Package: bvars (via r-universe)

June 8, 2026

Type Package

Title Bayesian Forecasting with Large Vector Autoregressions

Version 1.0

Date 2026-06-03

Maintainer Tomasz Woźniak <wozniak.tom@pm.me>

Description Provides fast and efficient procedures for Bayesian estimation and forecasting using state-of-the-art Vector Autoregressions. This package includes the model proposed by Chan (2020) <doi:10.1080/07350015.2018.1451336>, that is, a Bayesian Vector Autoregression with Minnesota priors and a flexible structure of the error term specification. The latter includes: conditional multivariate normal or Student's t distributions, as well as homoskedastic or heteroskedastic specifications with a common volatility modelled by centred or non-centred Stochastic Volatility. Additionally, the package facilitates predictive analyses using density forecasting and forecast-error variance decompositions. All this is complemented by simple workflows, useful plots and summary functions, and comprehensive documentation. The 'bvars' package aligns with R packages 'bsvars' by Woźniak (2024) <doi:10.32614/CRAN.package.bsvars>, 'bsvarSIGNs' by Wang & Woźniak (2025) <doi:10.32614/CRAN.package.bsvarSIGNs>, and 'bpvars' by Woźniak (2025) <doi:10.32614/CRAN.package.bpvars> regarding objects, workflows, and code structure, and they constitute an integrated toolset.

License GPL (>= 3)

Depends R (>= 4.1.0), RcppArmadillo, bsvars

Imports generics, Rcpp (>= 1.0.14), RcppProgress, RcppTN, R6

LinkingTo Rcpp, RcppArmadillo, RcppProgress, RcppTN, bsvars

URL https://bsvars.org/bvars/

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

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

NeedsCompilation yes

Author Rui Liu [aut] (ORCID: <https://orcid.org/0009-0008-9348-8581>), Andrés Ramirez Hassan [aut] (ORCID: <https://orcid.org/0000-0002-0467-7903>), Tomasz Woźniak [aut, cre, cph] (ORCID: <https://orcid.org/0000-0003-2212-2378>)

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

Date/Publication 2026-06-08 19:40:12 UTC

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

RemoteRef HEAD

RemoteSha d992ea03f00052568fe4aa799e033a08c06da027

Contents

`bvars-package` *Bayesian Forecasting with Large Vector Autoregressions*

Description

Provides fast and efficient procedures for Bayesian estimation and forecasting using state-of-the-art Vector Autoregressions. This package includes the model proposed by Chan (2020) <doi:10.1080/07350015.2018.1451336>, that is, a Bayesian Vector Autoregression with Minnesota priors and a flexible structure of the error term specification. The latter includes: conditional multivariate normal or Student's t distributions, as well as homoskedastic or heteroskedastic specifications with a common volatility modelled by centred or non-centred Stochastic Volatility. Additionally, the package facilitates predictive analyses using density forecasting and forecast-error variance decompositions. All this is complemented

by simple workflows, useful plots and summary functions, and comprehensive documentation. The 'bvars' package aligns with R packages 'bsvars' by Woźniak (2024) <doi:10.32614/CRAN.package.bsvars>, 'bsvarSIGNs' by Wang & Woźniak (2025) <doi:10.32614/CRAN.package.bsvarSIGNs>, and 'bpvars' by Woźniak (2025) <doi:10.32614/CRAN.package.bpvars> regarding objects, workflows, and code structure, and they constitute an integrated toolset.

Details

Models. All the BVAR models in this package are specified by two equations, including 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.

This package assumes that the error matrix follows a matrix normal distribution:

$$E \mid X \sim \mathcal{MN}_{N\times T}(\mathbf{0}, \mathbf{\Sigma}, \mathbf{\Omega})$$

where Σ is the `NxN` covariance matrix of the error term at time t, and Ω is a `TxT` diagonal matrix.

The diagonal elements of Ω determine the specification of the error term covariance structure. Specifically, the error term at time t follows the multivariate normal distribution

$$e_t \sim \mathcal{N}_N(\mathbf{0}, \sigma_t^2 \lambda_t \mathbf{\Sigma})$$

where the scalar processes σ_t^2 and λ_t determine the diagonal elements of Ω. The process σ_t^2 specifies conditional variance and includes three options:

$\sigma_t^2 = 1$ homoskedastic error term

σ_t^2 estimated and following non-centred stochastic volatility

σ_t^2 estimated and following centred stochastic volatility

The process λ_t specifies the conditional distribution of the error term and includes two options:

$\lambda_t = 1$ Gaussian error term specification

λ_t estimated and following a priori an inverse gamma 2 distribution $\mathcal{IG}2(\nu - 2, \nu)$, where $\nu > 2$ is a degrees of freedom parameter

Prior distributions. The autoregressive matrix A is assigned matrix-variate normal distribution:

$$\mathbf{A} \mid \underline{\mathbf{A}}, \mathbf{V}, \mathbf{\Sigma} \sim \mathcal{MN}_{N\times K}(\underline{\mathbf{A}}, \mathbf{\Sigma}, \mathbf{V})$$

with the mean matrix $\underline{\mathbf{A}}$, and covariance matrices $\mathbf{\Sigma}_{N\times N}$ and $\mathbf{V}_{K\times K}$ defining the row- and column-covariance structures.

This is complemented by the inverse Wishart prior for the error term covariance $\mathbf{\Sigma}$:

$$\mathbf{\Sigma} \mid \underline{\mathbf{S}}, \underline{\nu} \sim \mathcal{IW}(\underline{\mathbf{S}}, \underline{\nu})$$

with the scale matrix $\underline{\mathbf{S}}$ and degrees of freedom $\underline{\nu}$.

Note

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

Author(s)

Rui Liu <rl3023@columbia.edu>, Andres Ramirez Hassan <aramir21@gmail.com>, Tomasz Woźniak <wozniak.tom@pm.me>

References

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

See Also

Useful links:

- https://bsvars.org/bvars/
- Report bugs at https://github.com/bsvars/bvars/issues

Examples

```
# simple workflow
############################################################
spec = specify_bvar$new(us_macro_chan)        # specify the model
burn = estimate(spec, 5)                      # run the burn-in
post = estimate(burn, 10)                     # estimate the model

# workflow with the pipe |>
############################################################
us_macro_chan |>
  specify_bvar$new() |>
  estimate(S = 5) |>
  estimate(S = 10) -> post
```

`compute_fitted_values.PosteriorBVAR` *Computes posterior draws from data predictive density*

Description

Each of the draws from the posterior estimation of the BVAR model is transformed into a draw from the data predictive density.

Usage

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

Arguments

posterior	posterior estimation outcome - an object of class PosteriorBVAR 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)

Tomasz Woźniak <wozniak.tom@pm.me>

Examples

```
spec = specify_bvar$new(us_macro_chan)        # specify the model
burn = estimate(spec, 5)                      # run the burn-in
post = estimate(burn, 10)                     # estimate the model
fitt = compute_fitted_values(post)            # fitted values

# workflow with the pipe |>
############################################################
us_macro_chan |>
  specify_bvar$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_fitted_values() -> fitt
```

---

compute_shocks *Computes posterior draws of shocks*

Description

Each of the draws from the posterior estimation of models from the package **bvars** is transformed into a draw from the posterior distribution of the structural shocks.

Usage

```
compute_shocks(posterior)
```

Arguments

posterior	posterior estimation outcome 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 shocks.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

Examples

```
# simple workflow
############################################################
spec = specify_bvar$new(us_macro_chan)        # specify the model
burn = estimate(spec, 5)                      # run the burn-in
post = estimate(burn, 10)                     # estimate the model
shoc = compute_shocks(post)                   # compute shocks
plot(shoc)

# workflow with the pipe |>
############################################################
us_macro_chan |>
  specify_bvar$new() |>
  estimate(S = 5) |>
  estimate(S = 10) |>
  compute_shocks() |> plot()
```

---

compute_shocks.PosteriorBVAR

Computes posterior draws of shocks

Description

Each of the draws from the posterior estimation of the BAVR model is transformed into a draw from the posterior distribution of the shocks.

Usage

```
## S3 method for class 'PosteriorBVAR'
compute_shocks(posterior)
```

Arguments

posterior	posterior estimation outcome - an object of class PosteriorBVAR 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 shocks.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

Examples

```
# simple workflow
############################################################
spec = specify_bvar$new(us_macro_chan)        # specify the model
burn = estimate(spec, 5)                      # run the burn-in
post = estimate(burn, 10)                     # estimate the model
shoc = compute_shocks(post)                   # compute shocks

# workflow with the pipe |>
############################################################
us_macro_chan |>
  specify_bvar$new() |>
  estimate(S = 5) |>
  estimate(S = 10) |>
  compute_shocks() -> shoc
```

---

compute_variance_decompositions.PosteriorBVAR

Computes posterior draws of the forecast error variance decomposition

---

Description

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

Usage

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

Arguments

posterior	posterior estimation outcome obtained by running the estimate function.
horizon	a positive integer number denoting the forecast horizon for the forecast error variance decomposition 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)

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.

Examples

```
spec = specify_bvar$new(us_macro_chan)        # specify the model
burn = estimate(spec, 5)                      # run the burn-in
post = estimate(burn, 10)                     # estimate the model
fevd = compute_variance_decompositions(post, horizon = 4)

# workflow with the pipe |>
############################################################
us_macro_chan |>
  specify_bvar$new() |>
  estimate(S = 5) |>
  estimate(S = 10) |>
  compute_variance_decompositions(horizon = 4) -> fevd
```

---

estimate.BVAR *Bayesian Estimation via Gibbs sampler of a Bayesian VAR with a Flexible Error Term Specification*

Description

Estimates the model by Chan (2020) <doi:10.1080/07350015.2018.1451336>, that is, a Bayesian Vector Autoregression with Minnesota priors and a flexible structure of the error term specification. The latter includes: conditional multivariate normal or Student's t distributions, as well as homoskedastic or heteroskedastic specifications with a common volatility modelled by centred or non-centred Stochastic Volatility. The estimation is conducted via an efficient Gibbs sampler employing frontier numerical techniques and algorithms written in C++ for excellent computational speed.

Usage

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

Arguments

specification	an object of class BVAR generated using the specify_bvar$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

Value

An object of class PosteriorBVAR 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

Sigma an NxNxS array with the posterior draws for matrix Σ

V a KxKxS array with the posterior draws for the hyper-parameter matrix Σ

last_draw an object of class BVAR 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)

Rui Liu <rl3023@columbia.edu>, Andres Ramirez Hassan <aramir21@gmail.com> & Tomasz Woźniak <wozniak.tom@pm.me>

References

Barndorff-Nielsen, Blaesild, Jensen, Jorgensen (1982) Exponential Transformation Models, Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences, 379, 41—65, <doi:10.1098/rspa.1982.0004>.

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

Hamura, Irie, Sugasawa (2024) Gibbs Sampler for Matrix Generalized Inverse Gaussian Distributions, Journal of Computational and Graphical Statistics, 33(2), 331–340, <doi:10.1080/10618600.2023.2258186>.

Thabane, Safiul Haq (2004) On the Matrix-Variate Generalized Hyperbolic Distribution and Its Bayesian Applications, Statistics: A Journal of Theoretical and Applied Statistics, 38(6), 511–526, <doi:10.1080/02331880412331319279>.

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

See Also

specify_bvar, specify_posterior_bvar

Examples

```
# simple workflow
############################################################
spec = specify_bvar$new(us_macro_chan)        # specify the model
burn = estimate(spec, 5)                      # run the burn-in
post = estimate(burn, 10)                     # estimate the model

# workflow with the pipe |>
############################################################
us_macro_chan |>
  specify_bvar$new() |>
  estimate(S = 5) |>
  estimate(S = 10) -> post
```

`estimate.PosteriorBVAR` *Bayesian Estimation via Gibbs sampler of a Bayesian VAR with a Flexible Error Term Specification*

Description

Estimates the model by Chan (2020) <doi:10.1080/07350015.2018.1451336>, that is, a Bayesian Vector Autoregression with Minnesota priors and a flexible structure of the error term specification. The latter includes: conditional multivariate normal or Student's t distributions, as well as homoskedastic or heteroskedastic specifications with a common volatility modelled by centred or non-centred Stochastic Volatility. The estimation is conducted via an efficient Gibbs sampler employing frontier numerical techniques and algorithms written in C++ for excellent computational speed.

Usage

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

Arguments

`specification`	an object of class `PosteriorBVAR` generated using the `estimate.BVAR()` function. This setup facilitates the continuation of the MCMC sampling starting from the last draw of the previous run.
`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

Value

An object of class `PosteriorBVAR` 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

Sigma an `NxNxS` array with the posterior draws for matrix Σ

V a `KxKxS` array with the posterior draws for the hyper-parameter matrix Σ

`last_draw` an object of class `BVAR` 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)

Rui Liu <rl3023@columbia.edu>, Andres Ramirez Hassan <aramir21@gmail.com> & Tomasz Woźniak <wozniak.tom@pm.me>

References

Barndorff-Nielsen, Blaesild, Jensen, Jorgensen (1982) Exponential Transformation Models, Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences, 379, 41—65, <doi:10.1098/rspa.1982.0004>.

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

Hamura, Irie, Sugasawa (2024) Gibbs Sampler for Matrix Generalized Inverse Gaussian Distributions, Journal of Computational and Graphical Statistics, 33(2), 331–340, <doi:10.1080/10618600.2023.2258186>.

Thabane, Safiul Haq (2004) On the Matrix-Variate Generalized Hyperbolic Distribution and Its Bayesian Applications, Statistics: A Journal of Theoretical and Applied Statistics, 38(6), 511–526, <doi:10.1080/02331880412331319279>.

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

See Also

`specify_bvar`, `specify_posterior_bvar`

Examples

```
# simple workflow
############################################################
spec = specify_bvar$new(us_macro_chan)        # specify the model
burn = estimate(spec, 5)                      # run the burn-in
post = estimate(burn, 10)                     # estimate the model

# workflow with the pipe |>
############################################################
us_macro_chan |>
```

```
specify_bvar$new() |>
estimate(S = 5) |>
estimate(S = 10) -> post
```

`forecast.PosteriorBVAR`

Forecasting using Structural Vector Autoregression

Description

Samples from the joint predictive density of all of the dependent variables for the model by Chan (2020) <doi:10.1080/07350015.2018.1451336>, that is, a Bayesian Vector Autoregression with Minnesota priors and a flexible structure of the error term specification. The latter includes: conditional multivariate normal or Student's t distributions, as well as homoskedastic or heteroskedastic specifications with a common volatility modelled by centred or non-centred Stochastic Volatility.

Usage

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

Arguments

`object`	posterior estimation outcome - an object of class `PosteriorBVAR` 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

Value

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

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

forecast_mean an `NxhorizonxS` array with the mean of the predictive density

forecast_covariance an `NxNxhorizonxS` array with the covariance of the predictive density

Y an NxT matrix with the data on dependent variables

Author(s)

Rui Liu <rl3023@columbia.edu>, Andres Ramirez Hassan <aramir21@gmail.com> & Tomasz Woźniak <wozniak.tom@pm.me>

Examples

```
spec = specify_bvar$new(us_macro_chan)   # specify the model
burn = estimate(spec, 5)                      # run the burn-in
post = estimate(burn, 5)                     # estimate the model
pred = forecast(post, 4)                      # forecast 1 year ahead

# workflow with the pipe |>
############################################################
set.seed(123)
us_macro_chan |>
  specify_bvar$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  forecast(horizon = 4) -> pred
```

---

rmatnorm1 *Samples random numbers from the matrix-variate normal distribution*

Description

Samples random numbers from the matrix-variate normal distribution

Usage

```
rmatnorm1(M, V, S)
```

Arguments

`M`	a real-valued matrix of the expected values
`V`	a positive definite symmetric matrix of column-specific covariance
`S`	a positive definite symmetric matrix of row-specific covariance

Value

a matrix - a draw from the matrix-variate normal distribution

Examples

```
rmatnorm1(matrix(0, 2, 3), diag(2), diag(3))
```

---

specify_bvar *R6 Class representing the specification of the* BVAR *model*

Description

The class BVAR presents complete specification for the BVAR model.

Public fields

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

`prior` an object PriorBVAR with the prior specification.

`data_matrices` an object DataMatricesBSVAR with the data matrices.

`starting_values` an object StartingValuesBVAR with the starting values.

Methods

Public methods:

- specify_bvar$new()
- specify_bvar$get_normal()
- specify_bvar$get_homoskedastic()
- specify_bvar$get_centred_sv()
- specify_bvar$get_data_matrices()
- specify_bvar$get_prior()
- specify_bvar$get_starting_values()
- specify_bvar$clone()

Method `new()`**:** Create a new specification of the BVAR model.

Usage:

```
specify_bvar$new(
  data,
  p = 1L,
  exogenous = NULL,
  common_volatility = c("homoskedastic", "ncSV", "cSV"),
  distribution = c("norm", "t"),
  stationary = rep(FALSE, ncol(data))
)
```

Arguments:

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

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

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

`common_volatility` a character string specifying the common volatility component of the error term covariance matrix. It can take three values: `homoskedastic` - the model assumes homoskedastic errors, `ncSV` - the model assumes non-centred stochastic volatility, and `cSV` - the model assumes centred stochastic volatility.

`distribution` a character string specifying the conditional distribution of structural shocks. Value `"norm"` sets it to the normal distribution, while value `"t"` sets the Student-t distribution.

`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 complete specification for the `BVAR` model.

Method `get_normal()`**:** Returns the logical value of whether the conditional shock distribution is normal.

Usage:

```
specify_bvar$get_normal()
```

Examples:

```
spec = specify_bvar$new(us_macro_chan)
spec$get_normal()
```

Method `get_homoskedastic()`**:** Returns the logical value of whether the common volatility is homoskedastic.

Usage:

```
specify_bvar$get_homoskedastic()
```

Examples:

```
spec = specify_bvar$new(us_macro_chan)
spec$get_homoskedastic()
```

Method `get_centred_sv()`**:** Returns the logical value of whether the common volatility is centred Stochastic Volatility

Usage:

```
specify_bvar$get_centred_sv()
```

Examples:

```
spec = specify_bvar$new(us_macro_chan)
spec$get_centred_sv()
```

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

Usage:

```
specify_bvar$get_data_matrices()
```

Examples:

```
spec = specify_bvar$new(us_macro_chan)
spec$get_data_matrices()
```

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

Usage:

```
specify_bvar$get_prior()
```

Examples:

```
spec = specify_bvar$new(us_macro_chan)
spec$get_prior()
```

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

Usage:

```
specify_bvar$get_starting_values()
```

Examples:

```
spec = specify_bvar$new(us_macro_chan)
spec$get_starting_values()
```

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Examples

```
spec = specify_bvar$new(us_macro_chan)


## ------------------------------------------------
## Method `specify_bvar$get_normal`
## ------------------------------------------------

spec = specify_bvar$new(us_macro_chan)
spec$get_normal()


## ------------------------------------------------
## Method `specify_bvar$get_homoskedastic`
## ------------------------------------------------

spec = specify_bvar$new(us_macro_chan)
```

```
spec$get_homoskedastic()


## ------------------------------------------------
## Method `specify_bvar$get_centred_sv`
## ------------------------------------------------

spec = specify_bvar$new(us_macro_chan)
spec$get_centred_sv()


## ------------------------------------------------
## Method `specify_bvar$get_data_matrices`
## ------------------------------------------------

spec = specify_bvar$new(us_macro_chan)
spec$get_data_matrices()


## ------------------------------------------------
## Method `specify_bvar$get_prior`
## ------------------------------------------------

spec = specify_bvar$new(us_macro_chan)
spec$get_prior()


## ------------------------------------------------
## Method `specify_bvar$get_starting_values`
## ------------------------------------------------

spec = specify_bvar$new(us_macro_chan)
spec$get_starting_values()
```

`specify_posterior_bvar`

R6 Class Representing `PosteriorBVAR`

Description

The class `PosteriorBVAR` contains posterior output and the specification including the last MCMC draw for the BVAR model. 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 `BVAR` 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 collected in elements `A`, `Sigma`, and `V`.

Methods

Public methods:

- specify_posterior_bvar$new()
- specify_posterior_bvar$get_posterior()
- specify_posterior_bvar$get_last_draw()
- specify_posterior_bvar$clone()

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

Usage:

```
specify_posterior_bvar$new(specification_bvar, posterior_bvar)
```

Arguments:

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

`posterior_bvar` a list containing Bayesian estimation output collected in elements `A`, `Sigma`, and `V`.

Returns: A posterior output `PosteriorBVAR`.

Method `get_posterior()`**:** Returns a list containing Bayesian estimation output collected in elements `A`, `Sigma`, and `V`.

Usage:

```
specify_posterior_bvar$get_posterior()
```

Examples:

```
spec = specify_bvar$new(us_macro_chan)
post = estimate(spec, 5)
post$get_posterior()
```

Method `get_last_draw()`**:** Returns an object of class `BVAR` 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()`.

Usage:

```
specify_posterior_bvar$get_last_draw()
```

Examples:

```
spec = specify_bvar$new(us_macro_chan)
burn = estimate(spec, 5)
post = estimate(burn, 5)
```

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Examples

```
# This is a function that is used within estimate()
spec = specify_bvar$new(us_macro_chan)
post = estimate(spec, 5)
class(post)


## ------------------------------------------------
## Method `specify_posterior_bvar$get_posterior`
## ------------------------------------------------

spec = specify_bvar$new(us_macro_chan)
post = estimate(spec, 5)
post$get_posterior()


## ------------------------------------------------
## Method `specify_posterior_bvar$get_last_draw`
## ------------------------------------------------

spec = specify_bvar$new(us_macro_chan)
burn = estimate(spec, 5)
post = estimate(burn, 5)
```

`specify_prior_bvar` *R6 Class Representing* `PriorBVAR`

Description

The class `PriorBVAR` presents a prior specification for the BVAR model.

#' **The Model.** All the BVAR models in this package are specified by two equations, including 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.

This package assumes that the error matrix follows a matrix normal distribution:

$$E \mid X \sim \mathcal{MN}_{N \times T}(\mathbf{0}, \mathbf{\Sigma}, \mathbf{\Omega})$$

where Σ is the `NxN` covariance matrix of the error term at time t, and Ω is a `TxT` diagonal matrix.

The diagonal elements of Ω determine the specification of the error term covariance structure. Specifically, the error term at time t follows the multivariate normal distribution

$$e_t \sim \mathcal{N}_N(\mathbf{0}, \sigma_t^2 \lambda_t \mathbf{\Sigma})$$

where the scalar processes σ_t^2 and λ_t determine the diagonal elements of Ω. The process σ_t^2 specifies conditional variance and includes three options:

$\sigma_t^2 = 1$ homoskedastic error term

σ_t^2 estimated and following non-centred stochastic volatility

σ_t^2 estimated and following centred stochastic volatility

The process λ_t specifies the conditional distribution of the error term and includes two options:

$\lambda_t = 1$ Gaussian error term specification

λ_t estimated and following a priori an inverse gamma 2 distribution $\mathcal{IG}2(\nu - 2, \nu)$, where $\nu > 2$ is a degrees of freedom parameter

Prior distributions. The autoregressive matrix A is assigned matrix-variate normal distribution:

$$\mathbf{A} \mid \underline{\mathbf{A}}, \mathbf{V}, \mathbf{\Sigma} \sim \mathcal{MN}_{N \times K}(\underline{\mathbf{A}}, \mathbf{\Sigma}, \mathbf{V})$$

with the mean matrix $\underline{\mathbf{A}}$, and covariance matrices $\mathbf{\Sigma}_{N \times N}$ and $\mathbf{V}_{K \times K}$ defining the row- and column-covariance structures.

This is complemented by the inverse Wishart prior for the error term covariance $\mathbf{\Sigma}$:

$$\mathbf{\Sigma} \mid \underline{\mathbf{S}}, \underline{\nu} \sim \mathcal{IW}(\underline{\mathbf{S}}, \underline{\nu})$$

with the scale matrix $\underline{\mathbf{S}}$ and degrees of freedom $\underline{\nu}$.

Public fields

`A` a real-valued `NxK` matrix, the mean matrix A_0 of the matrix-variate normal prior distribution for the parameter matrix A.

`S` a `NxN` positive definite scale matrix S_0 of the Inverse Wishart prior distribution for the error term covariance matrix Σ.

`nu` a positive scalar, shape parameter ν_0 of the Inverse Wishart prior distribution for the error term covariance matrix Σ.

`Psi` a `KxK` scale matrix Ψ_0 of the matrix generalized inverse Gaussian distribution for the equation-specific prior covariance V

`Gamma` a `KxK` scale matrix Γ_0 of the matrix generalized inverse Gaussian distribution for the equation-specific prior covariance V

`lambda` a positive scalar shape parameter λ_0 of the matrix generalized inverse Gaussian distribution for the equation-specific prior covariance V

`sv_a` a positive scalar, the shape parameter of the gamma prior in the hierarchical prior for the common stochastic volatility.

`sv_s` a positive scalar, the scale parameter of the gamma prior in the hierarchical prior for the common stochastic volatility.

Methods

Public methods:

- `specify_prior_bvar$new()`
- `specify_prior_bvar$get_prior()`
- `specify_prior_bvar$clone()`

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

Usage:

```
specify_prior_bvar$new(
  N,
  p,
  d = 0,
  stationary = rep(FALSE, N),
  is_homoskedastic = TRUE
)
```

Arguments:

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

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

`d` a positive integer - the number of `exogenous` variables in the model.

`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.

`is_homoskedastic` a logical scalar - if `TRUE` the model assumes homoskedastic errors, otherwise it assumes stochastic volatility.

Returns: A new prior specification `PriorBVAR`.

Examples:

```
# a prior for 3-variable example with one lag and stationary data
prior = specify_prior_bvar$new(N = 3, p = 1, stationary = rep(TRUE, 3))
prior$A # show autoregressive prior mean
```

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

Usage:

```
specify_prior_bvar$get_prior()
```

Examples:

```
# a prior for 3-variable example with four lags
prior = specify_prior_bvar$new(N = 3, p = 4)
prior$get_prior() # show the prior as list
```

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Examples

```
prior = specify_prior_bvar$new(N = 3, p = 1)  # a prior for 3-variable example with one lag
prior$A                                       # show autoregressive prior mean
```

```
## ------------------------------------------------
## Method `specify_prior_bvar$new`
## ------------------------------------------------

# a prior for 3-variable example with one lag and stationary data
prior = specify_prior_bvar$new(N = 3, p = 1, stationary = rep(TRUE, 3))
prior$A # show autoregressive prior mean


## ------------------------------------------------
## Method `specify_prior_bvar$get_prior`
## ------------------------------------------------

# a prior for 3-variable example with four lags
prior = specify_prior_bvar$new(N = 3, p = 4)
prior$get_prior() # show the prior as list
```

`specify_starting_values_bvar`
R6 Class Representing `StartingValuesBVAR`

Description

The class `StartingValuesBVAR` presents starting values for the BVAR model.

Public fields

`A` an `NxK` matrix of starting values for the autoregressive matrix A.

`Sigma` an `NxN` matrix of starting values for the error term covariance Σ.

`V` a `KxK` matrix of starting values for the prior equation-specific covariance V of the hierarchical prior distribution for matrix A.

`h` an `T`-vector with the starting values of the log-volatility processes.

`rho` a scalalr for the SV autoregressive parameter.

`omega` a scalar for the SV process conditional standard deviation.

`sigma2v` a scalar for SV process conditional variances.

`S` a `T` integer vector with the auxiliary mixture component indicator.

`sigma2_omega` a scalar for the variance of the zero-mean normal prior for ω.

`s_` a positive scalar with the scale of the gamma prior of the hierarchical prior for σ_ω^2.

`lambda` a `T`-vetor of starting values for latent variable.

`df` a scalar greater than 2 with the starting value for the degrees of freedom parameter of the Student-t conditional distribution of error term.

Methods

Public methods:

- specify_starting_values_bvar$new()
- specify_starting_values_bvar$get_starting_values()
- specify_starting_values_bvar$set_starting_values()
- specify_starting_values_bvar$clone()

Method `new()`**:** Create new starting values `StartingValuesBVAR`.

Usage:

```
specify_starting_values_bvar$new(
  N,
  p,
  T,
  d = 0,
  is_homoskedastic = TRUE,
  is_normal = TRUE,
  ar_sigma2 = rep(1, N),
  kappa = c(0.2^2, 10^2)
)
```

Arguments:

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

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

`T` a positive integer - the number of time periods in the data.

`d` a positive integer - the number of `exogenous` variables in the model.

`is_homoskedastic` a logical scalar - if `TRUE` the model assumes homoskedastic errors, otherwise it assumes stochastic volatility.

`is_normal` a logical scalar - if `TRUE` the model assumes normal error term, otherwise, it assumes Student-t errors.

`ar_sigma2` a positive `N`-vector with the autoregressive variance estimates for each variable to be used in the Minnesota prior for the autoregressive parameters.

`kappa` a positive `2`-vector with the hyperparameters of the Minnesota prior for the autoregressive parameters - the first element is the overall tightness hyperparameter, while the second element is the tightness of the prior on the constant and exogenous variable coefficients.

Returns: Starting values `StartingValuesBVAR`.

Examples:

```
# starting values for a 3-variable BVAR model
sv = specify_starting_values_bvar$new(N = 3, p = 4, T = 100)
```

Method `get_starting_values()`**:** Returns the elements of the starting values `StartingValuesBVAR` as a `list`.

Usage:

```
specify_starting_values_bvar$get_starting_values()
```

Examples:

```
# starting values for a 3-variable BVAR model
sv = specify_starting_values_bvar$new(N = 3, p = 4, T = 100)
sv$get_starting_values()   # show starting values as list
```

Method `set_starting_values()`**:** Sets the elements of the starting values `StartingValuesBVAR` to provided values.

Usage:

```
specify_starting_values_bvar$set_starting_values(last_draw)
```

Arguments:

`last_draw` a list containing the last draw of elements `A` - a `KxN` matrix, `Sigma` - an `NxN` matrix, and `V` - a `KxK` matrix.

Returns: An object of class `StartingValuesBVAR` including the last draw of the current MCMC as the starting value to be passed to the continuation of the MCMC estimation using `estimate()`.

Examples:

```
# starting values for a 3-variable BVAR model
sv = specify_starting_values_bvar$new(N = 3, p = 4, T = 100)

# Modify the starting values by:
sv_list = sv$get_starting_values()   # getting them as list
sv_list$A <- matrix(rnorm(12), 3, 4) # modifying the entry
sv$set_starting_values(sv_list)      # providing to the class object
```

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Examples

```
# starting values for a 3-variable BVAR model.
sv = specify_starting_values_bvar$new(N = 3, p = 4, T = 100)


## ------------------------------------------------
## Method `specify_starting_values_bvar$new`
## ------------------------------------------------

# starting values for a 3-variable BVAR model
sv = specify_starting_values_bvar$new(N = 3, p = 4, T = 100)


## ------------------------------------------------
## Method `specify_starting_values_bvar$get_starting_values`
```

```
## ------------------------------------------------

# starting values for a 3-variable BVAR model
sv = specify_starting_values_bvar$new(N = 3, p = 4, T = 100)
sv$get_starting_values()   # show starting values as list


## ------------------------------------------------
## Method `specify_starting_values_bvar$set_starting_values`
## ------------------------------------------------

# starting values for a 3-variable BVAR model
sv = specify_starting_values_bvar$new(N = 3, p = 4, T = 100)

# Modify the starting values by:
sv_list = sv$get_starting_values()   # getting them as list
sv_list$A <- matrix(rnorm(12), 3, 4) # modifying the entry
sv$set_starting_values(sv_list)      # providing to the class object
```

`summary.PosteriorBVAR` *Provides posterior summary of VAR estimation*

Description

Provides posterior mean, standard deviations, as well as 5 and 95 percentiles of the parameters: autoregressive parameters $\mathbf{A}$, and the covariance matrix Σ.

Usage

```
## S3 method for class 'PosteriorBVAR'
summary(object, ...)
```

Arguments

`object`	an object of class PosteriorBVAR obtained using the `estimate()` function applied to a Bayesian VAR model specification set by function `specify_bvar$new()` containing draws from the posterior distribution of the parameters.
`...`	additional arguments affecting the summary produced.

Value

A list reporting the posterior mean, standard deviations, as well as 5 and 95 percentiles of the parameters: autoregressive parameters $\mathbf{A}$, and the covariance matrix Σ..

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

Examples

```
# simple workflow
############################################################
spec = specify_bvar$new(us_macro_chan)        # specify the model
burn = estimate(spec, 5)                      # run the burn-in
post = estimate(burn, 10)                     # estimate the model
summary(post)

# workflow with the pipe |>
############################################################
us_macro_chan |>
  specify_bvar$new() |>
  estimate(S = 5)  |>
  estimate(S = 10) |>
  summary()
```

`us_macro_chan` *A 20-variable US macroeconomic system for the period 1959 Q4 – 2013 Q4*

Description

A system of 20 US macroeconomic aggregates used by Chan (2020).

Usage

```
data(us_macro_chan)
```

Format

A matrix and a `ts` object with time series of 217 observations on 20 variables:

rgdp Real gross domestic product

cpi Consumer price index

FFR Effective Federal funds rate

m2 M2 money stock

pinc Personal income

rpce Real personal consumption expenditure

ip Industrial production index

UR Civilian unemployment rate

hs Housing starts

pci Producer price index

pce Personal consumption expenditures: chain-type price index

ahem Average hourly earnings: manufacturing

mi MI money stock

TMR10Y 10-Year Treasury constant maturity rate

rgpdi Real gross private domestic investment

aetnf All employees: total nonfarm

pmici ISM manufacturing: PMI composite index

noi ISM manufacturing: new orders index

bsro Business sector: real output per hour of all Persons

sp500 Real stock prices (S& P 500 index divided by PCE index

The series are used and described by Chan (2020) in Appendix B of Supplementary Materials available at <doi:10.1080/07350015.2018.1451336>.

Source

FRED Economic Database, Federal Reserve Bank of St. Louis, https://fred.stlouisfed.org/

References

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

Examples

```
data(us_macro_chan)   # upload the data
```

Index