17.67}$$

and

$$y^{(q_m)}_{T+i} = A'x_{T+i} + H'_{T+i}\xi^{(q_m)}_{T+i} + w_{T+i}, \quad i = 1, \dots, g^*, \tag{17.68}$$

while $\xi^{(q_m)}_T = \xi_T$.

The population mean of the controlled shocks, the states and the observables for fixed parameters can be directly obtained from (17.66)–(17.68) by letting, for example, $\xi_T = \xi^{(z)}_{T|T+g^*}$, $\eta^{(q-q_m)}_{T+i} = 0$ and $w_{T+i} = 0$ for all $i = 1, \dots, h$, while $\eta^{(q_m)}_{T+i} = 0$ for $i = g^* + 1, \dots, h$. However, to compare with the approaches in Sections 17.9.5–17.9.6 the population mean will be derived in a more complex manner.

For the conditioning sample and under fixed beliefs it holds for the state variables that

$$\xi_{T+i} = \bar{x}_{T+i} + \tilde{F}^i_{T-1}\xi_T + \sum_{j=0}^{i-1} \tilde{F}^j_{T-1}\tilde{B}_{T-1}\eta_{T+i-j}, \quad i = 1, \dots, g^*, \tag{17.69}$$

where

$$\bar{x}_{T+i} = \tilde{M}_{T-1}x_{T+i} + \tilde{F}_{T-1}\bar{x}_{T+i-1},$$

and $\bar{x}_T = 0$.

---

[140] Note the slight changes in notation with asterisks on z and g to avoid confusing the conditioning assumptions with the state variables and the number of conditioning periods with the dimension of the belief coefficients. The matrix M used in Section 12.2, however, will remain in use for the conditional forecasts although it also denotes the transitions for the anticiated shocks in Section 17.10 on the zero lower bound.

Stacking the g^* sets of equations for the state variables, we find that

$$
\begin{bmatrix} \xi_{T+g^*} \\ \xi_{T+g^*-1} \\ \vdots \\ \xi_{T+1} \end{bmatrix} = \begin{bmatrix} \bar{x}_{T+g^*} \\ \bar{x}_{T+g^*-1} \\ \vdots \\ \bar{x}_{T+1} \end{bmatrix} + \begin{bmatrix} \tilde{F}_{T-1}^{g^*} \\ \tilde{F}_{T-1}^{g^*-1} \\ \vdots \\ \tilde{F}_{T-1} \end{bmatrix} \xi_T + \begin{bmatrix} \tilde{B}_{T-1} & \tilde{F}_{T-1}\tilde{B}_{T-1} & \cdots & \tilde{F}_{T-1}^{g^*-1}\tilde{B}_{T-1} \\ 0 & \tilde{B}_{T-1} & & \tilde{F}_{T-1}^{g^*-2}\tilde{B}_{T-1} \\ \vdots & & \ddots & \vdots \\ 0 & 0 & & \tilde{B}_{T-1} \end{bmatrix} \begin{bmatrix} \eta_{T+g^*} \\ \eta_{T+g^*-1} \\ \vdots \\ \eta_{T+1} \end{bmatrix}.
$$

In terms of the observed variables, we therefore obtain

$$
\begin{bmatrix} y_{T+g^*} \\ y_{T+g^*-1} \\ \vdots \\ y_{T+1} \end{bmatrix} = \begin{bmatrix} A'x_{T+g^*} + H'_{T+g^*}\bar{x}_{T+g^*} \\ A'x_{T+g^*-1} + H'_{T+g^*-1}\bar{x}_{T+g^*-1} \\ \vdots \\ A'x_{T+1} + H'_{T+1}\bar{x}_{T+1} \end{bmatrix} + \begin{bmatrix} H'_{T+g^*}\tilde{F}_{T-1}^{g^*} \\ H'_{T+g^*-1}\tilde{F}_{T-1}^{g^*-1} \\ \vdots \\ H'_{T+1}\tilde{F}_{T-1} \end{bmatrix} \xi_T + \begin{bmatrix} w_{T+g^*} \\ w_{T+g^*-1} \\ \vdots \\ w_{T+1} \end{bmatrix}
$$

$$
+ \begin{bmatrix} H'_{T+g^*}\tilde{B}_{T-1} & H'_{T+g^*}\tilde{F}_{T-1}\tilde{B}_{T-1} & \cdots & H'_{T+g^*}\tilde{F}_{T-1}^{g^*-1}\tilde{B}_{T-1} \\ 0 & H'_{T+g^*-1}\tilde{B}_{T-1} & & H'_{T+g^*-1}\tilde{F}_{T-1}^{g^*-2}\tilde{B}_{T-1} \\ \vdots & & \ddots & \vdots \\ 0 & 0 & & H'_{T+1}\tilde{B}_{T-1} \end{bmatrix} \begin{bmatrix} \eta_{T+g^*} \\ \eta_{T+g^*-1} \\ \vdots \\ \eta_{T+1} \end{bmatrix},
$$

or

$$
Y_{T+g^*} = \bar{X}_{T+g^*} + \tilde{G}_{T-1}\xi_T + W_{T+g^*} + \tilde{D}_{T-1}N_{T+g^*}. \tag{17.70}
$$

Equation (12.15) is not much affected as it only links conditioning assumptions to observed variables and initial conditions. This means that

$$
Z^*_{T+g^*} = K'Y_{T+g^*} + U_T, \tag{17.71}
$$

where $Z^*_{T+g^*} = [z^{*\prime}_{T+g^*} \cdots z^{*\prime}_{T+1}]'$.

Under direct control of the shocks, we can define N_{T+g^*} in terms of the q_m controlled shocks and the $(q-q_m)$ free shocks for each time period in the conditioning sample as in Section 12.2.1. That is,

$$
N_{T+g^*} = \left(I_g^* \otimes \bar{M}_\perp\right)N_{T+g^*}^{(q-q_m)} + \left(I_g^* \otimes \bar{M}\right)N_{T+g^*}^{(q_m)}.
$$

Substituting this expression into equation (17.70) and using (17.71), the controlled shocks satisfy

$$
K'\tilde{D}_{T-1}\left(I_{g^*} \otimes \bar{M}\right)N_{T+g^*}^{(q_m)} = \tilde{k}_{T+g^*}^{(q_m)},
$$

where

$$
\tilde{k}_{T+g^*}^{(q_m)} = Z^*_{T+g^*} - U_T - K'\left(\bar{X}_{T+g^*} + \tilde{G}_{T-1}\xi_T + W_{T+g^*} + \left(I_{g^*} \otimes \bar{M}_\perp\right)N_{T+g^*}^{(q-q_m)}\right).
$$

The population mean of the structural shocks is therefore

$$
\tilde{\mu}_{N,T+g^*} = \left(I_{g^*} \otimes \bar{M}\right)\left[K'\tilde{D}_{T-1}\left(I_{g^*} \otimes \bar{M}\right)\right]^{-1}\left(Z^*_{T+g^*} - U_T - K'\left(\bar{X}_{T+g^*} + \tilde{G}_{T-1}\xi_{T|T+g^*}^{(z)}\right)\right)
$$

where $\xi_{T|T+g^*}^{(z)}$ is computed as in Section 17.9.7. It is also possible to replace this with $\xi_{T|T}$ as an approximation.

Given the population mean of the structural shocks, we now find that the population mean of the forecasted observed variables over the conditioning sample is

$$
E\left[Y_{T+g^*} | \mathcal{Y}_T, Z^*_{T+g^*}; \theta\right] = \bar{X}_{T+g^*} + \tilde{G}_{T-1}\xi_{T|T+g^*}^{(z)} + \tilde{D}_{T-1}\tilde{\mu}_{N,T+g^*}. \tag{17.72}
$$

To simulate from the conditional predictive distribution under the direct control of shocks method, we can follow the principles outlined in Section 12.2.1 taking the necessary changes due to adaptive learning into account. One such change concerns drawing the initial historical state ξ_T from $N(\xi_{T|T+g^*}^{(z)}, P_{T|T+g^*}^{(z)})$. The default behavior in YADA is to set $\xi_{T|T+g^*}^{(z)} = \xi_{T|T}$ and

$P_{T|T+g^*}^{(z)}) = P_{T|T}$, but the case of taking the conditioning assumptions into account for the smooth estimates is also supported. In the latter case, the procedure in Section 12.2.4 needs to be adjusted as described below in Section 17.9.7.

The forecast covariance matrix expression derived in Section 12.2.1 may be used for the fixed beliefs case as well, at least as an approximation, although it does not take variability of the belief coefficients into account and one needs to incorporate $\tilde{D}_{T-1}$ insytead of D in those expressions. A correction for this error can be obtained by using the simulated paths from the predictive distribution minus the covariance matrix in question and attributing this part to the belief coefficient uncertainty.

17.9.4.2. *Updated Beliefs*

Updating of beliefs for the unconditional forecasts in Section 17.9.2 is based on equation (17.63). Once the forecasts of the belief coefficients are available, the solution matrices over the forecast sample are given as in Section 17.3. Under conditional forecasts, however, addtional information may be used when updating the beliefs. First of all, this begs the question if we should take the information into account. If the answer is yes, then a second question is how precisely to achieve that.

One answer to the second question is to follow the approach in Section 17.9.7, where the Kalman filter is run over the forecast sample based on the conditioning assumptions. When beliefs are updated in that exercise, the solution matrices are also obtained from the filtering part of the procedure.

The approach in YADA is to follow the setup in Section 17.9.2 and update the beliefs purely through equation (17.63) and therefore not take implied assumptions about future state variables into account. This simplifies matters considerably, but neglects the possibility that some conditioning information can affect the belief coefficients, presumably if that information would imply some very large structural shocks. However, such information would also affect the smoothed values for the state variables at T and using the procedure in Section 17.9.7 may be more useful in that case. That approach is supported in YADA and may be sufficient given the overall uncertainty of the forecasts.

17.9.5. *Conditional Forecasts under Control of the Distribution of the Shocks*

For the Waggoner and Zha (1999) approach, we begin with the fixed beliefs case, and then discuss the changes when having updated beliefs. In line with Section 12.2.2, we first derive an equation similar to (12.14) which links the observed variables over the conditioning sample to the determininstic variables, the measurement errors and the structural shocks over the conditioning sample as well as the state variables in the last historical time period. Once this relationship has been obtained, we can make use of the conditioning assumptions as in (12.15).

To obtain the mean vector and the covariance matrix of the structural shocks conditional on ξ_T, W_{T+g^*} and θ we follow the approach in Section 12.2.2. Substituting Y_{T+g^*} from (17.70) into (17.71) and rearranging terms we obtain

$$K'\tilde{D}_{T-1}N_{T+g^*} = \tilde{k}_{T+g^*}, \tag{17.73}$$

where

$$\tilde{k}_{T+g^*} = Z_{T+g^*}^* - U_T - K'\left(\bar{X}_{T+g^*} + \tilde{G}_{T-1}\xi_T + W_{T+g^*}\right).$$

As is Waggoner and Zha (1999), the distribution of the structural shocks over the conditioning sample is normal with mean vector $\mu_{N,T+g^*}$ and idempotent covariance matrix $\Sigma_{N,T+g^*}$, where

$$
\begin{aligned}
\mu_{N,T+g^*} &= \tilde{D}_{T-1}'K\left(K'\tilde{D}_{T-1}\tilde{D}_{T-1}'K\right)^{-1}\tilde{k}_{T+g^*} \\
\Sigma_{N,T+g^*} &= I_{qg^*} - \tilde{D}_{T-1}'K\left(K'\tilde{D}_{T-1}\tilde{D}_{T-1}'K\right)^{-1}K'\tilde{D}_{T-1}.
\end{aligned}
\tag{17.74}
$$

The population mean of the structural shocks is therefore

$$\bar{\mu}_{N,T+g^*} = \tilde{D}'_{T-1}K\big(K'\tilde{D}_{T-1}\tilde{D}'_{T-1}K\big)^{-1}$$
$$\times \Big(Z^*_{T+g^*} - U_T - K'\big(\bar{X}_{T+g^*} + \tilde{G}_{T-1}\xi^{(z)}_{T|T+g^*}\big)\Big),$$

so that the predictive population mean of the observed variables is given by

$$E\big[Y_{T+g^*}|\mathcal{Y}_T, Z^*_{T+g^*};\theta\big] = \bar{X}_{T+g^*} + \tilde{G}_{T-1}\xi^{(z)}_{T|T+g^*} + \tilde{D}_{T-1}\bar{\mu}_{N,T+g^*}. \qquad (17.75)$$

Under updated beliefs, the same principle as in Section 17.9.4.2 applies. In terms of the matrix expression above, the formulas become somewhat more complicated, but as long as one keeps in mind the dating of the matrices the expressions can be derived without much complication. We leave that as an exercise for the reader.

17.9.6. *Conditional Forecasts under Conrol of the Distribution of a Subset of the Shocks*

Suppose that not all q structural shocks are used to guarantee that the q_m conditioning assumptions are satisfied in each time period of the conditioning sample. Instead we pick q_r shocks for that purpose and let the remaining $q - q_r$ shocks be free. Following the analysis in Section 12.2.3, it is required that $q_m \leq q_r \leq q$, where $q_r = q_m$ is identical to the direct control method in Section 17.9.4, while $q_r = q$ yields that case examined in Section 17.9.5.

As in Section 17.9.5 we focus on the fixed beliefs case. Following equation (12.34), the stacked vector of structural shocks N_{T+g^*} can be expressed in term of two parts, the $q_r g^*$ shocks used to ensure that the $q_m g^*$ conditioning assumptions are satisfied and the $(q - q_r)g^*$ free shocks. That is,

$$N_{T+g^*} = \big(I_{g^*} \otimes \bar{M}\big)N^{(q_r)}_{t+g^*} + \big(I_{g^*} \otimes \bar{M}_\perp\big)N^{(q-q_r)}_{t+g^*}, \qquad (17.76)$$

where $N^{(q_r)}_{t+g^*} = [\eta^{(q_r)\prime}_{T+g^*} \cdots \eta^{(q_r)\prime}_{T+1}]'$ and $N^{(q-q_r)}_{t+g^*} = [\eta^{(q-q_r)\prime}_{T+g^*} \cdots \eta^{(q-q_r)\prime}_{T+1}]'$, while $\eta^{(q_r)}_t = M'\eta_t$, $\eta^{(q-q_r)}_t = M'_\perp\eta_t$, $M'_\perp M = 0$, $\bar{M} = M(M'M)^{-1}$ and $\bar{M}_\perp = M_\perp(M'_\perp M_\perp)^{-1}$. The shocks are typically selected such that M and $M_\perp$ are specific columns of I_q such that, for example, $\bar{M} = M$. The $(q-q_r)$ free shocks are normally distributed with mean zero and covariance matrix $M'_\perp M_\perp$, typically an identify matrix.

Using equation (17.76) in (17.70) we obtain

$$Y_{T+g^*} = \bar{X}_{T+g^*} + \tilde{G}_{T-1}\xi_T + W_{T+g^*} + \tilde{D}_{T-1}\big(I_{g^*} \otimes \bar{M}_\perp\big)N^{(q-q_r)}_{T+g^*} + \tilde{D}_{T-1}\big(I_{g^*} \otimes \bar{M}\big)N^{(q_r)}_{T+g^*}. \qquad (17.77)$$

Combining equation (17.77) with (17.71) and rearranging terms gives

$$K'\tilde{D}_{T-1}\big(I_{g^*} \otimes \bar{M}\big)N^{(q_r)}_{T+g^*} = \tilde{k}^{(q_r)}_{T+g^*}, \qquad (17.78)$$

where

$$\tilde{k}^{(q_r)}_{T+g^*} = Z^*_{T+g^*} - U_T - K'\Big(\bar{X}_{T+g^*} + \tilde{G}_{T-1}\xi_T + W_{T+g^*} + \tilde{D}_{T-1}\big(I_{g^*} \otimes \bar{M}_\perp\big)N^{(q-q_r)}_{T+g^*}\Big).$$

To satisfy the restrictions in (17.78), the $q_r g^*$ shocks in $N^{(q_r)}_{T+g^*}$ are normal with mean and idempotent covariance matrix given by

$$\mu^{(q_r)}_{N,T+g^*} = \big(I_{g^*} \otimes \bar{M}'\big)\tilde{D}'_{T-1}K\big(K'\tilde{D}_{T-1}\big(I_{g^*} \otimes \bar{M}\bar{M}'\big)\tilde{D}'_{T-1}K\big)^{-1}\tilde{k}^{(q_r)}_{T+g^*}$$
$$\Sigma^{(q_r)}_{N,T+g^*} = I_{qg^*} - \big(I_{g^*} \otimes \bar{M}'\big)\tilde{D}'_{T-1}K\big(K'\tilde{D}_{T-1}\big(I_{g^*} \otimes \bar{M}\bar{M}'\big)\tilde{D}'_{T-1}K\big)^{-1} \qquad (17.79)$$
$$K'\tilde{D}_{T-1}\big(I_{g^*} \otimes \bar{M}\big).$$

The population mean of the structural shocks is therefore

$$\mu^*_{N,T+g^*} = \big(I_{g^*} \otimes \bar{M}\bar{M}'\big)\tilde{D}'_{T-1}K\big(K'\tilde{D}_{T-1}\big(I_{g^*} \otimes \bar{M}\bar{M}'\big)\tilde{D}'_{T-1}K\big)^{-1}$$
$$\times \Big(Z^*_{T+g^*} - U_T - K'\big(\bar{X}_{T+g^*} + \tilde{G}_{T-1}\xi^{(z)}_{T|T+g^*}\big)\Big),$$

while the predictive population mean of the observed variables over the conditioning sample is:

$$E\left[Y_{T+g^*}|\mathcal{Y}_T, Z^*_{T+g^*};\theta\right] = \bar{X}_{T+g^*} + \tilde{G}_{T-1}\xi^{(z)}_{T|T+g^*} + \tilde{D}_{T-1}\mu^*_{N,T+g^*}. \tag{17.80}$$

It is straightforward to show for all three conditioning methods that the mean predictions satisfy the conditioning assumptions. The same conclusion holds when simulating values for the shocks, measurement errors and state variables at T.

17.9.7. *Smooth Estimation of the State Variables using the Conditioning Assumptions Under Adaptive Learning*

Like in Section 12.2.4, we begin with the relationship between the conditioning assumtions, the observed variables and the initial condition for the conditioning data (here denoted also with an asterisk):

$$z^*_t = \begin{bmatrix} K'_1 & K'_{2,1} & \cdots & K_{2,g^*-1} \end{bmatrix} \begin{bmatrix} y_t \\ y_{t-1} \\ \vdots \\ y_{t-g^*+1} \end{bmatrix} + u^*_{t-g^*}, \quad t = T+1,\ldots,T+g^*,$$

or more compactly

$$z^*_t = K^{*\prime}Y_t + u^*_{t-g^*}. \tag{17.81}$$

The matrix K^* has dimension $ng^* \times q_m$. To derive a state-space representaion for z^*_t and ξ_t over the conditioning part of the forecast sample, we rewrite the measurement equation for y_t as follows

$$y_t = A'x_t + \begin{bmatrix} H'_t & I_n \end{bmatrix} \begin{bmatrix} \xi_t \\ w_t \end{bmatrix}$$

$$= A'x_t + H^{*\prime}_t\xi^*_t. \tag{17.82}$$

In other words, we treat the measurement errors as state variables. The state equation for the model variables can now be expressed as

$$\begin{bmatrix} \xi_t \\ w_t \end{bmatrix} = \begin{bmatrix} M_t \\ 0 \end{bmatrix} x_t + \begin{bmatrix} F_t & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \xi_{t-1} \\ w_{t-1} \end{bmatrix} + \begin{bmatrix} B_t\eta_t \\ w_t \end{bmatrix},$$

or

$$\xi^*_t = M^*_t x_t + F^*_t\xi^*_{t-1} + v^*_t, \tag{17.83}$$

where M_t, F_t and B_t will be replaced with suitable solution matrices over the forecast sample as further discussed below.

The state equation innovations v^*_t are normally distributed with zero mean and covariance matrix

$$Q^*_t = \begin{bmatrix} B_t B'_t & 0 \\ 0 & R \end{bmatrix}.$$

Let us next stack the measurement equation such that we can substitute for Y_t in (17.81). This means that

$$\begin{bmatrix} y_t \\ y_{t-1} \\ \vdots \\ y_{t-g^*+1} \end{bmatrix} = \begin{bmatrix} A'x_t \\ A'x_{t-1} \\ \vdots \\ A'x_{t-g^*+1} \end{bmatrix} + \begin{bmatrix} H^{*\prime}_t & 0 & \cdots & 0 \\ 0 & H^{*\prime}_{t-1} & & 0 \\ \vdots & & \ddots & \\ 0 & 0 & & H^{*\prime}_{t-g^*+1} \end{bmatrix} \begin{bmatrix} \xi^*_t \\ \xi^*_{t-1} \\ \vdots \\ \xi^*_{t-g^*+1} \end{bmatrix},$$

or

$$Y_t = X_t + \tilde{H}^{*\prime}_t\Xi^*_t, \quad t = T+1,\ldots,T+g^*. \tag{17.84}$$

Substituting for Y_t in (17.81), we obtain the following measurement equation for z_t^*

$$z_t^* = \begin{bmatrix} K^{*\prime} & I_{q_m} \end{bmatrix} \begin{bmatrix} X_t \\ u_{t-g^*}^* \end{bmatrix} + K^{*\prime}\tilde{H}_t^{*\prime}\Xi_t^*$$

$$= \tilde{A}'X_t^* + \tilde{H}_t'\Xi_t^*, \quad t = T+1,\ldots,T+g^*.$$

(17.85)

The state equation can also be stacked conformably, i.e.

$$\begin{bmatrix} \xi_t^* \\ \xi_{t-1}^* \\ \xi_{t-2}^* \\ \vdots \\ \xi_{t-g^*+1}^* \end{bmatrix} = \begin{bmatrix} M_t^* \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} x_t + \begin{bmatrix} F_t^* & 0 & \cdots & 0 & 0 \\ I_{r+n} & 0 & \cdots & 0 & 0 \\ 0 & I_{r+n} & & 0 & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & & I_{r+n} & 0 \end{bmatrix} \begin{bmatrix} \xi_{t-1}^* \\ \xi_{t-2}^* \\ \xi_{t-3}^* \\ \vdots \\ \xi_{t-g^*}^* \end{bmatrix} + \begin{bmatrix} v_t^* \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix},$$

or with the compact expression,

$$\Xi_t^* = \bar{M}_t x_t + \bar{F}_t \Xi_{t-1}^* + V_t^*, \quad t = T+1,\ldots,T+g^*.$$

(17.86)

Notice that V_t^* has zero mean and covariance matrix $\bar{Q}_t$, a positive semidefinite matrix of dimension $g^*(r+n) \times g^*(r+n)$ given by

$$\bar{Q}_t = \begin{bmatrix} Q_t^* & 0 \\ 0 & 0 \end{bmatrix}.$$

To compute $\Xi_{T+1|T}^*$ and its covariance matrix $\tilde{P}_{T+1|T}$ of the stacked system, we first let $M_{T+1} = \tilde{M}_{T-1}$, $F_{T+1} = \tilde{F}_{T-1}$ and $B_{T+1} = \tilde{B}_{T-1}$, i.e., the solution matrices used in iteration T of the Kalman filter in Section 17.5, thereby determining $\bar{M}_{T+1}$, $\bar{F}_{T+1}$, $\bar{B}_{T+1}$ and $\bar{Q}_{T+1}$. It now follows that

$$\Xi_{T+1|T}^* = \bar{M}_{T+1} x_{T+1} + \bar{F}_{T+1} \Xi_{T|T}^*,$$

where $\Xi_{T|T}^*$ is constructed from $\xi_{T-i|T}$ and $w_{T-i|T}$ for $i = 0, 1, \ldots, g^* - 1$. Furthermore,

$$\tilde{P}_{T+1|T} = \bar{F}_{T+1} \tilde{P}_{T|T} \bar{F}_{T+1}' + \bar{Q}_{T+1}.$$

The matrix $\tilde{P}_{T|T}$ is obtained by letting the block diagonal matrix elements be given by

$$P_{T-i|T}^* = \begin{bmatrix} P_{T-i|T} & 0 \\ 0 & R \end{bmatrix}, \quad i = 0, 1, \ldots, g^* - 1.$$

Notice that block (i, i) corresponds to rows (and columns) $i(r+n) + 1$ until $(i+1)(r+n)$ for $i = 0, 1, \ldots, g^* - 1$. The off-diagonal block matrices in $\tilde{P}_{T|T}$ for row and column block (i, j) are given by

$$\begin{bmatrix} \left(\prod_{k=i}^{j-1} \tilde{F}_{T-k-2}\right) P_{T-j+t|T} & 0 \\ 0 & 0 \end{bmatrix}, \quad \text{if } g^* - 1 \geq j > i \geq 0,$$

and to the transpose of this expression if $g^* - 1 \geq i > j \geq 0$. For instance, suppose $i = 0$ and $j = 1$, then block $(0, 1)$ corresponds to rows 1 until $(r+n)$ and columns $(1+r+n)$ until $2(r+n)$ of $\tilde{P}_{T|T}$.

With the initial values for $\Xi_{T+1|T}^*$ and $\tilde{P}_{T+1|T}$ at hand, the Kalman filter from Section 1.5 may be applied until period $T + g^*$. If beliefs are assumed to be fixed, the solution matrices are also kept fixed over the conditioning horizon such that step 4 of the filter is skipped. On the other hand, if beliefs are updated then the solution matrices $(\tilde{M}_T, \tilde{F}_T, \tilde{B}_T)$ are used as input when the filter is run for periods $T + 2, \ldots, T + g^*$ with updating of beliefs based on step 4 of the filter. Once the forecast and update estimates of the state variables, conditioning variables and their covariance matrices are available, the Kalman smoother from Section 1.6. may be employed to

generate $(\Xi^*_{t|T+g^*}, \tilde{P}_{t|T+g^*})$ for periods $t = T + g^* - 1, T + g^*$ only. From the previous of these two periods we can extract $\xi^{(z)}_{T|T+g^*}$ from rows $(g^* - 1)(r + n) + 1$ until $(g^* - 1)(r + n) + r$ of $\Xi^*_{T+g^*-1|T+g^*}$, while $P^{(z)}_{T|T+g^*}$ is given by the rows and columns of $\tilde{P}_{T+g^*-1|T+g^*}$ using the same range of integers.

17.10. *Adaptive Learning and the Zero Lower Bound*

Similar to the rational expectations case for the zero lower bound in Section 3.4, the DSGE model with adaptive learning can also be solved under this nonlinear constraint using anticipated shocks with some minor adjustments to the HLS approach for the RE case; see also Hebden et al. (2011) for details on their original approach.

As in Section 3.4.3, let $e_{p,r}$ be a p-dimensional vector with unity in position r and zeros elsewhere, i.e. the r:th column of I_p. Furthermore, $K_{p,r}$ is a $p \times (p - 1)$ matrix with the remaining columns of I_p. Making using of such generic selection vectors and matrices, the matrices of the DSGE model in (17.1) can be redefined as $H^{(m)}_i = K'_{p,m}H_i$, $h^{(m)}_i = e'_{p,m}H_iK_{p,r}K'_{p,r}$, $h^{(r)}_i = e'_{p,m}H_ie_{p,r}$ for $i = -1, 0, 1$, while $D^{(m)} = K'_{p,m}D$ and $d^{(m)} = e'_{p,m}D$. Let the m:th equation of the DSGE model be the unrestricted policy rule, while the restricted policy rate is located in position r of z_t and denoted by $\tilde{r}_t$. The unrestricted policy rate is denoted by $\hat{r}_t$.

The DSGE model with anticipated shocks can be expressed as

$$\begin{bmatrix} H^{(m)}_{-1} & 0 \\ h^{(m)}_{-1} & h^{(r)}_{-1} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} z_{t-1} \\ \hat{r}_{t-1} \end{bmatrix} + \begin{bmatrix} H^{(m)}_0 & 0 \\ h^{(m)}_0 & h^{(r)}_0 \\ e'_{p,r} & -1 \end{bmatrix} \begin{bmatrix} z_t \\ \hat{r}_t \end{bmatrix} + \begin{bmatrix} H^{(m)}_1 & 0 \\ h^{(m)}_1 & h^{(r)}_1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} E_t[z_{t+1}] \\ E_t[\hat{r}_{t+1}] \end{bmatrix} =$$

$$= \begin{bmatrix} D^{(m)} \\ d^{(m)} \\ 0 \end{bmatrix} \eta_t + \begin{bmatrix} 0 \\ 0 \\ e'_{T+1,1} \end{bmatrix} A_t,$$

or

$$H^{(e)}_{-1}Y_{t-1} + H^{(e)}_0 Y_t + H^{(e)}_1 E_t[Y_{t+1}] = D^{(e)}\eta_t + S^{(e)}A_t. \tag{17.87}$$

When solving the rewritten model under adaptive learning for forecasting purposes we ignore A_t as it is exogenous and assumed to be zero for all historical time periods. The solution of the model is obtained by first using $H^{(e)}_i$ and $D^{(e)}$ instead of H_i and D such that the transformation in equation (17.5) can be rewritten as

$$H^*_{-1}Y_{t-1} + H^*_0 Y_t + H^*_{1,f}E_t[Y^f_{t+1}] = D^{(e)}\eta_t + S^{(e)}A_t, \tag{17.88}$$

where Y^f_t is a vector with the forward looking variables only. If $h^r_1 = 0$ then Y^f_t is identical to z^f_t and the unrestricted policy rate is not forward looking. On the other hand, when $h^r_1 \neq 0$ and the policy rate is selected as a forward looking variable, then one must ensure that the restricted policy rate is not forward looking, while the unrestricted policy rate is.

Second, with the change of dimension of the vector of model variables, the G_j matrices in equation (17.14) need to reflect this. It is possible in YADA to chose if the lagged unrestricted or the lagged restricted policy rate is allowed to appear in the policy rule. This decision also affects how the G_j matrices are rewritten. Suppose $m = 2$ such that

$$q_{t-1,j} = G'_j Z_{t-1} = \begin{bmatrix} G_{j,k} \\ G_{j,1} \\ G_{j,2} \end{bmatrix}' \begin{bmatrix} x_t \\ z_{t-1} \\ z_{t-2} \end{bmatrix},$$

where $G_{j,k}$ is $k \times g_j$, while $G_{j,i}$ is $p \times g_j$ for $i = 1, 2$.

Suppose that the lagged unrestricted policy rate is allowed to enter the PLM. In that case, the G_j are rewritten such that

$$G'_{j,i} K_{p,r} K'_{p,r} z_{t-i} + G'_{j,i} e_{p,r} \hat{r}_{t-i} = G^{(e)'}_{j,i} Y_{t-i}, \quad i = 1, 2.$$

If instead the lagged restricted policy rate is allowed to enter the PLM, the matrices are rewritten such that

$$G'_{j,i} z_{t-i} + 0 \hat{r}_{t-i} = G^{(e)'}_{j,i} Y_{t-i}, \quad i = 1, 2.$$

Third, the solution approach discussed in Section 17.3 can now be applied to the transformed system in equation (17.88). If $m \leq 2$, the solution for Y_t given η_t and A_t can be expressed as

$$Y_t = \tilde{M}_t x_t + \tilde{F}_t Y_{t-1} + \tilde{B}^{(\eta)}_t \eta_t + \tilde{B}^{(A)}_t A_t, \tag{17.89}$$

where

$$\tilde{M}_t = -\left[H_0^* + H_{1,f}^* \tilde{\beta}_{t|t,1} \right]^{-1} H_{1,f}^* \tilde{\beta}_{t|t,0},$$

$$\tilde{F}_{t,1} = -\left[H_0^* + H_{1,f}^* \tilde{\beta}_{t|t,1} \right]^{-1} \left[H_{-1}^* + H_{1,f}^* \tilde{\beta}_{t|t,2} \right],$$

$$\tilde{B}^{(\eta)}_t = \left[H_0^* + H_{1,f}^* \tilde{\beta}_{t|t,1} \right]^{-1} D^{(e)},$$

$$\tilde{B}^{(A)}_t = \left[H_0^* + H_{1,f}^* \tilde{\beta}_{t|t,1} \right]^{-1} S^{(e)}.$$

To deal with the lower bound looking forward, we would like to express the system solution as in equation (3.28) for the predetermined variables, also taking x_t into account. This means that

$$\begin{bmatrix} \eta_t \\ Y_{t-1} \\ A_t \end{bmatrix} = \begin{bmatrix} 0 \\ \tilde{M}_{t-1} \\ 0 \end{bmatrix} x_{t-1} + \begin{bmatrix} 0 & 0 & 0 \\ \tilde{B}^{(\eta)}_{t-1} & \tilde{F}_{t-1} & \tilde{B}^{(A)}_{t-1} \\ 0 & 0 & M \end{bmatrix} \begin{bmatrix} \eta_{t-1} \\ Y_{t-2} \\ A_{t-1} \end{bmatrix} + \begin{bmatrix} I_q & 0 \\ 0 & 0 \\ 0 & I_{T+1} \end{bmatrix} \begin{bmatrix} \eta_t \\ \Phi_t \end{bmatrix},$$

or

$$V_t = M_{t-1} x_{t-1} + G_{t-1} V_{t-1} + C_1 \begin{bmatrix} \eta_t \\ \Phi_t \end{bmatrix}. \tag{17.90}$$

Using equation (17.89), equation (3.29) can be rewritten for the adaptive learning case as

$$Y_t = \tilde{M}_t x_t + \begin{bmatrix} \tilde{B}^{(\eta)}_t & \tilde{F}_t & \tilde{B}^{(A)}_t \end{bmatrix} \begin{bmatrix} \eta_t \\ Y_{t-1} \\ A_t \end{bmatrix} = \tilde{M}_t x_t + P_t V_t. \tag{17.91}$$

When projecting conditional on the model variables at t, the solution at t is used for $t+1$, i.e. the matrices $\tilde{M}_t$ and P_t. If we also assume that the belief coefficients are fixed when projecting, $P_{t+\tau,t} = P_t$ and similarly $\tilde{M}_{t+\tau,t} = \tilde{M}_t$ and $G_{t+\tau,t} = G_t$.[141] In terms of projections, this implies that

$$Y_{t+\tau,t} = \tilde{M}_t x_{t+\tau} + P_t \sum_{i=0}^{\tau-1} G_t^i M_t x_{t+\tau-1-i} + P_t G_t^\tau V_t, \quad \tau \geq 0.$$

The projected restricted and unrestricted policy rates are obtained by premultiplying $Y_{t+\tau,t}$ by $e'_{p+1,r}$ and $e'_{p+1,p+1}$, respectively; see equations (3.30) and (3.31) for the rational expectations case. The procedure in Sections 3.4.4–3.4.5 can now be applied to solve for the anticipated shocks in A_t from $\tau = 0, 1, \ldots, T$.

---

[141] Once the double Kalman filter and smoother is used, the relevant matrices are dated $t-1$ instead of t.

If the belief coefficients are instead updated as in Section 17.9.2. In that case, the projections of $Y_{t+\tau}$ are:

$$Y_{t+\tau,t} = \tilde{M}_{t+\tau,t} x_{t+\tau} + P_{t+\tau,t} \sum_{i=0}^{\tau-1} \left(\prod_{j=1}^{i} G_{t+\tau-j,t} \right) M_{t+\tau-1-i,t} x_{t+\tau-1-i}$$

$$+ P_{t+\tau,t} \left(\prod_{j=1}^{\tau} G_{t+\tau-j,t} \right) V_t, \quad \tau \geq 0.$$

These calculations can be simplified in practise. In the case of the fixed beliefs case, notice that for $\tau \geq 2$

$$\bar{x}_{t+\tau-1} = \sum_{i=0}^{\tau-1} G_t^i M_t x_{t+\tau-1-i}$$

$$= M_t x_{t+\tau-1} + G_t \bar{x}_{t+\tau-2},$$

while $\bar{x}_t = M_t x_t$. Similarly when beliefs are updated, it holds that

$$\bar{x}_{t+\tau-1} = \sum_{i=0}^{\tau-1} \left(\prod_{j=1}^{i} G_{t+\tau-j,t} \right) M_{t+\tau-1-i,t} x_{t+\tau-1-i}$$

$$= M_{t+\tau-1,t} x_{t+\tau-1} + G_{t+\tau-1,t} \bar{x}_{t+\tau-2},$$

while the initial condition is the same as for the fixed beliefs case.

Stochastic simulations are discussed in Section 3.4.6 and the algorithm reported there can also be used with the modifications to the solution algorithm that are provided above.

17.11. *YADA Code*

This section contains information about specific YADA functions that are used when applying adaptive learning to a DSGE model. The function are grouped into those dealing with parameters and solving the DSGE model, densities, estimation, sampling and tools.

17.11.1. *Parameters and Model Solving*

17.11.1.1. `AdaptiveLearningTransformDSGEModel`

The function `AdaptiveLearningTransformDSGEModel` deals with transforming the model as discussed in Section 17.1.1. The aim of the function is to deliver the matrices in equation (17.5) and to obtain this goal, 5 input variables are used: `Hlag`, `H0`, `Hexp`, `S`, and `Sperp`. The first three matrices are given by the three matrices on the state variables in the structural form equation (17.1), while the last two are simply S and $S_\perp$.

The output variables are given by `TransStatus`, `Hstarlag`, `Hstar0`, and `Hstar1f`. The first is a boolean variable which is unity if the transformation is successful and zero otherwise. The remaining three variables are the matrices from equation (17.5) that are pre-multiplied to the state variables for the lag and the contemporaneous cases and the forward looking variables.

17.11.1.2. `AdaptiveLearningInitialValuesDSGE`

The function `AdaptiveLearningInitialValuesDSGE` computes the initial values of the beliefs as discussed in Section 17.2. The required input variables are 11 in total and are given by: `F`, `B0`, `sigmarAL`, `sigmaeAL`, `SMatrix`, `G`, `m`, `x`, `f`, `MaximumIterationsValue`, and `ToleranceValue`. The first two variables are simply the solution of the DSGE model under rational expectations. The following two input variables are the adaptive learning parameters σ_r and σ_ε. The following input variable is simply the matrix S, which selects the f forward looking variables among the state variables. The cell array `G` has f elements with the matrices G_j that are used to determine the explanatory variables in the PLM for each one of the forward looking variables in equation (17.14). The lag order m of the PLM is given by `m` and as mentioned above it is at most equal to 2 in YADA. The matrix `x` is $k \times T$ and holds the estimation sample data for the deterministic

variables, while the integer f is the number of forward looking variables. The last two required input variables are used by the doubling algorithm as discussed in Section 5.17.1.

The output variables are given by: `betavec`, `Sigmau`, `R10`, `Sigmaeps`, `Gtilde`, and `Kgf`. The first variable is the initial value of the vector of belief coefficients, β, in equation (17.15), the second variable is the covariance matrix Σ_u in equation (17.21), while the third and fourth variables are given by $R_{1|0}$ and Σ_ε, which both depend on the so called GLS matrix in equation (17.22). The fifth output variable is given by $\tilde{G}$ in equation (17.16), while the last output variables is the $gf \times gf$ commutation matrix K_{gf} from Section 17.2.2.

The last two output variables do not depend on parameters and need therefore not be recomputed when parameters change. The function therefore accepts two optional input variables, `Gtilde` and `Kgf`, which are only submitted to the function when it has already been run before during the same session.

17.11.1.3. `AdaptiveLearningSolveDSGEModel`

The function `AdaptiveLearningSolveDSGEModel` solves the DSGE model under adaptive learning as in equation (17.30). It takes 9 input variables: `Hstar0`, `Hstarlag`, `Hstar1f`, `D`, `Gtilde`, `Kgf`, `m`, `k`, and `betatt`. The first three matrices are obtained via the model transformation function above, while `D` is simply the matrix on the shocks in the structural form. The fifth and sixth input variables are given by $\tilde{G}$ and K_{gf} and which are output variables of the initial values function. The next two input variables are simply the lag order of the PLM and the number of deterministic variables in the DSGE model, while the final input variable is the vector of belief coefficients.

The first three output variables are given by the matrices: `Mt1`, `Ft1`, and `Bt1`. They correspond to the matrices $\tilde{M}_t$, $\tilde{F}_t$, and $\tilde{B}_t$ in equation (17.30). In addition, the function provides one optional output variable denoted by `InvMat`. It holds the inverse of the matrix on z_t in equation (17.27) which is used when solving the model. This inverse is required when solving the model with anticipated shocks, as in equation (17.89).

17.11.2. *Density Functions*

17.11.2.1. `logPosteriorAdaptiveLearningPhiDSGE`

The function `logPosteriorAdaptiveLearningPhiDSGE` computes the log posterior kernel for the transformed parameters and takes 19 input variables. The 8 variables `phi`, `thetaIndex`, `UniformBounds`, `LowerBound`, `thetaPositions`, `thetaDist`, `PriorDist`, and `ModelParameter` are shared with the function used under rational expectations, i.e., `logPosteriorPhiDSGE` in Section 7.4. The following 6 variables are specific to adaptive learning and given by: `S`, `Sperp`, `m`, `G`, `Gtilde`, and `Kgf` and are discussed above. The `xbar` variable, a vector of dimension k equal to the number of deterministic variables and with 1 in the position of the constant and zero entries elsewhere. The remaining 4 input variables are: `DSGEModel`, `AIMData`, `OrderQZ`, and `NumWorkers` and these are shared with many of the functions in YADA.

The function provides 1 required and 2 optional output variables. The required is called `logPost` and gives the value of the log posterior kernel at the transformed parameter value. The optional variables are `logLike` and `logLikeT`, the log likelihood value for the full sample and a vector of the log likelihood value for individual time period, with the former being the sum of the latter.

17.11.2.2. `logPosteriorAdaptiveLearningThetaDSGE`

The function `logPosteriorAdaptiveLearningThetaDSGE` takes the same input variables as the previous function, exact that `phi`, the vector of transformed parameters, is replaced with `theta`, the vector of original parameters.

The output variables are essentially also shared with the previous function, except that `logPost` is the value of the log posterior kernel at the original parameter value.

17.11.2.3. `logLikelihoodAdaptiveLearningDSGE`

The function `logLikelihoodAdaptiveLearningDSGE` computes the log-likelihood of the DSGE model under adaptive learning. It takes 12 input variables: `ModelParameters`, `S`, `Sperp`, `m`, `G`, `Gtilde`, `Kgf`, `xbar`, `DSGEModel`, `AIMData`, `OrderQZ`, and `NumWorkers`. All these variables are also input variables for the log posterior kernel functions above.

The function gives 4 required and 1 optional output variable. The required variables are: `logLikeValue`, `Solution`, `mcode`, and `status`. The first variable is the value of the log-likelihood function at the value supplied to it; `Solution` is a structure with fields `A`, `H` and `R` from the measurement equation, and `F` and `B0` from the steady-state solution matrices under adaptive learning. Like previously, `mcode` is the solution status of the DSGE model under rational expectations. The variable `status` indicates if the log-likelihood function was calculated (unity) or not (zero), and is equal to -1 is the forecast covariance matrix of the adaptive learning based Kalman filter is not positive definite for some time period in the estimation sample. The optional output variable is `logLikeT`, a vector with the log-likelihood values for the individual time periods in the sample.

17.11.3. *Kalman Filter and Estimation Functions*

17.11.3.1. `AdaptiveLearningKalmanFilter`

The function `AdaptiveLearningKalmanFilter` computes the Kalman filtered values of the log-likelihood, the filter and forecast values of the state variables, the belief coefficients and the solution of the DSGE model under adaptive learning at the given value of the unknown parameters. It takes a total of 31 input variables and the first 7 are all related to the adaptive learning initialization of the filter: `m`, `SMatrix`, `betavec`, `Sigmau`, `R10`, `Sigmaeps`, and `rhoAL`. The last variable is the value of the ρ parameter in the state equation for the belief coefficients in (17.8). The next 4 input variables are given by the structural matrices in the transformed model (17.5), i.e., `Hstar0`, `Hstarlag`, `Hstar1f`, and `D`. The following 3 variables concern the steady-state solution matrices `Mt1`, `Ft1`, and `Bt1` which are used initially by the filter and thereafter the function needs `Gtilde` and `Kgf` to set up the explanatory variables for the PLM. The succeeding 5 variables concern data and sample and are given by `Y`, `X`, `FirstPeriod`, `LastPeriod`, and `xbar`. The matrices with data, `Y` and `X`, are assumed to contain all available data on the observed variables and the deterministic variables, respectively. The first and last period of the sample are given by the integer values of the serendipitously named `FirstPeriod` and `LastPeriod`. The `xbar` is dicsussed above for the log-likelihood function. The subsequent 3 variables are the matrices from the measurement equation `A`, `H`, and `R`. These are followed by 5 variables that are also used by the regular Kalman filter function in Section 5.17.1: `InitP`, `c`, `MaxIter`, `Tolerance`, and `StartPeriod`. The last 2 input variables are boolean and given by `AllowNaNs` and `NoSmoothedStatesForBeliefs`. The first indicates if the observed data has missing observations or not, while the second variable determines if the user wishes to use smoothed states for the second lag of the states variables included in the PLM or not.

The function has 2 required output variables, given by `LogLikeValue` and `BeliefStatus`. The former is simply the scalar value of the log-likelihood over the estimation sample, while the latter variable reports the status of the filter, with 1 being no errors and 0 being the other as indicated by the log-likelihood value being a NaN or not a real number.

The number of optional output variables is 15 the first among them is `lnLt`, a vector with the individual estimation sample values of the log-likelihood $(t|t-1)$. This variable is followed by `Yerror` and `Yhat`, a matrix with the one-step-ahead forecast errors of the observed variables and the one-step-ahead forecasts, respectively. The next two variables, `MSEY` and `InvMSEY`, are 3D matrices of dimension $n \times n \times T$ with the one-step-ahead forecast error covariance matrices of the observed variables and their inverses. The subsequent 5 output variables concern the state variables and the belief coefficients: `Ksitt1`, `Ptt1`, `Ksitt`, `Ptt`, and `betatt`. Next, the `Zlag` variable is a matrix of dimension $(k+r) \times T$ with the estimated explanatory variables in the PLM. It is followed by 3 variables contain the solution matrices over the estimation sample and are called `Mtilde`, `Ftilde`, and `Btilde`. The last output variable is given by the structure

`LastSolution` which has 6 fields. These are given by `betaTT`, `MTT`, `FTT`, `BTT`, `betavec`, and `M`. The first 4 fields are the solution estimates at the end of the last sample period, i.e., the values that would be used in period $T + 1$. The field `betavec` is the steady-state value of β, while `M` holds the $g \times g$ matrix Γ.

17.11.3.2. `AdaptiveLearningKalmanSmoother`

The function `AdaptiveLearningKalmanSmoother` computes smoothed estimates of the state variables, structural shocks and measurement errors. The function uses 12 input variables to fulfill its task: `Ksitt1`, `Ptt1`, `Yerror`, `InvMSEY`, `H`, `R`, `Ftilde`, `Btilde`, `FirstPeriod`, `LastPeriod`, `StartPeriod`, and `AllowNaNs`. All inputs are discussed above for the Kalman filter function.

The 6 output variables are given by: `KsitT`, `PtT`, `etatt`, `etatT`, `wtt`, and `wtT`. The first two output variables are the smooth estimates of the state variables along with their covariance matrices over the full estimation sample. The following two output variables are the update and the smooth estimates of the structural shocks, while the last two variables are the update and smooth estimates of the measurement errors.

17.11.3.3. `AdaptiveLearningPosteriorModeEstimation`

The function `AdaptiveLearningPosteriorModeEstimation` estimates the posterior mode of the DSGE model parameters for either the transformed or the original parameters when expectations are formed through the adaptive learning mechanism rather than as fully model consistent expectations. The function works similar to the regular posterior mode estimation function in Section 7.4. Since the adaptive learning approach involves three additional parameters to the ones in the DSGE model, the vector of parameters can include additional entries for the adaptive learning case.

17.11.4. *Sampling Functions*

17.11.4.1. `AdaptiveLearningDSGERWMPosteriorSampling`

The function `AdaptiveLearningDSGERWMPosteriorSampling` runs the RWM algorithm for sampling from the posterior density based on a normal proposal density and a single block of parameters. It works in the same basic way as the regular RWM posterior sampler for DSGE models in Section 8.7. The corresponding case is valid for the RWM sampler with a Student-t proposal density, `AdaptiveLearningDSGERWMStudentPosteriorSampling`, the fixed and random blocking functions, also based on either a normal or a Student-t proposal density and with function names that can be correctly surmised without further ado. There are therefore a total of six posterior samplers under adaptive learning that can be used with these MCMC based approaches.

17.11.4.2. `AdaptiveLearningDSGESlicePosteriorSampling`

The function `AdaptiveLearningDSGESlicePosteriorSampling` handles the slice sampler under adaptive learning and it works in much the same way as the rational expectations based function `DSGESlicePosteriorSampling` discussed in Section 8.7.

17.11.4.3. `AdaptiveLearningDSGESMCLikelihoodTemperingPosteriorSampler` & `AdaptiveLearningDSGESMCDataTemperingPosteriorSampler`

The function `AdaptiveLearningDSGESMCLikelihoodTemperingPosteriorSampler` operates the execution of the sequential Monte Carlo with likelihood tempering posterior sampler, while `AdaptiveLearningDSGESMCDataTemperingPosteriorSampler` takes care of the data tempering approach. Both functions work in much the same way as their rational expectations versions in Section 8.7.

17.11.4.4. `AdaptiveLearningDSGEISMitISEMPosteriorSampler`

The function `AdaptiveLearningDSGEISMitISEMPosteriorSampler` runs the importance sampler based on the MitISEM algorithm for an adaptive learning based DSGE model. Again, aside from adaptive learning peculiarities this function operates in the same way as its rational expectations version `DSGEISMitISEMPosteriorSampler`.

17.11.5. *Tools Functions*

17.11.5.1. `CalculateAdaptiveLearningStateVariables`

The function `CalculateAdaptiveLearningStateVariables` provides output on estimates of various unobserved variables and works in a very similar way as its rational expectations relation `CalculateDSGEStateVariables` in Section 11.18.3. At this stage, the function does not compute the covariance matrices of the update and smooth estimates of the structural shocks or the measurement errors. Furthermore, recursive estimation of the unobserved variables is also not suuported. These features, however, are planned for a future release of YADA.

17.11.5.2. `AdaptiveLearningIRFtheta`

The function `AdaptiveLearningIRFtheta` computes the impulse responses as discussed in Section 17.7 for a fixed value of the parameters. In addition to the adaptive learning responses for each time period in the estimation sample, it also computes the same functions under rational expectations. The function needs 7 input variables: `Ftilde`, `Btilde`, `H`, `F`, `B0`, `DSGEModel`, and `UseLevels`. The last variable is 1 if impulse responses for the levels should be calculated, 2 if they should be computed for annualized variables, and 0 if they should be kept in their original form.

The output variable is given by `IRStructure` which supports two fields, `RE` and `AL`. These contain the results from using rational expectations and adaptive learning, respectively. The former field has T elements which hold the sub-fields `Ksi` and `Y` for the responses of the state variables and observed variables, while the latter has dimention one and has the same named sub-fields. The `Ksi` and `Y` fields have dimensions $r \times q \times (h + 1)$ and $n \times q \times (h + 1)$, with h being the final period of the response horizon.

17.11.5.3. `AdaptiveLearningDSGEPredictionPathsTheta`

The function `AdaptiveLearningDSGEPredictionPathsTheta` computes unconditional forecasts for fixed parameter values as discussed in Sections 17.9.1 and 17.9.2 for the fixed and updated beliefs. Apart from this adaptive learning specific feature, the function works much the same way as its rational expectations relation `DSGEPredictionPathsTheta`, discussed in Section 12.11.

18. Required Input for YADA

In order to estimate a DSGE model YADA needs input from the user on the observed data, the measurement equation, the prior distribution, parameters that are defined from other parameters (such as the steady state), and the DSGE model. This section discusses all these topics through the example in Section 2.1. The *DSGE Data* tab in YADA is shown in Figure 9 indicating the various input files that are used to estimate the parameters of the An and Schorfheide model.

18.1. *Construction of the AiM Model File*

The AiM model file is simply a text file that is written using a syntax that the AiM parser can interpret. The code used for the An and Schorfheide model is listed in Table 1. In this case, the model has 6 state variables (cf. equation (2.1)), but an additional state variable has been included to account for the need of $\hat{y}_{t-1}$ in the measurement equation for ΔY_t in (2.2). Hence,

FIGURE 9: The DSGE Data tab on the YADA window.

$r = 7$ is the dimension of ξ_t. The model also has $q = 3$ shocks (the $\eta_{i,t}$ variables) which are listed among the variables, and one constant. The total number of variables (and equations) is therefore NumEq = 11. The names of the variables given in Table 1 will also appear in the string matrix that will be sent as input to the measurement equation function, i.e., the string matrix StateVarNames. Similarly, the names of the equations, e.g., EQ1Euler, will also show up in a string matrix that can be used to help YADA determine exactly which are the state equations of the structural form.

TABLE 1: The AiM model file code for the An and Schorfheide example in equation (2.1).

```
MODEL> ASmodel
ENDOG>
        yhat        _NOTD
        pihat       _NOTD
        chat        _NOTD
        rhat        _NOTD
        ghat        _NOTD
        zhat        _NOTD
        yhatlag     _NOTD
        one         _DTRM
        etaR        _NOTD
        etaG        _NOTD
        etaZ        _NOTD
EQUATION> EQ1Euler
EQTYPE>   IMPOSED
EQ>       yhat =      LEAD(yhat,1) + ghat - LEAD(ghat,1) - (1/tau)*rhat
                      + (1/tau)*LEAD(pihat,1) + (1/tau)*LEAD(zhat,1)
EQUATION> EQ2Phillips
EQTYPE>   IMPOSED
EQ>       pihat =     beta*LEAD(pihat,1) + kappa*yhat
                      - kappa*ghat
EQUATION> EQ3Consumption
EQTYPE>   IMPOSED
EQ>       chat =      yhat - ghat
EQUATION> EQ4MonPolicyRule
EQTYPE>   IMPOSED
EQ>       rhat =      rhoR*LAG(rhat,1) + (1-rhoR)*psi1*pihat + (1-rhoR)*psi2*yhat
                      - (1-rhoR)*psi2*ghat + sigmaR*etaR
EQUATION> EQ5GovConsumption
EQTYPE>   IMPOSED
EQ>       ghat =      rhoG*LAG(ghat,1) + sigmaG*etaG
EQUATION> EQ6Technology
EQTYPE>   IMPOSED
EQ>       zhat =      rhoZ*LAG(zhat,1) + sigmaZ*etaZ
EQUATION> EQ7YLag
EQTYPE>   IMPOSED
EQ>       yhatlag =   LAG(yhat,1)
EQUATION> EQ8OneDef
EQTYPE>   IMPOSED
EQ>       one =       0*LAG(one,1)
EQUATION> EQ9MonPolShock
EQTYPE>   IMPOSED
EQ>       etaR =      0*one
EQUATION> EQ10GovConsShock
EQTYPE>   IMPOSED
EQ>       etaG =      0*one
EQUATION> EQ11TechShock
EQTYPE>   IMPOSED
EQ>       etaZ =      0*one
END
```

It may be noted that the first and the third equation in Table 1 are written exactly as in An and Schorfheide (2007, equations 29 and 31) and not as in (2.1). These two ways of writing the log-linearized consumption Euler equation and the aggregate resource constraint are equivalent for this model. Furthermore, the use of the constant one is a simple trick which allows us to ensure

TABLE 2: An example of the required and optional data for the prior distribution file.

Model parameter	Status	Initial value	Prior type	Prior parameter 1	Prior parameter 2	Lower bound	Upper bound
tau	estimated	1.87500	gamma	2.000	0.50	1.000	
kappa	estimated	0.15000	gamma	0.200	0.10	0.000	
psi1	estimated	1.45830	gamma	1.500	0.25	0.000	
psi2	estimated	0.37500	gamma	0.500	0.25	0.000	
rhoR	estimated	0.50000	beta	0.500	0.20	0	1
rhoG	estimated	0.84620	beta	0.800	0.10	0	1
rhoZ	estimated	0.70590	beta	0.660	0.15	0	1
rA	estimated	0.50000	gamma	0.500	0.50	0.000	
piA	estimated	6.42860	gamma	7.000	2.00	0.000	
gammaQ	estimated	0.40000	normal	0.400	0.20		
sigmaR	estimated	0.00358	invgamma	0.004	4.00	0.000	
sigmaG	estimated	0.00859	invgamma	0.010	4.00	0.000	
sigmaZ	estimated	0.00447	invgamma	0.005	4.00	0.000	

that the iid shocks are indeed exogenous. Regarding the AiM notation used in the Table, _NOTD refers to *not data*, meaning that AiM treats the variable as an unobserved variable. Similarly, _DTRM means that the variable is deterministic. For more information, see, e.g., Zagaglia (2005, Section 4.1).

The AiM code in Table 1 is also found in the file AnSchorfheideModel.aim located in the sub-directory example\AnSchorfheide. The file can be parsed by AiM and, as mentioned in Section 3.5, this is handled by the function AiMInitialize. Similar aim files exist for the Lubik and Schorfheide (2007a) and the Smets and Wouters (2007) examples in sub-directories of the example directory in YADA.

18.2. *Specification of the Prior Distribution*

The file with the prior distribution data must be given by either a Lotus 1-2-3 spreadsheet (file extension .wk1) or an Excel spreadsheet (extension .xls). The An and Schorfheide example comes with a number of such prior distribution files, e.g., AnSchorfheidePrior.wk1. In addition, if the user wishes to make use of a system prior it needs to be setup as a matlab function, as discussed below in Section 18.2.10; see also Section 4.4 for a more theoretical discussion on system priors.

The prior distribution file should list all the parameters that are going to be estimated. It may also list parameters that are calibrated.[142] The 7 required column headers in this file are given by model parameter, status, initial value, prior type, prior parameter 1, prior parameter 2, and lower bound. The entries under the lower bound header are in fact ignored unless the prior distribution is gamma, inverted gamma, or left truncated normal. Furthermore, all the *headers* are case insensitive in YADA.

YADA also supports two optional headers: upper bound and prior parameter 3. The upper bound header is used for the beta distribution only. When YADA locates this header it will also take the lower bound for beta distributed parameters into account. If the header is missing YADA assumes that any beta distributed parameters have lower bound 0 and upper bound 1. The prior parameter 3 header is used by the Student-*t* distribution and should contain a positive interger, the number of degrees of freedom in (4.23). If this header is missing, then YADA will set the degrees of freedom parameter to unity for the Student-*t* prior.

---

[142] In Section 18.3, we discuss alternative and more flexible ways of specifying additional parameters that YADA needs to know about in order to solve the DSGE model.

18.2.1. *The Model Parameter Header*

The names of all the parameters that need to be estimated should be specified under this header. It is important that the names are exactly the same as in other input files, e.g., the AiM model file (see, Section 18.1). Since Matlab is case sensitive regarding variable and field names, the parameter names are case sensitive.

18.2.2. *The Status Header*

The status header reports if a parameter should be estimated or is to be viewed as calibrated. The valid entries are thus `es(timated)` and `calibrated`. In fact, as long as the status string is not empty, the parameter will be regarded as calibrated unless the first two letters of the status string are `es`. This entry is case insensitive.

18.2.3. *The Initial Value Header*

The initial value must be a scalar and it should be in the support of the prior distribution assigned to the parameter. Natural candidates as an initial value is either the mean or the mode of the prior distribution (if they exist).

18.2.4. *The Prior Type Header*

The prior type header determines the prior distribution of the parameter. The entry should be one of the following: `gamma`, `beta`, `invgamma`, `normal`, `truncnormal`, `uniform`, `student`, `cauchy`, `logistic`, `gumbel`, or `pareto`. This entry is case insensitive.

18.2.5. *The Prior Parameter 1 Header*

The prior parameter 1 header reports the first parameter for the prior distribution. This parameter is assumed to be the mean of the gamma, beta, normal, logistic, and Gumbel distributions, the location paremeter s for the inverted gamma distribution (see equation (4.11)), the location parameter μ for the left truncated normal (see equation (4.20)), the Student-t and Cauchy distributions, the shape parameter a for the Pareto distribution (see equation (4.34)), and the lower bound of the uniform distribution.

18.2.6. *The Prior Parameter 2 Header*

The prior parameter 2 header reports the second parameters for the prior distribution. This parameter is assumed to be the standard deviation of the gamma, beta, normal, logistic, and Gumbel distributions. For the inverted gamma distribution it is the q (degrees of freedom) parameter, for the left truncated normal, the Student-t and the Cauchy the scale paremeter σ, the location parameter b for the Pareto, and the upper bound for the uniform distribution.

18.2.7. *The Lower Bound Header*

The lower bound header is primarily used if the distribution is gamma, inverted gamma, and left truncated normal. For the left truncated normal the lower bound is, as in Section 4.2.7, the c parameter. The entries can for the other prior distributions be either empty or real numbers.

18.2.8. *The Upper Bound Header*

The upper bound header is optional and, when present, is only used by the beta distribution. If the upper bound header is missing or the lower bound is not specified, then the beta prior is assumed to have lower bound 0 and upper bound 1. When the header is present it will be ignored for parameters that do not have a beta prior.

18.2.9. *The Prior Parameter 3 Header*

The prior parameter 3 header is optional and, when present, used by the Student-*t*, the logistic, and the Pareto distributions. In the first case it measures the number of degrees of freedom, in the second the shape parameter, and in the the last case the origin parameter *c*; cf. Section 4.2.13. If this header is either missing or has no value, then YADA assumes that the corresponding Student-*t* prior has 1 degree of freedom, i.e., a Cauchy prior, that the shape parameter is 1 for the logistic, while the origin parameter for the Pareto is zero by default.

18.2.10. *System Prior File*

The system prior file is a matlab function which takes six required input variables and provides one required output variable. It computes the log height of the system prior density, where the latter is denoted by $p(\Phi_\omega|\theta, h)$ in Section 4.4.

The six input variables are given by: `theta`, `thetaPositions`, `ModelParameters`, `AIMData`, `DSGEModel`, and `CurrINI`. These inputs are sufficient to solve the DSGE model, in the event that this is needed. An instructive example is provided in the YADA distribution by the function `SystemPriorFile.m`, located in the subdirectory `example\AnSchorfheide`. YADA often sends the model solution to the system prior file through the `DSGEModel` field `Solution`, thereby making it possible to avoid solving the model too many times. It should be noted that, in principle, the system prior file has access to all YADA functions, such as the distribution functions in the directory `dist`. For example, the system prior file `SystemPriorFile.m` makes use of the inverted Gamma density (`logInvertedGammaPDF`) for the standard deviation of inflation.

Finally, the required output variable is given by the natural logarithm of the height (value) of the system prior density at the parameter value given by `theta`.

18.3. *Defining Additional Parameters*

Additional parameters can optionally be included as input for YADA. These can be parameters that are calibrated but not specified in the prior distribution file. They can also be parameters, such as β in (2.1), that are defined from other parameters, e.g., steady state parameters. YADA allows for input of two different types of additional parameters. The first is a function that only specifies parameters that should be initialized, while the second is a function with parameters that should be updated along with the estimated parameters. The β example concerns the latter type of additional parameters function.

For both types of additional parameters functions, YADA requires that the function takes as input the structure with model parameters, e.g., `ModelParameters` whose fields have the same names as the names specified in the prior distribution file and the AiM model file. As output, the function must also give the structure with model parameters. In the β example, the AiM code uses a parameter `beta`. Since this is a parameter updated when `rA` receives a new value, the function with parameters to update should include a line such as:

$$\text{ModelParameters.beta = 1/(1+(ModelParameters.rA/400)).}$$

See the file `MoreAsParameters.m` in the sub-directory `example\AnSchorfheide` for more details.

There are four names that may not be used for the parameters. They are: `YADA`, `YADAg`, `YADAh`, and `UserVariables`. The first is used internally to store the state equations matrices, F and B_0, through the field names `ModelParameters.YADA.F` and `ModelParameters.YADA.B0`, respectively. This means that these matrices are available inside the measurement equation file (discussed in Section 18.4). The next two are used internally to allow for parameters with the names `g` and `h`, respectively. The last is a reserved name that allows the user to pass on information from the parameter function. That information can be reused by the function itself, such as holding initial values for some numerical problem that the function solves (e.g., steady-state calculations). The field `ModelParameters.UserVariables` is viewed as a structure where the user can select his or her own field names. YADA, for its part, simply ignores the `UserVariables` field when dealing with parameters. YADA stores `ModelParameters`, along with a number of other variables, in a mat-file when it has finished the posterior mode estimation

routine. The user can therefore access data in `ModelParameters.UserVariables` via that file. To find it, simply look in the `mode` directory that the posterior mode estimation routine creates.

Any additional parameters files that you have specified are always checked for internal consistency before executing, e.g., the actual posterior mode estimation functions.

In Section 4.2 a number of prior distributions were discussed where the function with parameters that need to be updated can be utilized to support additional priors. For example, suppose that we have a parameter α whose prior should be a Weibull distribution with scale parameter $a = 3$ and shape parameter $b = 2$; cf. Section 4.2.3. We may then define the auxiliary parameter α_G with prior distribution $G(1, 1)$. The code

```
ModelParameters.alpha = 3*ModelParameters.alphaG^(1/2)
```

in the file with parameters to update ensures that the parameter α has a $W(3, 2)$ prior distribution. Other changes to the prior may, for instance, reflect a mirror image prior by multiplying the value of an auxiliary parameter by -1.

If a dynare model file is used instead of an aim model file for parsing, then it may be necessary to define one parameter per state/model variable. Specifically, if the model has been specified in non-linear form, the `compute_aim_matrices.m` file will require that the `ModelParameters` structure has fields with names that are identical to the state variables. YADA is distributed with an example of this for the An Schorfheide model and which is called `ASNonLinearDynare` in the `example` sub-directory of the YADA distribution.

18.4. *Setting up the Measurement Equation*

The measurement equation is specified as a function in a Matlab m-file. This file takes 7 input arguments. The first is the structure `ModelParameters`, whose fields have names given by the parameter names. Hence, if the model has a parameter called `rA`, then the `ModelParameters` structure has a field `ModelParameters.rA`. If `rA` is specified in the prior distribution file, then this field is automatically generated by YADA. If you need to refer to this parameter in the measurement equation file, then the syntax should follow this example. Note, that `ModelParameters` can take any name you wish in the measurement equation file since the name of the structure is local to the function.

As noted in Section 18.3, the state equation matrices F and B_0 can be accessed via the `ModelParameters` structure in the measurement equation file. Specifically, the following code inside the measurement equation file retrieves the current values for F and B_0

```
F = ModelParameters.YADA.F;
B0 = ModelParameters.YADA.B0;
```

These parameter matrices can, for instance, be useful when, as in Del Negro and Eusepi (2011), observed data on expectations are linked to the corresponding model variables.

A second input argument for this function is the string matrix called, e.g., `StateVarNames`. This matrix contains a name in each row that is equal to the name obtained from parsing the AiM model file. That is, it is equal to the `endog_` output from the `compute_aim_data` function that the AiM parser creates.

The third and fourth input variables for the measurement equation function are the string matrices `VariableNames` and `XVariableNames`. The rows of these matrices provide the names of the observed and the exogenous variables, respectively.

The last three input arguments for the measurement equation function are `n`, `r`, and `k`. These are equal to the dimension variables n, r, and k in (5.1).

The output from this function should be the three matrices A, H, and R; see, equation (5.1). The dimensions of these matrices should be exactly as they are specified in Section 5. That is, A is $k \times n$ if $k > 0$ and an empty matrix otherwise, H is typically an $r \times n$ matrix, while R is an $n \times n$ matrix with measurement error covariances. If there are no measurement errors in your model then $R = 0$.

The measurement equation from the An and Schorfheide model (cf. equation (2.2)), is specified in the file `MeasurementEqFile` in the sub-directory `example\AnSchorfheide`. This file shows how these matrices can be determined from the 7 input arguments to the function as

well as using the `loc` function for locating the position of variables within the `StateVarNames`, `VariableNames`, and `XVariableNames` string matrices. It may be noted that of the 7 input arguments only the first is always likely to be useful since it includes all the parameter values. The other 6 inputs are provided primarily for convenience. When you write your own measurement equation m-file, you have to make sure that all the 7 inputs are accepted. Whether you then make use of them or not in the function is, of course, up to you.

It is possible to let the measurement matrix H be time-varying. YADA handles this case by letting the H matrix exported from the measurement equation file be 3-dimensional, i.e., an $r \times n \times T_H$ matrix, where T_H is large enough for the estimation sample to be covered. YADA assumes that `H(:,:,t)` measures the same time period t as column t in the $n \times T$ matrix `StructureForData.Y.data`; cf. Section 18.5.

It is important to note that if there are many calls to, say, the `loc` function for setting values to the correct entries of the A, H, and R matrices, this may slow down both the posterior mode estimation phase and, even more importantly, when draws are taken from the posterior distribution via the random walk Metropolis algorithm. For smaller models, the additional time occupied by these calls may be negligible, but in larger models this may affect the computation time considerably. Hence, I would suggest that the `loc` function is used only for testing purposes in such cases, and that the entry numbers are hand-coded into the measurement equation file later on.

The measurement equation file is always checked for internal consistency before commencing with, e.g., posterior mode estimation.

18.5. *Reading Observed Data into YADA*

To estimate θ with YADA the data on $y_t = [\Delta Y_t \, \Pi_t \, I_t]'$ and $x_t = 1$ needs to be read from a file. Since the user may wish to transform the raw data prior to defining y_t by, e.g., taking logs, YADA requires that the construction of data is handled in a Matlab m-file.

As an example, consider the data construction file `DataConstFile.m` that is located in the sub-directory `example\AnSchorfheide`. It assumes that there are no inputs for the function. The requirements on its setup concerns the structuring of the output. Specifically, the data construction file should return a structure, named e.g., `StructureForData`. The actual name of the structure is not important as it is a local variable to the function. The fields of this structure, however, have required names and setup. The matrix with observed data should appear as `StructureForData.Y.data`. It should preferably be of dimension $n \times T$ with $n < T$; if not, YADA will take its transpose.

If you need to transform your data prior to using them for estimation, you can always do so in your data construction file. For instance, you may want to take natural logarithms of some of the variables in your data input file, you may wish to rescale some variables, take first differences, remove a linear trend, etc. All Matlab functions located on the `matlabpath` can be used for this purpose. It is important to note, however, that any files you instruct YADA to read data from should be specified in the data construction file with their full path. The reason is that YADA copies all the Matlab m-files that you specify on the DSGE Data tab (see Figure 9) to the directory `tmp` and executes them from there. That way, YADA avoids having to deal with temporary changes to the path. At the same time, a data file located in, e.g., the same directory as your data construction file will not be copied to the `tmp` directory. Hence, a command like

$$\texttt{wk1read(['data\textbackslash AnSchorfheideData.wk1']),}$$

will not work unless you manually create a directory `data` below YADA's working directory and copy the file `AnSchorfheideData.wk1` to this directory.

The working directory for YADA is always given by `pwd`, i.e., the directory where `YADA.m` is located. Hence, if you store your data file in a sub-directory to YADA's working directory, e.g., `example\AnSchorfheide\data` you can use the command `pwd` to set the root for the path where your data file is located. In this case,

$$\texttt{wk1read([pwd '\textbackslash example\textbackslash AnSchorfheide\textbackslash data\textbackslash AnSchorfheideData.wk1']).}$$

When you exit YADA, all files found in the `tmp` directory are automatically deleted.

The names of the observed variables should appear in the field `StructureForData.Y.names`. It should be given as a cell array with n string elements. In `DataConstFile.m`, $n = 3$ while

$$\text{StructureForData.Y.names = \{'YGR' 'INFL' 'INT'\}.}$$

The data for any exogenous variables should be given by `StructureForData.X.data`, a matrix of dimension $k \times T$, e.g., a vector with ones for a constant term. Similarly, the cell array `StructureForData.X.names` provides the names of these variables, e.g., being given by `{'const'}`. If the model has no exogenous variables, then these two fields should be empty.

Given that the model has exogenous variables, it is possible to add extra data on these variables to the entry `StructureForData.X.extradata`. This is an optional entry that if used should either be an empty matrix or a $k \times T_h$ matrix. YADA views this data on the exogenous variables as having been observed after the data in `StructureForData.X.data`. Hence, the extra data can be used in, for instance, out-of-sample forecasting exercises where it will be regarded as observations $T + 1$ until $T + T_h$.

Next, the field `StructureForData.sample` should contain a 4 dimensional vector with entries giving the start year, start period, end year and end period. For instance,

$$\text{StructureForData.sample = [1980 1 2004 4].}$$

This sample data refers to the data in the matrix `StructureForData.Y.data` for the observed variables and the matrix `StructureForData.X.data` for the exogenous variables. The sample used for estimation can be changed on the Settings tab in YADA.

YADA requires that the data frequency is specified as a string. Valid string entries are `quarterly`, `monthly`, and `annual`. The first letter of these strings are also permitted. The name of the field for the data frequency is simply `StructureForData.frequency`.

An optional field to `StructureForData.Y` with data is called actuals. This vector structure has the subfields `data` and `title`. These two fields provide the data and a data name for alternative actuals of the observed variables. These actuals may be used when computing the predictive likelihood and are thus relevant in the context of forecasting with real-time data; see, e.g., Croushore and Stark (2001) and Croushore (2011a,b).

Suppose that two alternative datasets of the observed variables have been loaded by the data construction file with *at least* as many observations as in the original data matrix. We may denote these two matrices as `DataMatrix1` and `DataMatrix2`, with dimensions $n \times T_1$ and $n \times T_2$, respectively, where $T_1, T_2 \geq T$, the number of observations in `StructureForData.Y.data`. These data may then be added as actuals for a forecasting exercise by the following commands

```
StructureForData.Y.actuals(1).data = DataMatrix1;
StructureForData.Y.actuals(1).title = 'First release';
StructureForData.Y.actuals(2).data = DataMatrix2;
StructureForData.Y.actuals(2).title = 'Annual revision';
```

The first set of actuals has here been given the name "first release", which indicates that in this example these data are the first release numbers of the observed variables. The second set of actuals has instead been given the name "annual revision", suggesting that they contain the numbers of the observed variables for the sample dates which were revised one year after the first release. The original data can, of course, also be used as actuals when forecasting and are in fact the default data for such exercises. Finally, notice that the first observation in both `DataMatrix1` and `DataMatrix2` is assumed to be taken from the same time period as the first observation in `StructureForData.Y.data`.

18.5.1. *Transformations of the Data*

It is possible to transform the data on the observed variables (`StructureForData.Y.data`) through YADA. Such transformations, however, will not be applied to the data for estimation purposes. Rather, in certain situations, such as forecasting, it may be interesting to transform the data based on user defined functions. The information YADA needs for such data transformations is assumed to be provided via the data construction file, through the sub-field `StructureForData.Y.transformation`. This field is not mandatory, but if it is missing then YADA will not be able to transform the observed variables.

The transformations are performed on a variable-by-variable basis. For this reason YADA assumes that the name of an observed variable provides a sub-field in the structure. For instance, in the `DataConstFile.m` there is a field `StructureForData.Y.transformation.YGR` for GDP growth. For each variable specific sub-field there are 6 sub-sub-fields that are needed. These are: `fcn`, `partial`, `annualizefcn`, `annualizepartial`, `initial`, and `x`. In addition, there are 3 sub-sub-fields for inverting the transformation. These are: `invertfcn`, `invertinitial`, and `invertx`. Finally, YADA also has 5 sub-sub-fields for dealing with exporting of data: `exportfcn`, `exportinitial`, `exportx`, `exporttitle`, and `exportname`.

The fields `fcn` and `annualizefcn` hold string vectors for transforming the data. The `inline` function is used by YADA and for the string vector to be operational it is therefore required that it holds valid Matlab syntax. The field `fcn` holds the general transformation function, while `annualizefcn`, as the name suggests, holds a particular function that is used when the transformation is assumed to provide an annualization of the data.

The fields `partial` and `annualizepartial` hold the first order partial derivatives of the functions in the `fcn` and `annualizefcn` string vectors with respect to the variable that should be transformed *times* a part of the variable in question. That is, this string vector should include the full first order Taylor expansion term for the variable to be transformed $((\partial f(x)/\partial x)x_j$, where $x = x_1 + \ldots + x_n$ and $j = 1, \ldots, n$). Since the partial and the variable are both vectors, the element-by-element product operator (`.*`) should be used. If the partial is a constant, then the partial should take into account that x is the variable to be transformed and x_j is a part of this variable. The `partial` string vector must have exactly as many variables as the `fcn` string vector *plus* one additional variable, which must be the last variable in the string vector. Apart from the last term all other variables in `fcn` and `partial` must appear in the same order. This means that if the `fcn` string is `'4*(S-MeanS)'`, then `partial` should be, e.g., `'(4-(0*(S-MeanS))).*PartS'`. The same rule holds for the `annualizepartial` string vector.

The field `initial` holds a vector of initial values for the variable that is to be transformed. These values will be prepended to the vector of data on the variable prior to it being transformed. The dating of the initial values is assumed to be the periods just before to the start of the full sample for the observed variables.

The field `x` holds a matrix with data on any additional variables required by the transformation. The dimension of this matrix should be $d_x \times T_x$, where d_x is the number of additional variables. Since it is possible that certain transformation functions require different dimensions of the various additional variables, YADA will look for `NaN` entries at the beginning of each row of `x` and remove such entries on a variable-by-variable basis before executing the transformation.

To illustrate these features, suppose that the transformation function for a variable `YGR` is:

cumsum(YGR)+0.2*TREND

and that one initial value is given for `YGR`. Data for the variable `TREND` is stored in `x`. Since one observation is prepended to the vector of data for `YGR`, the dimension of the data for `TREND` must match the dimension for `YGR`, i.e., $T_x = T + 1$. This means that `x` is a $1 \times (T + 1)$ vector with data on `TREND`. For instance, this may be the vector $[0 \ 1 \ \cdots \ T]$.

Suppose we instead consider the following transformation function:

YGR-diff(N)+0.2*TREND

and that the variable `N` is located in the first row of `x` and `TREND` in the second row. In this case, `N` needs to have one more element than `YGR` once initial values for latter have been taken into account. At the same time `TREND` should have the same number of elements as `YGR`. By letting the first element in the second row of `x` be `NaN` (while the first element of the first row is a real number), this transformation can be achieved.

Since the `inline` function in Matlab, when executed with one input argument, creates a function with an input ordering of the variables that follows the order in which the variables appear in the string vector provided to `inline`, YADA requires that the variable to be transformed appears first in the function string, and all the additional variables thereafter. The ordering of the additional variables is assumed to match the ordering of these variables in the matrix `x`.

Assuming that the transformation has been performed successfully, YADA checks if the dimension of the transformed variable is greater than that of the original variable. Should this be the case, data at the beginning of the variable created by the transformation are removed such that the dimensions match. In the event that the dimension of the variable created is smaller than the original, then YADA assumes that the transformation used up data at the beginning from a time perspective. This would, for example, be the case if the transformation uses the `diff` function and no initial values are made available.

The field `invertfcn` holds a string that describes how the function in the field `fcn` should be inverted. In analogy with the `fcn` field, the fields `invertinitial` and `invertx` hold initial values and data on all additional variables that are needed by the inversion function. Similarly, the function for exporting data is stored in the field `exportfcn`. Likewise, the information about initial values and additional variables required for the export transformation are located in the fields `exportinitial` and `exportx`. Finally, the fields `exporttitle` and `exportname` hold string vectors that makes it possible to use a different name of the variable in the file with exported data. The `exporttitle` string is written to the line above the `exportname` string.

The following example for GDP growth is found in the file `DataConstFile.m`:

```
StructureForData.Y.transformation.YGR.fcn = '100*(exp(YGR/100)-1)';
StructureForData.Y.transformation.YGR.partial = 'exp(YGR/100).*PartYGR';
StructureForData.Y.transformation.YGR.annualizefcn = ...
    '100*(exp((1/100)*(YGR(4:length(YGR))+YGR(3:length(YGR)-1)+...
    YGR(2:length(YGR)-2)+YGR(1:length(YGR)-3)))-1)';
StructureForData.Y.transformation.YGR.annualizepartial = ...
    'exp((1/100)*(YGR(4:length(YGR))+YGR(3:length(YGR)-1)+...
    YGR(2:length(YGR)-2)+YGR(1:length(YGR)-3))).*...
    (PartYGR(4:length(PartYGR))+PartYGR(3:length(PartYGR)-1)+...
    PartYGR(2:length(PartYGR)-2)+PartYGR(1:length(PartYGR)-3))';
StructureForData.Y.transformation.YGR.initial = [];
StructureForData.Y.transformation.YGR.x = [];
StructureForData.Y.transformation.YGR.invertfcn = ...
    '100*log(1+(YGR/100))';
StructureForData.Y.transformation.YGR.invertinitial = [];
StructureForData.Y.transformation.YGR.invertx = [];
StructureForData.Y.transformation.YGR.exportfcn = ...
    '100*(exp(cumsum(YGR)/100))';
StructureForData.Y.transformation.YGR.exportinitial = 0;
StructureForData.Y.transformation.YGR.exportx = [];
StructureForData.Y.transformation.YGR.exporttitle = 'Real GDP';
StructureForData.Y.transformation.YGR.exportname = 'DY';
```

Since YGR is the log first difference of GDP, the general function in the `fcn` field calculates the (quarterly) growth rate of GDP. The annualization function similarly provides the annual growth rate of GDP, while no initial data are supplied and the computations do not need any additional variables. The field `partial` (`annualizepartial`) is the first order partial derivative of the `fcn` (`annualizefcn`) function times a part of the variable that is transformed. The function is used when the transformation of the variable is applied to a linear decomposition of the variable, such as the observed variable decomposition in Section 11.8.

It is also possible to transform the observed variables via linear combinations of the transformations performed by the above setup. With three observed variables the following defines such a transformation matrix:

```
StructureForData.Y.TransMatrix = [1 0 0;0 1 0;0 -1 1];
```

The linear combinations of the individually transformed variables is therefore the first, the second, and the third minus the second (the real rate). Notice that the `TransMatrix` matrix will only be applied to the vector of observed variables *after* the individual transformation functions under the field `fcn` have been utilized. This implies that linear transformations of the observed

variables can only be performed when the transformation functions have been properly setup for *all* observed variables.

The field `invertfcn` inverts the calculation in `fcn`, while the field `exportfcn` gives the expression for calculating the levels data for `YGR` based on a constant initial value for the variable. The variables `INFL` and `INT` have their own transformation functions; see `DataConstFile.m` for details.

18.5.2. *Levels or First Differences*

To inform YADA about which variables appear in levels (like the interest rate) and which appear in first differences (output and inflation), the field `levels` should contain a vector whose elements are either 0 (first difference) or 1 (level). In our example this means that

$$\texttt{StructureForData.Y.levels = [0 0 1]}.$$

This information makes it possible for YADA to compute, e.g., the levels responses in all observed variables of a certain economic shock. If the levels field is missing from the data construction file, then YADA displays a message box to remind you. The file is, however, regarded as valid, with all observed variables in levels. Once you add the field to the data construction file, YADA will stop nagging you about the missing information.

18.5.3. *Simple Annualization*

Furthermore, to tell YADA how to "annualize" observed variables, the field `annual` should contain a vector whose elements are either 0 (do not annualize/is already annualized) or 1 (annualize). YADA annualizes a variable by adding to the current value the previous 3 (11) observations for quarterly (monthly) data. For instance, the `YGR` variable measures quarterly logged GDP (per capita) growth. Summing this variable over 4 consecutive quarters thus gives the annual logged GDP (per capita) growth. The inflation and interest rate variable are already assumed to be measured in annual terms. This means that we set:

$$\texttt{StructureForData.Y.annual = [1 0 0]}.$$

How the data frequency is specified is provided below. It may be noted that the `annual` field is optional, with the default being a zero vector, and that YADA does not display a nag screen when this field is missing.

In some situations you may wish that the annualized data should be multiplied by a constant. For example, the inflation equation in (2.2) "annualizes" quarter-to-quarter inflation by multiplying the quarterly price changes with 4. The annualized inflation series can therefore be calculated by adding four consecutive quarters of Π_t and dividing the sum with 4. For YADA to compute such an annualized inflation series field `annualscale` should be specified. Here, we may set

$$\texttt{StructureForData.Y.annualscale = [1 0.25 1]}.$$

Notice that YADA will only use elements of this vector on the variables that you allow YADA to annualize. This means that the second element of the vector `StructureForData.Y.annual` must be unity before YADA will apply the scaling constant 0.25 to inflation. Moreover, the scaling vector is optional and is by default equal to a unit vector.

18.5.4. *Zero Lower Bound*

In order to solve a log-linearized DSGE model subject to the zero lower bound, YADA uses the foward-back shooting algorithm developed by Hebden et al. (2011) and which is discussed in some detail in Section 3.4. To make use of this feature in YADA, the data construction file should hold two pieces of information. First, YADA needs to to be told which among the observed variable is the monetary policy rate. Second, optionally it also needs to have data on the zero lower bound. This information is optional since by default YADA will assume that the zero lower bound is indeed zero. However, it may well be that the lower bound is considered to be a positive or even a negative value. Moreover, it may not be the same for each time period. Although this may be more flexible than needed, YADA supports such lower bounds since it doesn not complicate the problem of solving the model in principle.

The field `Y` has two fields that contain the zero lower bound data (`ZLBdata`) and the position of the monetary policy rate among the observed variables (`policyrate`). For example,

```
StructureForData.Y.ZLBdata = 0.1;
StructureForData.Y.policyrate = 3;
```

means that the zero lower bound is set to 0.1, while the third variable in the field `Y.names` is the monetary policy rate. Whenever the zero lower bound is time varying, the `ZLBdata` field needs to be a vector where each element is measured at the same point in time as the corresponding row of the field `Y.data`. Note also that if this vector is too short, then YADA will extend it to be as long as needed using the value of the last element.

It is possible that a DSGE model has more than one interest rate, such as a short-term monetary policy rate and a long-term bond yield. At this stage, YADA does not allow for a zero lower bound on more than one interest rate.

18.5.5. *DSGE-VAR*

It is possible to estimate a DSGE-VAR model in YADA. Such models are discussed in some detail in Section 15. The data construction file can include one field which is used for estimating DSGE-VARs. Namely, the field `Lambda`, a vector with the *possible* values for the λ hyperparameter to use. YADA will use all values given in this field that satisfy the lower bound constraint on λ, i.e., all values such that $\lambda \geq \lambda_l \equiv (n(p+1)+k)/T$, where n is the number of observed (endogenous) variables, p is the lag order of the DSGE-VAR, k is the number of exogenous variables, and T is the effective sample size when estimating the model.

The `Lambda` field is optional and, when missing, YADA sets the vector equal to:

$$\text{StructureForData.Lambda} = [0.25\ 0.5\ 0.75\ 1\ 5\ \text{Inf}],$$

where the last entry is $\lambda = \infty$. In case the data construction file specifies a `Lambda` which does not have the `Inf` entry, then YADA will automatically add this entry to the vector. When applying the values of the `Lambda` vector to DSGE-VARs, YADA ensures that the sample size times the value is rounded to the closest integer.

18.5.6. *Bayesian VAR*

If you wish to estimate a Bayesian VAR model with YADA you may include one additional field in the `StructureForData.Y` structure. This field, denoted by `BVAR`, should provide the variable positions in the matrix `StructureForData.Y.data`. These variable numbers may be ordered in any way you wish and all observed variables need not appear. In the An and Schorfheide example we let:

$$\text{StructureForData.Y.BVAR} = [1\ 2\ 3].$$

If the field `BVAR` is missing, then YADA automatically assumes that all observed variables are to be included in any Bayesian VAR analysis you wish to perform. Hence, the field `BVAR` is also optional and YADA does not display any nag screen to inform you about this. Please note that Section 14 discussed in some detail the Bayesian VAR models that you can estimate with YADA.

For Bayesian VAR analysis you may also wish to specify which exogenous variables should be included. You do this in an analogous way to how observed variables are made available as endogenous variables in the VAR model. That is, you let `StructureForData.X.BVAR` be equal to a vector with integer values corresponding to the variable position in the matrix with exogenous variables, i.e., `StructureForData.X.data`. Also, you need not include all exogenous variables that the estimated DSGE model has. This feature is optional and by default YADA will include all exogenous variables in the Bayesian VAR model when you decide to estimate such a model.

18.5.7. *Conditional Forecasting Data*

If you wish to perform conditional forecasting you need to set up the entries needed for conditioning. All fields discussed for conditional forecasting are optional, but all are required if you wish to perform such forecasts. The mathematics behind conditional forecasting is explained in Section 12.2. Here I shall focus on definitions that can be made in the data construction file.

First, the variables that you wish to condition on are located in the field Z in the structure StructureForData. As in the case of the observed variables and the exogenous variables, the subfields data and names holds the data and the names of the conditioning information. It is important to note that the data subfield should contain an $m \times T_z$ matrix with T_z being the number of available observations of the conditioning information. It is assumed that the first column of this matrix is measured at the same point in time as the first column of the matrix with the observed variables. Since the latter has T observation, the value of T_z should be greater than T if you plan to use all the observed data for estimation and then perform conditional forecasting for periods $T + 1$ and onwards until T_z. Furthermore, if Z has some NaN values, then the corresponding conditioning assumption is skipped by YADA. The subfield names should hold a cell array with length equal to m, the number of variables to condition on. Note that you may select a subset of these variables through the YADA GUI.

Second, the field Z has two subfields that contain matrices K1 and K2 that are needed to map the observed variables into the conditioning information; cf. equation (12.6). The first matrix is required to be of full rank m, while the second can be a zero matrix as well as being 3D, where the third dimension covers the j-dimension for K_{2j}. Their common first and second dimension should be $n \times m$, where n as before is the number of observed variables and m is the number of conditioning variables in StructureForData.Z.data.

Since the mapping from the observed variables to the conditioning variables may require an initial value, YADA also needs data in the field U in the structure StructureForData. This field should contain the subfield data that provides an $m \times T_u$ matrix with initial conditions in the mapping; cf. (12.6) for details. Again, YADA assumes that the first observation in this matrix comes from the same time period as the first observation in the matrix with observed variables. This initial conditions data can be used provided that T_u is at least equal to the last period used for estimation of the parameters in the DSGE model. Finally, the conditioning data can also be linked with transformation function and the logic is the same as in the case of the transformation functions for the observed variable scenario data, i.e., through the field StructureForData.Z.transformation.

Apart from conditioning assumptions that are mapped into the observed variables, YADA also allows for conditioning assumptions that restrict the paths for linear combinations of the state variables over the forecast period. The field zeta covers the information needed to forecast subject to such assumptions. The data for the state variable assumptions are given by the $q_z \times T_z$ matrix StructureForData.zeta.data, while the names of these assumptions are given by the cell array of strings StructureForData.zeta.names. The zeta field also has the subfield K3 which is an $r \times q_z$ matrix which maps the current value of the r state variables into the current value of the q_z state variable assumptions; see Section 12.4 for details.

18.5.8. *Percentiles for Distributions*

It is possible to control the percentiles that are used for plotting confidence bands for certain distributions. This is currently handled by StructureForData.percentiles. This optional entry should be a vector with integer values greater than 0 and less than 100. If there are at least 2 valid entries YADA will make use of them. For instance, we may write

$$\text{StructureForData.percentiles} = [85\ 50\ 15\ 5\ 95].$$

Unless the values are already expressed in ascending order YADA will sort them. Moreover, if the vector has an odd number of entries then the middle entry is ignored. In the above example we thus have that YADA would treat the 5 dimensional vector as equal to

$$\text{StructureForData.percentiles} = [5\ 15\ 85\ 95].$$

YADA then assumes that the first and the last element may be used to construct the outer confidence band, while the second and the third are used for an inner confidence band.

Abramowitz, M. and Stegun, I. A. (1964), *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, Dover Publications, New York.

Adolfson, M., Anderson, M. K., Lindé, J., Villani, M., and Vredin, A. (2007a), "Modern Forecasting Models in Action: Improving Macroeconomic Analyses at Central Banks," *International Journal of Central Banking*, 3(4), 111–144.

Adolfson, M., Laséen, S., Lindé, J., and Svensson, L. E. O. (2008a), "Optimal Monetary Policy in an Operational Medium-Sized DSGE Model," Sveriges Riksbank Working Paper Series No. 225.

Adolfson, M., Laséen, S., Lindé, J., and Svensson, L. E. O. (2011), "Optimal Monetary Policy in an Operational Medium-Sized DSGE Model," *Journal of Money, Credit and Banking*, 43(7), 1287–1331.

Adolfson, M., Laséen, S., Lindé, J., and Villani, M. (2005), "Are Constant Interest Rate Forecasts Modest Policy Interventions? Evidence from a Dynamic Open-Economy Model," *International Finance*, 8(3), 509–544.

Adolfson, M., Laséen, S., Lindé, J., and Villani, M. (2007b), "Bayesian Estimation of an Open Economy DSGE Model with Incomplete Pass-Through," *Journal of International Economics*, 72(2), 481–511.

Adolfson, M., Laséen, S., Lindé, J., and Villani, M. (2008b), "Evaluating an Estimated New Keynesian Small Open Economy Model," *Journal of Economic Dynamics and Control*, 32(8), 2690–2721.

Adolfson, M., Lindé, J., and Villani, M. (2007c), "Bayesian Analysis of DSGE Models - Some Comments," *Econometric Reviews*, 26(2), 173–185.

Adolfson, M., Lindé, J., and Villani, M. (2007d), "Forecasting Performance of an Open Economy DSGE Model," *Econometric Reviews*, 26(2), 289–328.

Aknouche, A. and Hamdi, F. (2007), "Periodic Chandrasekhar Recursions," *arXiv:0711.3857v1*.

Amisano, G. and Geweke, J. (2017), "Prediction Using Several Macroeconomic Models," *The Review of Economics and Statistics*, 99(5), 912–925.

An, S. and Schorfheide, F. (2007), "Bayesian Analysis of DSGE Models," *Econometric Reviews*, 26(2), 113–172, with discussion, p. 173–219.

Anderson, B. D. O. and Moore, J. M. (1979), *Optimal Filtering*, Prentice-Hall, Englewook Cliffs.

Anderson, E. W., Hansen, L. P., McGrattan, E. R., and Sargent, T. J. (1996), "Mechanics of Forming and Estimating Dynamic Linear Economies," in H. M. Amman, D. A. Kendrick, and J. Rust (Editors), *Handbook of Computational Economics*, 171–252, Elsevier.

Anderson, G. and Moore, G. (1983), "An Efficient Procedure for Solving Linear Prefect Foresight Models," Mimeo, Board of Governors of the Federal Reserve System.

Anderson, G. and Moore, G. (1985), "A Linear Algebraic Procedure for Solving Linear Perfect Foresight Models," *Economics Letters*, 17(3), 247–252.

Anderson, G. S. (1999), "The Anderson-Moore Algorithm: A MATLAB Implementation," Mimeo, Board of Governors of the Federal Reserve System.

Anderson, G. S. (2008), "Solving Linear Rational Expectations Models: A Horse Race," *Computational Economics*, 31(2), 95–113.

Anderson, G. S. (2010), "A Reliable and Computationally Efficient Algorithm for Imposing the Saddle Point Property in Dynamic Models," *Journal of Economic Dynamics and Control*, 34(3), 472–489.

Andrés, J., López-Salido, J. D., and Vallés, J. (2006), "Money in an Estimated Business Cycle Model of the Euro Area," *The Economic Journal*, 116(511), 457–477.

Andrews, D. W. K. (1991), "Heteroskedasticity and Autocorrelation Consistent Covariance Matrix Estimation," *Econometrica*, 59(3), 817–858.

Andrle, M. (2010a), "A Note on Identification Patterns in DSGE Models," ECB Working Paper Series No. 1235.

Andrle, M. (2010b), "Correlation Decompositions," Manuscript, IMF.

Andrle, M. and Beneš, J. (2013), "System Priors: Formulating Priors about DSGE Models' Properties," IMF Working Paper No. 13/257.

Ansley, C. F. and Kohn, R. (1982), "A Geometrical Derivation of the Fixed Smoothing Algorithm," *Biometrika*, 69(2), 486–487.

Ansley, C. F. and Kohn, R. (1985), "Estimation, Filtering, and Smoothing in State Space Models with Incompletely Specified Initial Conditions," *The Annals of Statistics*, 13(4), 1286–1316.

Ansley, C. F. and Kohn, R. (1990), "Filtering and Smoothing in State Space Models with Partially Diffuse Initial Conditions," *Journal of Time Series Analysis*, 11(4), 275–293.

Arnold, III., W. F. and Laub, A. J. (1984), "Generalized Eigenproblem Algorithms and Software for Algebraic Riccati Equations," *Proceedings of the IEEE*, 72(12), 1746–1754.

Aruoba, S. B. and Schorfheide, F. (2011), "Sticky Prices versus Monetary Frictions: An Estimation of Policy Trade-offs," *American Economic Journal: Macroeconomics*, 3(1), 60–90.

Ascari, G. and Ropele, T. (2012a), "Disinflation in a DSGE Perspective: Sacrifice Ratio or Welfare Gain Ratio?" *Journal of Economic Dynamics and Control*, 36(2), 169–182.

Ascari, G. and Ropele, T. (2012b), "Sacrifice Ratio in a Medium-Scale New Keynesian Model," *Journal of Money, Credit and Banking*, 44(2–3), 457–467.

Baştürk, N., Grassi, S., Hoogerheide, L., Opschoor, A., and van Dijk, H. K. (2017), "The R Package MitISEM: Efficient and Robust Simulation Procedures for Bayesian Inference," *Journal of Statistical Software*, 79(1), 1–40.

Baştürk, N., Grassi, S., Hoogerheide, L., and van Dijk, H. K. (2016), "Parallelization Experience with Four Canonical Econometric Models using ParMitISEM," *Econometrics*, 4(1), 11.

Balakrishnan, N. and Leung, M. Y. (1988), "Order Statistics from the Type I Generalized Logistic Distribution," *Communications in Statistics - Simulation and Computation*, 17(1), 25–50.

Bartlett, M. S. (1957), "A Comment on D. V. Lindley's Statistical Paradox," *Biometrika*, 44(3–4), 533–534.

Bauwens, L., Lubrano, M., and Richard, J. F. (1999), *Bayesian Inference in Dynamic Econometric Models*, Oxford University Press, Oxford.

Berger, J. O. (1985), *Statistical Decision Theory and Bayesian Analysis*, Springer-Verlag, New York, 2nd edition.

Berkowitz, J. (2001), "Testing Density Forecasts, with Applications to Risk Management," *Journal of Business & Economic Statistics*, 19(4), 465–474.

Bernanke, B. S., Gertler, M., and Gilchrist, S. (1999), "The Financial Accelerator in a Quantitative Business Cycle Framework," in J. B. Taylor and M. Woodford (Editors), *Handbook of Macroeconomics*, volume 1C, 1341–1393, North Holland, Amsterdam.

Bernardo, J. M. (1976), "Psi (Digamma) Function," *Applied Statistics*, 25(3), 315–317, algorithm AS 103.

Bernardo, J. M. (1979), "Expected Information as Expected Utility," *The Annals of Statistics*, 7(3), 686–690.

Bernardo, J. M. and Smith, A. F. M. (2000), *Bayesian Theory*, John Wiley, Chichester.

Beyer, A. and Farmer, R. E. A. (2004), "On the Indeterminacy of New-Keynesian Economics," ECB Working Paper Series No. 323.

Beyer, A. and Farmer, R. E. A. (2007), "Testing for Indeterminacy: An Application to U.S. Monetary Policy: Comment," *American Economic Review*, 97(1), 524–529.

Blanchard, O. J. and Kahn, C. M. (1980), "The Solution of Linear Difference Models under Rational Expectations," *Econometrica*, 48(5), 1305–1311.

Bonaldi, J. P. (2010), "Identification Problems in the Solution of Linearized DSGE Models," Borradores de Economia 593, Banco de la Republica de Colombia.

Box, G. E. P. (1980), "Sampling and Bayes' Inference in Scientific Modelling and Robustness," *Journal of the Royal Statistical Society Series A*, 143(4), 383–430.

Brooks, S. P. and Gelman, A. (1998), "General Methods for Monitoring Convergence of Iterative Simulations," *Journal of Computational and Graphical Statistics*, 7(4), 434–455.

Buiter, W. B. (1982), "Predetermined and Non-Predetermined Variables In Rational Expectations Models," *Economics Letters*, 10(1–2), 49–54.

Burmeister, E. (1980), "On Some Conceptual Issues in Rational Expectations Modeling," *Journal of Money, Credit and Banking*, 12(4, Part 2), 800–816.

Cagan, P. (1965), *Determinants and Effects of Changes in the Stock of Money 1875–1960*, National Bureau of Economic Research, Cambridge, MA.

Calvo, G. A. (1983), "Staggered Prices in a Utility-Maximizing Framework," *Journal of Monetary Economics*, 12(3), 383–398.

Canova, F. and Sala, L. (2009), "Back to Square One: Identification Issues in DSGE Models," *Journal of Monetary Economics*, 56(4), 431–449.

Carpenter, J., Clifford, P., and Fearnhead, P. (1999), "An Improved Particle Filter for Non-linear Prolems," *IEE Proceedings—Radar, Sonar and Navigation*, 146(1), 2–7.

Carter, C. K. and Kohn, R. (1994), "On Gibbs Sampling for State Space Models," *Biometrika*, 81(3), 541–553.

Casella, G. and George, E. I. (1992), "Explaining the Gibbs Sampler," *The American Statistician*, 46(3), 167–174.

Chan, J. C. C. and Eisenstat, E. (2015), "Marginal Likelihood Estimation with the Cross-Entropy Method," *Econometric Reviews*, 34(3), 256–285.

Chari, V. V., Kehoe, P. J., and McGrattan, E. R. (2009), "New Keynesian Models: Not Yet Useful for policy Analysis," *American Economic Journal: Macroeconomics*, 1(1), 242–266.

Chen, M.-H. and Shao, Q.-M. (1999), "Monte Carlo Estimation of Bayesian Credible and HPD Intervals," *Journal of Computational and Graphical Statistics*, 8(1), 69–92.

Chib, S. (1995), "Marginal Likelihood from the Gibbs Output," *Journal of the American Statistical Association*, 90(432), 1313–1321.

Chib, S. and Greenberg, E. (1995), "Understanding the Metropolis-Hastings Algorithm," *The American Statistician*, 49(4), 327–335.

Chib, S. and Jeliazkov, I. (2001), "Marginal Likelihood from the Metropolis-Hastings Output," *Journal of the American Statistical Association*, 96(453), 270–281.

Chib, S. and Ramamurthy, S. (2010), "Tailored Randomized Block MCMC Methods with Applications to DSGE Models," *Journal of Econometrics*, 155(1), 19–38.

Chopin, N. (2002), "A Sequential Particle Filter Method for Static Models," *Biometrika*, 89(3), 539–551.

Chopin, N. (2004), "Central Limit Theorem for Sequential Monte Carlo Methods and its Application to Bayesian Inference," *The Annals of Statistics*, 32(6), 2385–2411.

Christiano, L. J. (2002), "Solving Dynamic Equilibrium Models by a Method of Undetermined Coefficients," *Computational Economics*, 20(1–2), 21–55.

Christiano, L. J., Eichenbaum, M., and Evans, C. (2005), "Nominal Rigidities and the Dynamic Effects of a Shock to Monetary Policy," *Journal of Political Economy*, 113(1), 1–45.

Christiano, L. J., Motto, R., and Rostagno, M. (2003), "The Great Depression and the Friedman-Schwartz Hypothesis," *Journal of Money, Credit and Banking*, 35(6), 1119–1197.

Christiano, L. J., Motto, R., and Rostagno, M. (2010), "Financial Factors in Economic Fluctuations," ECB Working Paper Series No. 1192.

Christiano, L. J., Trabandt, M., and Walentin, K. (2011), "Introducing Financial Frictions and Unemployment into a Small Open Economy Model," *Journal of Economic Dynamics and Control*, 35(12), 1999–2041.

Christiano, L. J. and Vigfusson, R. J. (2003), "Maximum Likelihood in the Frequency Domain: The Importance of Time-to-Plan," *Journal of Monetary Economics*, 50(4), 789–815.

Christoffel, K., Coenen, G., and Warne, A. (2008), "The New Area-Wide Model of the Euro Area: A Micro-Founded Open-Economy Model for Forecasting and Policy Analysis," ECB Working Paper Series No. 944.

Christoffel, K., Coenen, G., and Warne, A. (2011), "Forecasting with DSGE Models," in M. P. Clements and D. F. Hendry (Editors), *The Oxford Handbook of Economic Forecasting*, 89–127, Oxford University Press, New York.

Clarida, R., Galí, J., and Gertler, M. (2000), "Monetary Policy Rules and Macroeconomic Stability: Evidence and Some Theory," *Quartely Journal of Economics*, 115(1), 147–180.

Coenen, G. and Warne, A. (2014), "Risks to Price Stability, the Zero Lower Bound and Forward Guidance: A Real-Time Assessment," *International Journal of Central Banking*, 10(2), 7–54.

Connor, R. J. and Mosimann, J. E. (1969), "Concepts of Independence for Proportions with a Generalization of the Dirichlet Distribution," *Journal of the American Statistical Association*, 64(325), 194–206.

Consolo, A., Favero, C. A., and Paccagnini, A. (2009), "On the Statistical Identification of DSGE Models," *Journal of Econometrics*, 150(1), 99–115.

Cowles, M. K. and Carlin, B. P. (1996), "Markov Chain Monte Carlo Convergence Diagnostics: A Comparative Review," *Journal of the American Statistical Association*, 91(434), 883–904.

Creal, D. (2007), "Sequential Monte Carlo Samplers for Bayesian DSGE Models," Unpublished Manuscript, Vrije Universitiet Amsterdam.

Creal, D. (2012), "A Survey of Sequential Monte Carlo Methods for Economics and Finance," *Econometric Reviews*, 31(3), 245–296.

Croushore, D. (2011a), "Forecasting with Real-Time Data Vintages," in M. P. Clements and D. F. Hendry (Editors), *The Oxford Handbook of Economic Forecasting*, 247–267, Oxford University Press, New York.

Croushore, D. (2011b), "Frontiers of Real-Time Data Analysis," *Journal of Economic Literature*, 49(1), 72–100.

Croushore, D. and Stark, T. (2001), "A Real-Time Data Set for Macroeconomists," *Journal of Econometrics*, 105(1), 111–130.

Dawid, A. P. (1984), "Statistical Theory: The Prequential Approach," *Journal of the Royal Statistical Society, Series A*, 147(2), 278–292.

De Graeve, F. (2008), "The External Finance Premium and the Macroeconomy: US Post-WWII Evidence," *Journal of Economic Dynamics and Control*, 32(11), 3415–3440.

De Jong, P. (1988), "A Cross-Validation Filter for Time Series Models," *Biometrika*, 75(3), 594–600.

De Jong, P. (1989), "Smoothing and Interpolation with the State-Space Model," *Journal of the American Statistical Association*, 84(408), 1085–1088.

De Jong, P. (1991), "The Diffuse Kalman Filter," *The Annals of Statistics*, 19(2), 1073–1083.

De Jong, P. and Chu-Chun-Lin, S. (2003), "Smoothing with an Unknown Initial Condition," *Journal of Time Series Analysis*, 24(2), 141–148.

De Jong, P. and Shephard, N. (1995), "The Simulation Smoother for Time Series Models," *Biometrika*, 82(2), 339–350.

Deistler, M., Dunsmuir, W., and Hannan, E. J. (1978), "Vector Linear Time Series Models: Corrections and Extensions," *Advances in Applied Probability*, 10(2), 360–372.

DeJong, D. N., Ingram, B. F., and Whiteman, C. H. (2000), "A Bayesian Approach to Dynamic Macroeconomics," *Journal of Econometrics*, 98(2), 203–223.

Del Moral, P., Doucet, A., and Jasra, A. (2006), "Sequential Monte Carlo Samplers," *Journal of the Royal Statistical Society Series B*, 68(3), 411–436.

Del Negro, M. and Eusepi, S. (2011), "Fitting Observed Inflation Expectations," *Journal of Economic Dynamics and Control*, 35(12), 2105–2131.

Del Negro, M., Eusepi, S., Giannoni, M., Sbordone, A., Tambalotti, A., Cocci, M., Hasegawa, R., and Linder, M. H. (2013), "The FRBNY DSGE Model," Federal Reserve Bank of New York Staff Reports No. 647.

Del Negro, M., Giannoni, M. P., and Schorfheide, F. (2015), "Inflation in the Great Recession and New Keynesian Models," *American Economic Journal: Macroeconomics*, 7(1), 168–196.

Del Negro, M. and Schorfheide, F. (2004), "Priors from General Equilibrium Models," *International Economic Review*, 45(2), 643–673.

Del Negro, M. and Schorfheide, F. (2006), "How Good Is What You've Got? DSGE-VAR as a Toolkit for Evaluating DSGE Models," *Federal Reserve Bank of Atlanta Economic Review*, 91(2), 21–37.

Del Negro, M. and Schorfheide, F. (2008), "Forming Priors for DSGE Models (and How It Affects the Assessment of Nominal Rigidities)," *Journal of Monetary Economics*, 55(7), 1191–1208.

Del Negro, M. and Schorfheide, F. (2009), "Monetary Policy Analysis with Potentially Misspecified Models," *American Economic Review*, 99(4), 1415–1450.

Del Negro, M. and Schorfheide, F. (2013), "DSGE Model-Based Forecasting," in G. Elliott and A. Timmermann (Editors), *Handbook of Economic Forecasting*, volume 2, 57–140, North Holland, Amsterdam.

Del Negro, M., Schorfheide, F., Smets, F., and Wouters, R. (2007), "On the Fit of New-Keynesian Models," *Journal of Business & Economic Statistics*, 25(2), 123–143, with discussion, p. 143–162.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977), "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society Series B*, 39(1), 1–38.

Diebold, F., Gunther, T. A., and Tay, A. S. (1998), "Evaluating Density Forecasts with Applications to Financial Risk Management," *International Economic Review*, 39(4), 863–883.

Douc, R., Cappé, O., and Moulines, E. (2005), "Comparison of Resampling Schemes for Particle Filtering," in *Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis, ISPA 2005*, IEEE, conference Location: Zagreb, Crotia.

Doucet, A., Briers, M., and Sénécal, S. (2006), "Efficient Block Sampling Strategies for Sequential Monte Carlo Methods," *Journal of Computational and Graphical Statistics*, 15(3), 693–711.

Doucet, A. and Johansen, A. M. (2011), "A Tutorial on Particle Filtering and Smoothing: Fifteen Years Later," in D. Crisan and B. Rozovskiĭ (Editors), *The Oxford Handbook of Nonlinear Filtering*, volume 12, 656–704, Oxford University Press, Oxford.

Dunsmuir, W. (1979), "A Central Limit Theorem for Parameter Estimation in Stationary Vector Time Series and Its Application to Models for a Sginal Observed with Noise," *The Annals of Statistics*, 7(3), 490–605.

Dunsmuir, W. and Hannan, E. J. (1976), "Vector Linear Time Series Models," *Advances in Applied Probability*, 8(2), 339–364.

Durbin, J. and Koopman, S. J. (1997), "Monte Carlo Maximum Likelihood Estimation in Non-Gaussian State Space Models," *Biometrika*, 84(3), 669–684.

Durbin, J. and Koopman, S. J. (2002), "A Simple and Efficient Simulation Smoother for State Space Time Series Analysis," *Biometrika*, 89(3), 603–615.

Durbin, J. and Koopman, S. J. (2012), *Time Series Analysis by State Space Methods*, Oxford University Press, Oxford, 2nd edition.

Durham, G. and Geweke, J. (2014), "Adaptive Sequential Posterior Simulators for Massively Parallel Computing Environments," in I. Jeliazkov and D. Poirier (Editors), *Advances in Econometrics*, volume 34, chapter 6, 1–44, Emerald Group Publishing Limited.

Efron, B. (1986), "Why Isn't Everyone a Bayesian?" *The American Statistician*, 40(1), 1–5, with discussion, p. 5–11.

Eklund, J. and Karlsson, S. (2007), "Forecast Combinations and Model Averaging using Predictive Measures," *Econometric Reviews*, 26(2–4), 329–363.

Erceg, C. J. and Lindé, J. (2013), "Fiscal Consolidation in a Currency Union: Spending Cuts vd. Tax Hikes," *Journal of Economic Dynamics and Control*, 37(2), 422–445.

Evans, G. W. and Honkapohja, S. (2001), *Learning and Expectations in Macroeconomics*, Princeton University Press, Princeton.

Fagan, G., Lothian, J. R., and McNelis, P. D. (2013), "Was the Gold Standard Really Destabilizing?" *Journal of Applied Econometrics*, 28(2), 231–249.

Fang, K.-T., Kotz, S., and Ng, K. W. (1990), *Symmetric Multivariate and Related Distributions*, Chapman & Hall, London.

Fasano, G. and Franceschini, A. (1987), "A Multidimensional Version of the Kolmogorov-Smirnov Test," *Monthly Notices of the Royal Astronomical Society*, 225(1), 115–170.

Faust, J. and Gupta, A. (2012), "Posterior Predictive Analysis for Evaluation of DSGE Model," NBER Working Paper Series No. 17906.

Fernández-Villaverde, J. (2010), "The Econometrics of DSGE Models," *SERIEs*, 1(1–2), 3–49.

Fernández-Villaverde, J. and Rubio-Ramírez, J. F. (2004), "Comparing Dynamic Equilibrium Models to Data: A Bayesian Approach," *Journal of Econometrics*, 123(1), 153–187.

Fernández-Villaverde, J. and Rubio-Ramírez, J. F. (2005), "Estimating Dynamic Equilibrium Economies: Linear versus Non-Linear Likelihood," *Journal of Applied Econometrics*, 20(7), 891–910.

Fernández-Villaverde, J., Rubio-Ramírez, J. F., Sargent, T. J., and Watson, M. W. (2007), "ABCs (and Ds) of Understanding VARs," *American Economic Review*, 97(3), 1021–1026.

Fishburn, P. C. (1977), "Mean-Risk Analysis with Risk Associated with Below-Target Returns," *American Economic Review*, 67(2), 116–126.

Fisher, J. D. M. (2015), "On the Structural Interpretation of the Smets-Wouters "Risk Premium" Shock," *Journal of Money, Credit and Banking*, 47(2–3), 511–516.

Franchi, M. and Paruolo, P. (2012), "On ABCs (and Ds) of VAR Representations of DSGE Models," Manuscript, University of Tome "La Sapienza".

Frühwirth-Schnatter, S. (1994), "Data Augmentation and Dynamic Linear Models," *Journal of Time Series Analysis*, 15(2), 183–202.

Fuhrer, J. C. (1994), "Optimal Monetary Policy and the Sacrifice Ratio," *Federal Reserve Bank of Boston, Conference Series*, 38(June), 43–69.

Fuhrer, J. C. and Madigan, B. F. (1997), "Monetary Policy when Interest Rates are Bounded at Zero," *Review of Economics and Statistics*, 79(4), 573–585.

Fuller, W. A. (1976), *Introduction to Statistical Time Series*, John Wiley, New York.

Galí, J. (1999), "Technology, Employment, and the Business Cycle: Do Technology Shocks Explain Aggregate Fluctuations?" *American Economic Review*, 89(1), 249–271.

Galí, J. and Gertler, M. (1999), "Inflation dynamics: A structural econometric analysis," *Journal of Monetary Economics*, 44(2), 195–222.

Galí, J. and Gertler, M. (2007), "Macroeconomic Modeling for Monetary Policy Evaluation," *Journal of Economic Perspectives*, 21(4), 25–45.

Galí, J. and Monacelli, T. (2005), "Monetary Policy and Exchange Rate Volatility in a Small Open Economy," *Review of Economic Studies*, 72(3), 707–734.

Galí, J., Smets, F., and Wouters, R. (2012), "Unemployment in an Estimated New Keynesian Model," in D. Acemoglu and M. Woodford (Editors), *NBER Macroeconomics Annual 2011*, 329–360, University of Chicago Press.

Gelfand, A. and Dey, D. (1994), "Bayesian Model Choice: Asymptotics and Exact Calculations," *Journal of the Royal Statistical Society Series B*, 56(3), 501–514.

Gelfand, A. E. and Smith, A. F. M. (1990), "Sampling Based Approaches to Calculating Marginal Densities," *Journal of the American Statistical Association*, 85(410), 398–409.

Gelman, A. (1996), "Inference and Monitoring Convergence," in W. R. Gilks, S. Richardson, and D. J. Spiegelhalter (Editors), *Markov Chain Monte Carlo in Practice*, 131–143, Chapman & Hall/CRC, Boca Raton.

Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (2004), *Bayesian Data Analysis*, Chapman & Hall/CRC, Boca Raton, 2nd edition.

Gelman, A., Meng, X.-L., and Stern, H. (1996a), "Posterior Predictive Assessment of Model Fitness via Realized Discrepancies," *Statistica Sinica*, 6(4), 733–760, with discussions, p. 760–808.

Gelman, A., Roberts, G. O., and Gilks, W. R. (1996b), "Efficient Metropolis Jumping Rules," in J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith (Editors), *Bayesian Statistics 5*, 599–607, Oxford University Press, Oxford.

Gelman, A. and Rubin, D. B. (1992), "Inference from Iterative Simulations Using Multiple Sequences," *Statistical Science*, 7(4), 457–511.

Geman, S. and Geman, D. (1984), "Stochastic Relaxation, Gibbs Distributions and the Bayesian Restoration of Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6), 721–741.

Geweke, J. (1988), "Antithetic Acceleration of Monte Carlo Integration in Bayesian Inference," *Journal of Econometrics*, 38(1–2), 73–89.

Geweke, J. (1989a), "Bayesian Inference in Econometric Models Using Monte Carlo Integration," *Econometrics*, 57(6), 1317–1340.

Geweke, J. (1989b), "Exact Predictive Densities for Linear Models with ARCH Disturbances," *Journal of Econometrics*, 40(1), 63–86.

Geweke, J. (1999), "Using Simulation Methods for Bayesian Econometric Models: Inference, Development, and Communication," *Econometric Reviews*, 18(1), 1–73.

Geweke, J. (2005), *Contemporary Bayesian Econometrics and Statistics*, John Wiley, Hoboken.

Geweke, J. (2007), "Bayesian Model Comparison and Validation," *American Economic Review*, 97(2), 60–64.

Geweke, J. (2010), *Complete and Incomplete Econometric Models*, Princeton University Press, Princeton.

Geweke, J. and Amisano, G. (2010), "Comparing and Evaluating Bayesian Predictive Distributions of Asset Returns," *International Journal of Forecasting*, 26(2), 216–230.

Geweke, J. and Amisano, G. (2011), "Optimal Prediction Pools," *Journal of Econometrics*, 164(1), 130–141.

Geweke, J. and Amisano, G. (2012), "Prediction and Misspecified Models," *American Economic Review*, 102(3), 482–486.

Geweke, J. and Whiteman, C. H. (2006), "Bayesian Forecasting," in G. Elliott, C. W. Granger, and A. Timmermann (Editors), *Handbook of Economic Forecasting*, 3–80, North Holland, Amsterdam, volume 1.

Geyer, C. J. (1992), "Practical Markov Chain Monte Carlo," *Statistical Science*, 7(4), 473–511.

Gilks, W. R. and Berzuini, C. (2001), "Following a Moving Target—Monte Carlo Inference for Dynamic Bayesian Models," *Journal of the Royal Statistical Society Series B*, 63(1), 127–146.

Gill, J. and King, G. (2004), "What to Do When Your Hessian Is Not Invertible: Alternatives to Model Respecifcation in Nonlinear Estimation," *Sociological Methods & Research*, 33(1), 54–87.

Giordani, P., Pitt, M., and Kohn, R. (2011), "Bayesian Inference for Time Series State Space Models," in J. Geweke, G. Koop, and H. van Dijk (Editors), *The Oxford Handbook of Bayesian Econometrics*, 61–124, Oxford University Press, New York.

Gómez, V. (2007), "Wiener-Kolmogorov Filtering and Smoothing for Multivariate Series with State-Space Structure," *Journal of Time Series Analysis*, 28(3), 361–385.

Gneiting, T., Balabdaoui, F., and Raftery, A. E. (2007), "Probabilistic Forecasts, Calibration and Sharpness," *Journal of the Royal Statistical Society Series B*, 69(2), 243–268.

Gneiting, T. and Raftery, A. E. (2007), "Strictly Proper Scoring Rules, Prediction, and Estimation," *Journal of the American Statistical Association*, 102(477), 359–378.

Gneiting, T., Raftery, A. E., Westveld III, A. H., and Goldman, T. (2005), "Calibrated Probabilistic Forecasting Using Ensemble Model Output Statistics and Minimum CRPS Estimation," *Monthly Weather Review*, 133(5), 1098–1118.

Gneiting, T., Stanberry, L. I., Grimit, E. P., Held, L., and Johnson, N. A. (2008), "Assessing Probabilistic Forecasts of Multivariate Quantities, with an Application to Ensemble Predictions of Surface Winds," *TEST*, 17(2), 211–235.

Golub, G. H. and van Loan, C. F. (1983), *Matrix Computations*, Johns Hopkins University Press, Baltimore, 1st edition.

Good, I. J. (1952), "Rational Decisions," *Journal of the Royal Statistical Society Series B*, 14(1), 107–114.

Gordon, N. J., Salmond, D. J., and Smith, A. F. M. (1993), "Novel Approach to Nonlinear/non-Gaussian Bayesian State Estimation," *Radar and Signal Processing, IEE Proceedings-F*, 140(2), 107–113.

Gouriéroux, C., Monfort, A., and Renault, E. (1993), "Indirect Inference," *Journal of Applied Econometrics*, 8(S1), S85–S188.

Hamilton, J. D. (1994), *Time Series Analysis*, Princeton University Press, Princeton.

Harvey, A. C. (1989), *Forecasting, Structural Time Series Models and the Kalman Filter*, Cambridge University Press, Cambridge.

Harvey, A. C. and Pierse, R. G. (1984), "Estimating Missing Observations in Economic Time Series," *Journal of the American Statistical Association*, 79(385), 125–131.

Hastings, W. K. (1970), "Monte Carlo Sampling Methods Using Markov Chains and Their Applications," *Biometrika*, 57(1), 97–109.

Havil, J. (2003), *Gamma: Exploring Euler's Constant*, Princeton Science Library, Princeton, N. J.

Hebden, J., Lindé, J., and Svensson, L. E. O. (2011), "Optimal Monetary Policy in the Hybrid New Keynesian Model under the Zero Lower Bound," Mimeo, Federal Reserve Board.

Herbst, E. and Schorfheide, F. (2014), "Sequential Monte Carlo Sampling for DSGE Models," *Journal of Applied Econometrics*, 29(7), 1073–1098.

Herbst, E. and Schorfheide, F. (2016), *Bayesian Estimation of DSGE Models*, Princeton University Press, Princeton.

Herbst, E. P. (2015), "Using the "Chandrasekhar Recursions" for Likelihood Evaluation of DSGE Models," *Computational Economics*, 45(4), 693–705.

Hol, J. D., Schön, T. B., and Gustafsson, F. (2006), "On Resampling Algorithms for Particle Filters," *Proceedings of the IEEE Nonlinear Statistical Signal Processing Workshop*, Cambridge, U. K., Sept. 2006, 79–82.

Holthausen, D. M. (1981), "A Risk-Return Model with Risk and Return Measured as Deviations from a Target Return," *American Economic Review*, 71(1), 182–188.

Hoogerheide, L., Opschoor, A., and van Dijk, H. K. (2012), "A Class of Adaptive Importance Sampling Weighted EM Algorithms for Efficient and Robust Posterior and Predictive Simulation," *Journal of Econometrics*, 171(2), 101–120.

Ingram, B. F. and Whiteman, C. H. (1994), "Supplanting the 'Minnesota' Prior — Forecasting Macroeconomic Time Series Using Real Business Cycle Model Priors," *Journal of Monetary Economics*, 34(3), 497–510.

Ireland, P. (2004), "Money's Role in the Monetary Business Cycle," *Journal of Money, Credit and Banking*, 36(6), 969–984.

Iskrev, N. (2008), "Evaluating the Information Matrix in Linearized DSGE Models," *Economics Letters*, 99(3), 607–610.

Iskrev, N. (2010), "Local Identification in DSGE Models," *Journal of Monetary Economic*, 57(2), 189–202.

Johnson, N., Kotz, S., and Balakrishnan, N. (1995), *Continuous Univariate Distributions*, volume 2, John Wiley, New York, 2nd edition.

Jordan, A., Krüger, F., and Lerch, S. (2019), "Evaluating Probabilistic Forecasts with scoringRules," *Journal of Statistical Software*, 90(12), 1–37.

Kadiyala, K. R. and Karlsson, S. (1997), "Numerical Methods for Estimation and Inference in Bayesian VAR-Models," *Journal of Applied Econometrics*, 12(2), 99–132.

Kalman, R. E. (1960), "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D), 35–45.

Kalman, R. E. and Bucy, R. S. (1961), "New Results in Linear Filtering and Prediction Theory," *Transactions of the ASME–Journal of Basic Engineering*, 83(Series D), 95–108.

Kass, R. E. and Raftery, A. E. (1995), "Bayes Factors," *Journal of the American Statistical Association*, 90(430), 773–795.

Kilian, L. and Manganelli, S. (2007), "Quantifying the Risk of Deflation," *Journal of Money, Credit and Banking*, 39(2–3), 561–590.

Kilian, L. and Manganelli, S. (2008), "The Central Banker as a Risk Manager: Estimating the Federal Reserve's Preferences under Greenspan," *Journal of Money, Credit and Banking*, 40(6), 1103–1129.

Kimball, M. S. (1995), "The Quantitative Analytics of the Basic Nonmonetarist Model," *Journal of Money, Credit and Banking*, 27(4), 1241–77.

King, R. G. (2000), "The New IS-LM Model: Language, Logic, and Limits," *Federal Reserve Bank of Richmond Economic Quarterly*, 86(3), 45–103.

King, R. G. and Watson, M. W. (1996), "Money, Prices, Interest Rates and the Business Cycle," *Review of Economics and Statistics*, 78(1), 35–53.

King, R. G. and Watson, M. W. (1998), "The Solution of Singular Linear Difference Systems under Rational Expectations," *International Economic Review*, 39(4), 1015–1026.

Kitagawa, G. (1996), "Monte Carlo Filter and Smoother for Non-Gaussian Nonlinear State Space Models," *Journal of Computational and Graphical Statistics*, 5(1), 1–25.

Klein, A. and Neudecker, H. (2000), "A Direct Derivation of the Exact Fisher Information Matrix of Gaussian Vector State Space Models," *Linear Algebra and its Applications*, 321(1–3), 233–238.

Klein, A. and Spreij, P. (2006), "An Explicit Expression for the Fisher Information Matrix of a Multiple Time Series Process," *Linear Algebra and its Applications*, 417(1), 140–149.

Klein, A. and Spreij, P. (2009), "Matrix Differential Calculus Applied to Multiple Stationary Time Series and an Extended Whittle Formula for Information Matrices," *Linear Algebra and its Applications*, 430(2–3), 674–691.

Klein, P. (2000), "Using the Generalized Schur Form to Solve a Multivariate Linear Rational Expectations Model," *Journal of Economic Dynamics and Control*, 24(10), 1405–1423.

Kloek, T. and van Dijk, H. K. (1978), "Bayesian Estimates of Equation System Parameters: An Application of Integration by Monte Carlo," *Econometrica*, 46(1), 1–20.

Kohn, R. and Ansley, C. F. (1983), "Fixed Interval Estimation in State Space Models when some of the Data are Missing or Aggregated," *Biometrika*, 70(3), 683–688.

Kohn, R. and Ansley, C. F. (1989), "A Fast Algorithm for Signal Extraction, Influence and Cross-Validation in State Space Models," *Biometrika*, 76(1), 65–79.

Komunjer, I. and Ng, S. (2011), "Dynamic Identification of Dynamic Stochastic General Equilibrium Models," *Econometrica*, 79(6), 1995–2032.

Kong, A., Liu, J. S., and Wong, W. H. (1994), "Sequential Imputations and Bayesian Missing Data Problems," *Journal of the American Statistical Association*, 89(425), 278–288.

Koopman, S. J. (1993), "Disturbance Smoother for State Space Models," *Biometrika*, 80(1), 117–126.

Koopman, S. J. (1997), "Exact Initial Kalman Filtering and Smoothing for Nonstationary Time Series Models," *Journal of the American Statistical Association*, 92(440), 1630–1638.

Koopman, S. J. and Durbin, J. (2000), "Fast Filtering and Smoothing for State Space Models," *Journal of Time Series Analysis*, 21(3), 281–296.

Koopman, S. J. and Durbin, J. (2003), "Filtering and Smoothing of State Vector for Diffuse State-Space Models," *Journal of Time Series Analysis*, 24(1), 85–98.

Koopman, S. J. and Harvey, A. (2003), "Computing Observation Weights for Signal Extraction and Filtering," *Journal of Economic Dynamics and Control*, 27(7), 1317–1333.

Krishnamurthy, A. and Vissing-Jorgensen, A. (2012), "The Aggregate Demand for Treasury Debt," *Journal of Political Economy*, 120(2), 233–267.

Kullback, S. and Leibler, R. A. (1951), "On Information and Sufficiency," *The Annals of Mathematical Statistics*, 22(1), 79–86.

Kydland, F. E. and Prescott, E. C. (1982), "Time to Build and Aggregate Fluctuations," *Econometrica*, 50(6), 1345–1370.

Landsman, Z. M. and Valdez, E. A. (2003), "Tail Conditional Expectations for Elliptical Distributions," *North American Actuarial Journal*, 7(4), 55–71.

Lanne, M. and Luoto, J. (2018), "Data-Driven Identification Constraints for DSGE Models," *Oxford Bulletin of Economics and Statistics*, 80(2), 23–258.

Laséen, S. and Svensson, L. E. O. (2011), "Anticipated Alternative Policy Rate Paths in Policy Simulations," *International Journal of Central Banking*, 7(3), 1–33.

Leeper, E. M., Plante, M., and Traum, N. (2010), "Dynamics of Fiscal Financing in the United States," *Journal of Econometrics*, 156(2), 304–321.

Leeper, E. M., Walker, T. B., and Yang, S.-C. S. (2013), "Fiscal Foresight and Information Flows," *Econometrica*, 81(3), 1115–1145.

Leeper, E. M. and Zha, T. (2003), "Modest Policy Interventions," *Journal of Monetary Economics*, 50(8), 1673–1700.

Lees, K., Matheson, T., and Smith, C. (2011), "Open Economy Forecasting with a DSGE-VAR: Head to Head with the RBNZ Published Forecasts," *International Journal of Forecasting*, 27(2), 512–528.

Li, T., Bolić, M., and Djurić, P. M. (2015), "Resampling Methods for Particle Filtering: Classification, Implementation, and Strategies," *IEEE Signal Processing Magazine*, 32(3), 70–86.

Lindé, J., Smets, F., and Wouters, R. (2016), "Challenges for Central Banks' Macro Models," in J. B. Taylor and H. Uhlig (Editors), *Handbook of Macroeconomics*, volume 2B, 2185–2262, North Holland, Amsterdam.

Lindley, D. V. (1957), "A Statistical Paradox," *Biometrika*, 44(1–2), 187–192.

Little, R. J. (2006), "Calibrated Bayes: A Bayes/Frequentist Roadmap," *The American Statistician*, 60(3), 213–223.

Liu, J. S. and Chen, R. (1995), "Blind Deconvolution via Sequential Imputations," *Journal of the American Statistical Association*, 90(430), 567–576.

Liu, J. S. and Chen, R. (1998), "Sequential Monte Carlo Methods for Dynamic Systems," *Journal of the American Statistical Association*, 93(443), 1032–1044.

Loudin, J. D. and Miettinen, H. E. (2003), "A Multivariate Method for Comparing N-dimenional Distributions," *PHYSTAT2003 Proceedings*, 207–210.

Lubik, T. A. and Schorfheide, F. (2004), "Testing for Indeterminacy: An Application to U. S. Monetary Policy," *American Economic Review*, 94(1), 190–217.

Lubik, T. A. and Schorfheide, F. (2007a), "Do Central Banks Respond to Exchange Rate Movements? A Structural Investigation," *Journal of Monetary Economics*, 54(4), 1069–1087.

Lubik, T. A. and Schorfheide, F. (2007b), "Testing for Indeterminacy: An Application to U.S. Monetary Policy: Reply," *American Economic Review*, 97(1), 530–533.

Lucas, Jr., R. E. (1976), "Econometric Policy Evaluation: A Critique," in K. Brunner and A. H. Meltzer (Editors), *Carnegie-Rochester Series on Public Policy, Vol. 1*, 19–46, North-Holland, Amsterdam.

Machina, M. J. and Rothschild, M. (1987), "Risk," in J. Eatwell, M. Millgate, and P. Newman (Editors), *The New Palgrave Dictionary of Economics*, 203–205, MacMillan, London.

Magnus, J. R. and Neudecker, H. (1988), *Matrix Differential Calculus with Applications in Statistics and Econometrics*, John Wiley, Chichester.

Mankiw, N. G. (2006), "The Macroeconomist as Scientist and Engineer," *Journal of Economic Perspectives*, 20(4), 29–46.

Mankiw, N. G. and Reis, R. (2002), "Sticky Information versus Sticky Prices: A Proposal to Replace the New Keynesian Phillips Curve," *Quarterly Journal of Economics*, 117(4), 1295–1328.

Marsaglia, G., Tsang, W. W., and Wang, J. (2003), "Evaluating Kolmogorov's Distribution," *Journal of Statistical Software*, 8(18), 1–4.

McCallum, B. T. and Nelson, E. (1999), "An Optimizing IS-LM Specification for Monetary Policy and Business Cycle Analysis," *Journal of Money, Credit and Banking*, 31(3), 296–316.

Meinhold, R. J. and Singpurwalla, N. D. (1983), "Understanding the Kalman Filter," *The American Statistician*, 37(2), 123–127.

Meng, X.-L. (1994), "Posterior Predictive *p*-values," *The Annals of Statistics*, 22(3), 1142–1160.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953), "Equations of State Calculations by Fast Computing Machines," *Journal of Chemical Physics*, 21(6), 1087–1091.

Meyer-Gohde, A. (2010), "Linear Rational-Expectations Models with Lagged Expectations: A Synthetic Method," *Journal of Economic Dynamics and Control*, 34(5), 984–1002.

Mikhail, W. M. (1972), "Simulating the Small-Sample Properties of Econometric Estimators," *Journal of the American Statistical Association*, 67(339), 620–624.

Mira, A. and Tierney, L. (2002), "Efficiency and Convergence Properties of Slice Samplers," *Scandinavian Journal of Statistics*, 29(1), 1–12.

Morf, M. and Kailath, T. (1975), "Square Root Algorithms for Least Squares Estimation," *IEEE Transaction on Automatic Control*, 20(4), 487–497.

Morf, M., Sidhu, G. S., and Kailath, T. (1974), "Some New Algorithms for Recursive Estimation in Constant, Linear, Discrete-Time Systems," *IEEE Transaction on Automatic Control*, 19(4), 315–323.

Neal, R. M. (2003), "Slice Sampling," *The Annals of Statistics*, 31(3), 705–741, with discussion, p. 742–767.

Newey, W. K. and West, K. D. (1987), "A Simple, Positive Semi-Definite, Heteroskedasticity and Autocorrelation Consistent Covariance Matrix," *Econometrica*, 55(3), 703–708.

Otrok, C. (2001), "On Measuring the Welfare Cost of Business Cycles," *Journal of Monetary Economics*, 47(1), 61–92.

Phillips, P. C. B. (1996), "Econometric Model Determination," *Econometrica*, 64(4), 763–812.

Poirier, D. J. (1988), "Frequentist and Subjectivist Perspectives on the Problem of Model Building in Economics," *Journal of Economic Perspectives*, 2, 121–144, with discussion, p. 145–170.

Raftery, A. E. (1996), "Hypothesis Testing and Model Selection," in W. R. Gilks, S. Richardson, and D. J. Spiegelhalter (Editors), *Markov Chain Monte Carlo in Practice*, 163–187, Chapman & Hall/CRC, Boca Raton.

Rao, C. R. (1973), *Linear Statistical Inference and Its Applications*, John Wiley, New York, 2nd edition.

Ratto, M. (2008), "Analysing DSGE Models with Global Sensitivity Analysis," *Computational Economics*, 31(2), 115–139.

Reifschneider, D. and Williams, J. (2000), "Three Lessons for Monetary Policy in a Low-Inflation Era," *Journal of Money, Credit and Banking*, 32(4, Part 2), 936–966.

Roberts, G. O. (1996), "Markov Chain Concepts Related to Sampling Algorithms," in W. R. Gilks, S. Richardson, and D. J. Spiegelhalter (Editors), *Markov Chain Monte Carlo in Practice*, 45–57, Chapman & Hall/CRC, Boca Raton.

Roberts, G. O., Gelman, A., and Gilks, W. R. (1997), "Weak Convergence and Optimal Scaling of Random Walk Metropolis Algorithms," *The Annals of Applied Probability*, 7(1), 110–120.

Roberts, G. O. and Rosenthal, J. S. (1999), "Convergence of Slice Sampler Markov Chains," *Journal of the Royal Statistical Society Series B*, 61(3), 643–660.

Roberts, G. O. and Tweedie, R. L. (1996), "Geometric Convergence and Central Limit Theorems for Multidimensional Hastings and Metropolis Algorithms," *Biometrika*, 83(1), 95–110.

Robertson, J. C., Tallman, E. W., and Whiteman, C. H. (2005), "Forecasting Using Relative Entropy," *Journal of Money, Credit and Banking*, 37(3), 383–401.

Rosenblatt, M. (1952), "Remarks on a Multivariate Transformation," *The Annals Of Mathematical Statistics*, 23(3), 470–472.

Rotemberg, J. J. and Woodford, M. (1997), "An Optimization-Based Econometric Framework for the Evaluation of Monetary Policy," in B. S. Bernanke and J. J. Rotemberg (Editors), *NBER Macroeconomics Annual 1997*, volume 12, 297–346, MIT Press.

Rothenberg, T. J. (1971), "Identification in Parametric Models," *Econometrica*, 39(3), 577–591.

Rubin, D. B. (1984), "Bayesianly Justifiable and Relevant Frequency Calculations for the Applied Statistician," *The Annals of Statistics*, 12(4), 1151–1172.

Rubin, D. B. (1991), "EM and Beyond," *Psychometrika*, 56(2), 241–254.

Sargent, T. J. (1987), *Macroeconomic Theory*, Academic Press, Orlando, Florida, 2nd edition.

Sargent, T. J. (1989), "Two Models of Measurement and the Investment Accelerator," *Journal of Political Economy*, 97(2), 251–287.

Sargent, T. J. (1993), *Bounded Rationality in Macroeconomics*, Clarendon Press, Oxford.

Schmitt-Grohé, S. (2010), "Comment on Paul Beaudry and Bernd Lucke 'Letting Different Views about Business Cycles Compete'," in D. Acemuglo, K. Rogoff, and M. Woodford (Editors), *NBER Macroeconomics Annual 2009*, volume 24, 475–490, The Univeristy of Chicago Press.

Schmitt-Grohé, S. and Uribe, M. (2012), "What's News in Business Cycles," *Econometrica*, 80(6), 2733–2764.

Schneider, B. E. (1978), "Trigamma Function," *Applied Statistics*, 27(1), 97–99, algorithm AS 121.

Schorfheide, F. (2000), "Loss Function-Based Evaluation of DSGE Models," *Journal of Applied Econometrics*, 15(6), 645–670.

Sims, C. A. (2002), "Solving Linear Rational Expectations Models," *Computational Economics*, 20(1–2), 1–20.

Sims, C. A., Waggoner, D. F., and Zha, T. (2008), "Methods for Inference in Large Multiple-Equation Markov-Switching Models," *Journal of Econometrics*, 146(1), 255–274.

Sims, C. S. (2003), "Implications of Rational Inattention," *Journal of Monetary Economics*, 50(3), 665–690.

Sims, E. R. (2012), "News, Non-Invertibility, and Structural VARs," in N. Balke, F. Canova, F. Milani, and M. A. Wynne (Editors), *DSGE Models in Macroeconomics: Estimation, Evaluation, and New Developments (Advances in Econometrics)*, volume 28, 81–135, Emerald Group Publishing.

Slobodyan, S. and Wouters, R. (2012), "Learning in a Medium-Scale DSGE Model with Expectations Based on Small Forecasting Models," *American Economic Journal: Macroeconomics*, 4(2), 65–101.

Smets, F., Warne, A., and Wouters, R. (2014), "Professional Forecasters and Real-Time Forecasting with a DSGE Model," *International Journal of Forecasting*, 30(4), 981–995.

Smets, F. and Wouters, R. (2003), "An Estimated Stochastic Dynamic General Equilibrium Model for the Euro Area," *Journal of the European Economic Association*, 1(5), 1123–1175.

Smets, F. and Wouters, R. (2004), "Forecasting with a Bayesian DSGE Model: An Application to the Euro Area," *Journal of Common Market Studies*, 42(4), 841–867.

Smets, F. and Wouters, R. (2005), "Comparing Shocks and Frictions in U.S. and Euro Area Business Cycles: A Bayesian DSGE Approach," *Journal of Applied Econometrics*, 20(2), 161–183.

Smets, F. and Wouters, R. (2007), "Shocks and Frictions in US Business Cycles: A Bayesian DSGE Approach," *American Economic Review*, 97(3), 586–606.

Smith, Jr., A. A. (1993), "Estimating Nonlinear Time-Series Models Using Simulated Vector Autoregressions," *Journal of Applied Econometrics*, 8(S1), S63–S84.

Smith, J. Q. (1985), "Diagnostic Checks of Non-Standard Time Series Models," *Journal Of Forecasting*, 4(3), 283–291.

Strachan, R. and van Dijk, H. K. (2011), "Divergent Priors and well Behaved Bayes Factors," Tinbergen Institute Discussion Papers 11-006/4, Tinbergen Institute.

Svensson, L. E. O. and Woodford, M. (2003), "Indicator Variables for Optimal Policy," *Journal of Monetary Economics*, 50(3), 691–720.

Tanner, M. A. and Wong, W. H. (1987), "The Calculation of Posterior Distributions by Data Augmentation," *Journal of the American Statistical Association*, 82(394), 528–540, with discussion, p. 541–550.

Thompson, P. A. and Miller, R. B. (1986), "Sampling the Future: A Bayesian Approach to Forecasting from Univariate Time Series Models," *Journal of Business & Economic Statistics*, 4(4), 427–436.

Thomson, W. (1891), *Popular Lectures and Addresses, Volume 1*, Macmillan, London, 2nd edition, lecture on "Electrical Units of Measurement", 3rd of May, 1883.

Tierney, L. (1994), "Markov Chains for Exploring Posterior Distributions," *The Annals of Statistics*, 22(4), 1701–1728, with discussion, p. 1728–1762.

Tierney, L. and Kadane, J. B. (1986), "Accurate Approximations for Posterior Moments and Marginal Densities," *Journal of the American Statistical Association*, 81(393), 82–86.

Villani, M. (2001), "Bayesian Prediction with Cointegrated Vector Autoregressions," *International Journal of Forecasting*, 17(4), 585–605.

Villani, M. (2005), "Bayesian Reference Analysis of Cointegration," *Econometric Theory*, 21(2), 326–357.

Villani, M. (2009), "Steady-State Priors for Vector Autoregressions," *Journal of Applied Econometrics*, 24(4), 630–650.

Waggoner, D. F. and Zha, T. (1999), "Conditional Forecasts in Dynamic Multivariate Models," *Review of Economics and Statistics*, 81(4), 639–651.

Walker, A. M. (1964), "Asymptotic Properties of Least-Squares Estimates of Parameters of the Spectrum of a Stationary Non-Deterministic Time-Series," *Journal of the Australian Mathematical Society*, 4(3), 363–384.

Wallis, K. F. (1986), "Forecasting with an Econometric Model: The Ragged Edge Problem," *Journal of Forecasting*, 5(1), 1–13.

Warne, A. (2006), "Bayesian Inference in Cointegrated VAR Models: With Applications to the Demand for Euro Area M3," ECB Working Paper Series No. 692.

Warne, A. (2025), "Extenting YADA — A Guide to The User Interface, The Controls and The Data Structures," Manuscript, European Central Bank. Available with the YADA distribution.

Warne, A., Coenen, G., and Christoffel, K. (2013), "Predictive Likelihood Comparisons with DSGE and DSGE-VAR Models," ECB Working Paper No. 1536.

Warne, A., Coenen, G., and Christoffel, K. (2017), "Marginalized Predictive Likelihood Comparisons of Linear Gaussian State-Space Models with Applications to DSGE, DSGE-VAR and VAR Models," *Journal of Applied Econometrics*, 32(1), 103–119.

Whittle, P. (1953), "The Analysis of Multiple Time Series," *Journal of the Royal Statistical Society Series B*, 15(1), 125–139.

Winkler, R. L. and Murphy, A. H. (1968), "'Good' Probability Assessors," *Journal of Applied Meteorology*, 7(5), 751–758.

Woodford, M. (2003), *Interest and Prices: Foundations of a Theory of Monetary Policy*, Princeton University Press, Princeton.

Yu, B. and Mykland, P. (1998), "Looking at Markov Samplers Through CUSUM Path Plots: A Simple Diagnostic Idea," *Statistics and Computing*, 8(3), 275–286.

Zadrozny, P. A. (1989), "Analytic Derivatives for Estimation of Linear Dynamic Models," *Computers and Mathematics with Applications*, 18(6–7), 539–553.

Zadrozny, P. A. (1992), "Errata to 'Analytical Derivatives for Estimation of Linear Dynamic Models'," *Computers and Mathematics with Applications*, 24(8–9), 289–290.

Zagaglia, P. (2005), "Solving Rational-Expectations Models through the Anderson-Moore Algorithm: An Introduction to the Matlab Implementation," *Computational Economics*, 26(1), 91–106.

Zellner, A. (1971), *An Introduction to Bayesian Inference in Econometrics*, John Wiley, New York.