Department of Economics
Working Paper No. 296

BVAR: Bayesian Vector Autoregressions with Hierarchical Prior Selection in R

Nikolas Kuschnig
Lukas Vashold

October 2019

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

BVAR: Bayesian Vector Autoregressions with Hierarchical Prior Selection in R

Nikolas Kuschnig
Vienna University of Economics and Business

Lukas Vashold
Vienna University of Economics and Business

Abstract

Vector autoregression (VAR) models are widely used models for multivariate time series analysis, but often suffer from their dense parameterization. Bayesian methods are commonly employed as a remedy by imposing shrinkage on the model coefficients via informative priors, thereby reducing parameter uncertainty. The subjective choice of the informativeness of these priors is often criticized and can be alleviated via hierarchical modeling. This paper introduces **BVAR**, an R package dedicated to the estimation of Bayesian VAR models in a hierarchical fashion. It incorporates functionalities that permit addressing a wide range of research problems while retaining an easy-to-use and transparent interface. It features the most commonly used priors in the context of multivariate time series analysis as well as an extensive set of standard methods for analysis. Further functionalities include a framework for defining custom dummy-observation priors, the computation of impulse response functions, forecast error variance decompositions and forecasts.

Keywords: Vector autoregression, VAR, Bayesian, multivariate, hierarchical, R, package.

1. Introduction

Vector autoregression (VAR) models are widely used in multivariate time series analysis across various disciplines (e.g., Crespo Cuaresma, Feldkircher, and Huber 2016; Enders and Sandler 1993; Wild, Eichler, Friederich, Hartmann, Zipfel, and Herzog 2010). Popularized by Sims (1980), VAR models have become a staple of empirical macroeconomic research (Kilian and Lütkepohl 2017). However, the large number of parameters and the limited temporal availability of macroeconomic datasets may lead to over-parameterization problems (Koop and Korobilis 2010), that can be circumvented via a Bayesian approach. Informative priors impose additional structure on the model and shrink it towards parsimonious, yet proven, benchmarks. The result are models with reduced parameter uncertainty and significantly enhanced out-of-sample forecasting performance (Koop 2013). The choice of these priors and their informativeness poses a challenge and remains the fulcrum of discussion and criticism. Giannone, Lenza, and Primiceri (2015) provide a data-based, theoretically grounded approach of setting prior informativeness in the spirit of hierarchical modeling. They alleviate the subjectivity of setting hyperparameters and demonstrate remarkable performance in common analyses. **BVAR** is the first R package implementing these hierarchical Bayesian VAR models and provides a complete and easy-to-use toolkit for estimation and analysis.

The field of Bayesian statistical software is a vivid one – increasing computational power

has given rise to Markov chain Monte Carlo (MCMC) methods. Established software built on Gibbs sampling, a variant of the Metropolis-Hastings algorithm, include **BUGS** (Lunn, Thomas, Best, and Spiegelhalter 2000; Lunn, Spiegelhalter, Thomas, and Best 2009) and **JAGS** (Plummer 2003). **Stan** (Carpenter, Gelman, Hoffman, Lee, Goodrich, Betancourt, Brubaker, Guo, Li, and Riddell 2017), an imperative probabilistic programming language for statistical models, uses the No-U-Turn Sampler (Hoffman and Gelman 2014), a variant of Hamiltonian Monte Carlo method, for improved robustness and efficiency. These packages provide flexible and extensible tools for Bayesian inference and are available cross-platform with interfaces to several languages (e.g., to R and Python). A range of other packages provide similar or more specific implementations of MCMC algorithms – including **PyMC3** (Salvatier, Wiecki, and Fonnesbeck 2016) for Python and **MCMCglmm** (Hadfield 2010), **greta** (Golding *et al.* 2018) as well as **bvarsv** (Krueger 2015) for R. Integrated nested Laplace approximation (Rue, Martino, and Chopin 2009) is an alternative approach to approximate Bayesian inference with significant computational advantages, that is widely used in spatial modeling (see e.g., Blangiardo, Cameletti, Baio, and Rue 2013; Bivand, Gómez-Rubio, and Rue 2015) and has an R implementation in **R-INLA** (Rue, Martino, Lindgren *et al.* 2015). Despite this variety of software for Bayesian inference, there are no comprehensive options dedicated to Bayesian VAR modeling. The **vars** (Pfaff 2008) package represents a cornerstone in the field of frequentist multivariate time series analyses with R, offering a complete set of VAR-related functionalities. However, currently there exists no equivalent for paradigmatically different Bayesian VAR models, in spite of their popularity.

Aforementioned packages cover a wide range of possible applications and provide powerful and efficient tools for modeling and subsequent analysis. However, work with Bayesian VARs is frequently done via ad hoc MATLAB and R scripts. As a result, code gets recycled and pieced together frequently, without any central repository or version control facilitating reproducibility (Ram 2013), while many other scripts are only ever used once. In this paper we present **BVAR**, a software package for R that reconciles the advantages of more complete toolkits for Bayesian inference and the specificity of ad hoc scripts. The package implements the hierarchical modeling approach proposed by Giannone *et al.* (2015) to choose the informativeness of commonly used priors. The current version implements functionalities for a range of common analyses and allows for the use of existing frameworks, such as **coda** (Plummer, Best, Cowles, and Vines 2006) for MCMC diagnostics.

BVAR is licensed under the GNU General Public License 3 and is openly available on the Comprehensive R Archive Network (CRAN, `https://cran.r-project.org/package=BVAR`) and on GitHub (`https://github.com/nk027/bvar`).

The remainder of this paper is structured as follows. Section 2 describes the econometric framework. Section 3 provides an overview of the **BVAR** package and Section 4 demonstrates its use with an example. Section 5 concludes.

2. Econometric framework

In this Section we briefly introduce VAR models in the Bayesian context and outline the approach to hierarchical prior selection proposed by Giannone *et al.* (2015). For a detailed explanation and more information on Bayesian estimation of VAR models we refer to Koop and Korobilis (2010) and Kilian and Lütkepohl (2017).

2.1. Model specification

VAR models are a generalization of univariate autoregressive (AR) models and are commonly resorted to as tools for investigating the effects of economic shocks. They are based on the notion of dynamic behavior between lagged values of all variables in the model. A VAR model of finite order p may be referred to as VAR(p) model and can be expressed as:

$$\boldsymbol{y}_t = \boldsymbol{a}_o + \boldsymbol{A}_1\boldsymbol{y}_{t-1} + \cdots + \boldsymbol{A}_p\boldsymbol{y}_{t-p} + \boldsymbol{\epsilon}_t \text{ with } \boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \boldsymbol{\Sigma}) \tag{1}$$

where $\boldsymbol{y}_t$ is an $M \times 1$ vector of endogenous variables, $\boldsymbol{a}_0$ is an $M \times 1$ vector of constants, $\boldsymbol{A}_p$ is an $M \times M$ matrix of coefficients and $\boldsymbol{\epsilon_t}$ is a $M \times 1$ vector of exogenous shocks. The number of coefficients to be estimated is $M + M^2p$ and hence rises drastically with the number of included variables and/or lags. This is referred to as the curse of dimensionality and is especially problematic for frequentist estimation. In the Bayesian setting prior beliefs are imposed on the model parameters, circumventing the curse and allowing for larger models with significantly improved out-of-sample prediction accuracy to be estimated (see e.g., Bańbura, Giannone, and Reichlin 2010; Koop 2013).

2.2. Prior specification

Properly informing prior beliefs is critical and naturally open to criticism. Giannone *et al.* (2015) propose setting prior parameters in a data-based fashion, i.e., by treating them as additional parameters. They do so by integrating out the marginal likelihood (ML) of the model and using it as a decision criterion for exploring the parameter space. In their paper the authors theoretically ground this approach and show its accuracy in the estimation of impulse response functions. Via empirical examples Giannone *et al.* (2015) also demonstrate that forecasting accuracy is superior to standard VAR models, performing as well as factor models. Their approach has since been used extensively (see e.g., Alquist, Kilian, and Vigfusson 2013; Miranda-Agrippino and Rey 2015; Altavilla, Giannone, and Lenza 2014).

They consider prior distributions of the commonly used Gaussian-inverse-Wishart family:

$$\beta|\boldsymbol{\Sigma} \sim \mathcal{N}(\boldsymbol{b}, \boldsymbol{\Sigma} \otimes \boldsymbol{\Omega}) \tag{2}$$

$$\boldsymbol{\Sigma} \sim \mathcal{IW}(\boldsymbol{\Psi}, \boldsymbol{d}) \tag{3}$$

where $\boldsymbol{b}$, $\boldsymbol{\Omega}$, $\boldsymbol{\Psi}$ and $\boldsymbol{d}$ are functions of a lower-dimensional vector of hyperparameters $\boldsymbol{\gamma}$. Due to the conjugacy of Equations 2 and 3 the ML of the model can be computed efficiently in closed form as a function of $\boldsymbol{\gamma}$ (see Giannone *et al.* 2015, p. 439). In their paper the authors consider three specific priors – the so-called Minnesota (Litterman) prior, which is used as a baseline, the sum-of-coefficients prior and the single-unit-root prior.

The Minnesota prior (Litterman 1980) essentially imposes the hypothesis that individual variables all follow random walk processes. This parsimonious specification typically performs well in forecasts of macroeconomic time series (Kilian and Lütkepohl 2017, p. 356) and is often used as a benchmark to evaluate accuracy. It is characterized by the following moments:

$$\mathbb{E}[(\boldsymbol{A}_s)_{ij}|\boldsymbol{\Sigma}] = \begin{cases} 1 & \text{if } i = j \text{ and } s = 1 \\ 0 & \text{otherwise} \end{cases}$$

$$cov\left((\boldsymbol{A}_s)_{ij}, (\boldsymbol{A}_r)_{kl}|\boldsymbol{\Sigma}\right) = \begin{cases} \lambda^2 \frac{1}{s^\alpha} \frac{\boldsymbol{\Sigma}_{ik}}{\psi_j/(d-M-1)} & \text{if } l = j \text{ and } r = s \\ 0 & \text{otherwise} \end{cases}$$

The key parameter is λ and controls the tightness of the prior, i.e., it weighs the relative importance of prior and data. For $\lambda \to 0$ the prior is imposed exactly, while as $\lambda \to \infty$ the posterior estimates will approach the ordinary least squares (OLS) estimates. Governing the variance decay with increasing lag order, α controls the punishment of more distant observations. Finally, ψ controls the prior's standard deviation on lags of variables other than the dependent.

Refinements of the Minnesota prior are often implemented as additional priors trying to "reduce the importance of the deterministic component implied by VAR models estimated conditioning on the initial observations" (Giannone *et al.* 2015, p. 440). The first of these, the sum-of-coefficients (SOC) prior (Doan, Litterman, and Sims 1984), imposes the notion that a no-change forecast is optimal at the beginning of a time series. It is implemented via Theil mixed estimation by adding artificial dummy-observations on top of the data matrix, which are constructed as follows:

$$\begin{aligned} \underset{M\times M}{\boldsymbol{y}^+} &= diag\begin{pmatrix} \bar{\boldsymbol{y}} \\ \mu \end{pmatrix} \\ \underset{M\times(1+MP)}{\boldsymbol{x}^+} &= [\boldsymbol{0}, \boldsymbol{y}^+, \ldots, \boldsymbol{y}^+] \end{aligned}$$

where $\bar{\boldsymbol{y}}$ is a $M \times 1$ vector of the averages over the first p observations of each variable. The key parameter μ controls the variance and hence the tightness of the prior, i.e., for $\mu \to \infty$ the prior becomes uninformative. For $\mu \to 0$ the model is pulled towards a form with as many unit roots as variables and no cointegration. This motivates the single-unit-root (SUR) prior (Sims 1993; Sims and Zha 1998), which allows for cointegration relationships in the data. The prior pushes the variables either towards their unconditional mean or towards the presence of at least one unit root. Its associated dummy observations are:

$$\begin{aligned} \underset{1\times M}{\boldsymbol{y}^{++}} &= \frac{\bar{\boldsymbol{y}}}{\delta} \\ \underset{1\times(1+Mp)}{\boldsymbol{x}^{++}} &= \left[\frac{1}{\delta}, \boldsymbol{y}^{++}, \ldots, \boldsymbol{y}^{++}\right] \end{aligned}$$

where $\bar{\boldsymbol{y}}$ is again defined as above. Similarly to before δ is the key parameter, governing the tightness of the SUR prior.

Setting the parameters of these priors has been discussed extensively and a number of heuristics have been proposed (see e.g., Litterman 1980; Doan *et al.* 1984; Bańbura *et al.* 2010). Giannone *et al.* (2015) note that from a Bayesian perspective this choice of parameters is

conceptually identical to the inference on any other parameter of the model. They show that it is possible to treat the model as a hierarchical one, with the marginal likelihood of the data, given the prior parameters, available in closed form for VAR models with conjugate priors (Giannone *et al.* 2015, p. 437). Estimating these hyperparameters via maximization of the ML is an empirical Bayes method with clear frequentist interpretation (Giannone *et al.* 2015).

3. The BVAR package

BVAR implements the hierarchical approach of Giannone *et al.* (2015) into R (R Core Team 2019) and hands the user an easy-to-handle and flexible tool for modeling hierarchical Bayesian VAR models. It can be primarily seen as a toolkit for modern, (macro-)economic multivariate time series analysis. Its hierarchical approach to prior selection serves as a safeguard against problematic parameter choices. Due to its ease of use and flexible nature **BVAR** is ideal for economic analyses in the fashion of Bańbura *et al.* (2010) or Koop (2013) and may be used for consistency checks of similar models written in single-use scripts. It can also serve as an introduction to the Bayesian paradigm in multivariate economic time series modeling.

The package has no dependencies outside base R and **mvtnorm** (Genz, Bretz, Miwa, Mi, Leisch, Scheipl, and Hothorn 2019) and is thus available cross-platform and even on minimal installations. It is implemented in native R for transparency and in order to lower the bar for contributions and/or adaptations. Regardless, a functional approach to the package's structure facilitates future ports of computationally intensive steps to e.g., C. The complete documentation, helper functions to access the multitude of settings and use of standard methods for analysis make the package easy to operate, without sacrificing flexibility.

BVAR features extensive customization options regarding priors to be employed, their parameters and their hierarchical treatment. The Minnesota prior is used as baseline; all of its parameters are adjustable and can be treated hierarchically. Options to include the SOC and/or SUR priors are also readily available. Furthermore, the flexible implementation allows users to specify their own custom priors, as long as they are implemented via Theil mixed estimation, i.e., as dummy-observation priors. Appropriate starting values for hyperparameters are obtained from `optim()` (R Core Team 2019), using the limited-memory BFGS quasi-Newton method. Further options are devoted to the MCMC and specifically the Metropolis-Hastings (MH) algorithm. Naturally, the number of burned and saved draws are adjustable and thinning may be employed. To properly explore the posterior distributions the proposal range of prior parameters is vital and can thus be set individually. Suitable acceptance rates may be achieved by enabling automatic proposal rate adjustments during the burn-in phase, with variable target and adjustment rates.

The primary application for Bayesian VAR models is the structural analysis of (macro-)economic systems using impulse response functions (IRF). These functions serve as a representation of shocks hitting the economic system and are used to analyze the reaction of the model's variables. The exact propagation of these shocks is of great interest, but proper identification is necessary, in order for them to be interpretable in a meaningful way. **BVAR** currently features two of the most common identification schemes – namely short-term zero restrictions and sign restrictions. The former is also known as recursive identification and is achieved via Cholesky decomposition of the variance-covariance matrix $\mathbf{\Sigma}$ (see Kilian and

Lütkepohl 2017, Chapter 8). This approach is computationally cheap and achieves exact identification without the need for too much theory-based presumptions about variable behavior. Only the order of the variables is pivotal, as contemporaneous reactions of certain variables are limited. Sign restrictions (see Kilian and Lütkepohl 2017, Chapter 13) are another popular means of identification, that is available in package, following the approach of Rubio-Ramirez, Waggoner, and Zha (2010). This identification scheme makes some presumptions about the behavior of variables following a certain shock necessary. With increasing dimension of the model this may quickly become challenging. Additionally, identification via sign restrictions comes at the cost of increased parameter uncertainty and a loss of precision for resulting IRF. Another question that can be tackled using VAR models is which variables drive the path of a certain variable after a shock. To help analyze this, forecast error variance decompositions (FEVD) are implemented. They allow for a more detailed structural analysis of the processes determining the behavior of the economic system. Bayesian VAR models also perform very well in forecasting exercises. They have proven to be superior to many other methods (see e.g., Carriero, Kapetanios, and Marcellino 2009; Koop 2013), without needing to induce particular restrictions on the parameters of the model like structural models. At the time, **BVAR** can be used to conduct unconditional forecasts that rival the ones obtained by factor models (Giannone *et al.* 2015). Conditional forecasts or scenario analyses, i.e., where the future path of one or more of the endogenous variables is assumed to be known, will be implemented in the future.

Estimation of Bayesian VAR models using the package can easily be customized via a variety of helper functions and arguments. The default values were chosen to provide a sensible starting point and allow for step-by-step adoption of the package. Analyses are readily accessible to R users – options for plotting traces, densities, residuals, forecasts and impulse responses are available alongside implementations of generic functions, including `summary()`, `predict()`, `irf()` and many more. Final and intermediate outputs are provided in an idiomatic format and feature `print()` methods for a transparent research process. As a result, existing frameworks may be used for further analysis – e.g., **coda** (Plummer *et al.* 2006) for checking convergence properties or **ggplot2** (Hadley 2016) for plotting.

In addition to the features introduced above **BVAR** includes the FRED-QD dataset (McCracken and Ng 2016), licensed under a modified Open Data Commons Attribution License (ODC-BY 1.0). It constitutes one of the largest databases for macroeconomic variables describing the US economy in the post-war period and is perfectly suited to multivariate time series analyses. The dataset features 248 macroeconomic indicators on a quarterly basis, going back as far as Q1 1959, 234 of which are included in the package. It is updated on a regular basis, with the version currently included in the package ranging until Q4 2018. FRED-QD lends itself to studies of a wide range of economic phenomena and is regularly used in benchmarking exercises for newly developed models (e.g., Huber, Koop, and Onorante 2019).

4. An applied example

In this Section we demonstrate the functionalities of **BVAR** with a short, applied example, using a subset of the included FRED-QD dataset (McCracken and Ng 2016). We go through a typical workflow of (1) preparing the data, (2) configuring priors and other aspects of the model, (3) estimation of the model, and finally (4) assessing outputs and plotting results. Further possible applications and examples are available in the Appendix.

4.1. Data preparation

The main function `bvar()` expects input data to be coercible to a numeric matrix, without any missing values. We retrieve six time series from the FRED-QD database – real gross domestic product (GDP) in billions of 2012 dollars, industrial production as an index, total non-farm employment in thousands of persons, the average weekly hours worked in the goods-producing industry, the consumer price index (CPI) for urban consumers as well as the effective federal funds rate in percent. As we want to end up with stable AR processes we transform all variables, except the federal funds rate to achieve stationarity. For GDP and CPI we use yearly log-differences, for industrial production, non-farm employment and the average weekly hours quarterly ones. Figure 1 provides an overview of the transformed time series.

```
R> set.seed(42)
R> library("BVAR")
R> data("fred_qd")
R> df <- fred_qd[, c("GDPC1", "INDPRO", "PAYEMS",
+    "CES0600000007", "CPIAUCSL", "FEDFUNDS")]
R> for (i in c("GDPC1", "CPIAUCSL"))
+    df[5:nrow(df), i] <- diff(log(df[, i]), lag = 4) * 100
R> for(i in c("INDPRO", "PAYEMS", "CES0600000007"))
+    df[2:nrow(df), i] <- diff(log(df[, i]), lag = 1) * 100
R> df <- df[5:nrow(df), ]
```

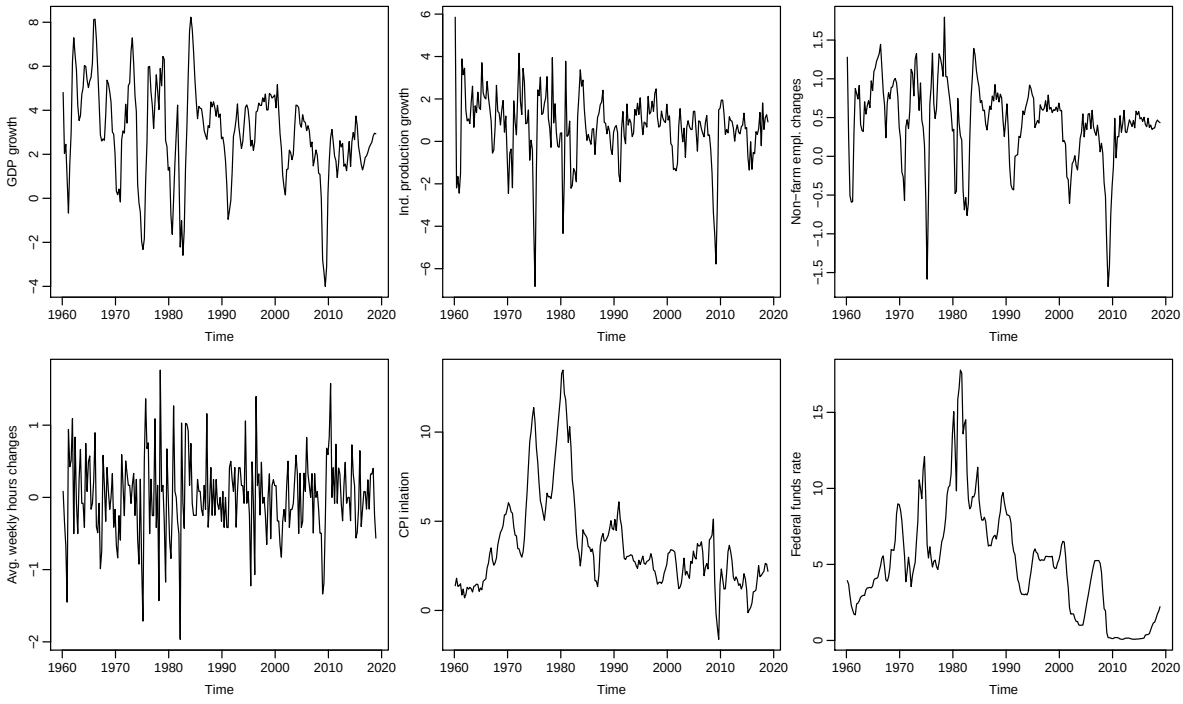

Figure 1: Transformed time series under consideration.

Naturally this selection can be extended with other variables from FRED-QD or other sources.

Also see Chapters 2 and 19 in Kilian and Lütkepohl (2017) for more information on variable transformation.

4.2. Prior setup and further configurations

After preparing the data we need to specify priors, as discussed in Section 2, and adjust other configurations of the model. Note that functions related to the ex-ante setup have the prefix `bv_`. Thus users have a quick and easy way of accessing options and their documentations, which facilitates their use and lowers entry barriers for new users. Methods and functions for analysis in contrast, stick closely to idiomatic R.

As noted, the Minnesota prior is used as a baseline and its key parameter λ is included in the hierarchical modelling exercise per default. We use `bv_minnesota()` to specify prior distributions for the parameters λ and α, as well as upper and lower bounds for proposal values. Following Giannone *et al.* (2015) the distributions are Gamma densities and the bounds are used to discard implausible or impossible values, that are drawn from the Gaussian proposal distribution used in the MH step. The argument `var` specifies the prior variance on the constant term of the model and is generally set to be rather diffuse, i.e., to a large value. We leave $\mathbf{\Psi}$ to be set automatically to the square root of the innovations variance after fitting AR(p) models to the variables.

```
R> mn <- bv_minnesota(
+    lambda = bv_lambda(mode = 0.2, sd = 0.4, min = 0.0001, max = 5),
+    alpha = bv_alpha(mode = 2, sd = 0.25, min = 1, max = 3),
+    var = 1e07)
```

With the SOC and SUR priors we also include the two pre-constructed dummy-observation priors. The priors of the key parameters are also assumed to be Gamma distributed and specification works in the same way as for λ and α. Creation of custom dummy-observation priors is done similarly via `bv_dummy()`, only requiring an additional function to construct the observations (see Appendix A for a demonstration).

```
R> soc <- bv_soc(mode = 1, sd = 1, min = 1e-04, max = 50)
R> sur <- bv_sur(mode = 1, sd = 1, min = 1e-04, max = 50)
```

Once the priors are specified we provide them to `bv_priors()`, making use of the ellipsis argument (`...`) for any dummy-observation priors. Via the argument `hyper` we choose which prior parameters should be treated hierarchically. Its default setting `"auto"` includes λ and the key parameters of all provided dummy-observation priors, which is equivalent to providing the character vector `c("lambda", "soc", "sur")` in our case. Prior parameters not included in the hierarchical step are treated as fixed and set equal to their `mode` parameter.

```
R> priors <- bv_priors(hyper = "auto", mn = mn, soc = soc, sur = sur)
```

Adjustments to the calculation of IRF or forecasts can be made via the functions `bv_irf()` and `bv_fcast()`. The horizon to consider can be adjusted for both. FEVD are calculated in an extra step for the IRF and may be toggled via `fevd`. Identification of the shocks is toggled

via `identification`; it is then performed via Cholesky decomposition unless sign restrictions are provided (see Appendix B for an example). To skip calculation of either and speed up estimation, the argument may be set to `NULL`; both can be calculated ex-post as well.

```
R> irfs  <- bv_irf(horizon = 12, fevd = TRUE, identification = TRUE)
R> fcasts <- NULL
```

Finally, we adjust the MH step to achieve a suitable acceptance rate and thus, explore the posterior distributions of the model's parameters properly. This is done via the `bv_metropolis()` function with the primary argument `scale_hess`, a numeric scalar or vector used for scaling the inverse Hessian that is used to draw proposals for hierarchically treated parameters. This can be complemented by setting `adjust_acc = TRUE`, enabling automatic scale adjustment during the burn-in period. Automatic adjustment is done iteratively by `acc_change` percent until an acceptance rate between `acc_lower` and `acc_upper` is reached.

```
R> mh <- bv_metropolis(scale_hess = 0.005, adjust_acc = TRUE,
+    acc_lower = 0.25, acc_upper = 0.35, acc_change = 0.02)
```

This variety of available settings allows users to tailor their models and all of their components to individual needs. This is necessary for addressing an extensive set of different research questions. However, much simpler and quicker utilization is possible as well – the default settings should suffice for a wide range of applications. This enables users to (1) focus on critical parts of their model and (2) use **BVAR** with ease and gradually fine-tune their models.

4.3. Estimation of the model

At the core of **BVAR** is its main function `bvar()`. After preparing the data and optionally adjusting the various settings of the model it is ready to be estimated. Besides the objects with settings, we provide the number of lags to include in our model and some options regarding the MCMC iterations. In `n_save` we define the total number of iterations, in `n_burn` we set the number of initial iterations to discard and via `n_thin` we denote the denominator of the fraction of draws to store. Furthermore, `verbose = TRUE` prompts printing of intermediate results and enables a progress bar during the MCMC step.

```
R> run <- bvar(df, lags = 5, n_draw = 25000, n_burn = 10000, n_thin = 1,
+    priors = priors, mh = mh, fcast = fcasts, irf = irfs, verbose = TRUE)

Optimisation concluded.
Posterior marginal likelihood: -1123.907
Parameters: lambda = 0.52; soc = 0.9; sur = 0.69
  |==================================================| 100%
Finished after 2.58 mins.
```

The return value of the function is an object of class `bvar` – a named list with several outputs. These always include the primary parameters of interest, i.e., posterior draws of the VAR coefficients, posterior draws of the variance-covariance matrix and posterior draws of the

hyperparameters that were treated hierarchically. Other content includes the values of the marginal likelihood for each draw, optimized starting values of the prior parameters obtained via `optim`, prior settings provided and the ones set automatically, as well as the original call to the `bvar` function. A variety of meta information is included as well, e.g., the number of accepted draws, the variable names and the time spent calculating. Some outputs are only appended if they were actually calculated – namely IRF, FEVD and forecasting results.

4.4. Assessing the results

BVAR provides `print()`, `plot()` and `summary()` methods for objects of type `bvar` and derivatives. The `print()` method provides some meta information, details on hierarchically treated prior parameters and results of optimization via `optim()`. The `summary()` method mimics its counterpart in **vars** (Pfaff 2008) and includes information regarding the log-likelihood of the estimated model, the coefficients of the VAR model and the variance-covariance matrix of its residuals. These are also available via standard methods for `logLik()`, `coef()` and `vcov()`. Further standard methods, such as `fitted()`, `density()` and `residuals()` are available. The assessment of IRF, FEVD and forecasting results works similarly and is discussed later on.

```
R> summary(run)

Bayesian VAR consisting of 231 observations, 6 variables and 5 lags.
Time spent calculating: 2.58 mins
Hyperparameters: lambda, soc, sur
Hyperparameter values after optimisation: 0.517, 0.897, 0.686
Iterations (burnt / thinning): 25000 (10000 / 1)
Accepted draws (rate): 4997 (0.333)

Numeric array (dimensions 31, 6) of coefficient values from a BVAR.
Median values:
                     GDPC1 INDPRO PAYEMS CES0600000007 CPIAUCSL FEDFUNDS
const                0.522  0.278  0.034         0.138    0.063   -0.308
GDPC1-lag1           0.867  0.051  0.036         0.036    0.015   -0.058
INDPRO-lag1          0.210  0.495  0.054         0.072    0.091    0.096
PAYEMS-lag1          0.616  0.482  0.782         0.295    0.090    0.422
CES0600000007-lag1 -0.181  0.047 -0.024        -0.198   -0.069    0.070
CPIAUCSL-lag1      -0.216 -0.049 -0.032         0.019    1.156   -0.054
FEDFUNDS-lag1       0.004 -0.123 -0.036        -0.007    0.140    1.047
GDPC1-lag2          0.009  0.031  0.010        -0.051   -0.052    0.103
INDPRO-lag2        -0.059 -0.124 -0.065        -0.055   -0.043   -0.105
PAYEMS-lag2         0.065 -0.211  0.013        -0.096   -0.004    0.225
CES0600000007-lag2  0.083  0.105  0.039        -0.004   -0.019    0.081
CPIAUCSL-lag2       0.147 -0.112  0.027        -0.027   -0.183    0.178
FEDFUNDS-lag2      -0.212 -0.118 -0.030        -0.101   -0.053   -0.305
GDPC1-lag3         -0.075  0.022 -0.017         0.015    0.016    0.030
INDPRO-lag3         0.109  0.162  0.027         0.014   -0.004    0.043
PAYEMS-lag3        -0.160 -0.208 -0.017        -0.160    0.059   -0.018
```

```
CES0600000007-lag3 -0.038 -0.033 -0.006         0.032   -0.076   -0.082
CPIAUCSL-lag3       0.034 -0.003 -0.005        -0.036    0.015    0.005
FEDFUNDS-lag3       0.133  0.168  0.040         0.063    0.056    0.216
GDPC1-lag4         -0.160  0.038  0.009         0.017    0.070   -0.003
INDPRO-lag4        -0.144 -0.097 -0.025        -0.013   -0.022   -0.093
PAYEMS-lag4        -0.186 -0.060  0.005        -0.084   -0.024   -0.214
CES0600000007-lag4 -0.132 -0.050 -0.014        -0.037   -0.032   -0.003
CPIAUCSL-lag4      -0.044  0.081  0.009         0.039   -0.194   -0.070
FEDFUNDS-lag4       0.027  0.031  0.008         0.055   -0.111    0.011
GDPC1-lag5          0.136 -0.054 -0.009        -0.026   -0.039   -0.010
INDPRO-lag5         0.037  0.025  0.008        -0.014   -0.009    0.018
PAYEMS-lag5         0.055  0.158  0.045         0.016    0.037    0.118
CES0600000007-lag5 -0.054 -0.060 -0.023         0.025   -0.002    0.018
CPIAUCSL-lag5       0.052  0.082  0.011         0.001    0.159    0.045
FEDFUNDS-lag5       0.044 -0.004  0.003        -0.022   -0.028   -0.065

Numeric array (dimensions 6, 6) of variance-covariance values from a BVAR.
Median values:
              GDPC1 INDPRO PAYEMS CES0600000007 CPIAUCSL FEDFUNDS
GDPC1         0.546  0.479  0.108         0.181    0.030    0.112
INDPRO        0.479  1.254  0.216         0.318    0.072    0.313
PAYEMS        0.108  0.216  0.065         0.071    0.024    0.058
CES0600000007 0.181  0.318  0.071         0.233    0.032    0.082
CPIAUCSL      0.030  0.072  0.024         0.032    0.298    0.101
FEDFUNDS      0.112  0.313  0.058         0.082    0.101    0.560

Log-Likelihood: -944.9026
```

Convergence is essential for the stability of MCMC algorithms and hence requires special attention. Beside a suitable acceptance rate of the MH step, trace and density plots of parameters are commonly used to assess convergence. The `plot()` method provides both plots for the ML and hierarchically treated parameters by default (see Figure 2). Note that burnt draws are not included and parameter boundaries are plotted as dashed gray lines. The plotting method also provides a `type` argument to choose between traces, densities or both, as well as arguments to subset plotted hyperparameters (`vars`) or plot coefficient values instead (`vars_response` and `vars_impulse`). See the code below for an example with λ and Figure 3 for the corresponding plot. Visual inspection of Figures 2 and 3 indicate convergence of the key hyperparameters. No glaring outliers are recognizable and there is moderate shrinkage imposed by the Minnesota prior, as most of λ's probability mass is in the range between 0.45 and 0.60. However, one might want to employ additional convergence diagnostics. This is facilitated via a method for **coda**'s (Plummer *et al.* 2006) generic `as.mcmc()` function. See Appendix C for an illustration of the use of further diagnostics, e.g. across multiple chains.

```
R> plot(run)
```

```
R> plot(run, type = "full", vars = "lambda", mfrow = c(2, 1))
```

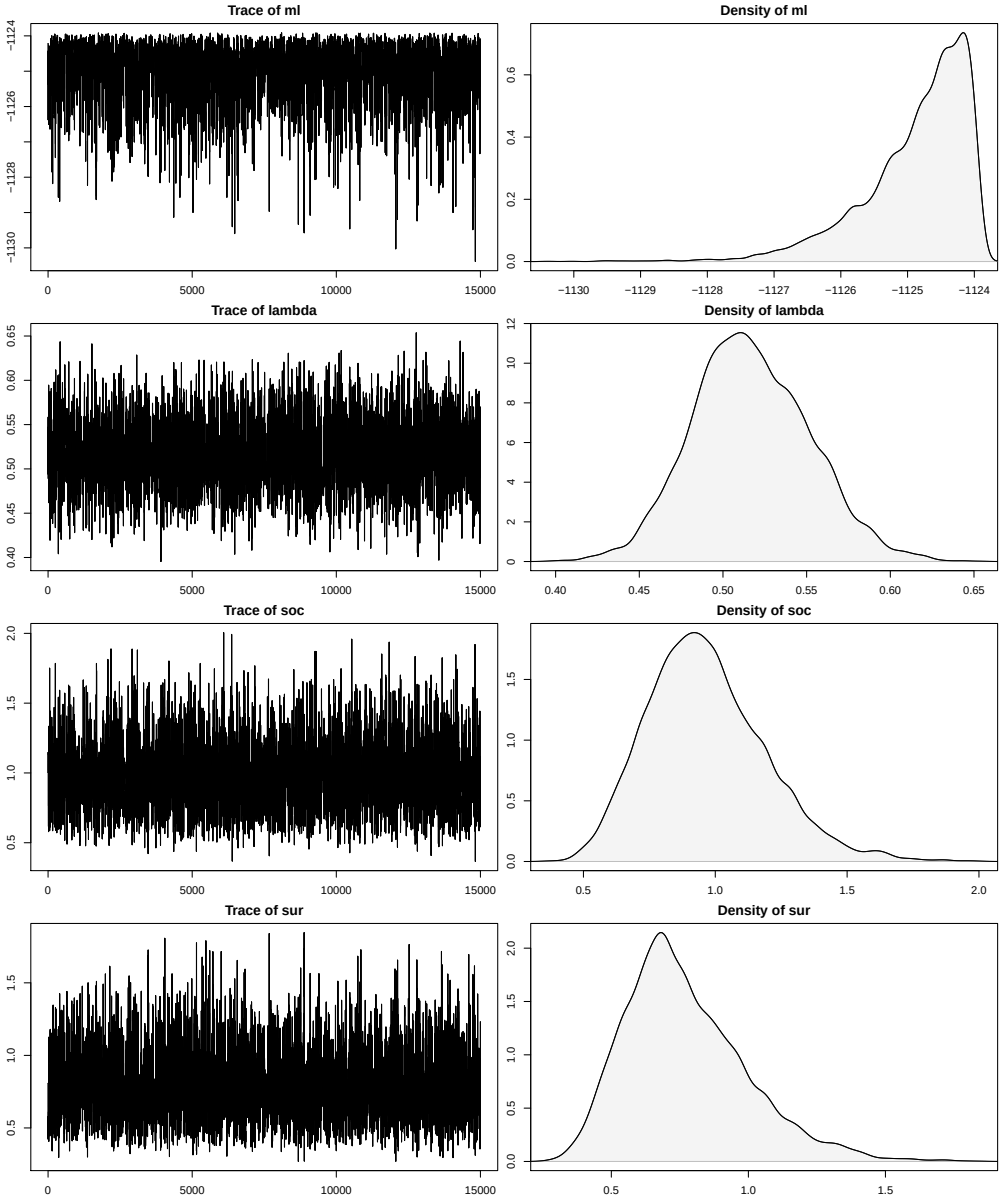

Figure 2: Trace and density plots of all hierarchically treated parameters and the ML.

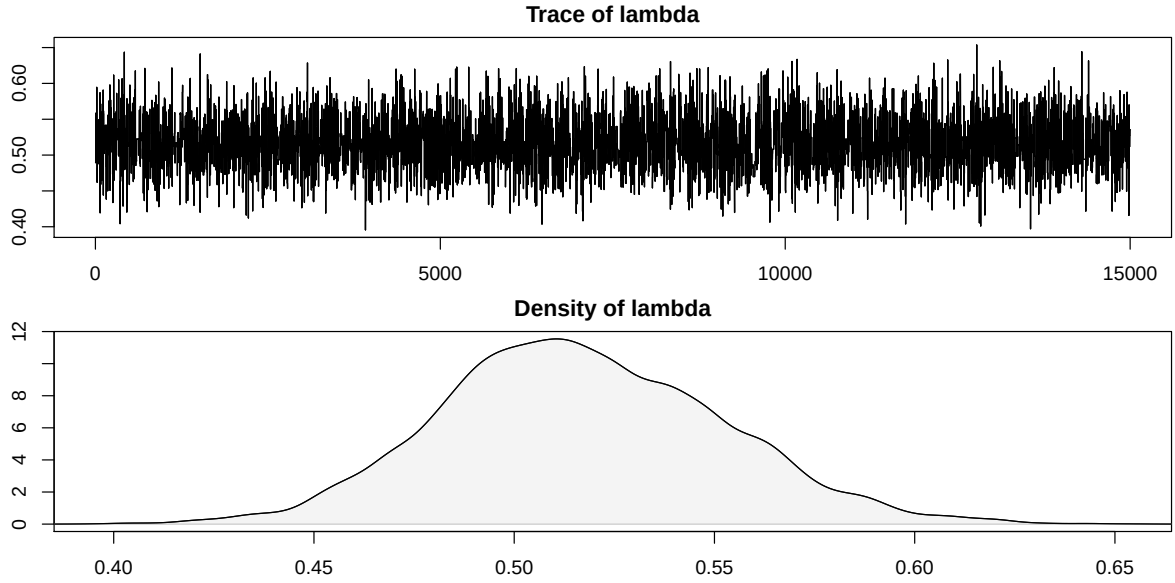

Figure 3: Trace and density plots of λ, the key parameter of the Minnesota prior.

Structural analysis through interpretation of impulse responses is facilitated by **BVAR** in a straightforward fashion. The generic function `irf()` can be used to retrieve or compute IRF from a `bvar` object. The resulting object has methods for plotting, printing and summarizing itself. The `plot()` method has options to subset the plots to specific impulses and/or responses via name or position. An example can be seen below, with the associated output in Figure 4.

```
R> plot(irf(run), vars_impulse = c("GDPC1", "FEDFUNDS"),
+    vars_response = c(1:5))
```

Forecast error variance decompositions constitute another important tool for structural analysis. FEVD draws are provided as an array in the `bvar` object. One way of summarizing them is computing the median over all saved draws and time periods, as demonstrated below using the generic function `fevd()`.

```
R> fevd(run)

Numeric array (dimensions 6, 6) of FEVD values from a BVAR.
Median values:
                GDPC1 INDPRO PAYEMS CES0600000007 CPIAUCSL FEDFUNDS
GDPC1          0.8457 0.0254 0.0050         0.0020   0.0315   0.0821
INDPRO         0.5393 0.3530 0.0040         0.0033   0.0281   0.0544
PAYEMS         0.3811 0.1033 0.1181         0.0080   0.0878   0.2860
CES0600000007 0.3963 0.1171 0.0275         0.2346   0.0271   0.1747
CPIAUCSL       0.0532 0.0313 0.0058         0.0029   0.8236   0.0683
FEDFUNDS       0.0927 0.0601 0.0059         0.0045   0.1739   0.6521
```

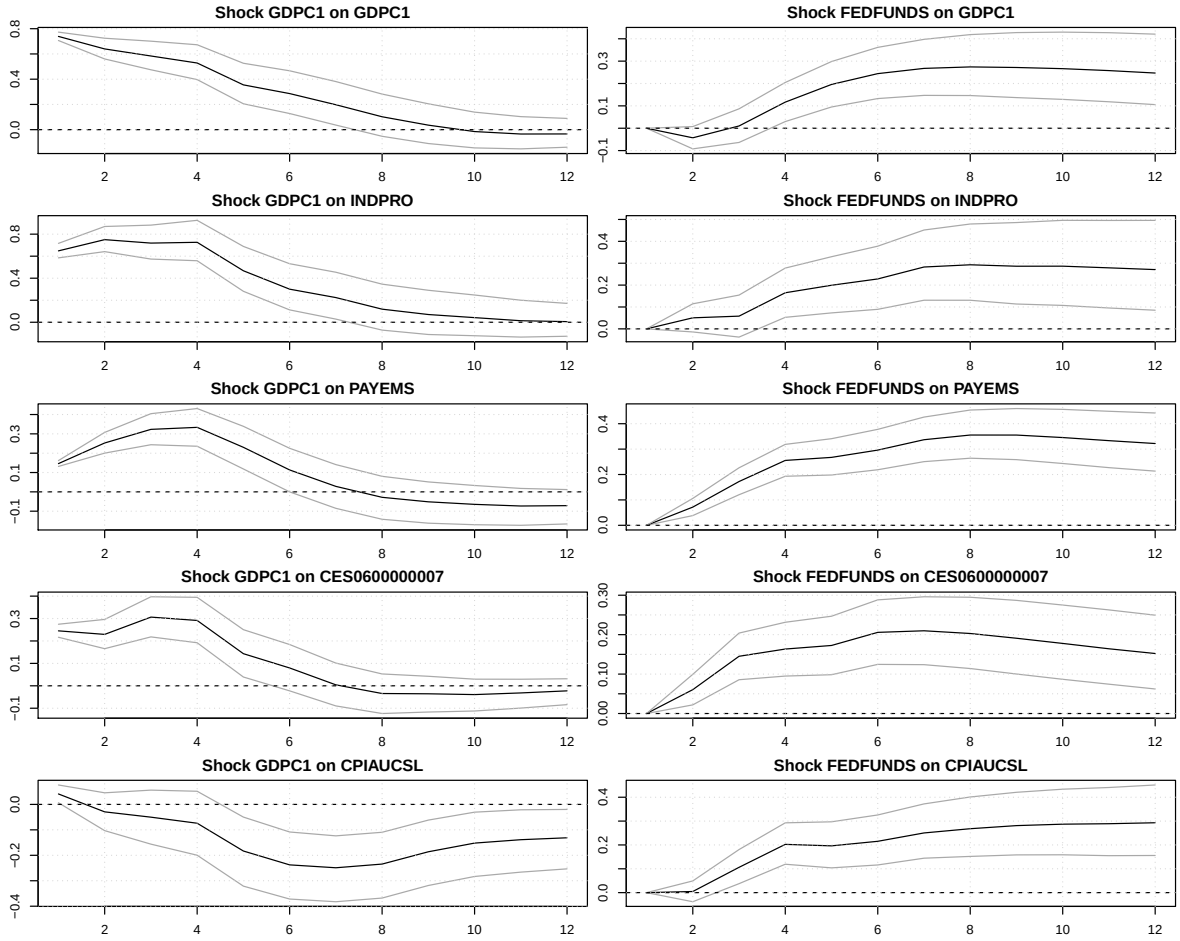

Figure 4: Impulse responses of GDP growth, industrial production, non-farm employment, average weekly hours worked and CPI inflation and to an aggregate demand shock (left panels) and a monetary policy shock (right panels). Grey lines denote 16^{th} and 84^{th} credible sets.

To aid in forecasting exercises **BVAR** provides a `predict()` method with associated methods for plotting, printing and summarizing. Calculation of forecasts is possible within `bvar()` itself or ex-post, using `predict()`. In both cases settings are provided through `bv_fcast()`, although the ellipsis argument may be used for the latter. In the example below we add forecasts to our `bvar` object from before, by assigning the output of `predict()` to `run[["fcast"]]`. We then use the `plot()` method to visualize forecasts for two of the contained variables, by providing their names (alternatively their positions) to `vars`. See Figure 5 for the associated output.

```
R> run[["fcast"]] <- predict(run, horizon = 8)
R> plot(predict(run), vars = c("GDPC1", "FEDFUNDS"),
+    orientation = "vertical")
```

Both `predict()` and `irf()` either retrieve existing forecasts or IRF or calculate them based on stored iterations in the provided `bvar` object. As such they can be used to compute results

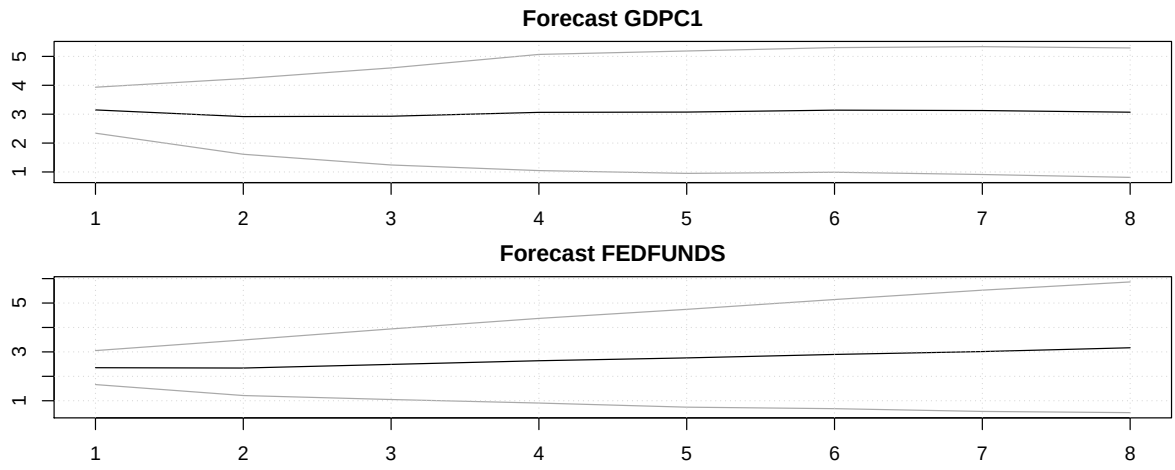

Figure 5: Unconditional forecasts for GDP growth and the federal funds rate. Grey lines denote 16^{th} and 84^{th} credible sets.

using different settings without the need for re-estimating the whole model. Furthermore, the argument `conf_bands` allows the adjustment of credible intervals surrounding the median forecast / IRF, that are used in other methods. See the example below for an ex-post calculation of IRF with increased horizon and adjusted credible intervals. The result is then plotted and can be seen in Figure 6.

```
R> plot(irf(run, conf_bands = 0.05, horizon = 20, fevd = TRUE),
+    vars_impulse = c("GDPC1", "FEDFUNDS"), vars_response = c(1:5))
```

This concludes the brief demonstration of the current functionality via an applied example. Further methods are available and described in the package documentation.

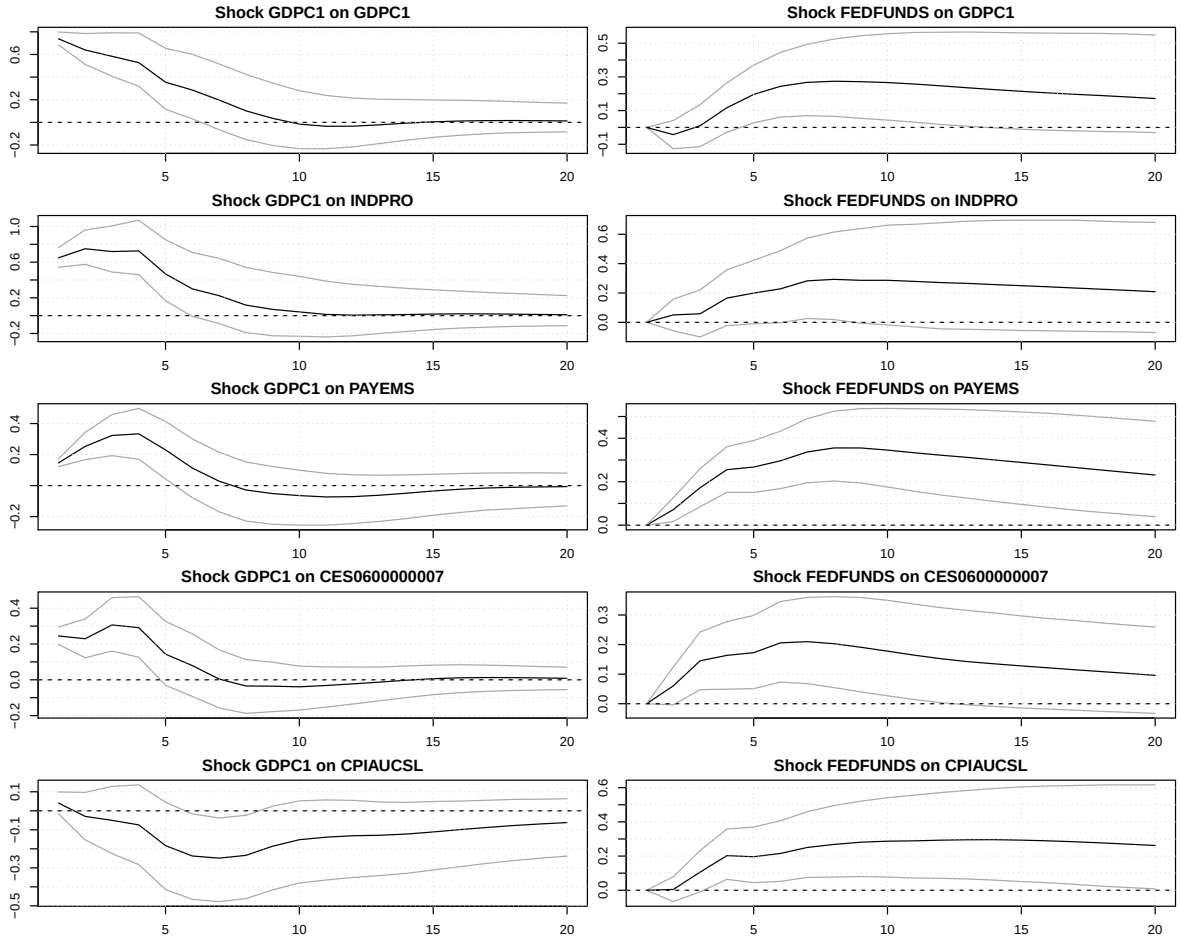

Figure 6: Ex-post computed impulse responses of GDP growth, industrial production, non-farm employment, average weekly hours worked and CPI inflation to an aggregate demand shock (left panels) and a monetary policy shock (right panels) for a horizon of 20 quarters. Grey lines denote 5^{th} and 95^{th} credible sets.

5. Conclusion

This article introduced the **BVAR** package that implements hierarchical estimation of Bayesian vector autoregressions in R. It offers a flexible, yet structured and transparent way to assess a wide range of research questions in the field of multivariate time series analysis. By mitigating the need for subjective choices, regarding prior specifications it counteracts some of the main criticism of Bayesian methods in general. Through the provision of several methods and functionalities the package allows for the quick assessment of various model outputs. With the aid of an applied example we illustrated the usage of the package and explained its implementation and configuration.

BVAR has a lower cost of entry than more general software for Bayesian statistics and includes tools for subsequent analysis of generated VAR models. It is useful for a range of issues and may be applied to a wider range of research questions than similar packages focussing on Bayesian VAR modelling. The idiomatic implementation in R makes the package easy to use and extensible.

Computational details

The results in this paper were obtained using R 3.6.1 with **BVAR** 0.2.1 and the **mvtnorm** 1.0-11 package as singular dependency. The machine used is an Apple MacBook Pro Mid-2012 running MacOS High Sierra Version 10.13 with an Intel Core i7-2.9 GHz and 8 GB RAM. The scripts used were also tested on a machine running Ubuntu 18.04 with an Intel i7-7500U and 16 GB RAM. R itself and all packages used are available from the Comprehensive R Archive Network (CRAN) at `https://CRAN.R-project.org/`.

Acknowledgments

We would like to thank Victor Maus and Florian Huber for helpful comments and encouragement, Michael McCracken and the St. Louis Federal Reserve for their work on FRED-QD, and the JSS editorial team for improvement suggestions. Our thanks also go out to Gregor Kastner, Casper Engelen, and Daran Demirol.

References

Alquist R, Kilian L, Vigfusson RJ (2013). "Forecasting the Price of Oil." In G Elliott, A Timmermann (eds.), *Handbook of Economic Forecasting*, volume 2 of *Handbook of Economic Forecasting*, pp. 427–507. Elsevier. doi:10.1016/B978-0-444-53683-9.00008-6. URL http://www.sciencedirect.com/science/article/pii/B9780444536839000086.

Altavilla C, Giannone D, Lenza M (2014). "The Financial and Macroeconomic Effects of OMT Announcements." *CEPR Discussion Papers*. URL https://ssrn.com/abstract=2501497.

Bańbura M, Giannone D, Reichlin L (2010). "Large Bayesian Vector Auto Regressions." *Journal of Applied Econometrics*, **25**(1), 71–92. doi:10.1002/jae.1137.

Bivand R, Gómez-Rubio V, Rue H (2015). "Spatial Data Analysis with **R-INLA** with Some Extensions." *Journal of Statistical Software*, **63**(20), 1–31. ISSN 1548-7660. doi:10.18637/jss.v063.i20. URL https://www.jstatsoft.org/v063/i20.

Blangiardo M, Cameletti M, Baio G, Rue H (2013). "Spatial and Spatio-Temporal Models with **R-INLA**." *Spatial and Spatio-temporal Epidemiology*, **4**, 33–49. doi:10.1016/j.sste.2012.12.001.

Carpenter B, Gelman A, Hoffman MD, Lee D, Goodrich B, Betancourt M, Brubaker M, Guo J, Li P, Riddell A (2017). "Stan: A Probabilistic Programming Language." *Journal of Statistical Software*, **76**(1), 1–32. ISSN 1548–7660. doi:10.18637/jss.v076.i01.

Carriero A, Kapetanios G, Marcellino M (2009). "Forecasting Exchange Rates with a Large Bayesian VAR." *International Journal of Forecasting*, **25**(2), 400–417.

Crespo Cuaresma J, Feldkircher M, Huber F (2016). "Forecasting with Global Vector Autoregressive Models: A Bayesian Approach." *Journal of Applied Econometrics*, **31**(7), 1371–1391. doi:10.1002/jae.2504.

Doan T, Litterman R, Sims C (1984). "Forecasting and Conditional Projection Using Realistic Prior Distributions." *Econometric Reviews*, **3**(1), 1–100.

Enders W, Sandler T (1993). "The Effectiveness of Antiterrorism Policies: A Vector-Autoregression-Intervention Analysis." *American Political Science Review*, **87**(4), 829–844.

Gelman A, Rubin DB (1992). "Inference from Iterative Simulation Using Multiple Sequences." *Statistical Science*, **7**(4), 457–472. doi:10.1214/ss/1177011136.

Genz A, Bretz F, Miwa T, Mi X, Leisch F, Scheipl F, Hothorn T (2019). ***mvtnorm***: *Multivariate Normal and t Distributions*. R package version 1.0-11, URL https://CRAN.R-project.org/package=mvtnorm.

Geweke J (1992). "Evaluating the Accuracy of Sampling-Based Approaches to the Calculations of Posterior Moments." *Bayesian Statistics*, **4**, 641–649.

Giannone D, Lenza M, Primiceri GE (2015). "Prior Selection for Vector Autoregressions." *Review of Economics and Statistics*, **97**(2), 436–451. doi:10.1162/REST_a_00483.

Golding N, *et al.* (2018). ***Greta**: Simple and Scalable Statistical Modelling in R*. R package Version 0.3.0, URL https://CRAN.R-project.org/package=greta.

Hadfield JD (2010). "MCMC Methods for Multi-Response Generalized Linear Mixed Models: The **MCMCglmm** R Package." *Journal of Statistical Software*, **33**(2), 1–22. doi:10.18637/jss.v033.i02. URL http://www.jstatsoft.org/v33/i02/.

Hadley W (2016). ***ggplot2**: Elegant Graphics for Data Analysis*. Springer-Verlag New York. ISBN 978-3-319-24277-4. URL https://ggplot2.tidyverse.org.

Hoffman MD, Gelman A (2014). "The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo." *Journal of Machine Learning Research*, **15**(1), 1593–1623. URL http://jmlr.org/papers/v15/hoffman14a.html.

Huber F, Koop GM, Onorante L (2019). "Inducing Sparsity and Shrinkage in Time-Varying Parameter Models." *arXiv preprint*. URL https://arxiv.org/abs/1905.10787.

Kilian L, Lütkepohl H (2017). *Structural Vector Autoregressive Analysis*. Cambridge University Press.

Koop GM (2013). "Forecasting with Medium and Large Bayesian VARs." *Journal of Applied Econometrics*, **28**(2), 177–203. doi:10.1002/jae.1270.

Koop GM, Korobilis D (2010). "Bayesian Multivariate Time Series Methods for Empirical Macroeconomics." *Foundations and Trends in Econometrics*, **3**(4), 267–358. doi:10.1561/0800000013.

Krueger F (2015). ***bvarsv**: Bayesian Analysis of a Vector Autoregressive Model with Stochastic Volatility and Time-Varying Parameters*. R package version 1.1, URL https://CRAN.R-project.org/package=bvarsv.

Litterman RB (1980). "A Bayesian Procedure for Forecasting with Vector Autoregressions." *MIT Working Paper*.

Lunn DJ, Spiegelhalter D, Thomas A, Best N (2009). "The BUGS Project: Evolution, Critique and Future Directions." *Statistics in Medicine*, **28**(25), 3049–3067. doi:10.1002/sim.3680.

Lunn DJ, Thomas A, Best N, Spiegelhalter D (2000). "WinBUGS – A Bayesian Modelling Framework: Concepts, Structure, and Extensibility." *Statistics and Computing*, **10**(4), 325–337. doi:10.1023/a:1008929526011.

McCracken MW, Ng S (2016). "FRED-MD: A Monthly Database for Macroeconomic Research." *Journal of Business & Economic Statistics*, **34**(4), 574–589. doi:10.1080/07350015.2015.1086655.

Miranda-Agrippino S, Rey H (2015). "World Asset Markets and the Global Financial Cycle." *NBER Working Papers*, (21722). URL https://ssrn.com/abstract=2691240.

Pfaff B (2008). "VAR, SVAR and SVEC models: Implementation Within R Package **vars**." *Journal of Statistical Software*, **27**(4), 1–32. ISSN 1548-7660. doi:10.18637/jss.v027.i04. URL https://www.jstatsoft.org/v027/i04.

Plummer M (2003). "JAGS: A Program for Analysis of Bayesian Graphical Models Using Gibbs Sampling." In *Proceedings of the 3rd International Workshop on Distributed Statistical Computing*. ISSN 1609-395X. URL http://www.ci.tuwien.ac.at/Conferences/DSC-2003/.

Plummer M, Best N, Cowles K, Vines K (2006). "**CODA**: Convergence Diagnosis and Output Analysis for MCMC." *R News*, **6**(1), 7–11. URL https://journal.r-project.org/archive/.

Ram K (2013). "Git Can Facilitate Greater Reproducibility and Increased Transparency in Science." *Source Code for Biology and Medicine*, **8**(1), 7. doi:10.1186/1751-0473-8-7.

R Core Team (2019). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Rubio-Ramirez JF, Waggoner DF, Zha T (2010). "Structural Vector Autoregressions: Theory of Identification and Algorithms for Inference." *Review of Economic Studies*, **77**(2), 665–696. doi:10.1111/j.1467-937X.2009.00578.x.

Rue H, Martino S, Chopin N (2009). "Approximate Bayesian Inference for Latent Gaussian Models by Using Integrated Nested Laplace Approximations." *Journal of the Royal Statistical Society*, **71**(2), 319–392. doi:10.1111/j.1467-9868.2008.00700.x.

Rue H, Martino S, Lindgren F, *et al.* (2015). ***R-INLA**: Approximate Bayesian Inference Using Integrated Nested Laplace Approximations*. R package version 3.6, URL http://www.r-inla.org/.

Salvatier J, Wiecki TV, Fonnesbeck C (2016). "Probabilistic Programming in Python Using **PyMC3**." *PeerJ Computer Science*, **2**, e55. ISSN 2376–5992. doi:10.7717/peerj-cs.55.

Sims CA (1980). "Macroeconomics and Reality." *Econometrica: Journal of the Econometric Society*, pp. 1–48.

Sims CA (1993). "A Nine-Variable Probabilistic Macroeconomic Forecasting Model." In *Business Cycles, Indicators and Forecasting*, pp. 179–212. University of Chicago Press. URL https://www.nber.org/chapters/c7192.

Sims CA, Zha T (1998). "Bayesian Methods for Dynamic Multivariate Models." *International Economic Review*, pp. 949–968. URL https://www.jstor.org/stable/2527347.

Wild B, Eichler M, Friederich HC, Hartmann M, Zipfel S, Herzog W (2010). "A Graphical Vector Autoregressive Modelling Approach to the Analysis of Electronic Diary Data." *BMC Medical Research Methodology*, **10**(1), 28. doi:10.1186/1471-2288-10-28.

A. Construction of custom dummy priors

This Section demonstrates the construction of custom dummy priors using `bv_dummy`. As an exemplary case the SOC prior is reconstructed without using the helper function `bv_soc()`.

Most importantly, `bv_dummy()` has to be provided with a function to construct artificial observations. This function takes three parameters – the data in matrix format, an integer with the number of lags and the current value of the prior parameter. The function then returns a list with two numeric matrices, `X` and `Y` with artificial observations to stack on top of the data matrix and the lagged data matrix. For the SOC prior this follows the procedure stated in Section 2.2 and is done as follows:

```
R> add_soc <- function(Y, lags, par) {
+    soc <- if(lags == 1) {diag(Y[1, ]) / par} else {
+      diag(colMeans(Y[1:lags, ])) / par
+    }
+    Y_soc <- soc
+    X_soc <- cbind(rep(0, ncol(Y)),
+      matrix(rep(soc, lags), nrow = ncol(Y)))
+    return(list("Y" = Y_soc, "X" = X_soc))
+  }
```

This function is then passed to `bv_dummy()` via the argument `fun`. Similar to the constructor functions of the other prior parameters, values determining their prior distribution and suitable boundaries need to be provided as well. The output is then passed to the ellipsis parameter of `bv_priors()` with a name.

```
R> soc <- bv_dummy(mode = 1, sd = 1, min = 0.0001, max = 50, fun = add_soc)
R> priors_dum <- bv_priors(hyper = "auto", soc = soc)
```

The resulting object is then provided to the main function `bvar`, which extends the VAR with the new dummy prior. Draws of this prior are stored similarly to ones of the Minnesota prior and can be analyzed in the same way.

B. Identification via sign restrictions

In this Section we show how to perform identification of the VAR via sign restrictions. For the sake of conciseness the example below is reduced to include only three variables, a subset from the ones utilized in the main part of the paper.

The variables under investigation constitute a prototypical monetary VAR model – covering GDP growth, CPI inflation and the federal funds rate. The variables and the transformations applied are equal to the ones described in Section 4.1.

```
R> data("fred_qd")
R> df <- fred_qd[, c("GDPC1", "CPIAUCSL", "FEDFUNDS")]
R> for(i in c("GDPC1", "CPIAUCSL"))
+    df[5:nrow(df), i] <- diff(log(df[, i]), lag = 4) * 100
R> df <- df[5:nrow(df), ]
```

To identify impulse responses via sign restrictions one has to come up with suitable signs corresponding to the expected responses of all variables following a shock from any variable. These expectations are usually formed by and derived from economic theory. To apply this kind of identification in **BVAR** the argument `sign_restr` of the function `bv_irf()` has to be provided with a matrix of such sign restrictions. An element SR_{ij} of this matrix is then set to 1 (-1) if the contemporaneous response of variable i to a shock from variable j is an increase (decrease). For an agnostic view 0 is assigned to an element, imposing no restrictions on the sign of the contemporaneous reaction. The sign restrictions may also be visualized using the associated print method.

```
R> signs <- matrix(c(1, 1, 1, 0, 1, 1, -1, -1, 1), ncol = 3)
R> irf_signs <- bv_irf(horizon = 12, fevd = TRUE,
+    identification = TRUE, sign_restr = signs)
```

The resulting object is then provided to the main function `bvar()`, which calculates impulse response functions based on suitable shocks. These shocks are drawn following an algorithm proposed by Rubio-Ramirez *et al.* (2010), which increases computation time. Outputs are again accessed in the usual way and include information on the chosen sign restrictions. Figure 7 provides a visualization of obtained impulse responses. As can be discerned, shocks identified via sign restrictions allow for contemporaneous effects between all variables, other than ones identified by recursive identification. Furthermore, the instantaneous impacts are in accordance with the sign restrictions. However, one should note that the credible sets surrounding the median impulse response tend to get inflated, due to the additional uncertainty introduced by drawing random orthogonal matrices.

```
R> run_signs <- bvar(small_VAR, lags = 5, n_draw = 25000, n_burn = 10000,
+    priors = priors, mh = mh, fcast = fcasts, irf = irf_signs)
R> print(run_signs)

Bayesian VAR consisting of 231 observations, 3 variables and 5 lags.
Time spent calculating: 2.24 mins
Hyperparameters: lambda, soc, sur
Hyperparameter values after optimisation: 0.743, 0.388, 0.29
Iterations (burnt / thinning): 25000 (10000 / 1)
Accepted draws (rate): 3739 (0.249)

R> print(irf(run_signs))

Impulse response object from `bvar()`.
Horizon: 12
Identification: Sign restrictions
Chosen restrictions:
                        Shock to
                 Var1  Var2  Var3
Response of Var1    +     0     -
```

```
            Var2    +     +     -

            Var3    +     +     +
FEVD: TRUE
Variables: 3
Iterations: 15000
```

```
R> plot(irf(run_signs))
```

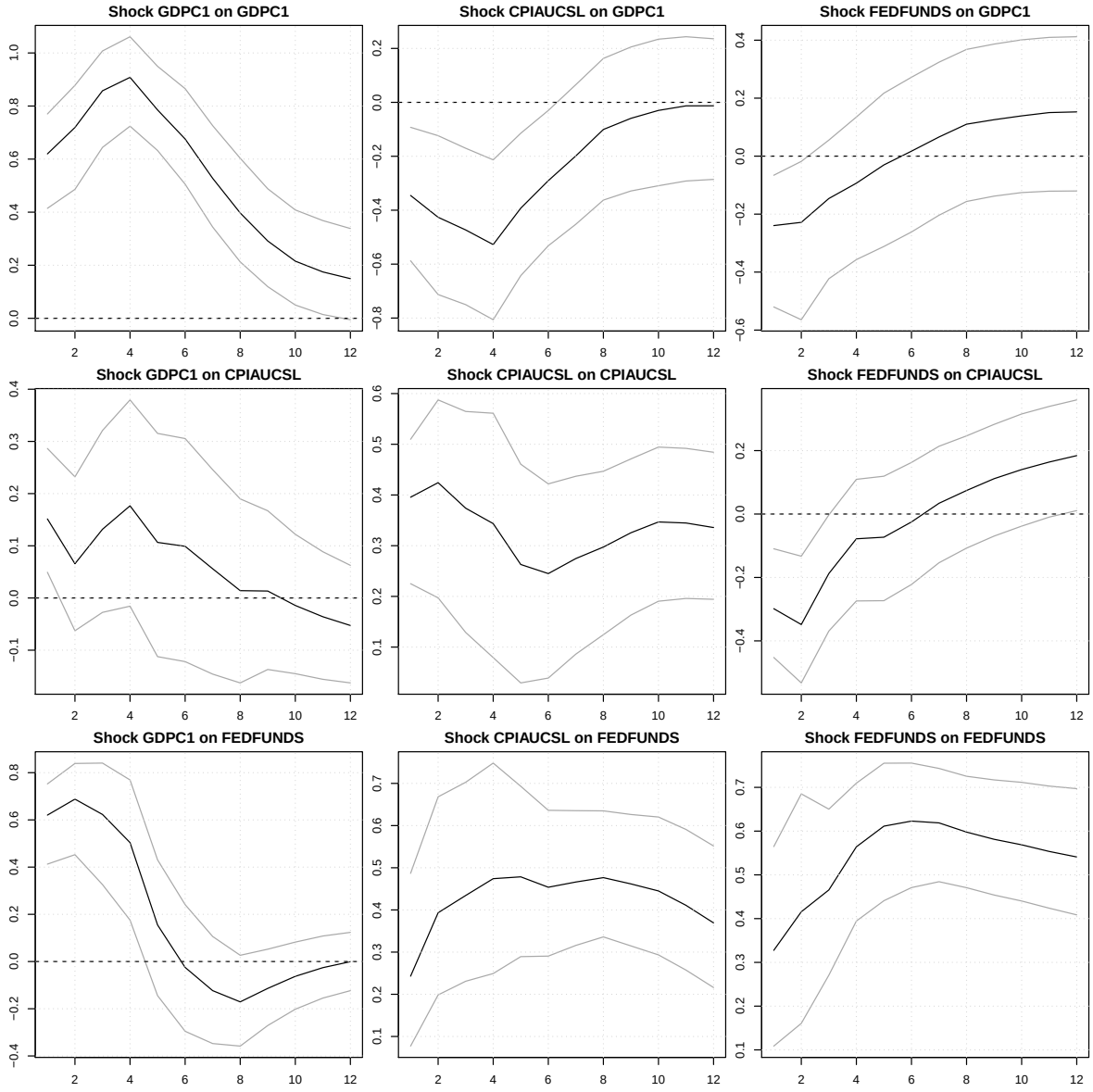

Figure 7: Impulse responses from a prototypical monetary VAR using sign restrictions to achieve identification.

C. Convergence assessment and parallelization

Bayesian methodology relies heavily on the convergence behavior of parameters. **BVAR** provides several functions and methods to facilitate assessing convergence. In this Section we present these functionalities and the use of **coda** (Plummer *et al.* 2006), which specializes in output assessment from Markov Chain Monte Carlo (MCMC) simulations.

First and foremost, the convergence of `bvar` objects can be assessed visually with the `plot()` method (see Figure 2). By default the method plots traces and densities of the marginal likelihood and hyperparameters, which can be narrowed down via the `vars` argument. The arguments `vars_response` and `vars_impulse` can be used to assess the behavior of specific coefficients. Furthermore, the method can be used to plot the results of multiple chains, i.e. `bvar` objects, by providing a list of them to the `chains` argument. The need for multiple chains makes parallelization attractive, which we demonstrate using the **parallel** (R Core Team 2019) package. We replicate the `run` object from the main part of the paper thrice and use the four resulting chains to assess the convergence of λ:

```
R> library("parallel")
R> n_cores <- 3
R> cl <- makeCluster(n_cores)
R> runs <- parLapply(cl, list(df, df, df),
+    function(x) {
+      library("BVAR")
+      bvar(x, lags = 5,
+      n_draw = 25000, n_burn = 10000, n_thin = 1,
+      priors = bv_priors(soc = bv_soc(), sur = bv_sur()),
+      mh = bv_mh(scale_hess = 0.005, adjust_acc = TRUE, acc_change = 0.02),
+      irf = bv_irf(horizon = 12, fevd = TRUE, identification = TRUE),
+      fcast = NULL, verbose = FALSE)
+    })
R> stopCluster(cl)
R> plot(run, type = "full", vars = "lambda", chains = runs)
```

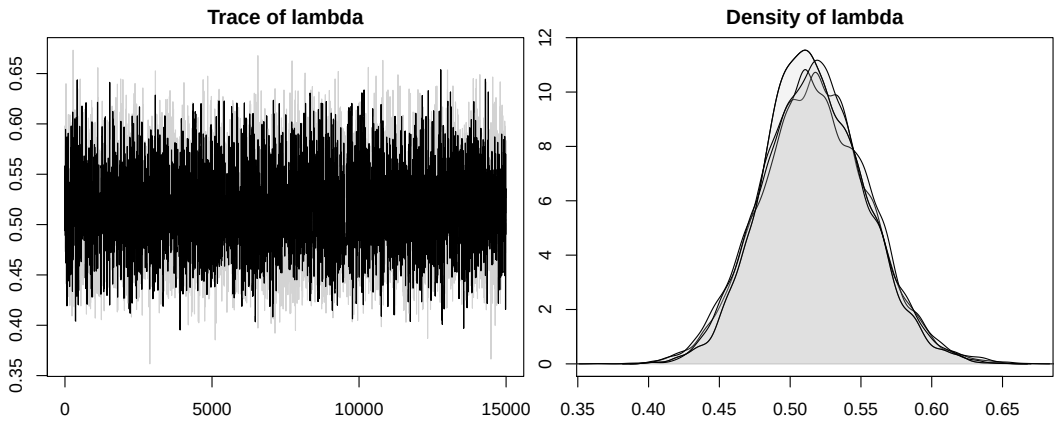

Figure 8: Plots of λ for separate runs.

A quick inspection of Figure 8 indicates proper convergence of the parameter. However, visual assessment may be not sufficient. One may want to evaluate the convergence behavior of parameters by means of diagnostic statistics, such as the one proposed by Geweke (1992). The **coda** (Plummer *et al.* 2006) package provides an implementation of this statistic, that evaluates convergence of a chain by testing for equality of means within certain parts of a Markov chain. To evaluate our chain we prepare our `bvar` object by calling the `as.mcmc()` method on it. This method works similarly to `plot`, including the same arguments `vars`, `vars_response` and `vars_impulse` to subset hyperparameters or coefficients:

```
R> library("coda")
R> run_mcmc <- as.mcmc(run)
R> geweke.diag(run_mcmc)

Fraction in 1st window = 0.1
Fraction in 2nd window = 0.5

     ml  lambda       soc       sur
1.47292  1.11516  0.01498  -0.19418
```

The values displayed constitute standard Z-scores and indicate proper within-chain convergence of all three hyperparameters. Still, one may be interested in the behavior across chains. One diagnostic to properly assess between-chain convergence was proposed by Gelman and Rubin (1992) and is also available in **coda** (Plummer *et al.* 2006). This diagnostic relies on multiple chains, leading us to use `runs`, the list of `bvar` objects we created earlier. Note that the chains should not differ in priors or other settings. To apply the `gelman.diag()` function we need to convert our objects to an `mcmc.list` object – we do so by providing our list of `bvar` objects to the `chains` argument of the `as.mcmc()` method:

```
R> runs_mcmc <- as.mcmc(run, chains = runs)
R> gelman.diag(runs_mcmc, autoburnin = FALSE)

Potential scale reduction factors:

       Point est. Upper C.I.
ml           1.01       1.02
lambda       1.00       1.00
soc          1.00       1.00
sur          1.00       1.00

Multivariate psrf

1.01
```

Affiliation:

Nikolas Kuschnig
Vienna University of Economics and Business
Institute for Ecological Economics
Global Resource Use
Welthandelsplatz 1
1020 Vienna, Austria
E-mail: nikolas.kuschnig@wu.ac.at