Package: bsvarSIGNs (via r-universe)

October 20, 2026

Type Package

Title Bayesian SVARs with Sign, Zero, and Narrative Restrictions

Version 3.0

Date 2026-08-21

Maintainer Xiaolei Wang <adamwang15@gmail.com>

Description Implements state-of-the-art algorithms for the Bayesian analysis of Structural Vector Autoregressions (SVARs) identified by sign, zero, and narrative restrictions. The core model is based on a flexible Vector Autoregression with estimated hyper-parameters of the Minnesota prior and the dummy observation priors as in Giannone, Lenza, Primiceri (2015) <doi:10.1162/REST_a_00483> extended by the COVID-specific heteroskedasticity proposed by Lenza, Primiceri (2022) <doi:10.1002/jae.2895>. The sign restrictions are implemented employing the methods proposed by Rubio-Ramírez, Waggoner & Zha (2010) <doi:10.1111/j.1467-937X.2009.00578.x>, while identification through sign and zero restrictions follows the approach developed by Arias, Rubio-Ramírez, & Waggoner (2018) <doi:10.3982/ECTA14468>. Furthermore, our toolset provides algorithms for identification via sign and narrative restrictions, in line with the methods introduced by Antolín-Díaz and Rubio-Ramírez (2018) <doi:10.1257/aer.20161852>. Users can also estimate a model with sign, zero, and narrative restrictions imposed 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 including the vignette by Wang & Woźniak (2024) <doi:10.48550/arXiv.2501.16711>. The 'bsvarSIGNs' package is aligned regarding objects, workflows, and code structure with the R packages 'bsvars' by Woźniak (2024) <doi:10.32614/CRAN.package.bsvars>, 'bvars' by Liu, Ramirez

Hassan, Woźniak (2026) <doi:10.32614/CRAN.package.bvars>, and 'bpvars' by Woźniak (2026) <doi:10.32614/CRAN.package.bpvars>, and they constitute an integrated toolset. It was granted the Di Cook Open-Source Statistical Software Award by the Statistical Society of Australia in 2024.

License GPL (>= 3)

Imports Rcpp (>= 1.0.12), RcppProgress, R6, parallel, generics

LinkingTo Rcpp, RcppArmadillo, RcppProgress, bsvars

Depends R (>= 4.1.0), RcppArmadillo, bsvars

Suggests knitr, tinytest

URL https://bsvars.org/bsvarSIGNs/, https://github.com/bsvars/bsvarSIGNs

BugReports https://github.com/bsvars/bsvarSIGNs/issues

Encoding UTF-8

LazyData true

VignetteBuilder knitr

Config/roxygen2/version 8.0.0

RoxygenNote 7.3.2

NeedsCompilation yes

Author Xiaolei Wang [aut, cre] (ORCID: <https://orcid.org/0009-0005-6192-9061>), Tomasz Woźniak [aut] (ORCID: <https://orcid.org/0000-0003-2212-2378>), Fei Shang [ctb] (ORCID: <https://orcid.org/0000-0003-1908-3275>, Corrected code for forecasting)

Repository https://cran.r-universe.dev

Date/Publication 2026-08-21 08:20:32 UTC

RemoteUrl https://github.com/cran/bsvarSIGNs

RemoteRef HEAD

RemoteSha 5d129388e44b0fce8778d3b5bca676117abf6388

Contents

bsvarSIGNs-package 3
compute_conditional_sd.PosteriorBSVARSIGN 6
compute_fitted_values.PosteriorBSVARSIGN 7
compute_historical_decompositions.PosteriorBSVARSIGN 8
compute_impulse_responses.PosteriorBSVARSIGN 10
compute_structural_shocks.PosteriorBSVARSIGN 11
compute_variance_decompositions.PosteriorBSVARSIGN 12
estimate.BSVARSIGN 14
forecast.PosteriorBSVARSIGN 16

monetary 18
optimism 19
specify_bsvarSIGN 20
specify_forecasts 26
specify_identification_bsvarSIGN 27
specify_narrative 29
specify_posterior_bsvarSIGN 30
specify_prior_bsvarSIGN 33

Index **36**

`bsvarSIGNs-package` *Bayesian Estimation of Structural Vector Autoregressions Identified by Sign, Zero, and Narrative Restrictions*

Description

Implements state-of-the-art algorithms for the Bayesian analysis of Structural Vector Autoregressions (SVARs) identified by sign, zero, and narrative restrictions. The core model is based on a flexible Vector Autoregression with estimated hyper-parameters of the Minnesota prior and the dummy observation priors as in Giannone, Lenza, Primiceri (2015) <doi:10.1162/REST_a_00483> extended by the COVID-specific heteroskedasticity proposed by Lenza, Primiceri (2022) <doi:10.1002/jae.2895>. The sign restrictions are implemented employing the methods proposed by Rubio-Ramírez, Waggoner & Zha (2010) <doi:10.1111/j.1467-937X.2009.00578.x>, while identification through sign and zero restrictions follows the approach developed by Arias, Rubio-Ramírez, & Waggoner (2018) <doi:10.3982/ECTA14468>. Furthermore, our toolset provides algorithms for identification via sign and narrative restrictions, in line with the methods introduced by Antolín-Díaz and Rubio-Ramírez (2018) <doi:10.1257/aer.20161852>. Users can also estimate a model with sign, zero, and narrative restrictions imposed 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 including the vignette by Wang & Woźniak (2024) <doi:10.48550/arXiv.2501.16711>. The 'bsvarSIGNs' package is aligned regarding objects, workflows, and code structure with the R packages 'bsvars' by Woźniak (2024) <doi:10.32614/CRAN.package.bsvars>, 'bvars' by Liu, Ramirez Hassan, Woźniak (2026) <doi:10.32614/CRAN.package.bvars>, and 'bpvars' by Woźniak (2026) <doi:10.32614/CRAN.package.bpvars>, and they constitute an integrated toolset. It was granted the Di Cook Open-Source Statistical Software Award by the Statistical Society of Australia in 2024.

Details

Models. All the SVAR models in this package are specified by two equations, including the reduced form equation:

$$y_t = Ax_t + \epsilon_t$$

where y_t is an `N`-vector of dependent variables, x_t is a `K`-vector of explanatory variables, ϵ_t is an `N`-vector of reduced form error terms, and A is an `NxK` matrix of autoregressive slope coefficients and parameters on deterministic terms in x_t.

The structural equation is given by:

$$B\epsilon_t = u_t$$

where u_t is an N-vector of structural shocks, and B is an NxN matrix of contemporaneous relationships.

Finally, all of the models share the following assumptions regarding the structural shocks `u_t`, namely, joint conditional normality given the past observations collected in matrix `x_t`, and temporal and contemporaneous independence. The latter implies zero correlations and autocorrelations.

Identification. The identification of the SVAR model is achieved by imposing:

- sign restrictions on the structural matrix B,
- sign and zero restrictions on the zero-horizon impulse responses $\Theta_0 = B^{-1}$,
- sign restrictions on the impulse responses at other horizons Θ_i for $i = 1, 2, \ldots$,
- sign restrictions on selected structural shocks u_t,
- two types of sign restrictions on the historical decompositions.

These restrictions determine the sampling algorithms of the structural matrix B defined as

$$B = Q'L$$

where Q is an NxN orthogonal matrix and L is a lower-triangular matrix $L = chol(\Sigma)^{-1}$, and Σ is the NxN conditional covariance matrix of the reduced-form error term ϵ_t. Consult the original papers by Rubio-Ramírez, Waggoner & Zha (2010), Arias, Rubio-Ramírez, & Waggoner (2018) and Antolín-Díaz and Rubio-Ramírez (2018) for more details.

Prior distributions. All the models feature a hierarchical Minnesota prior following the specification proposed by Giannone, Lenza, Primiceri (2015) and featuring:

- appropriate handling of unit-root non-stationary variables through the prior mean of the autoregressive coefficients A,
- normal prior shrinkage exhibiting exponential decay in the lag order of the autoregressive matrices,
- sum-of-coefficients and dummy-initial-observation prior,
- estimated shrinkage hyper-parameters,
- inverse-Wishart prior for the reduced-form covariance matrix Σ,
- estimated diagonal elements of the inverse-Wishart prior scale matrix.

Note

This package is currently in active development. Your comments, suggestions and requests are warmly welcome!

Author(s)

Xiaolei Wang <adamwang15@gmail.com> & Tomasz Woźniak <wozniak.tom@pm.me>

References

Antolín-Díaz & Rubio-Ramírez (2018) Narrative Sign Restrictions for SVARs, American Economic Review, 108(10), 2802-29, <doi:10.1257/aer.20161852>.

Arias, Rubio-Ramírez, & Waggoner (2018), Inference Based on Structural Vector Autoregressions Identified With Sign and Zero Restrictions: Theory and Applications, Econometrica, 86(2), 685-720, <doi:10.3982/ECTA14468>.

Giannone, Lenza, Primiceri (2015) Prior Selection for Vector Autoregressions, Review of Economics and Statistics, 97(2), 436-451, <doi:10.1162/REST_a_00483>.

Lenza, Primiceri (2022) How to estimate a vector autoregression after march 2020. Journal of Applied Econometrics 37 (4), 688–699, <doi:10.1002/jae.2895>.

Liu, Ramirez Hassan, Woźniak (2026) bvars: Bayesian Forecasting with Large Vector Autoregressions. R package version 1.0, <doi:10.32614/CRAN.package.bvars>.

Rubio-Ramírez, Waggoner & Zha (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>.

Wang, Xiaolei and Woźniak, Tomasz (2025). Bayesian Analyses of Structural Vector Autoregressions with Sign, Zero, and Narrative Restrictions Using the R Package bsvarSIGNs, <doi:10.48550/arXiv.2501.16711>.

Woźniak (2024) bsvars: Bayesian Estimation of Structural Vector Autoregressive Models. R package version 3.2, <doi:10.32614/CRAN.package.bsvars>.

Woźniak T (2026) bpvars: Forecasting with Bayesian Panel Vector Autoregressions. R package version 2.0, <doi:10.32614/CRAN.package.bpvars>.

Examples

```
# investigate the effects of the optimism shock
data(optimism)

# specify identifying restrictions:
# + no effect on productivity (zero restriction)
# + positive effect on stock prices (positive sign restriction)
sign_irf       = matrix(c(0, 1, rep(NA, 23)), 5, 5)

# specify the model
specification  = specify_bsvarSIGN$new(optimism * 100,
                                       p        = 4,
                                       sign_irf = sign_irf)

# estimate the model
posterior      = estimate(specification, S = 100)

# compute and plot impulse responses
irf            = compute_impulse_responses(posterior, horizon = 40)
plot(irf, probability = 0.68)
```

`compute_conditional_sd.PosteriorBSVARSIGN` *Computes posterior draws of structural shock conditional standard deviations*

Description

Each of the draws from the posterior estimation of models is transformed into a draw from the posterior distribution of the structural shock conditional standard deviations.

Usage

```
## S3 method for class 'PosteriorBSVARSIGN'
compute_conditional_sd(posterior)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVARSIGN` obtained by running the `estimate` function.

Value

An object of class `PosteriorSigma`, that is, an `NxTxS` array with attribute `PosteriorSigma` containing `S` draws of the structural shock conditional standard deviations.

Author(s)

Xiaolei Wang <adamwang15@gmail.com> and Tomasz Woźniak <wozniak.tom@pm.me>

See Also

`estimate.BSVARSIGN`

Examples

```
# upload data
data(optimism)

# specify the model and set seed
set.seed(123)

# + no effect on productivity (zero restriction)
# + positive effect on stock prices (positive sign restriction)
sign_irf       = matrix(c(0, 1, rep(NA, 23)), 5, 5)
specification  = specify_bsvarSIGN$new(optimism, sign_irf = sign_irf)

# estimate the model
posterior      = estimate(specification, 10)
```

```
# compute structural shocks' conditional standard deviations
sigma          = compute_conditional_sd(posterior)

# workflow with the pipe |>
############################################################
set.seed(123)
optimism |>
  specify_bsvarSIGN$new(sign_irf = sign_irf) |>
  estimate(S = 10) |>
  compute_conditional_sd() -> csd
```

`compute_fitted_values.PosteriorBSVARSIGN`
Computes posterior draws from data predictive density

Description

Each of the draws from the posterior estimation of models from packages **bsvars** or **bsvarSIGNs** is transformed into a draw from the data predictive density.

Usage

```
## S3 method for class 'PosteriorBSVARSIGN'
compute_fitted_values(posterior)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVARSIGN` obtained by running the `estimate` function.

Value

An object of class `PosteriorFitted`, that is, an `NxTxS` array with attribute `PosteriorFitted` containing `S` draws from the data predictive density.

Author(s)

Xiaolei Wang <adamwang15@gmail.com> and Tomasz Woźniak <wozniak.tom@pm.me>

See Also

`estimate.BSVARSIGN`, `summary`, `plot`

Examples

```
# upload data
data(optimism)

# specify the model and set seed
set.seed(123)

# + no effect on productivity (zero restriction)
# + positive effect on stock prices (positive sign restriction)
sign_irf       = matrix(c(0, 1, rep(NA, 23)), 5, 5)
specification  = specify_bsvarSIGN$new(optimism, sign_irf = sign_irf)

# estimate the model
posterior      = estimate(specification, 10)

# compute draws from in-sample predictive density
fitted         = compute_fitted_values(posterior)

# workflow with the pipe |>
############################################################
set.seed(123)
optimism |>
  specify_bsvarSIGN$new(sign_irf = sign_irf) |>
  estimate(S = 20) |>
  compute_fitted_values() -> fitted
```

`compute_historical_decompositions.PosteriorBSVARSIGN`
Computes posterior draws of historical decompositions

Description

Each of the draws from the posterior estimation of models from packages **bsvars** or **bsvarSIGNs** is transformed into a draw from the posterior distribution of the historical decompositions. IMPORTANT! The historical decompositions are interpreted correctly for covariance stationary data. Application to unit-root non-stationary data might result in non-interpretable outcomes.

Usage

```
## S3 method for class 'PosteriorBSVARSIGN'
compute_historical_decompositions(posterior, show_progress = TRUE)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVARSIGN` obtained by running the `estimate` function.
`show_progress`	a logical value, if `TRUE` the estimation progress bar is visible

Value

An object of class PosteriorHD, that is, an NxNxTxS array with attribute PosteriorHD containing S draws of the historical decompositions.

Author(s)

Xiaolei Wang <adamwang15@gmail.com> and Tomasz Woźniak <wozniak.tom@pm.me>

References

Kilian, L., & Lütkepohl, H. (2017). Structural VAR Tools, Chapter 4, In: Structural vector autoregressive analysis. Cambridge University Press.

See Also

estimate.BSVARSIGN, summary, plot

Examples

```
# upload data
data(optimism)

# specify the model and set seed
set.seed(123)

# + no effect on productivity (zero restriction)
# + positive effect on stock prices (positive sign restriction)
sign_irf       = matrix(c(0, 1, rep(NA, 23)), 5, 5)
specification  = specify_bsvarSIGN$new(optimism, sign_irf = sign_irf)

# estimate the model
posterior      = estimate(specification, 10)

# compute historical decompositions
hd             = compute_historical_decompositions(posterior)

# workflow with the pipe |>
############################################################
set.seed(123)
optimism |>
  specify_bsvarSIGN$new(sign_irf = sign_irf) |>
  estimate(S = 10) |>
  compute_historical_decompositions() -> hd
```

`compute_impulse_responses.PosteriorBSVARSIGN` *Computes posterior draws of impulse responses*

Description

Each of the draws from the posterior estimation of models from packages **bsvars** or **bsvarSIGNs** is transformed into a draw from the posterior distribution of the impulse responses.

Usage

```
## S3 method for class 'PosteriorBSVARSIGN'
compute_impulse_responses(posterior, horizon, standardise = FALSE)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVARSIGN` obtained by running the `estimate` function.
`horizon`	a positive integer number denoting the forecast horizon for the impulse responses computations.
`standardise`	a logical value. If `TRUE`, the impulse responses are standardised so that the variables' own shocks at horizon 0 are equal to 1. Otherwise, the parameter estimates determine this magnitude.

Value

An object of class PosteriorIR, that is, an `NxNx(horizon+1)xS` array with attribute PosteriorIR containing `S` draws of the impulse responses.

Author(s)

Xiaolei Wang <adamwang15@gmail.com> and Tomasz Woźniak <wozniak.tom@pm.me>

References

Kilian, L., & Lütkepohl, H. (2017). Structural VAR Tools, Chapter 4, In: Structural vector autoregressive analysis. Cambridge University Press.

See Also

`estimate.BSVARSIGN`, `summary`, `plot`

Examples

```
# upload data
data(optimism)

# specify the model and set seed
set.seed(123)

# + no effect on productivity (zero restriction)
# + positive effect on stock prices (positive sign restriction)
sign_irf       = matrix(c(0, 1, rep(NA, 23)), 5, 5)
specification  = specify_bsvarSIGN$new(optimism, sign_irf = sign_irf)

# estimate the model
posterior      = estimate(specification, 10)

# compute impulse responses 2 years ahead
irf           = compute_impulse_responses(posterior, horizon = 8)

# workflow with the pipe |>
############################################################
set.seed(123)
optimism |>
  specify_bsvarSIGN$new(sign_irf = sign_irf) |>
  estimate(S = 10) |>
  compute_impulse_responses(horizon = 8) -> ir
```

---

compute_structural_shocks.PosteriorBSVARSIGN

Computes posterior draws of structural shocks

Description

Each of the draws from the posterior estimation of models from packages **bsvars** or **bsvarSIGNs** is transformed into a draw from the posterior distribution of the structural shocks.

Usage

```
## S3 method for class 'PosteriorBSVARSIGN'
compute_structural_shocks(posterior)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVARSIGN` obtained by running the `estimate` function.

Value

An object of class `PosteriorShocks`, that is, an `NxTxS` array with attribute `PosteriorShocks` containing `S` draws of the structural shocks.

Author(s)

Xiaolei Wang <adamwang15@gmail.com> and Tomasz Woźniak <wozniak.tom@pm.me>

See Also

`estimate.BSVARSIGN`, `summary`, `plot`

Examples

```
# upload data
data(optimism)

# specify the model and set seed
set.seed(123)

# + no effect on productivity (zero restriction)
# + positive effect on stock prices (positive sign restriction)
sign_irf       = matrix(c(0, 1, rep(NA, 23)), 5, 5)
specification  = specify_bsvarSIGN$new(optimism, sign_irf = sign_irf)

# estimate the model
posterior      = estimate(specification, 10)

# compute structural shocks
shocks         = compute_structural_shocks(posterior)

# workflow with the pipe |>
############################################################
set.seed(123)
optimism |>
  specify_bsvarSIGN$new(sign_irf = sign_irf) |>
  estimate(S = 20) |>
  compute_structural_shocks() -> ss
```

---

compute_variance_decompositions.PosteriorBSVARSIGN

Computes posterior draws of the forecast error variance decomposition

---

Description

Each of the draws from the posterior estimation of the model is transformed into a draw from the posterior distribution of the forecast error variance decomposition.

Usage

```
## S3 method for class 'PosteriorBSVARSIGN'
compute_variance_decompositions(posterior, horizon)
```

Arguments

posterior	posterior estimation outcome - an object of class PosteriorBSVARSIGN obtained by running the estimate function.
horizon	a positive integer number denoting the forecast horizon for the impulse responses computations.

Value

An object of class PosteriorFEVD, that is, an NxNx(horizon+1)xS array with attribute PosteriorFEVD containing S draws of the forecast error variance decomposition.

Author(s)

Xiaolei Wang <adamwang15@gmail.com> and Tomasz Woźniak <wozniak.tom@pm.me>

References

Kilian, L., & Lütkepohl, H. (2017). Structural VAR Tools, Chapter 4, In: Structural vector autoregressive analysis. Cambridge University Press.

See Also

compute_impulse_responses.PosteriorBSVARSIGN, estimate.BSVARSIGN, summary, plot

Examples

```
# upload data
data(optimism)

# specify the model and set seed
set.seed(123)

# + no effect on productivity (zero restriction)
# + positive effect on stock prices (positive sign restriction)
sign_irf       = matrix(c(0, 1, rep(NA, 23)), 5, 5)
specification  = specify_bsvarSIGN$new(optimism, sign_irf = sign_irf)

# estimate the model
posterior      = estimate(specification, 10)

# compute forecast error variance decomposition 2 years ahead
fevd           = compute_variance_decompositions(posterior, horizon = 8)

# workflow with the pipe |>
############################################################
set.seed(123)
```

```
optimism |>
  specify_bsvarSIGN$new(sign_irf = sign_irf) |>
  estimate(S = 10) |>
  compute_variance_decompositions(horizon = 8) -> fevd
```

`estimate.BSVARSIGN` *Bayesian estimation of a Structural Vector Autoregression with traditional and narrative sign restrictions via Gibbs sampler*

Description

Estimates Bayesian Structural Vector Autoregression model using the Gibbs sampler proposed by Waggoner & Zha (2003) with traditional sign restrictions following Rubio-Ramírez, Waggoner & Zha (2010) and narrative sign restrictions following Antolín-Díaz & Rubio-Ramírez (2018). Additionally, the parameter matrices A and B follow a Minnesota prior and generalised-normal prior distributions respectively with the matrix-specific overall shrinkage parameters estimated using a hierarchical prior distribution.

Given sign restrictions, in each Gibbs sampler iteration, the sampler draws rotation matrix Q uniformly from the space of `NxN` orthogonal matrices and checks if the sign restrictions are satisfied. If a valid Q is found within `max_tries` (defined in `specify_bsvarSIGN`), the sampler saves the current A and B draw and proceeds to the next iteration. Otherwise, the sampler then proceeds to next iteration without saving the current A and B draw. If a narrative sign restriction is given, the posterior draws are resampled with `algorithm 1` in Antolín-Díaz & Rubio-Ramírez (2018).

See section **Details** for the model equations.

Usage

```
## S3 method for class 'BSVARSIGN'
estimate(specification, S, thin = 1, show_progress = TRUE)
```

Arguments

`specification`	an object of class BSVARSIGN generated using the `specify_bsvarSIGN$new()` function.
`S`	a positive integer, the number of posterior draws to be generated
`thin`	a positive integer, specifying the frequency of MCMC output thinning
`show_progress`	a logical value, if `TRUE` the estimation progress bar is visible

Details

The Structural VAR model is given by the reduced form equation:

$$Y = AX + E$$

where Y is an `NxT` matrix of dependent variables, X is a `KxT` matrix of explanatory variables, E is an `NxT` matrix of reduced form error terms, and A is an `NxK` matrix of autoregressive slope coefficients and parameters on deterministic terms in X.

The structural equation is given by

$$BE = U$$

where U is an NxT matrix of structural form error terms, and B is an NxN matrix of contemporaneous relationships. More specifically,

$$B = Q'P$$

where Q is an NxN rotation matrix and P is an NxN lower triangular matrix.

Finally, the structural shocks, U, are temporally and contemporaneously independent and jointly normally distributed with zero mean and unit variances.

Value

An object of class PosteriorBSVARSIGN containing the Bayesian estimation output and containing two elements:

posterior a list with a collection of S draws from the posterior distribution generated via Gibbs sampler containing:

A an NxKxS array with the posterior draws for matrix A

B an NxNxS array with the posterior draws for matrix B

hyper a 5xS matrix with the posterior draws for the hyper-parameters of the hierarchical prior distribution

skipped an integer of the total skipped iterations, the Gibbs sampler performs a total of S+skipped iterations, when the sampler does not find a valid rotation matrix Q within max_tries, the current iteration is skipped (i.e. the current draw of A,B is not saved). A message is shown when skipped/(skipped+S/thin) > 0.05, where S/thin is the total number of draws returned.

last_draw an object of class BSVARSIGN with the last draw of the current MCMC run as the starting value to be passed to the continuation of the MCMC estimation using estimate().

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>, Xiaolei Wang <adamwang15@gmail.com>

References

Antolín-Díaz & Rubio-Ramírez (2018) Narrative Sign Restrictions for SVARs, American Economic Review, 108(10), 2802-29, <doi:10.1257/aer.20161852>.

Arias, Rubio-Ramírez, & Waggoner (2018), Inference Based on Structural Vector Autoregressions Identified With Sign and Zero Restrictions: Theory and Applications, Econometrica, 86(2), 685-720, <doi:10.3982/ECTA14468>.

Giannone, Lenza, Primiceri (2015) Prior Selection for Vector Autoregressions, Review of Economics and Statistics, 97(2), 436-451 <doi:10.1162/REST_a_00483>.

Rubio-Ramírez, Waggoner & Zha (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>.

Examples

```
# investigate the effects of the optimism shock
data(optimism)

# specify identifying restrictions:
# + no effect on productivity (zero restriction)
# + positive effect on stock prices (positive sign restriction)
sign_irf       = matrix(c(0, 1, rep(NA, 23)), 5, 5)

# specify the model and set seed
set.seed(123)
specification  = specify_bsvarSIGN$new(optimism * 100,
                                       p        = 12,
                                       sign_irf = sign_irf)

# estimate the model
posterior      = estimate(specification, S = 10)
```

`forecast.PosteriorBSVARSIGN` *Forecasting using Structural Vector Autoregression*

Description

Samples from the joint predictive density of all of the dependent variables for models from packages **bsvars**, **bsvarSIGNs** or **bvarPANELs** at forecast horizons from 1 to `horizon` specified as an argument of the function. Also facilitates forecasting using models with exogenous variables and conditional forecasting given projected future trajcetories of (some of the) variables.

Usage

```
## S3 method for class 'PosteriorBSVARSIGN'
forecast(
  object,
  horizon = 1,
  exogenous_forecast = NULL,
  conditional_forecast = NULL,
  ...
)
```

Arguments

`object`	posterior estimation outcome - an object of class `PosteriorBSVARSIGN` obtained by running the `estimate` function.
`horizon`	a positive integer, specifying the forecasting horizon.

`exogenous_forecast`
: a matrix of dimension `horizon` x `d` containing forecasted values of the exogenous variables.

`conditional_forecast`
: a `horizon` x `N` matrix with forecasted values for selected variables. It should only contain `numeric` or `NA` values. The entries with `NA` values correspond to the values that are forecasted conditionally on the realisations provided as `numeric` values.

`...`
: Not used here.

Value

A list of class `Forecasts` containing the draws from the predictive density and data. The output list includes element:

forecasts an `NxhorizonxS` array with the draws from predictive density

Y an NxT matrix with the data on dependent variables

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me> and Xiaolei Wang <adamwang15@gmail.com>

See Also

`estimate.BSVARSIGN`, `summary`, `plot`

Examples

```
# + no effect on productivity (zero restriction)
# + positive effect on stock prices (positive sign restriction)
sign_irf       = matrix(c(0, 1, rep(NA, 23)), 5, 5)
specification  = specify_bsvarSIGN$new(optimism, sign_irf = sign_irf)

# estimate the model
posterior      = estimate(specification, 10)

# sample from predictive density 1 year ahead
predictive     = forecast(posterior, 4)

# workflow with the pipe |>
############################################################
optimism |>
  specify_bsvarSIGN$new(sign_irf = sign_irf) |>
  estimate(S = 20) |>
  forecast(horizon = 4) -> predictive

# conditional forecasting 2 quarters ahead conditioning on
#  provided future values for the Gross Domestic Product
############################################################
cf         = matrix(NA , 2, 5)
# # conditional forecasts equal to the last consumption observation
```

```
cf[,3]     = tail(optimism, 1)[3]
predictive = forecast(posterior, 2, conditional_forecast = cf)

# workflow with the pipe |>
############################################################
optimism |>
  specify_bsvarSIGN$new(sign_irf = sign_irf) |>
  estimate(S = 10) |>
  forecast(horizon = 2, conditional_forecast = cf) -> predictive
```

`monetary` *A 6-variable US monetary policy data, from 1965 Jan to 2007 Aug*

Description

A sample data to identify monetary policy shock.

Usage

```
data(monetary)
```

Format

A matrix and a `ts` object with time series of over two hundred observations on 5 variables:

gdpc1 monthly real gross domestic product

gdpdef monthly gross domestic product: implicit price deflator

cprindex monthly consumer price index

totresns monthly reserves of depository institutions

bognonbr monthly non-borrowed reserves of depository institutions

fedfunds monthly federal funds effective rate

Source

Replication package, https://www.aeaweb.org/articles?id=10.1257/aer.20161852

References

Antolín-Díaz & Rubio-Ramírez (2018) Narrative Sign Restrictions for SVARs, American Economic Review, 108(10), 2802-29, <doi:10.1257/aer.20161852>.

`optimism` *A 5-variable US business cycle data, from 1955 Q1 to 2004 Q4*

Description

A sample data to identify optimism shock.

Usage

```
data(optimism)
```

Format

A matrix and a `ts` object with time series of over two hundred observations on 5 variables:

productivity quarterly factor-utilization-adjusted total factor productivity

stock_prices quarterly end-of-period S&P 500 divided by CPI

consumption quarterly real consumption expenditures on nondurable goods and services

real_interest_rate quarterly real interest rate

hours_worked quarterly hours of all persons in the non-farm business sector

The series are as described by Beaudry, Nam and Wang (2011) in section 2.2.

Source

Replication package, https://www.econometricsociety.org/publications/econometrica/2018/03/01/inference-based-structural-vector-autoregressions-identified

References

Arias, Jonas E., Juan F. Rubio-Ramírez, and Daniel F. Waggoner. "Inference based on structural vector autoregressions identified with sign and zero restrictions: Theory and applications." Econometrica 86, no. 2 (2018): 685-720. <doi:10.3982/ECTA14468>

Beaudry, Paul, Deokwoo Nam, and Jian Wang. Do mood swings drive business cycles and is it rational?. No. w17651. National Bureau of Economic Research, 2011. <doi:10.3386/w17651>

specify_bsvarSIGN *R6 Class representing the specification of the BSVARSIGN model*

Description

The class BSVARSIGN presents complete specification for the Bayesian Structural VAR model with sign and narrative restrictions.

Public fields

`p` a non-negative integer specifying the autoregressive lag order of the model.

`identification` an object IdentificationBSVARSIGN with the identifying restrictions.

`prior` an object PriorBSVARSIGN with the prior specification.

`data_matrices` an object DataMatricesBSVARSIGN with the data matrices.

`starting_values` an object StartingValuesBSVARSIGN with the starting values.

`num_foreign_vars` a non-negative integer specifying the number of foreign variables.

`mc.cores` number of cores to use for parallel computing.

Methods

Public methods:

- `specify_bsvarSIGN$new()`
- `specify_bsvarSIGN$get_data_matrices()`
- `specify_bsvarSIGN$no_dummy_observations()`
- `specify_bsvarSIGN$estimate_hyper()`
- `specify_bsvarSIGN$get_identification()`
- `specify_bsvarSIGN$get_prior()`
- `specify_bsvarSIGN$get_starting_values()`
- `specify_bsvarSIGN$clone()`

Method `new()`**:** Create a new specification of the Bayesian Structural VAR model with sign and narrative restrictions BSVARSIGN.

Usage:

```
specify_bsvarSIGN$new(
  data,
  p = 1L,
  sign_irf,
  sign_narrative,
  sign_structural,
  max_tries = Inf,
  exogenous = NULL,
  foreign = NULL,
  stationary = NULL,
```

```
  hyper_mu = TRUE,
  hyper_delta = TRUE,
  hyper_lambda = TRUE,
  hyper_psi = TRUE,
  hyper_covid = NULL,
  mc.cores = 1
)
```

Arguments:

`data` a `(T+p)xN` matrix with time series data.

`p` a positive integer providing model's autoregressive lag order.

`sign_irf` a `NxNxH` array - sign and zero restrictions on the impulse response functions, ±1 for positive/negative sign restriction 0 for zero restrictions and NA for no restrictions, the `h`-th slice `NxN` matrix contains the restrictions on the `h-1` horizon.

`sign_narrative` a list of objects of class "narrative" - narrative sign restrictions.

`sign_structural` a `NxN` matrix with entries ±1 or NA - sign restrictions on the contemporaneous relations `B` between reduced-form errors `E` and structural shocks `U` where `BE=U`.

`max_tries` a positive integer with the maximum number of iterations for finding a rotation matrix Q that would satisfy sign restrictions

`exogenous` a `(T+p)xd` matrix of exogenous variables.

`foreign` a matrix of foreign variables for a Small Open Economy (SOE) model. Defaults to NULL.

`stationary` an `N` logical vector - its element set to `FALSE` sets the prior mean for the autoregressive parameters of the `N`th equation to the white noise process, otherwise to random walk.

`hyper_mu` whether to estimate the hyper-parameter in the sum-of-coefficients dummy prior.

`hyper_delta` whether to estimate the hyper-parameter in the single-unit-root dummy prior.

`hyper_lambda` whether to estimate the hyper-parameter of the shrinkage in the Minnesota prior.

`hyper_psi` whether to estimate the hyper-parameter of the variances in the Minnesota prior.

`hyper_covid` NULL or positive integer indicating the start of the COVID-19 pandemic.

`mc.cores` number of cores to use for parallel computing. Default is 1. We recommend setting it to `parallel::detectCores() - 1`.

Returns: A new complete specification for the Bayesian Structural VAR model BSVARSIGN.

Method `get_data_matrices()`**:** Returns the data matrices as the DataMatricesBSVAR object.

Usage:

```
specify_bsvarSIGN$get_data_matrices()
```

Examples:

```
# specify a model with the optimism data and 4 lags

data(optimism)
spec = specify_bsvarSIGN$new(
   data = optimism,
   p = 4
)
```

```
# get the data matrices
spec$get_data_matrices()
```

Method `no_dummy_observations()`**:** Sets the sum-of-coefficients and single-unit-root dummy observations to zero (removes the dummy observation prior).

Usage:

```
specify_bsvarSIGN$no_dummy_observations()
```

Examples:

```
# specify the model
data(optimism)
spec = specify_bsvarSIGN$new(optimism, p = 4)
spec$no_dummy_observations() # remove dummy observations
```

Method `estimate_hyper()`**:** Estimates hyper-parameters with adaptive Metropolis algorithm.

Usage:

```
specify_bsvarSIGN$estimate_hyper(S = 10000, burn_in = S/2)
```

Arguments:

`S` number of MCMC draws.

`burn_in` number of burn-in draws.

Examples:

```
# specify the model and set seed
set.seed(123)
data(optimism)
spec = specify_bsvarSIGN$new(optimism, p = 4)

# estimate hyper parameters with adaptive Metropolis algorithm
spec$estimate_hyper(S = 10)

# trace plot
hyper = t(spec$prior$hyper)[, 4:8]
colnames(hyper) = paste("psi", 1:5, sep = "")
plot.ts(hyper)
```

Method `get_identification()`**:** Returns the identifying restrictions as the IdentificationBSVARSIGN object.

Usage:

```
specify_bsvarSIGN$get_identification()
```

Examples:

```
# specify a model with the optimism data and 4 lags
data(optimism)
spec = specify_bsvarSIGN$new(
```

```
   data = optimism,
   p = 4
)

# get the identifying restrictions
spec$get_identification()
```

Method `get_prior()`**:** Returns the prior specification as the PriorBSVAR object.

Usage:

```
specify_bsvarSIGN$get_prior()
```

Examples:

```
# specify a model with the optimism data and 4 lags

data(optimism)
spec = specify_bsvarSIGN$new(
   data = optimism,
   p = 4
)

# get the prior specification
spec$get_prior()
```

Method `get_starting_values()`**:** Returns the starting values as the StartingValuesBSVAR object.

Usage:

```
specify_bsvarSIGN$get_starting_values()
```

Examples:

```
# specify a model with the optimism data and 4 lags

data(optimism)
spec = specify_bsvarSIGN$new(
   data = optimism,
   p = 4
)

# get the starting values
spec$get_starting_values()
```

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
specify_bsvarSIGN$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

See Also

estimate.BSVARSIGN, specify_posterior_bsvarSIGN

Examples

```
# specify a model with the optimism data and 4 lags

data(optimism)
specification = specify_bsvarSIGN$new(
   data = optimism,
   p = 4
)


## ------------------------------------------------
## Method `specify_bsvarSIGN$get_data_matrices`
## ------------------------------------------------

# specify a model with the optimism data and 4 lags

data(optimism)
spec = specify_bsvarSIGN$new(
   data = optimism,
   p = 4
)

# get the data matrices
spec$get_data_matrices()


## ------------------------------------------------
## Method `specify_bsvarSIGN$no_dummy_observations`
## ------------------------------------------------

# specify the model
data(optimism)
spec = specify_bsvarSIGN$new(optimism, p = 4)
spec$no_dummy_observations() # remove dummy observations


## ------------------------------------------------
## Method `specify_bsvarSIGN$estimate_hyper`
## ------------------------------------------------

# specify the model and set seed
set.seed(123)
data(optimism)
spec = specify_bsvarSIGN$new(optimism, p = 4)

# estimate hyper parameters with adaptive Metropolis algorithm
spec$estimate_hyper(S = 10)
```

```
# trace plot
hyper = t(spec$prior$hyper)[, 4:8]
colnames(hyper) = paste("psi", 1:5, sep = "")
plot.ts(hyper)


## ------------------------------------------------
## Method `specify_bsvarSIGN$get_identification`
## ------------------------------------------------

# specify a model with the optimism data and 4 lags
data(optimism)
spec = specify_bsvarSIGN$new(
   data = optimism,
   p = 4
)

# get the identifying restrictions
spec$get_identification()


## ------------------------------------------------
## Method `specify_bsvarSIGN$get_prior`
## ------------------------------------------------

# specify a model with the optimism data and 4 lags

data(optimism)
spec = specify_bsvarSIGN$new(
   data = optimism,
   p = 4
)

# get the prior specification
spec$get_prior()


## ------------------------------------------------
## Method `specify_bsvarSIGN$get_starting_values`
## ------------------------------------------------

# specify a model with the optimism data and 4 lags

data(optimism)
spec = specify_bsvarSIGN$new(
   data = optimism,
   p = 4
)

# get the starting values
spec$get_starting_values()
```

`specify_forecasts` *R6 Class Representing Forecasts*

Description

R6 class representing draws from the predictive density of a Bayesian Structural Vector Autoregression model.

Details

The class contains the following objects:

`forecasts` An `N` x `horizon` x `S` array containing draws from the predictive density.

`forecast_mean` An `N` x `horizon` x `S` array containing the conditional means of the predictive density.

`forecast_covariance` An `N` x `N` x `horizon` x `S` array containing the conditional covariance matrices of the predictive density.

`Y` An `N` x `T` matrix containing the data on the dependent variables used for estimation.

The method `as_list()` returns the contents of the `Forecasts` object as a list.

Value

An object of class `Forecasts`.

Public fields

`forecasts` An `N` x `horizon` x `S` numeric array containing draws from the predictive density.

`forecast_mean` An `N` x `horizon` x `S` numeric array containing the conditional means of the predictive density.

`forecast_covariance` An `N` x `N` x `horizon` x `S` numeric array containing the conditional covariance matrices of the predictive density.

`Y` An `N` x `T` numeric matrix containing the data on the dependent variables used for estimation.

Methods

Public methods:

- `specify_forecasts$new()`
- `specify_forecasts$get_forecasts()`
- `specify_forecasts$clone()`

Method `new()`**:** Creates a new `Forecasts` object from the output of the forecasting procedure.

Usage:

```
specify_forecasts$new(output, Y)
```

Arguments:

`output` A list containing the forecasting output, including `forecasts`, `forecast_mean`, and `forecast_cov`.

`Y` An `N` x `T` matrix containing the data on the dependent variables.

Returns: An object of class `Forecasts`.

Method `get_forecasts()`**:** Converts the `Forecasts` object to a list.

Usage:

```
specify_forecasts$get_forecasts()
```

Returns: A list containing `forecasts`, `forecast_mean`, `forecast_covariance`, and `Y`.

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
specify_forecasts$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
spec = specify_bsvarSIGN$new(optimism * 100)
post = estimate(spec, 5)
fore = forecast(post, 4)
apply(fore$forecasts, 1:2, mean) # compute mean forecasts
```

`specify_identification_bsvarSIGN` *R6 Class Representing IdentificationBSVARSIGN*

Description

The class IdentificationBSVARSIGN presents the identifying restrictions for the Bayesian Structural VAR models with sign and narrative restrictions.

Public fields

`VB` a list of `N` matrices determining the unrestricted elements of matrix B.

`sign_irf` a `NxNxH` array of sign restrictions on the impulse response functions.

`sign_narrative` a `ANYx6` matrix of narrative sign restrictions.

`sign_structural` a `NxN` matrix of sign restrictions on contemporaneous relations.

`max_tries` a positive integer with the maximum number of iterations for finding a rotation matrix Q that would satisfy sign restrictions.

Methods

Public methods:

- specify_identification_bsvarSIGN$new()
- specify_identification_bsvarSIGN$get_identification()
- specify_identification_bsvarSIGN$set_identification()
- specify_identification_bsvarSIGN$clone()

Method `new()`**:** Create new identifying restrictions IdentificationBSVARSIGN.

Usage:

```
specify_identification_bsvarSIGN$new(
  N,
  sign_irf,
  sign_narrative,
  sign_structural,
  max_tries = Inf
)
```

Arguments:

`N` a positive integer - the number of dependent variables in the model.

`sign_irf` a `NxNxH` array - sign and zero restrictions on the impulse response functions, ±1 for positive/negative sign restriction 0 for zero restrictions and NA for no restrictions, the `h`-th slice `NxN` matrix contains the restrictions on the `h-1` horizon.

`sign_narrative` a list of objects of class "narrative" - narrative sign restrictions.

`sign_structural` a `NxN` matrix with entries ±1 or NA - sign restrictions on the contemporaneous relations `B` between reduced-form errors `E` and structural shocks `U` where `BE=U`.

`max_tries` a positive integer with the maximum number of iterations for finding a rotation matrix Q that would satisfy sign restrictions.

Returns: Identifying restrictions IdentificationBSVARSIGN.

Method `get_identification()`**:** Returns the elements of the identification pattern IdentificationBSVARSIGN as a `list`.

Usage:

```
specify_identification_bsvarSIGN$get_identification()
```

Method `set_identification()`**:** Set new starting values StartingValuesBSVARSIGN.

Usage:

```
specify_identification_bsvarSIGN$set_identification(
  N,
  sign_irf,
  sign_narrative,
  sign_structural
)
```

Arguments:

`N` a positive integer - the number of dependent variables in the model.

`sign_irf` a `NxNxH` array - sign and zero restrictions on the impulse response functions, ±1 for positive/negative sign restriction 0 for zero restrictions and NA for no restrictions, the `h`-th slice `NxN` matrix contains the restrictions on the `h-1` horizon.

`sign_narrative` a list of objects of class "narrative" - narrative sign restrictions.

`sign_structural` a `NxN` matrix with entries ±1 or NA - sign restrictions on the contemporaneous relations `B` between reduced-form errors `E` and structural shocks `U` where `BE=U`.

`max_tries` a positive integer with the maximum number of iterations for finding a rotation matrix Q that would satisfy sign restrictions

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
specify_identification_bsvarSIGN$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
# recursive specification for a 5-variable system
specify_identification_bsvarSIGN$new(N = 5)

# specify sign restrictions of the first shock on the contemporaneous IRF
# + no effect on the first variable
# + positive effect on the second variable
sign_irf = matrix(c(0, 1, rep(NA, 23)), 5, 5)
specify_identification_bsvarSIGN$new(N = 5, sign_irf = sign_irf)
```

---

specify_narrative *vector specifying one narrative restriction*

Description

The class narrative specifies a single narrative restriction.

Usage

```
specify_narrative(
  start,
  periods = 1,
  type = "S",
  sign = 1,
  shock = 1,
  var = NA
)
```

Arguments

start	positive integer - the period in which the narrative starts (greater than the number of lags).
periods	positive integer - the number of periods the narrative restriction lasts.
type	character - the type of the narrative restriction (one of "S", "A", "B"), where: "S" for restrictions on structural shocks; "A" for type A restrictions on historical decomposition, i.e. if the absolute value of the historical decomposition of shock to var is the greatest/least among all shocks; "B" for type B restrictions on historical decomposition, i.e. if the absolute value of the historical decomposition of shock to var is the greater/less than the sum of all other shocks.
sign	integer - the sign of the narrative restriction (1 or -1).
shock	positive integer - the index of the shock to which the narrative restriction applies.
var	positive integer - the index of the variable to which the narrative restriction applies.

Value

An object of class `narrative` specifying one narrative restriction.

Examples

```
# specify a narrative restriction
narrative       = specify_narrative(
                    start = 166,
                    periods = 1,
                    type = "S",
                    sign = 1,
                    shock = 1,
                    var = 6
                  )
# use it to specify the model
specification   = specify_bsvarSIGN$new(monetary, sign_narrative = list(narrative))
```

---

specify_posterior_bsvarSIGN

R6 Class Representing PosteriorBSVARSIGN

Description

The class PosteriorBSVARSIGN contains posterior output and the specification including the last MCMC draw for the Bayesian Structural VAR model with sign and narrative restrictions. Note that due to the thinning of the MCMC output the starting value in element `last_draw` might not be equal to the last draw provided in element `posterior`.

Public fields

`last_draw` an object of class BSVARSIGN with the last draw of the current MCMC run as the starting value to be passed to the continuation of the MCMC estimation using `estimate()`.

`posterior` a list containing Bayesian estimation output including: an `NxNxS` array `B`, an `NxKxS` array `A`, and a `5xS` matrix `hyper`.

Methods

Public methods:

- `specify_posterior_bsvarSIGN$new()`
- `specify_posterior_bsvarSIGN$get_posterior()`
- `specify_posterior_bsvarSIGN$is_normalised()`
- `specify_posterior_bsvarSIGN$clone()`

Method `new()`**:** Create a new posterior output PosteriorBSVARSIGN.

Usage:

```
specify_posterior_bsvarSIGN$new(specification_bsvarSIGN, posterior_bsvarSIGN)
```

Arguments:

`specification_bsvarSIGN` an object of class BSVARSIGN with the last draw of the current MCMC run as the starting value.

`posterior_bsvarSIGN` a list containing Bayesian estimation output collected in elements an `NxNxS` array `B`, an `NxKxS` array `A`, and a `5xS` matrix `hyper`.

Returns: A posterior output PosteriorBSVARSIGN.

Method `get_posterior()`**:** Returns a list containing Bayesian estimation output collected in elements an `NxNxS` array `B`, an `NxKxS` array `A`, and a `5xS` matrix `hyper`.

Usage:

```
specify_posterior_bsvarSIGN$get_posterior()
```

Examples:

```
data(optimism)
specification  = specify_bsvarSIGN$new(optimism)
set.seed(123)
estimate       = estimate(specification, 50)
estimate$get_posterior()
```

Method `is_normalised()`**:** Returns `TRUE` if the posterior has been normalised using `normalise_posterior()` and `FALSE` otherwise.

Usage:

```
specify_posterior_bsvarSIGN$is_normalised()
```

Examples:

```
data(optimism)
specification  = specify_bsvarSIGN$new(optimism)
set.seed(123)
estimate       = estimate(specification, 20)

# check normalisation status afterwards
posterior$is_normalised()
```

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
specify_posterior_bsvarSIGN$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

See Also

estimate.BSVARSIGN, specify_bsvarSIGN

Examples

```
# This is a function that is used within estimate()
data(optimism)
specification  = specify_bsvarSIGN$new(optimism, p = 4)
set.seed(123)
posterior      = estimate(specification, 50)
class(posterior)


## ------------------------------------------------
## Method `specify_posterior_bsvarSIGN$get_posterior`
## ------------------------------------------------

data(optimism)
specification  = specify_bsvarSIGN$new(optimism)
set.seed(123)
estimate       = estimate(specification, 50)
estimate$get_posterior()


## ------------------------------------------------
## Method `specify_posterior_bsvarSIGN$is_normalised`
## ------------------------------------------------

data(optimism)
specification  = specify_bsvarSIGN$new(optimism)
set.seed(123)
estimate       = estimate(specification, 20)

# check normalisation status afterwards
posterior$is_normalised()
```

`specify_prior_bsvarSIGN` *R6 Class Representing PriorBSVAR*

Description

The class PriorBSVARSIGN presents a prior specification for the homoskedastic bsvar model.

Public fields

`p` a positive integer - the number of lags.

`hyper` a `(N+3)xS` matrix of hyper-parameters $\mu, \delta, \lambda, \psi$.

`A` a `NxK` normal prior mean matrix for the autoregressive parameters.

`V` a `KxK` matrix determining the normal prior column-specific covariance for the autoregressive parameters.

`S` an `NxN` matrix determining the inverted-Wishart prior scale of error terms covariance matrix.

`nu` a positive scalar greater than `N+1` - the shape of the inverted-Wishart prior for error terms covariance matrix.

`data` an `TxN` matrix of observations.

`Y` an `NxT` matrix of dependent variables.

`X` an `KxT` matrix of independent variables.

`Ysoc` an `NxN` matrix with the sum-of-coefficients dummy observations.

`Xsoc` an `KxN` matrix with the sum-of-coefficients dummy observations.

`Ysur` an `NxN` matrix with the single-unit-root dummy observations.

`Xsur` an `KxN` matrix with the single-unit-root dummy observations.

`mu.scale` a positive scalar - the shape of the gamma prior for μ.

`mu.shape` a positive scalar - the shape of the gamma prior for μ.

`delta.scale` a positive scalar - the shape of the gamma prior for δ.

`delta.shape` a positive scalar - the shape of the gamma prior for δ.

`lambda.scale` a positive scalar - the shape of the gamma prior for λ.

`lambda.shape` a positive scalar - the shape of the gamma prior for λ.

`psi.scale` a positive scalar - the shape of the inverted gamma prior for ψ.

`psi.shape` a positive scalar - the shape of the inverted gamma prior for ψ.

`covid` NULL or a positive integer indicating the start of the COVID-19 pandemic.

Methods

Public methods:

- specify_prior_bsvarSIGN$new()
- specify_prior_bsvarSIGN$get_prior()
- specify_prior_bsvarSIGN$clone()

Method `new()`**:** Create a new prior specification PriorBSVAR.

Usage:

```
specify_prior_bsvarSIGN$new(
  data,
  p,
  exogenous = NULL,
  stationary = rep(FALSE, ncol(data))
)
```

Arguments:

`data` the `TxN` data matrix of observations.

`p` a positive integer - the autoregressive lag order of the SVAR model.

`exogenous` a `Txd` matrix of exogenous variables.

`stationary` an `N` logical vector - its element set to `FALSE` sets the prior mean for the autoregressive parameters of the `N`th equation to the white noise process, otherwise to random walk.

Returns: A new prior specification PriorBSVARSIGN.

Examples:

```
# a prior for 5-variable example with one lag and stationary data
data(optimism)
prior = specify_prior_bsvarSIGN$new(optimism, p = 1)
prior$B # show autoregressive prior mean
```

Method `get_prior()`**:** Returns the elements of the prior specification PriorBSVAR as a `list`.

Usage:

```
specify_prior_bsvarSIGN$get_prior()
```

Examples:

```
# a prior for 5-variable example with four lags
prior = specify_prior_bsvar$new(N = 5, p = 4)
prior$get_prior() # show the prior as list
```

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
specify_prior_bsvarSIGN$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
# a prior for 5-variable example with one lag
data(optimism)
prior = specify_prior_bsvarSIGN$new(optimism, p = 1)
prior$A  # show autoregressive prior mean


## ------------------------------------------------
## Method `specify_prior_bsvarSIGN$new`
## ------------------------------------------------

# a prior for 5-variable example with one lag and stationary data
data(optimism)
prior = specify_prior_bsvarSIGN$new(optimism, p = 1)
prior$B # show autoregressive prior mean


## ------------------------------------------------
## Method `specify_prior_bsvarSIGN$get_prior`
## ------------------------------------------------

# a prior for 5-variable example with four lags
prior = specify_prior_bsvar$new(N = 5, p = 4)
prior$get_prior() # show the prior as list
```

Index

∗ **datasets**
monetary, 18
optimism, 19

bsvarSIGNs *(*bsvarSIGNs-package*)*, 3
bsvarSIGNs-package, 3

compute_conditional_sd.PosteriorBSVARSIGN, 6
compute_fitted_values.PosteriorBSVARSIGN, 7
compute_historical_decompositions.PosteriorBSVARSIGN, 8
compute_impulse_responses.PosteriorBSVARSIGN, 10, *13*
compute_structural_shocks.PosteriorBSVARSIGN, 11
compute_variance_decompositions.PosteriorBSVARSIGN, 12

estimate.BSVARSIGN, *6, 7*, *9, 10*, *12, 13*, 14, *17*, *24*, *32*

forecast.PosteriorBSVARSIGN, 16

monetary, 18

optimism, 19

plot, *7*, *9, 10*, *12, 13*, *17*

specify_bsvarSIGN, 20, *32*
specify_forecasts, 26
specify_identification_bsvarSIGN, 27
specify_narrative, 29
specify_posterior_bsvarSIGN, *24*, 30
specify_prior_bsvarSIGN, 33
summary, *7*, *9, 10*, *12, 13*, *17*