Bayesian Analyses of Structural Vector Autoregressions with Sign, Zero, and Narrative Restrictions Using the R Package bsvarSIGNs (Version 2.0)

Xiaolei Wang
University of Melbourne

Tomasz Woźniak
University of Melbourne

Abstract

The R package **bsvarSIGNs** implements state-of-the-art algorithms for the Bayesian analysis of Structural Vector Autoregressions identified by sign, zero, and narrative restrictions. It offers fast and efficient estimation thanks to the deployment of frontier econometric and numerical techniques and algorithms written in C++. The core model is based on a flexible Vector Autoregression with estimated hyper-parameters of the Minnesota prior and the dummy observation priors. The structural model can be identified by sign, zero, and narrative restrictions, including a novel solution making it possible to use the three types of restrictions at once. The package facilitates predictive and structural analyses using impulse responses, forecast error variance and historical decompositions, forecasting and conditional forecasting, as well as analyses of structural shocks and fitted values. All this is complemented by colourful plots, user-friendly summary functions, and comprehensive documentation. The package was granted the Di Cook Open-Source Statistical Software Award by the Statistical Society of Australia in 2024.

Keywords: Bayesian inference, Structural VARs, Gibbs sampler, sign restrictions, zero restrictions, narrative restrictions, forecasting, structural analysis, R.

1. Introduction

Following the seminal developments proposed by Sims (1980) and Doan, Litterman, and Sims (1984) Structural Vector Autoregressions (SVARs) have become go-to models for empirical research in macroeconomics and finance. This fact is attributed to the model's capacity to accommodate basic features of macroeconomic aggregates, relative simplicity of implementation, and interpretability facilitating a wide range of applications (see, for instance, Kilian and Lütkepohl 2017). The SVARs are models that treat all the variables and endogenously determined in a structural, stochastic, dynamic system capturing the data serial autocorrelation, as well as their temporal and contemporaneous interdependencies. Additional assumptions of linearity and conditional Gaussianity lead to analytical tractability and parsimonious estimation (see, for instance, Lütkepohl 2005; Karlsson 2013). Finally, they can be used to investigate Granger causality, dynamic causal effects of well-isolated shocks, network connections, and forecast future values of the variables of interest. All these features, make them indispensable for decision-making at economic governance and financial institutions.

Bayesian approach applied to SVARs facilitates inference in finite samples, which is particularly important due to a relatively small number of observations given the large dimension of the

parameter space in many applications. In this context, the prior distribution of parameters of the model that must be specified to apply Bayes' rule provides an additional tool to handle identification, sparsity, regularisation, and improve the forecasting performance. Additionally, the reliability of numerical methods used for Bayesian estimation has driven the incorporation of a large number of variables, sophisticated identification strategies, and non-linearity in the model specification. All this comes at a cost of computational complexity to be paid for estimation, inference, and forecasting. Following these developments, macroeconometrics as an academic field became increasingly computational and characterised by data-driven application-specific modelling approaches.

An important development facilitating Bayesian analysis of Vector Autoregressions (VARs) was the application of a matrix-variate normal inverse Wishart prior distribution for the model parameters, namely the autoregressive parameters matrix and error term covariance matrix, proposed by Drèze and Morales (1976) for the multivariate regression model and adapted to VARs by Kadiyala and Karlsson (1997). This natural-conjugate prior combined with the assumption of conditional normality of the error term result in a joint posterior distribution of these parameters being matrix-variate normal inverse Wishart (see, e.g., Woźniak 2016). This posterior distribution has a known analytical form, is easy to sample from, and has become the basis for multiple modelling approaches (see, for instance, Bańbura, Giannone, and Reichlin 2010; Koop 2013; Chan 2020).

This paper and the corresponding R package **bsvarSIGNs** by Wang and Woźniak (2024) provides fast and efficient algorithms for Bayesian analysis of SVARs. It takes advantage of this convenient model formulation and applies frontier econometric and numerical techniques and code written in C++ to ensure blazingly fast computations. Additionally, it offers a great flexibility in choosing the model identification pattern, modifying the prior assumptions, and accessing interpretable tabulated or plotted outputs. Therefore, the package makes it possible to benefit from the best of the two facilities: the convenience of data analysis using R and the computational speed using code written in C++.

More specifically the **bsvarSIGNs** package builds upon a Bayesian VAR model relying on the conditional prior and posterior distribution for the parameter of the model being the normal-inverse Wishart distribution. The models can be further extended by hierarchical prior distribution as in Giannone, Lenza, and Primiceri (2015) featuring the Minnesota and dummy-observation priors by Doan *et al.* (1984), explicit specification of a prior distribution for the rotation of the structural system as in Rubio-Ramírez, Waggoner, and Zha (2010). These additional model features are introduced through their corresponding marginal prior distributions and they are treated as given in the conditional prior specification of the VAR parameters. This construction of the joint prior and posterior distributions allow the models to benefit from both: the analytical convenience of the natural conjugate prior distribution granting computational efficiency and the flexibility of the structural model specification leading to its empirical relevance.

This convenient model construction facilitates the application of a variety of methods for the identification of structural shocks. The **bsvarSIGNs** package implements the identification of the SVARs using sign restrictions as in Rubio-Ramírez *et al.* (2010), zero and sign restrictions proposed by Arias, Rubio-Ramírez, and Waggoner (2018), and narrative sign restrictions introduced by Antolín-Díaz and Rubio-Ramírez (2018). We also prove that all of these restrictions can be used simultaneously and include this novel possibility in the package algorithms.

All of these models are set identified up to a rotation matrix such that the sign, zero, and narrative restrictions hold. Appropriate handling of the orthogonal matrix rotating the structural system is essential as this rotation is not updated with the information from the data and is featured in the asymptotic posterior distribution of the model parameters as

Figure 1: The hexagonal package logo features an impulse response that can be fully reproduced using the **bsvarSIGNs** package following a script available at: github.com/bsvars/hex/blob/main/bsvarSIGNs/bsvarSIGNs.R

shown by Baumeister and Hamilton (2015). Following Rubio-Ramírez *et al.* (2010), all the models provided in the **bsvarSIGNs** package sample the rotation matrix from the uniform distribution over the space of orthogonal matrices, namely the Haar distribution by Stewart (1980). Subsequently, this draw, together with the VAR model parameters, is accepted or not based on whether the sign, zero, or narrative restrictions hold. When the zero and narrative restrictions are used, the accepted draws are resampled using importance weights as in Arias *et al.* (2018) and Antolín-Díaz and Rubio-Ramírez (2018). This approach ensures that the posterior distribution of the model parameters is consistent with the restrictions imposed on the structural shocks and that the coverage of the subspace of orthogonal matrices consistent with the restrictions is optimal.

Additionally, the **bsvarSIGNs** package is aligned regarding objects, workflows, and code structure with the R package **bsvars** by Woźniak (2024a,b) for the Bayesian SVARs identified by exclusion restrictions, heteroskedasticity and non-normality, and they constitute an integrated toolset with more information available at https://bsvars.org. Altogether, these packages offer an unmatched number of SVAR models that will be appropriate for the estimation of the effects of monetary and fiscal policies, analysis of labour, financial, and housing markets, and many other essential applications.

The package implements also a wide range of tools for structural and predictive analyses. The former encompasses the methods comprehensively revised by (Kilian and Lütkepohl 2017, Chapter 4: SVAR Tools) and include the impulse response functions, forecast error variance decompositions, historical decompositions, as well as the basic analysis of fitted values, and structural shocks. The predictive analysis includes Bayesian forecasting implemented through an algorithm sampling from the predictive density of the unknown future values of the dependent variables, and conditional forecasting of a number of variables given other variables' projections. Methods `summary()` and `plot()` support the user in the interpretations and visualisation of such analyses.

The **bsvarSIGNs** package is written for R (R Core Team 2021). However, all of the algorithms are written in C++ implemented thanks to the **Rcpp** by Eddelbuettel, François, Allaire, Ushey, Kou, Russel, Chambers, and Bates (2011), Eddelbuettel, Francois, Allaire, Ushey, Kou, Russell, Ucar, Bates, and Chambers (2024a), and Eddelbuettel (2013). The essential parts of our algorithms rely on linear algebra operations and random matrices generators provided by the C++ library **Armadillo** by Sanderson and Curtin (2016) through the **RcppArmadillo** package by Eddelbuettel, Francois, Bates, Ni, and Sanderson (2024b) and Eddelbuettel and Sanderson

(2014). We use the progress bar from package **RcppProgress** by Forner (2020). Finally, the **bsvarSIGNs** package provides a range of methods for the generics defined by the **bsvars** package by Woźniak (2024a) that use classes from the **R6** package by Chang (2021).

The **bsvarSIGNs** package offers a range of improvements relative to existing software packages. Our algorithms are faster than those from competing packages for the same models or procedures, and we offer a greater range of model specifications and interpretable outputs to report. As we have already pointed out this package is highly-compatible with the **bsvars** package by Woźniak (2024a) in terms of the workflows, objects, and code structure. These two packages differ in terms of the model structures leading to different classes of Bayesian estimation algorithms. While the **bsvarSIGNs** package provides models that lead to independent draws from the posterior distribution of the parameters, the **bsvars** package relies on Monte Carlo Markov Chain methods producing dependent samples.

The most similar existing software packages offering Bayesian SVARs are the R package **BVAR** by Kuschnig and Vashold (2021) and the **BEAR** toolbox by Dieppe and van Roye (2024) for MATLAB. The former implements the same hierarchical model and identification through sign and zero restrictions. The latter implements the sign restrictions and a selection of other structural models. None of them implements narrative restrictions or the possibility of using sign, zero, and narrative restrictions at once. Woźniak (2024b) revises other relevant packages for VAR and SVAR models comprehensively. Here, we only mention those relying on Bayesian inference such as **bvarsv** by Krueger (2015) and **FAVAR** by Chen and Chen (2022) implementing specific SVARs from influential studies by Primiceri (2005) and Bernanke, Boivin, and Eliasz (2005) respectively, and those applying hierarchical modelling to reduced-form VAR models, such as **bvartools** by Mohr (2022), **bayesianVARs** by Gruber (2024), and **BGVAR** by Boeck, Feldkircher, and Huber (2020).

In this context, the **bsvarSIGNs** package offers a unique combination of features significantly enhancing the existing tools for Bayesian SVAR analysis. This is achieved by the appropriate VAR model specification, providing a number of structural identification strategies aligned with that specification, application of dedicated numerical methods, writing algorithms in C++, designing user-friendly workflows, offering a range of methods for structural and predictive empirical analyses and ways of tabular and visual presentation of the results. All this makes the **bsvarSIGNs** package an indispensable tool for empirical macroeconomic and financial research.

2. Workflows for SVAR analysis

The methods proposed in this paper are implemented in the R package **bsvarSIGNs** version 2.0 by Wang and Woźniak (2024). The package offers a simple workflow for the model specification, estimation, forecasting, computation of interpretable quantities, and their summaries and visualizations. This section first presents the basic workflow of the package visualized in Figure 2 and then focuses on its particular steps.

Figure 2 illustrates the basic workflow of the **bsvarSIGNs** package that is determined by the statistical procedures. Working with the package begins with uploading the package which uploads the sample data sets as well, including the matrix `optimism` used by Arias *et al.* (2018). This data matrix is provided as argument to the function `specify_bsvarSIGN` that sets the objects necessary to work with the model, such as the data matrices for dependent and explanatory variables, prior hyper-parameters that are fixed, identification restrictions, the VAR model lag order set to the default value `p = 1`, starting values. They are set as convenient **R6** objects.

The estimation proceeds in two steps. In the first optional one, the hyper-parameters can be estimated using function `estimate_hyper` being a method of the **R6** objects `spec` generated

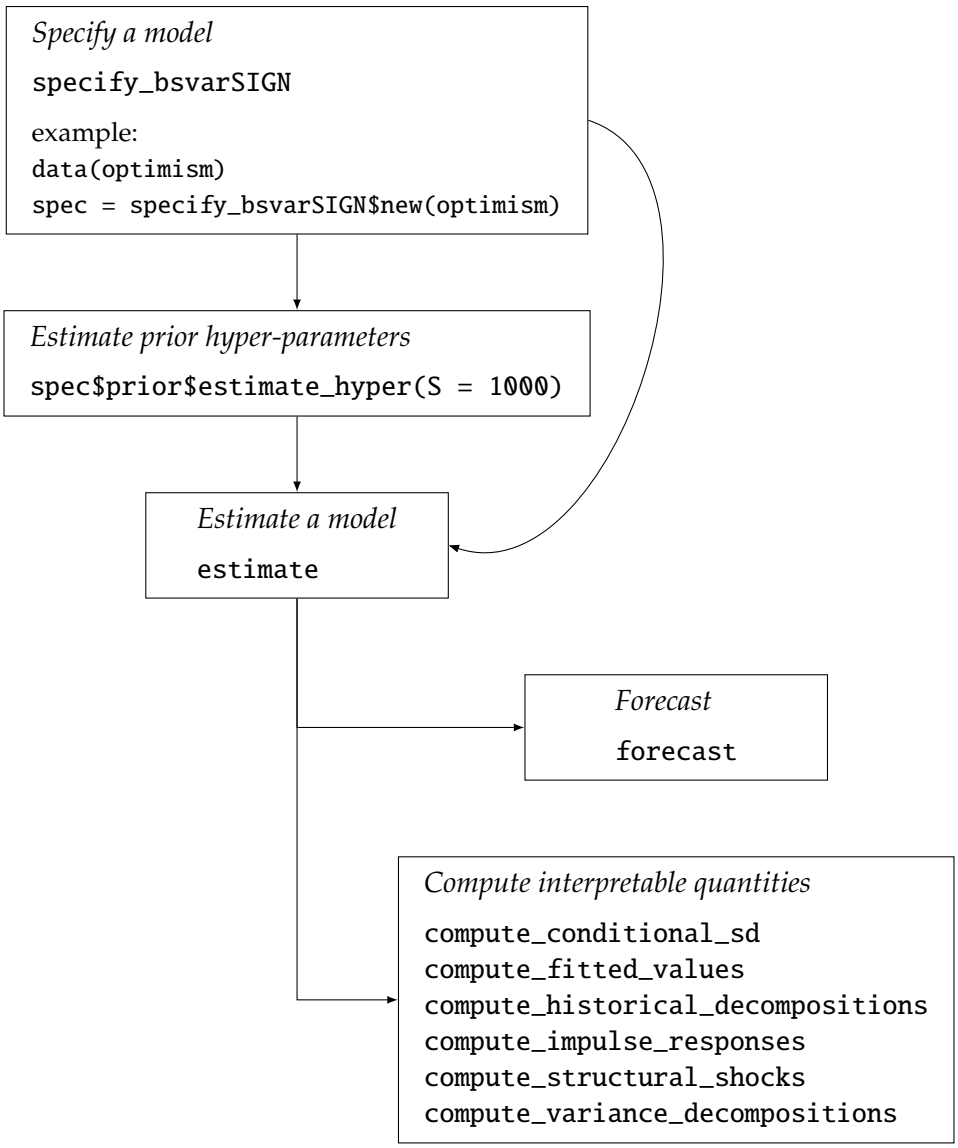


Figure 2: Workflow for SVAR analysis

in the model specification by running `specify_bsvarSIGN`. The hyper-parameter estimation is done marginally on the autoregressive parameters and the covariance matrix following the model structure proposed by Giannone *et al.* (2015). In this step, we use the Adaptive Random Walk Metropolis-Hastings algorithm by Andrieu and Moulines (2006) and Atchadé and Fort (2010) as presented by Kalli and Griffin (2018). It offers scalability, fast convergence, and higher sampling efficiency than the original solution used by Giannone *et al.* (2015). If the user decides not to estimate these hyper-parameters and skip this step, the default values of hyper-parameters proposed by Giannone *et al.* (2015) are used.

Subsequently, for each of the hyper-parameter draws, or given the fixed, that is, not estimated hyper-parameter values, the corresponding draws of the autoregressive parameters and the covariance matrix are sampled independently from the matrix-variate normal inverse Wishart distribution (see Woźniak 2016) using method `estimate.BSVARSIGN`. Here the autoregressive parameters are drawn equation-by-equation as in Carriero, Chan, Clark, and Marcellino (2022) significantly speeding up the computations.

Finally, given the draws of the parameters of the model, the user can compute the corresponding

posterior draws from the predictive density of the future unknown values using method `forecast.PosteriorBSVARSIGN`, or those for a selection of interpretable quantities such as impulse responses, historical decompositions, forecast error variance decompositions, as well as conditional standard deviations, fitted values, and structural shocks using the methods listed on the bottom of Figure 2. Each of these computations can be summarised in terms of the corresponding posterior distribution mean, standard deviation, and quantiles, and visualized using the methods `summary` and `plot` respectively.

2.1. The Basic Workflow

Upload the package to the R environment by executing the following line:

```
R> library(bsvarSIGNs)
```

The package includes sample data object formatted so that they can be provided directly as arguments of functions. The `optimism` data set, downloaded from the replication package of Arias *et al.* (2018), contains the time series formatted as `matrix`. Upload this data object to the memory and display several first observations by

```
R> data(optimism)
R> head(optimism)
```

```
     productivity stock_prices consumption real_interest_rate hours_worked
[1,]    0.2172072    -11.28895   -4.331866        0.008022252    -7.599184
[2,]    0.2129482    -11.17647   -4.324255        0.021877748    -7.592445
[3,]    0.2069894    -11.11805   -4.318455        0.018811208    -7.580577
[4,]    0.2003908    -11.08317   -4.301382        0.012867114    -7.572133
[5,]    0.1945243    -11.02211   -4.293948        0.024503568    -7.577512
[6,]    0.2002138    -11.06306   -4.291474        0.000876887    -7.577491
```

To impose sign restrictions on the impulse response functions, the user needs to specify a $N \times N \times h$ sign restrictions array where h is number of periods. When only imposing restrictions on the contemporaneous responses, a $N \times N$ matrix is also accepted. This sign restriction array only accepts -1, 0, 1 and NA values. Specify -1 if the user wants to impose a negative sign restriction, 0 for a zero restriction, 1 for a positive sign restriction, and NA if the user does not want to impose any restrictions. In each matrix of the restriction array, one can interpret rows as variables and columns as structural shocks. For example, to identify the optimism shock as in Arias *et al.* (2018), the user can impose restrictions on contemporaneous impulse responses of the first shock by:

```
R> sign_irf = matrix(c(0, 1, rep(NA, 23)), 5, 5)
R> sign_irf
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]    0   NA   NA   NA   NA
[2,]    1   NA   NA   NA   NA
[3,]   NA   NA   NA   NA   NA
[4,]   NA   NA   NA   NA   NA
[5,]   NA   NA   NA   NA   NA
```

Specify the model by running a simple function that specifies the model and all the required values, such as the number of lags, data matrices, prior distribution parameters, and some

characteristics for the specific steps of the estimation algorithm. For instance, the code below will use data matrix `optimism` (by setting the first argument `data = optimism`), specify a model with 4 lags (`p = 4`), and the Minnesota prior with prior mean for the autoregressive parameters reflecting unit-root non-stationary of the variables (the default value of the argument `stationary` that is not explicitly stated below), sign restrictions on the contemporaneous impulse responses (`sign_irf = sign_irf`), and save the model specification in the object `spec` that is of class `BSVARSIGN`.

```
R> spec = specify_bsvarSIGN$new(optimism, p = 4, sign_irf = sign_irf)
```

This object contains all the necessary information for the estimation of the model. In particular, following Giannone *et al.* (2015) it includes hyper-parameters μ for the sum-of-coefficients prior, δ for the single-unit-root prior, as well as λ and ψ for the Minnesota prior. By default, they are fixed to the values used by Doan *et al.* (1984).

In a fully Bayesian way, the user can also specify hyper-prior distributions for these hyper-parameters, and sample from their posterior distributions with an adaptive Metropolis algorithm using `spec$prior$estimate_hyper()`. To sample 5,000 burn-in draws and 10,000 posterior draws in the returned sample for all hyper-parameters, the user can run the following code:

```
R> set.seed(123)
R> spec$prior$estimate_hyper(
+   S = 15000,
+   burn_in = 5000,
+   mu = TRUE,
+   delta = TRUE,
+   lambda = TRUE,
+   psi = TRUE
+ )
```

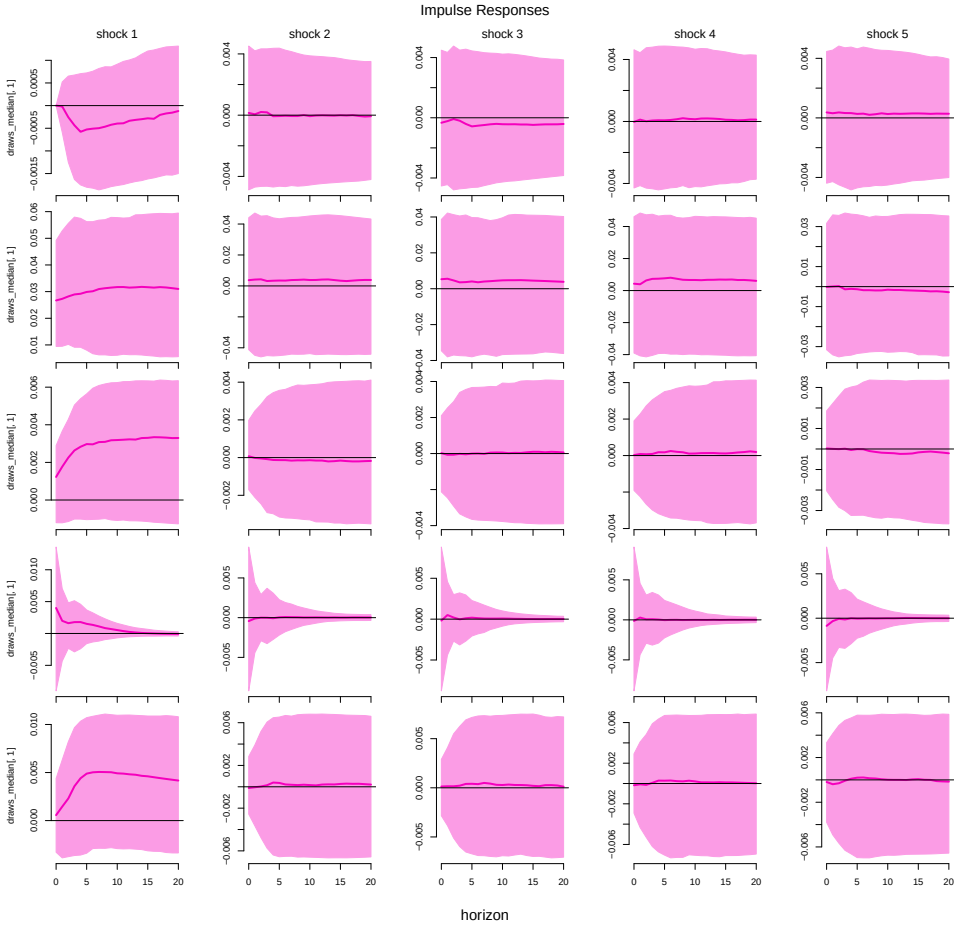
Given hyper-parameters, the VAR model exploits the normal inverse Wishart conjugate prior distribution, which allows for independent sampling from the posterior distribution of the same form without relying on MCMC methods. This is done by passing the `spec` object as the first argument to the function `estimate()`, the following code samples 10,000 independent draws from the posterior distribution and displays the progress bar:

```
R> set.seed(123)
R> post = estimate(spec, S = 10000)

************************************************|
 bsvarSIGNs: Bayesian Structural VAR with sign,   |
             zero and narrative restrictions      |
************************************************|
 Progress of simulation for 10000 independent draws
 Press Esc to interrupt the computations
************************************************|
```

One way to interpret the structural parameters is to compute the impulse response functions of the structural shocks. The user can compute the impulse response functions and plot them by running the following code:

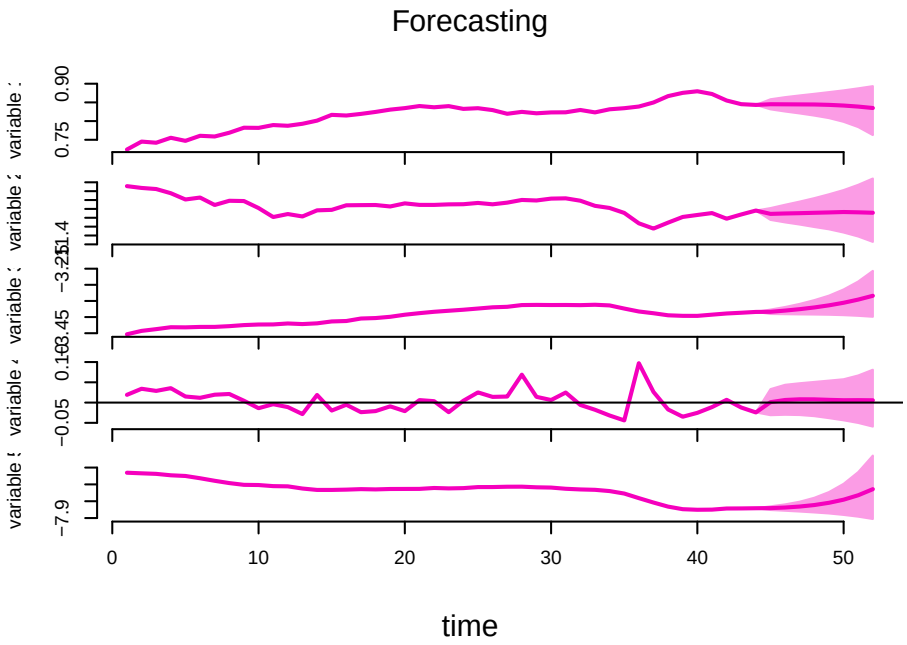
```
R> irf = compute_impulse_responses(post, horizon = 20)
R> plot(irf, probability = 0.68, col = "#F500BD")
```



This IRF plots replicates the results of Arias *et al.* (2018) and shows the response of the variables to a one standard deviation optimism shock. On impact, we see the optimism shock does not affect productivity and there is significant positive response of stock prices by construction, which can be seen in the plot from the fifth row and the first column. The response of the other variables, consumption, real interest rate, and hours worked, are not significantly different from 0.

On the other hand, the user can also compute the forecasts from the reduced-form parameters and plot them by running the following code:

```
R> fore = forecast(post, horizon = 8)
R> plot(fore, data_in_plot = 0.2, col = "#F500BD")
```



In what follows, we focus on particular elements of the workflow describe above, and other interpretations of the model result. Including details on specifying the model, estimating hyper-parameters, estimating the model, structural analysis and predictive analysis.

2.2. Install the Package

The package is available on CRAN and can be installed using the following command:

```
R> install.packages("bsvarSIGNs")
```

To install the development version from GitHub repository github.com/bsvars/bsvarSIGNs, use the following command:

```
R> devtools::install_github("bsvars/bsvarSIGNs")
```

The correct functioning of the **bsvarSIGNs** package requires the installation of the R package **bsvars** by Woźniak (2024a), which will proceed automatically during the installation of the **bsvarSIGNs** package.

Before every use of the package load it to the memory by running the command:

```
R> library(bsvarSIGNs)
```

2.3. Upload Data

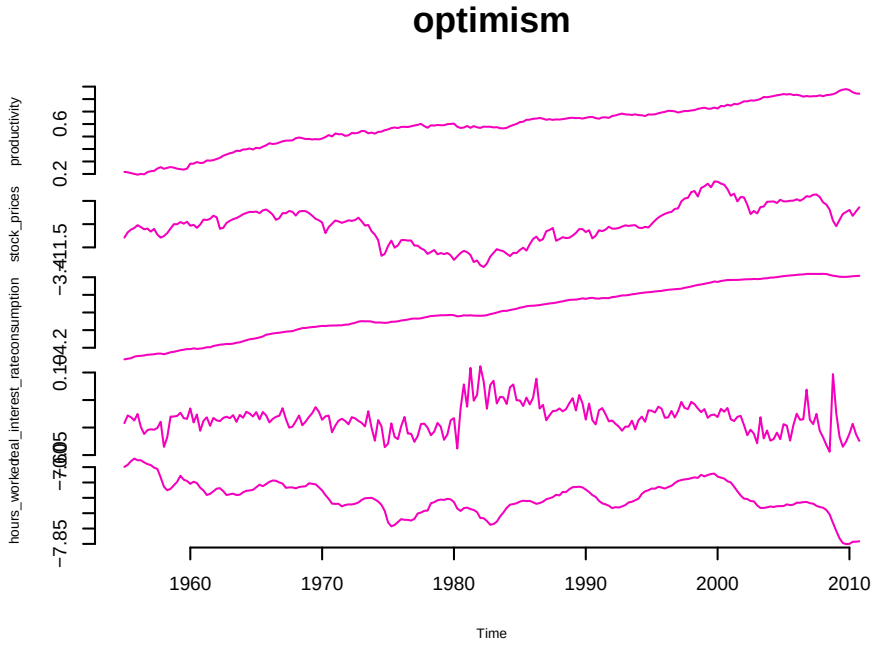
This package includes the `optimism` data set, downloaded from the replication package of Arias *et al.* (2018). It contains the quarterly observations from 1955 Q1 to 2004 Q4 of five variables in the United States:

1. Productivity: quarterly factor-utilization-adjusted total factor productivity.

2. Stock prices: quarterly end-of-period S&P 500 divided by CPI.
3. Consumption: quarterly real consumption expenditures on nondurable goods and services.
4. Real interest rate: quarterly real interest rate.
5. Hours worked: quarterly hours of all persons in the non-farm business sector.

To use the time series, load and visualise them by running the code:

```
R> data(optimism)
R> plot(optimism, col = "#F500BD", nc = 1, bty = "n", cex.lab = 0.45)
```



Finally, more info about the data and object is available in the documentation accessible by running `?optimism`.

2.4. Specify the Model

In specifying the model, there are two main steps. The first step is to specify the sign restrictions. This package allows for three types of restrictions: (i) restrictions on the impulse response functions, (ii) narrative restrictions, and (iii) restrictions on the structural matrix B.

To specify (i) restrictions on the impulse response functions, the user needs to specify the sign restrictions array as described in Section 2.1. For example, to identify the optimism shock as in Arias *et al.* (2018), the user can impose restrictions on contemporaneous impulse responses of the first shock by:

```
R> sign_irf = matrix(c(0, 1, rep(NA, 23)), 5, 5)
R> sign_irf
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]    0   NA   NA   NA   NA
[2,]    1   NA   NA   NA   NA
```

```
[3,]   NA   NA   NA   NA   NA
[4,]   NA   NA   NA   NA   NA
[5,]   NA   NA   NA   NA   NA
```

To specify (ii) narrative restrictions, the user needs to create a list of `narrative` objects using the function `specify_narrative()`. As in Antolín-Díaz and Rubio-Ramírez (2018), we allow for narrative restrictions on the structural shocks and the historical decompositions. For example, to impose a negative optimism shock during the financial crisis in 2008 Q3, the user can specify the narrative restriction by:

```
R> date_fc        = which(time(optimism) == 2008.5)
R> narrative1     = specify_narrative(start = date_fc, sign = -1)
R> sign_narrative = list(narrative1)
```

To specify (iii) restrictions on the structural matrix, the user needs to create a $N \times N$ matrix. Similar to (i), this matrix only accepts values `-1`, `1` and `NA`. Use `-1` to impose a negative sign restriction, `1` for a positive sign restriction, and `NA` if the user does not want to impose any restrictions for the particular period.

The second step in specifying the model is to create the specification object using `specify_bsvarSIGN$new()`, which takes the data object, the number of lags of the VAR model, the specified sign restrictions, and other additional information. For example, to specify a VAR model with 4 lags, sign restrictions on the impulse response functions and narrative restrictions for structural shocks, the user can run the following code:

```
R> spec = specify_bsvarSIGN$new(
+   data           = optimism,
+   p              = 4,
+   sign_irf       = sign_irf,
+   sign_narrative = sign_narrative
+ )
```

Finally, the documentation for the model specification can be accessed by running `?specify_bsvarSIGN` with details on its particular parts, such as the function generating the model specification `specify_bsvarSIGN$new()`, available following the links in the documentation just displayed.

2.5. Estimate Hyper-Parameters

This specification object `spec` contains all the necessary information for the estimation of the model. In particular, following Giannone *et al.* (2015) it includes hyper-parameters μ for the sum-of-coefficients prior, δ for the single-unit-root prior, and λ and ψ for the Minnesota prior. By default, they are fixed as $\mu = 1$, $\delta = 1$, $\lambda = 0.2$, and ψ are fixed as the OLS error variance estimates (see Doan *et al.* 1984).

Before sampling the hyper-parameters, the user can display and alter the prior distribution of the hyper-parameters by changing the default values. The hyper prior of ψ follows an inverse gamma distribution and the hyper prior of other hyper-parameters follows a gamma distribution. For example, to display the prior distribution of the shrinkage parameter λ in the Minnesota prior, the user can run the following code:

```
R> spec$prior$lambda.shape
```

```
[1] 1.370156
```

```
R> spec$prior$lambda.scale
```

```
[1] 0.5403124
```

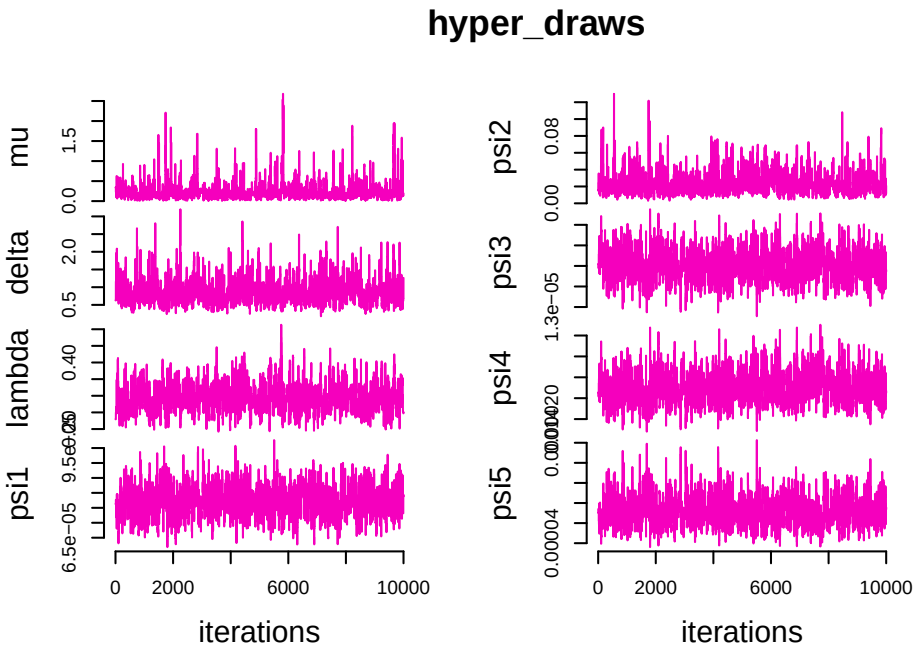
These values can be changed by simply assigning a new value to them. For example, `spec$prior$lambda.scale = 0.5` will change the scale parameter of the prior distribution of λ accordingly.

Given the specification object `spec`, we are now ready to estimate the hyper-parameters. Calling the function `spec$prior$estimate_hyper()` will sample from the posterior distribution of the hyper-parameters with an adaptive Metropolis algorithm. The user can specify the total number of draws from the posterior distribution by setting the argument `S`, and the number of burn-in draws by setting the argument `burn_in` to be discarded, then the algorithm will return `S - burn_in` draws of the hyper-parameters. It is also possible to only estimate a subset of the hyper-parameters by changing the argument of specific hyper-parameters to `TRUE` or `FALSE`. For example, to sample 15,000 draws for all the hyper-parameters and discard the first 5,000 draws as burn-in, the user can run the following code:

```
R> set.seed(123)
R> spec$prior$estimate_hyper(
+   S = 15000,
+   burn_in = 5000,
+   mu = TRUE,
+   delta = TRUE,
+   lambda = TRUE,
+   psi = TRUE
+ )
```

```
**************************************************|
 Adaptive Metropolis MCMC: hyper parameters       |
**************************************************|
```

```
R> hyper_draws = t(spec$prior$hyper)
R> colnames(hyper_draws) = c("mu", "delta", "lambda", paste0("psi", 1:ncol(optimism)))
R> plot.ts(hyper_draws, col = "#F500BD", bty = "n", xlab = "iterations")
```



The package also allows for more flexible specifications of the prior distribution, which can be achieved by modifying the elements of the `spec$prior` list. For example, if the user wishes to replicate the results in Arias *et al.* (2018), the specification should indicate fixed hyper-parameters and turn off the sum-of-coefficients and single-unit-root dummy observation priors. To do so, the user should set the corresponding dummy observation matrices to have zero columns prior to running the previous code:

```
R> spec$prior$Ysoc = matrix(NA, nrow(spec$prior$Ysoc), 0)
R> spec$prior$Xsoc = matrix(NA, nrow(spec$prior$Xsoc), 0)
R> spec$prior$Ysur = matrix(NA, nrow(spec$prior$Ysur), 0)
R> spec$prior$Xsur = matrix(NA, nrow(spec$prior$Xsur), 0)
```

2.6. Estimate the Model

Given hyper-parameters, whether fixed or a sample from the posterior distribution, the user can then sample the VAR parameters. The VAR models considered in the package all falls in the class of normal-inverse Wishart conjugate prior models, which allows for independent sampling from the posterior distribution without relying on Monte Carlo Markov Chain methods. This is done by passing the `spec` object as the first argument to the function `estimate()`, the following code samples 10,000 independent draws from the posterior distribution:

```
R> set.seed(123)
R> post = estimate(spec, S = 10000)

************************************************|
 bsvarSIGNs: Bayesian Structural VAR with sign,   |
             zero and narrative restrictions      |
************************************************|
 Progress of simulation for 10000 independent draws
 Press Esc to interrupt the computations
************************************************|
```

To assess the efficiency of the sampling algorithm, the user can print out the effective sample size of the draws by running the following code:

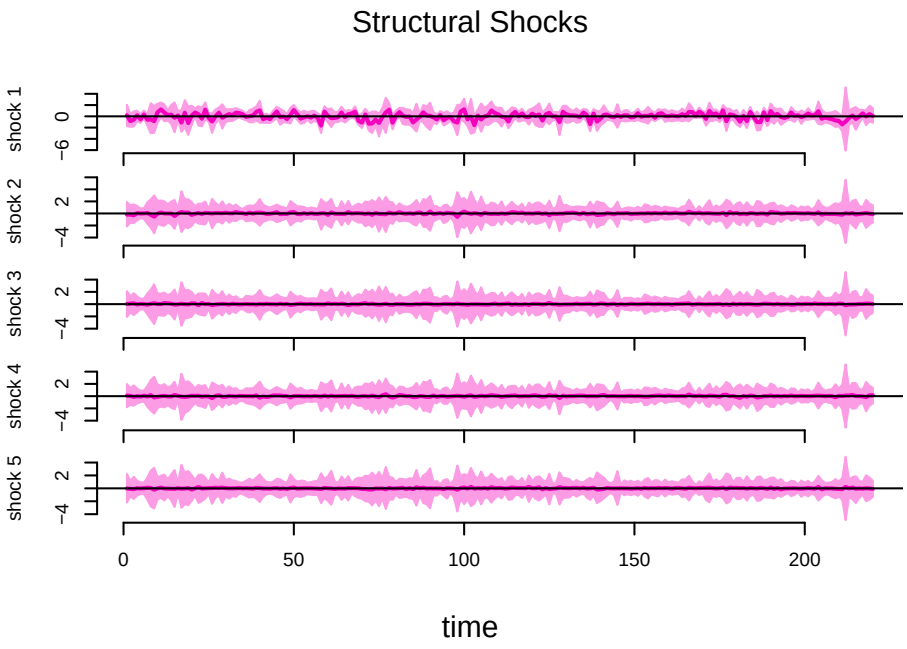
```
R> post$posterior$ess
```

```
[1] 257.3824
```

2.7. Report Structural Analysis

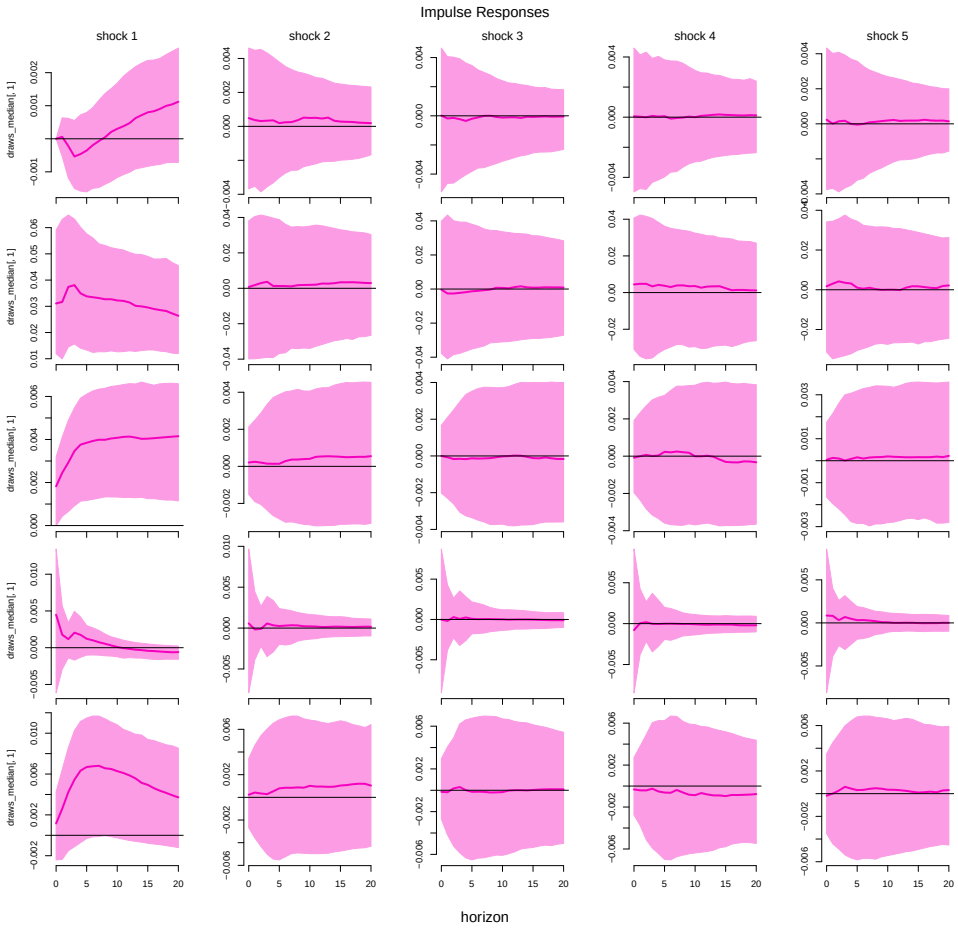
The method `compute_structural_shocks()` computes the structural shocks by appropriate rotations of the reduced-form shocks based on the specified restrictions. To visualize the structural shocks, the user can run the following code:

```
R> ss = compute_structural_shocks(post)
R> plot(ss, col = "#F500BD")
```



To compute the impulse response functions, use the method `compute_impulse_responses()` with the posterior object `post` as the first argument, and specify the number of horizons `horizon`. For example, to compute the impulse response functions on impact and for 20 periods ahead, and visualize the result, run the following code:

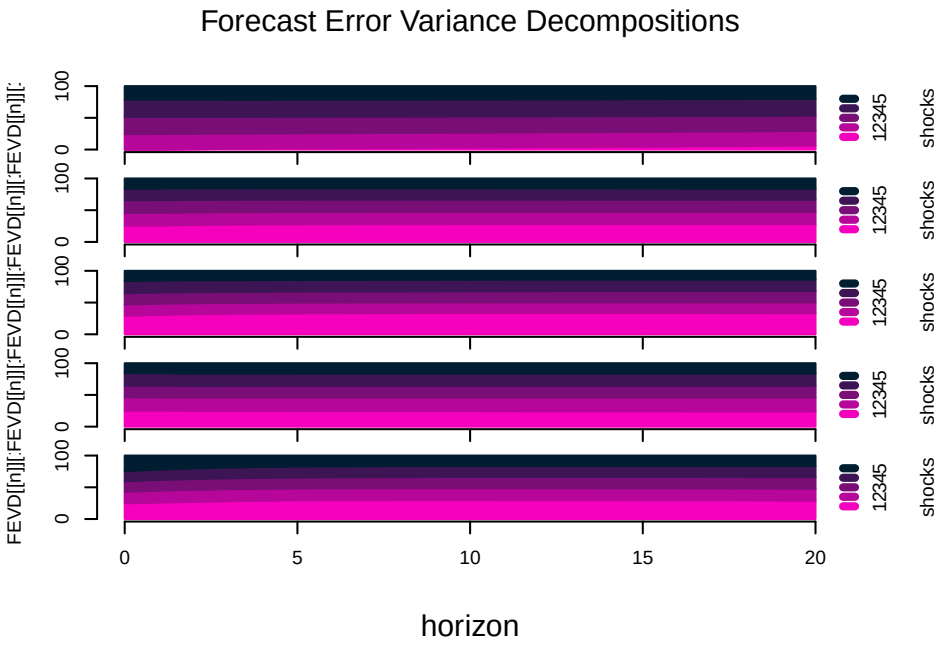
```
R> irf = compute_impulse_responses(post, horizon = 20)
R> plot(irf, probability = 0.68, col = "#F500BD")
```



This plot verifies the sign restrictions imposed on the contemporaneous responses of the variables to the identified structural shocks. The optimism shock (shock 1) has no contemporaneous effect on productivity (variable 1) but has positive impact on stock prices (variable 2). The effect of the optimism shock on both consumption and hours worked is positive but not significant in the short run. However, a noticeable sharpening of the identification of these effects is observed in a model with the additional narrative restriction relative to the model with only sign and zero restrictions reported in Section 2.1.

To compute the forecast error variance decompositions, use the method `compute_variance_decompositions()` with the posterior object `post` as the first argument. The user can then visualize the forecast error variance decompositions by running the following code:

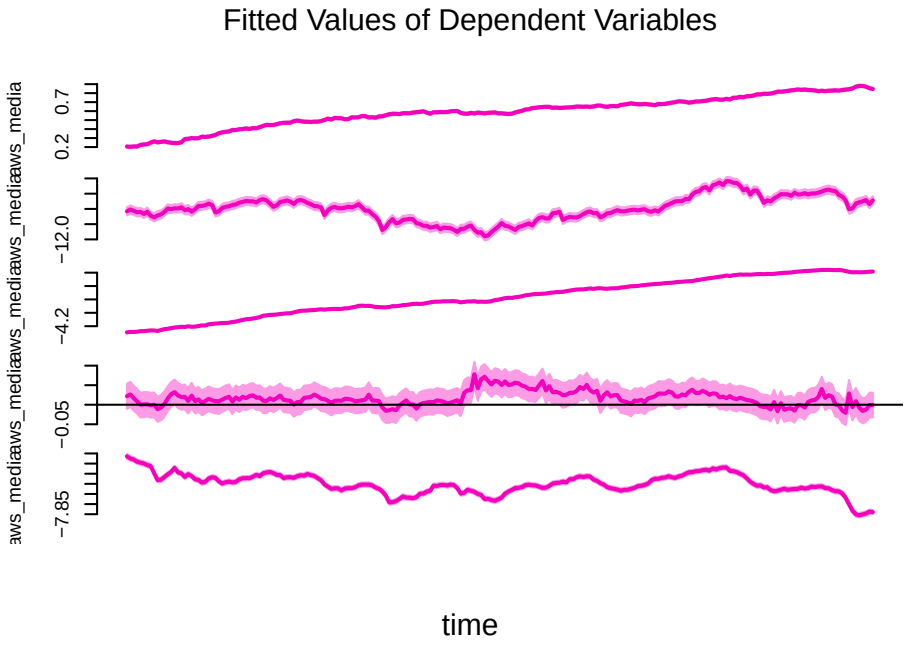
```
R> fevd = compute_variance_decompositions(post, horizon = 20)
R> fc   = colorRampPalette(c("#F500BD", "#001D31"))
R> cols = fc(5)
R> plot(fevd, cols = cols)
```



2.8. Report Predictive Analysis

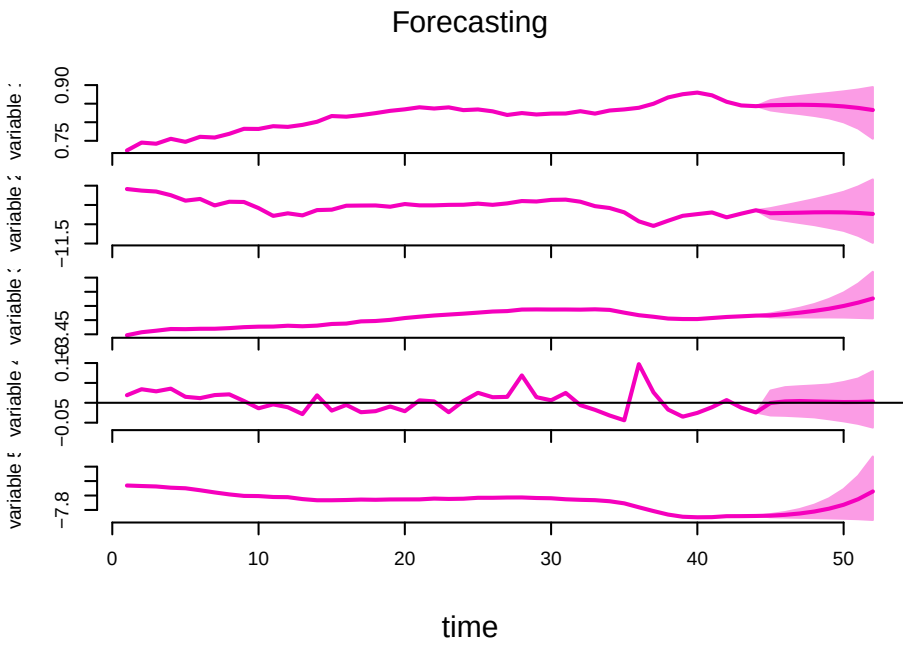
The method `compute_fitted_values()` computes the fitted values from the reduced-form VAR model. More precisely, the method samples random draws from the sample data density. TO perform th sampling and visualise the in-sample model fit, the user can run the following code:

```
R> fit = compute_fitted_values(post)
R> plot(fit, col = "#F500BD")
```



To compute an 2-year out-of-sample forecast, the user can pass the posterior object `post` as the first argument to the method `forecast()` and specify the forecast horizon by setting, for instancce, `horizon = 8`. In visualizing the forecast, it is possible to focus on the recent periods by setting the argument `data_in_plot` to be the desired percentage of the last observations to be included in the plot. For example, to visualize the forecast and the most recent 20% of the data, the user can run the following code:

```
R> fore = forecast(post, horizon = 8)
R> plot(fore, data_in_plot = 0.2, col = "#F500BD")
```



The forecasts, and other interpretable outputs, can be also presented using the `summary()` function, which provides a detailed table of the forecasts. For example, to print out the forecast of productivity (variable 1), the user can run the following code:

```
R> summary(fore)$variable1
```

```
 ================================================|
 bsvars: Bayesian Structural Vector Autoregressions|
 ================================================|
   Posterior summary of forecasts                 |
 ================================================|

       mean          sd 5% quantile 95% quantile
1 0.8458038 0.008377681   0.8317997    0.8595328
2 0.8463867 0.011949591   0.8264638    0.8661054
3 0.8467017 0.014756666   0.8220349    0.8707619
4 0.8463071 0.017534978   0.8168991    0.8749356
5 0.8449208 0.020845072   0.8105439    0.8788074
6 0.8422181 0.025390293   0.7998666    0.8831610
7 0.8374115 0.032229660   0.7828341    0.8881583
8 0.8297283 0.042756979   0.7555310    0.8947330
```

3. Conclusions

The package **bsvarSIGNs** provides a comprehensive set of tools for estimating and analysing Bayesian structural vector autoregressive models with sign, zero, and narrative restrictions. The package is designed to be user-friendly and flexible, allowing users to specify a wide range of prior distributions and restrictions. It implements efficient algorithms with C++ for estimation and inference, including an adaptive Metropolis algorithm for sampling the hyper-parameters and independent sampling for the VAR parameters. The package also provides a set of functions to summarise and visualise the results, including structural shocks, impulse response functions, historical decompositions, forecast error variance decompositions, fitted values, and forecasts.

References

Andrieu C, Moulines É (2006). "On the ergodicity properties of some adaptive MCMC algorithms." *The Annals of Applied Probability*, **16**(3), 1462 – 1505. URL https://doi.org/10.1214/105051606000000286.

Antolín-Díaz J, Rubio-Ramírez JF (2018). "Narrative Sign Restrictions for SVARs." *American Economic Review*, **108**(10), 2802–29. doi:10.1257/aer.20161852.

Arias JE, Rubio-Ramírez JF, Waggoner DF (2018). "Inference Based on Structural Vector Autoregressions Identified With Sign and Zero Restrictions: Theory and Applications." *Econometrica*, **86**(2), 685–720. doi:https://doi.org/10.3982/ECTA14468.

Atchadé Y, Fort G (2010). "Limit theorems for some adaptive MCMC algorithms with subgeometric kernels." *Bernoulli*, **16**(1), 116 – 154. doi:10.3150/09-BEJ199.

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

Baumeister C, Hamilton JD (2015). "Sign Restrictions, Structural Vector Autoregressions, and Useful Prior Information." *Econometrica*, **83**(5), 1963–1999. doi:10.3982/ecta12356.

Bernanke BS, Boivin J, Eliasz P (2005). "Measuring the Effects of Monetary Policy: A Factor-Augmented Vector Autoregressive (FAVAR) Approach*." *The Quarterly Journal of Economics*, **120**(1), 387–422. doi:10.1162/0033553053327452.

Boeck M, Feldkircher M, Huber F (2020). *BGVAR: Bayesian Global Vector Autoregressions*. R package version 2.5.7, URL https://CRAN.R-project.org/package=BGVAR.

Carriero A, Chan J, Clark TE, Marcellino M (2022). "Corrigendum to "Large Bayesian vector autoregressions with stochastic volatility and non-conjugate priors" [J. Econometrics 212 (1) (2019) 137–154]." *Journal of Econometrics*, **227**(2), 506–512. doi:https://doi.org/10.1016/j.jeconom.2021.11.010.

Chan JCC (2020). "Large Bayesian VARs: A Flexible Kronecker Error Covariance Structure." *Journal of Business & Economic Statistics*, **38**(1), 68–79. doi:10.1080/07350015.2018.1451336.

Chang W (2021). ***R6***: *Encapsulated Classes with Reference Semantics*. R package version 2.5.1, URL https://CRAN.R-project.org/package=R6.

Chen P, Chen C (2022). *FAVAR: Bayesian Analysis of a FAVAR Model*. R package version 0.1.3, URL https://CRAN.R-project.org/package=FAVAR.

Dieppe A, van Roye B (2024). *The BEAR toolbox*. MatLab library version 5.2.1, URL https://github.com/european-central-bank/BEAR-toolbox/.

Doan T, Litterman RB, Sims CA (1984). "Forecasting and Conditional Projection Using Realistic Prior Distributions." *Econometric Reviews*, **3**(1), 1–100. doi:10.1080/07474938408800053.

Drèze JH, Morales JA (1976). "Bayesian Full Information Analysis of Simultaneous Equations." *Journal of the American Statistical Association*, **71**(356), 919–923. doi:10.1080/01621459.1976.10480969.

Eddelbuettel D (2013). *Seamless R and C++ Integration with **Rcpp***. Springer, New York, NY. doi:10.1007/978-1-4614-6868-4.

Eddelbuettel D, François R, Allaire J, Ushey K, Kou Q, Russel N, Chambers J, Bates D (2011). "**Rcpp**: Seamless R and C++ integration." *Journal of Statistical Software*, **40**(8), 1–18. doi:10.18637/jss.v040.i08.

Eddelbuettel D, Francois R, Allaire J, Ushey K, Kou Q, Russell N, Ucar I, Bates D, Chambers J (2024a). *Rcpp: Seamless R and C++ Integration*. R package version 1.0.13, URL https://CRAN.R-project.org/package=Rcpp.

Eddelbuettel D, Francois R, Bates D, Ni B, Sanderson C (2024b). *RcppArmadillo: 'Rcpp' Integration for the 'Armadillo' Templated Linear Algebra Library*. R package version 14.0.2-1, URL https://CRAN.R-project.org/package=RcppArmadillo.

Eddelbuettel D, Sanderson C (2014). "**RcppArmadillo**: Accelerating R with high-performance C++ linear algebra." *Computational Statistics & Data Analysis*, **71**, 1054–1063. doi:10.1016/j.csda.2013.02.005.

Forner K (2020). ***RcppProgress**: An Interruptible Progress Bar with OpenMP Support for C++ in R Packages*. R package version 0.4.2, URL https://CRAN.R-project.org/package=RcppProgress.

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

Gruber L (2024). *bayesianVARs: MCMC Estimation of Bayesian Vectorautoregressions*. R package version 0.1.3, URL https://CRAN.R-project.org/package=bayesianVARs.

Kadiyala KR, Karlsson S (1997). "Numerical Methods for Estimation and Inference in Bayesian VAR-Models." *Journal of Applied Econometrics*, **12**(2), 99–132.

Kalli M, Griffin JE (2018). "Bayesian nonparametric vector autoregressive models." *Journal of econometrics*, **203**(2), 267–282. doi:10.1016/j.jeconom.2017.11.009.

Karlsson S (2013). "Forecasting with Bayesian Vector Autoregression." In *Handbook of Economic Forecasting*, volume 2, pp. 791–897. Elsevier. doi:10.1016/B978-0-444-62731-5.00015-4.

Kilian L, Lütkepohl H (2017). *Structural Vector Autoregressive Analysis*. Cambridge University Press, Cambridge. doi:10.1017/9781108164818.

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

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.

Kuschnig N, Vashold L (2021). "**BVAR**: Bayesian Vector Autoregressions with Hierarchical Prior Selection in R." *Journal of Statistical Software*, **100**(14), 1–27. doi:10.18637/jss.v100.i14.

Lütkepohl H (2005). *New introduction to multiple time series analysis*. Springer, New York, Berlin. doi:10.1007/978-3-540-27752-1.

Mohr FX (2022). ***bvartools**: Bayesian Inference of Vector Autoregressive and Error Correction Models*. R package version 0.2.1, URL https://CRAN.R-project.org/package=bvartools.

Primiceri GE (2005). "Time Varying Structural Vector Autoregressions and Monetary Policy." *The Review of Economic Studies*, **72**(3), 821–852. doi:10.1111/j.1467-937X.2005.00353.x.

R Core Team (2021). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

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

Sanderson C, Curtin R (2016). "**Armadillo**: a template-based C++ library for linear algebra." *Journal of Open Source Software*, **1**(2), 26. doi:10.21105/joss.00026.

Sims CA (1980). "Macroeconomics and Reality." *Econometrica*, **48**(1), 1–48. doi:10.2307/1912017.

Stewart GW (1980). "The efficient generation of random orthogonal matrices with an application to condition estimators." *SIAM Journal on Numerical Analysis*, **17**(3), 403–409. doi:10.1137/0717034.

Wang X, Woźniak T (2024). *bsvarSIGNs: Bayesian SVARs with Sign, Zero, and Narrative Restrictions*. doi:10.32614/CRAN.package.bsvarSIGNs. R package version 1.0.1, URL https://CRAN.R-project.org/package=bsvarSIGNs.

Woźniak T (2016). "Bayesian Vector Autoregressions." *Australian Economic Review*, **49**(3), 365–380. doi:10.1111/1467-8462.12179.

Woźniak T (2024a). *bsvars: Bayesian Estimation of Structural Vector Autoregressive Models*. doi:10.32614/CRAN.package.bsvars. R package version 3.2, URL https://CRAN.R-project.org/package=bsvars.

Woźniak T (2024b). "Fast and Efficient Bayesian Analysis of Structural Vector Autoregressions Using the R Package bsvars." *University of Melbourne Working Paper*. doi:10.48550/arXiv.2410.15090.

Affiliation:

Xiaolei Wang
University of Melbourne
Department of Economics
111 Barry Street
3053 Carlton, VIC, Australia
E-mail: adam.wang@unimelb.edu.au
URL: https://github.com/adamwang15

Tomasz Woźniak

University of Melbourne
Department of Economics
111 Barry Street
3053 Carlton, VIC, Australia
E-mail: tomasz.wozniak@unimelb.edu.au
URL: https://github.com/donotdespair