Package: bsvars (via r-universe)

September 29, 2026

Type Package

Title Bayesian Estimation of Structural Vector Autoregressive Models

Description Provides fast and efficient procedures for Bayesian analysis of Structural Vector Autoregressions. This package estimates a wide range of models, including homo-, heteroskedastic, and non-normal specifications. Structural models can be identified by adjustable exclusion restrictions, time-varying volatility, or non-normality, and include exclusion restrictions on autoregressive parameters. They all include a flexible three-level equation-specific local-global hierarchical prior distribution for the estimated level of shrinkage for autoregressive and structural parameters. Additionally, the package facilitates predictive and structural analyses such as impulse responses, forecast error variance and historical decompositions, forecasting, verification of heteroskedasticity, non-normality, and hypotheses on autoregressive parameters, as well as analyses of structural shocks, volatilities, and fitted values. Beautiful plots, informative summary functions, and extensive documentation including the vignette by Woźniak (2025) <doi:10.48550/arXiv.2410.15090> complement all this. The implemented techniques align closely with those presented in Lütkepohl, Shang, Uzeda, & Woźniak (2025) <doi:10.1016/j.jeconom.2025.106107>, Lütkepohl & Woźniak (2020) <doi:10.1016/j.jedc.2020.103862>, and Song & Woźniak (2021) <doi:10.1093/acrefore/9780190625979.013.174> and they embed many popular models proposed by other authors. The 'bsvars' package is aligned regarding objects, workflows, and code structure with the R packages 'bsvarSIGNs' by Wang & Woźniak (2025) <doi:10.32614/CRAN.package.bsvarSIGNs>, '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.

Version 4.0.0.9000

Date 2026-08-24

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

Depends R (>= 4.1.0)

Imports Rcpp (>= 1.0.7), RcppProgress (>= 0.1), RcppTN, GIGrvg, R6, stochvol, generics

Suggests knitr, tinytest

LinkingTo Rcpp, RcppProgress, RcppArmadillo, RcppTN

License GPL (>= 3)

URL https://bsvars.org/bsvars/

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

Encoding UTF-8

LazyData true

VignetteBuilder knitr

Config/roxygen2/version 8.0.0

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

Date/Publication 2026-09-29 06:52:48 UTC

RemoteUrl https://github.com/bsvars/bsvars

RemoteRef HEAD

RemoteSha 0606b87499a6bc8077dc5a07b7c026fef7aa65e9

Contents

bsvars-package 6
compute_conditional_sd 10
compute_conditional_sd.PosteriorBSVAR 11
compute_conditional_sd.PosteriorBSVAREXH 12
compute_conditional_sd.PosteriorBSVARHMSH 13
compute_conditional_sd.PosteriorBSVARMIX 14
compute_conditional_sd.PosteriorBSVARMSH 15
compute_conditional_sd.PosteriorBSVARSV 16
compute_conditional_sd.PosteriorBSVART 17
compute_fitted_values 18
compute_fitted_values.PosteriorBSVAR 19
compute_fitted_values.PosteriorBSVAREXH 20
compute_fitted_values.PosteriorBSVARHMSH 21
compute_fitted_values.PosteriorBSVARMIX 22
compute_fitted_values.PosteriorBSVARMSH 23
compute_fitted_values.PosteriorBSVARSV 24
compute_fitted_values.PosteriorBSVART 25
compute_historical_decompositions 26
compute_historical_decompositions.PosteriorBSVAR 27
compute_historical_decompositions.PosteriorBSVAREXH 28

compute_historical_decompositions.PosteriorBSVARHMSH 29
compute_historical_decompositions.PosteriorBSVARMIX 30
compute_historical_decompositions.PosteriorBSVARMSH 32
compute_historical_decompositions.PosteriorBSVARSV 33
compute_historical_decompositions.PosteriorBSVART 34
compute_impulse_responses 35
compute_impulse_responses.PosteriorBSVAR 36
compute_impulse_responses.PosteriorBSVAREXH 38
compute_impulse_responses.PosteriorBSVARHMSH 39
compute_impulse_responses.PosteriorBSVARMIX 40
compute_impulse_responses.PosteriorBSVARMSH 42
compute_impulse_responses.PosteriorBSVARSV 43
compute_impulse_responses.PosteriorBSVART 44
compute_regime_probabilities 46
compute_regime_probabilities.PosteriorBSVAREXH 47
compute_regime_probabilities.PosteriorBSVARHMSH 48
compute_regime_probabilities.PosteriorBSVARMIX 50
compute_regime_probabilities.PosteriorBSVARMSH 51
compute_structural_shocks 53
compute_structural_shocks.PosteriorBSVAR 54
compute_structural_shocks.PosteriorBSVAREXH 55
compute_structural_shocks.PosteriorBSVARHMSH 56
compute_structural_shocks.PosteriorBSVARMIX 57
compute_structural_shocks.PosteriorBSVARMSH 58
compute_structural_shocks.PosteriorBSVARSV 59
compute_structural_shocks.PosteriorBSVART 60
compute_variance_decompositions 61
compute_variance_decompositions.PosteriorBSVAR 62
compute_variance_decompositions.PosteriorBSVAREXH 63
compute_variance_decompositions.PosteriorBSVARHMSH 64
compute_variance_decompositions.PosteriorBSVARMIX 65
compute_variance_decompositions.PosteriorBSVARMSH 67
compute_variance_decompositions.PosteriorBSVARSV 68
compute_variance_decompositions.PosteriorBSVART 69
estimate 70
estimate.BSVAR 73
estimate.BSVAREXH 75
estimate.BSVARHMSH 77
estimate.BSVARMIX 80
estimate.BSVARMSH 82
estimate.BSVARSV 85
estimate.BSVART 88
estimate.PosteriorBSVAR 90
estimate.PosteriorBSVAREXH 92
estimate.PosteriorBSVARHMSH 94
estimate.PosteriorBSVARMIX 97
estimate.PosteriorBSVARMSH 99
estimate.PosteriorBSVARSV 102

estimate.PosteriorBSVART . . . 105
forecast.PosteriorBSVAR . . . 107
forecast.PosteriorBSVAREXH . . . 109
forecast.PosteriorBSVARHMSH . . . 111
forecast.PosteriorBSVARMIX . . . 112
forecast.PosteriorBSVARMSH . . . 114
forecast.PosteriorBSVARSV . . . 116
forecast.PosteriorBSVART . . . 118
normalise . . . 120
normalise.PosteriorBSVAR . . . 121
normalise.PosteriorBSVAREXH . . . 122
normalise.PosteriorBSVARHMSH . . . 124
normalise.PosteriorBSVARMIX . . . 125
normalise.PosteriorBSVARMSH . . . 126
normalise.PosteriorBSVARSV . . . 127
normalise.PosteriorBSVART . . . 128
plot.Forecasts . . . 130
plot.PosteriorFEVD . . . 131
plot.PosteriorFitted . . . 132
plot.PosteriorHD . . . 134
plot.PosteriorIR . . . 135
plot.PosteriorRegimePr . . . 136
plot.PosteriorShocks . . . 138
plot.PosteriorSigma . . . 139
plot_ribbon . . . 141
specify_bsvar . . . 142
specify_bsvar_exh . . . 146
specify_bsvar_hmsh . . . 149
specify_bsvar_mix . . . 153
specify_bsvar_msh . . . 155
specify_bsvar_sv . . . 159
specify_bsvar_t . . . 163
specify_data_matrices . . . 165
specify_forecasts . . . 166
specify_identification_bsvars . . . 168
specify_posterior_bsvar . . . 170
specify_posterior_bsvar_exh . . . 175
specify_posterior_bsvar_hmsh . . . 179
specify_posterior_bsvar_mix . . . 183
specify_posterior_bsvar_msh . . . 187
specify_posterior_bsvar_sv . . . 192
specify_posterior_bsvar_t . . . 197
specify_prior_bsvar . . . 201
specify_prior_bsvar_exh . . . 204
specify_prior_bsvar_mix . . . 206
specify_prior_bsvar_msh . . . 207
specify_prior_bsvar_sv . . . 209
specify_prior_bsvar_t . . . 211

specify_starting_values_bsvar . . . 212
specify_starting_values_bsvar_exh . . . 215
specify_starting_values_bsvar_hmsh . . . 218
specify_starting_values_bsvar_mix . . . 221
specify_starting_values_bsvar_msh . . . 222
specify_starting_values_bsvar_sv . . . 225
summary.Forecasts . . . 228
summary.PosteriorBSVAR . . . 229
summary.PosteriorBSVAREXH . . . 230
summary.PosteriorBSVARHMSH . . . 231
summary.PosteriorBSVARMIX . . . 232
summary.PosteriorBSVARMSH . . . 234
summary.PosteriorBSVARSV . . . 235
summary.PosteriorBSVART . . . 236
summary.PosteriorFEVD . . . 237
summary.PosteriorFitted . . . 238
summary.PosteriorHD . . . 239
summary.PosteriorIR . . . 241
summary.PosteriorRegimePr . . . 242
summary.PosteriorShocks . . . 243
summary.PosteriorSigma . . . 244
summary.SDDRautoregression . . . 245
summary.SDDRidMIX . . . 246
summary.SDDRidMSH . . . 247
summary.SDDRidSV . . . 249
summary.SDDRidT . . . 250
summary.SDDRvolatility . . . 251
us_fiscal_cond_forecasts . . . 252
us_fiscal_ex . . . 253
us_fiscal_ex_forecasts . . . 254
us_fiscal_lsuw . . . 255
us_fiscal_sww . . . 256
verify_autoregression . . . 257
verify_autoregression.PosteriorBSVAR . . . 258
verify_autoregression.PosteriorBSVAREXH . . . 260
verify_autoregression.PosteriorBSVARHMSH . . . 262
verify_autoregression.PosteriorBSVARMIX . . . 263
verify_autoregression.PosteriorBSVARMSH . . . 265
verify_autoregression.PosteriorBSVARSV . . . 267
verify_autoregression.PosteriorBSVART . . . 268
verify_identification . . . 270
verify_identification.PosteriorBSVAR . . . 272
verify_identification.PosteriorBSVAREXH . . . 273
verify_identification.PosteriorBSVARHMSH . . . 274
verify_identification.PosteriorBSVARMIX . . . 276
verify_identification.PosteriorBSVARMSH . . . 277
verify_identification.PosteriorBSVARSV . . . 279
verify_identification.PosteriorBSVART . . . 280

verify_normality 282
verify_normality.PosteriorBSVAREXH 283
verify_normality.PosteriorBSVARHMSH 285
verify_normality.PosteriorBSVARMIX 287
verify_normality.PosteriorBSVARMSH 288
verify_normality.PosteriorBSVARSV 290
verify_volatility 291
verify_volatility.PosteriorBSVAR 293
verify_volatility.PosteriorBSVAREXH 294
verify_volatility.PosteriorBSVARHMSH 296
verify_volatility.PosteriorBSVARMIX 298
verify_volatility.PosteriorBSVARMSH 299
verify_volatility.PosteriorBSVARSV 301

Index **304**

`bsvars-package` *Bayesian Estimation of Structural Vector Autoregressive Models*

Description

Provides fast and efficient procedures for Bayesian analysis of Structural Vector Autoregressions. This package estimates a wide range of models, including homo-, heteroskedastic, and non-normal specifications. Structural models can be identified by adjustable exclusion restrictions, time-varying volatility, or non-normality, and include exclusion restrictions on autoregressive parameters. They all include a flexible three-level equation-specific local-global hierarchical prior distribution for the estimated level of shrinkage for autoregressive and structural parameters. Additionally, the package facilitates predictive and structural analyses such as impulse responses, forecast error variance and historical decompositions, forecasting, verification of heteroskedasticity, non-normality, and hypotheses on autoregressive parameters, as well as analyses of structural shocks, volatilities, and fitted values. Beautiful plots, informative summary functions, and extensive documentation including the vignette by Woźniak (2025) <doi:10.48550/arXiv.2410.15090> complement all this. The implemented techniques align closely with those presented in Lütkepohl, Shang, Uzeda, & Woźniak (2025) <doi:10.1016/j.jeconom.2025.106107>, Lütkepohl & Woźniak (2020) <doi:10.1016/j.jedc.2020.103862>, and Song & Woźniak (2021) <doi:10.1093/acrefore/9780190625979.013.174> and they embed many popular models proposed by other authors. The 'bsvars' package is aligned regarding objects, workflows, and code structure with the R packages 'bsvarSIGNs' by Wang & Woźniak (2025) <doi:10.32614/CRAN.package.bsvarSIGNs>, 'bvars' by Liu, Ramirez Hassan, Woźniak (2026) <doi:10.32614/CRAN.package.bvars>, and 'bpvars' by Woźniak (2026) <doi:10.32614/CRAN.package.bpv and they constitute an integrated toolset.

Details

Models. All the SVAR 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.

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.

Finally, all of the models share assumptions regarding the structural shocks U, namely, temporal and contemporaneous independence. They imply zero correlations and autocorrelations.

The various SVAR models estimated differ by the specification of structural shocks variances. The different models include:

- homoskedastic model with unit variances
- heteroskedastic model with non-centred Stochastic Volatility process for variances
- heteroskedastic model with centred Stochastic Volatility process for variances
- heteroskedastic model with stationary Markov switching in the variances
- heteroskedastic model with sparse Markov switching in the variances where the number of heteroskedastic components is estimated
- heteroskedastic model with stationary heterogeneous Markov switching in the variances, where each shock volatility has its own Markov process
- heteroskedastic model with sparse heterogeneous Markov switching in the variances where the number of heteroskedastic components is estimated
- heteroskedastic model with exogenous heteroskedastic regime changes in the variances
- a model with Student-t distributed structural shocks with estimated equation-specific degrees-of-freedom parameter
- non-normal model with a finite mixture of normal components and component-specific variances
- non-normal model with a sparse mixture of normal components and component-specific variances where the number of heteroskedastic components is estimated

The structural shocks can be either normally or Student-t distributed, where in the latter case the shock-specific degrees of freedom parameters are estimated.

Prior distributions. All the models feature a Minnesota prior for autoregressive parameters in matrix A and a generalised-normal distribution for the structural matrix B. Both of these distributions feature a 3-level equation-specific local-global hierarchical prior that make the shrinkage estimation flexible improving the model fit and its forecasting performance.

Estimation algorithm. The models are estimated using frontier numerical methods making the Gibbs sampler fast and efficient. The estimation follows closely Lütkepohl, Shang, Uzeda, & Woźniak (2025). The sampler of the structural matrix follows Waggoner & Zha (2003), whereas that for autoregressive parameters follows Chan, Koop, Yu (2022). The specification of Markov switching heteroskedasticity is inspired by Song & Woźniak (2021), and that of Stochastic Volatility model by Kastner & Frühwirth-Schnatter (2014). The identification problems are considered in Lütkepohl, Shang, Uzeda, & Woźniak (2025) and Lütkepohl & Woźniak (2020).

Identification verification. The structural shocks can be identified through heteroskedasticity or non-normality following Lütkepohl, Shang, Uzeda, & Woźniak (2025) and Lütkepohl & Woźniak (2020). The package provides functions to verify both, homoskedasticity and normality of the structural shocks, which facilitates making probabilistic statements regarding the identification. Additionally, the package makes it possible to verify linear restrictions on autoregressive parameters.

Note

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

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Chan, J.C.C., Koop, G, and Yu, X. (2024) Large Order-Invariant Bayesian VARs with Stochastic Volatility. *Journal of Business & Economic Statistics*, **42**, doi:10.1080/07350015.2023.2252039.

Kastner, G. and Frühwirth-Schnatter, S. (2014) Ancillarity-Sufficiency Interweaving Strategy (ASIS) for Boosting MCMC Estimation of Stochastic Volatility Models. *Computational Statistics & Data Analysis*, **76**, 408–423, doi:10.1016/j.csda.2013.01.002.

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

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2025) Partial identification of structural vector autoregressions with non-centred stochastic volatility. *Journal of Econometrics* **256**, 106107, doi:10.1016/j.jeconom.2025.106107.

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Song, Y., and Woźniak, T. (2021) Markov Switching Heteroskedasticity in Time Series Analysis. In: *Oxford Research Encyclopedia of Economics and Finance*. Oxford University Press, doi:10.1093/acrefore/9780190625979.013.174.

Waggoner, D.F., and Zha, T., (2003) A Gibbs sampler for structural vector autoregressions. *Journal of Economic Dynamics and Control*, **28**, 349–366, doi:10.1016/S01651889(02)001689.

Wang X, Woźniak T (2025). bsvarSIGNs: Bayesian SVARs with Sign, Zero, and Narrative Restrictions. R package version 2.0, doi:10.32614/CRAN.package.bsvarSIGNs.

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

See Also

Useful links:

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

Examples

```
spec  = specify_bsvar_sv$new(         # specify the model
          us_fiscal_lsuw,
          exogenous = us_fiscal_ex
        )
burn  = estimate(spec, 5)             # run the burn-in
post  = estimate(burn, 5)             # estimate the model
irf   = compute_impulse_responses(    # compute impulse responses
          post,
          horizon = 2
         )

# compute forecast error variance decomposition one year ahead
fevd  = compute_variance_decompositions(post, horizon = 4)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_sv$new(exogenous = us_fiscal_ex) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_variance_decompositions(horizon = 4) -> fevds

# conditional forecasting using a model with exogenous variables
############################################################
us_fiscal_lsuw |>
  specify_bsvar_sv$new(exogenous = us_fiscal_ex) |>
  estimate(S = 5) |>
  estimate(S = 5) -> post

 post |> forecast(
    horizon = 8,
    exogenous_forecast = us_fiscal_ex_forecasts,
    conditional_forecast = us_fiscal_cond_forecasts
  ) -> pred

  pred |> summary()
  pred |> plot(probability = 0.68)

# estimation of a model with exogeneity restrictions on the  autoregressive matrix
############################################################
A = matrix(TRUE, 3, 7)
A[1,3] = A[1,6] = FALSE
us_fiscal_lsuw |>
  specify_bsvar_sv$new(p = 2, A = A) |>
  estimate(S = 5) |>
  estimate(S = 5) -> post
post |> summary()
```

compute_conditional_sd *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

```
compute_conditional_sd(posterior)
```

Arguments

posterior	posterior estimation outcome obtained by running the estimate function. The interpretation depends on the normalisation of the shocks using function normalise(). Verify if the default settings are appropriate.

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)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

estimate, normalise, summary

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)
sigma          = compute_conditional_sd(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_conditional_sd() -> csd
```

`compute_conditional_sd.PosteriorBSVAR` *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 'PosteriorBSVAR'
compute_conditional_sd(posterior)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVAR` 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)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

`estimate`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)
sigma          = compute_conditional_sd(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_conditional_sd() -> csd
```

`compute_conditional_sd.PosteriorBSVAREXH` *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 'PosteriorBSVAREXH'
compute_conditional_sd(posterior)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVAREXH` 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)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

`estimate`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar_exh$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)
csd     = compute_conditional_sd(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_exh$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_conditional_sd() -> csd
```

`compute_conditional_sd.PosteriorBSVARHMSH` *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 'PosteriorBSVARHMSH'
compute_conditional_sd(posterior)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVARHMSH` 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)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

`estimate`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)
csd     = compute_conditional_sd(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_hmsh$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_conditional_sd() -> csd
```

`compute_conditional_sd.PosteriorBSVARMIX` *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 'PosteriorBSVARMIX'
compute_conditional_sd(posterior)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVARMIX` 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)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

`estimate`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, M = 2)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)
csd     = compute_conditional_sd(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_mix$new(M = 2) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_conditional_sd() -> csd
```

`compute_conditional_sd.PosteriorBSVARMSH` *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 'PosteriorBSVARMSH'
compute_conditional_sd(posterior)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVARMSH` 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)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

`estimate`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, M = 2)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)
csd     = compute_conditional_sd(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new(M = 2) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_conditional_sd() -> csd
```

`compute_conditional_sd.PosteriorBSVARSV` *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 'PosteriorBSVARSV'
compute_conditional_sd(posterior)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVARSV` 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)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

`estimate`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar_sv$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)
csd     = compute_conditional_sd(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_sv$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_conditional_sd() -> csd
```

`compute_conditional_sd.PosteriorBSVART` *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 'PosteriorBSVART'
compute_conditional_sd(posterior)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVART` 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)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

`estimate`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar_t$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)
csd     = compute_conditional_sd(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_t$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_conditional_sd() -> csd
```

`compute_fitted_values` *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

```
compute_fitted_values(posterior)
```

Arguments

`posterior`	posterior estimation outcome 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>

See Also

`estimate`, `summary`

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new(p = 1) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_fitted_values() -> fitted
```

compute_fitted_values.PosteriorBSVAR *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 'PosteriorBSVAR'
compute_fitted_values(posterior)
```

Arguments

posterior	posterior estimation outcome - an object of class PosteriorBSVAR 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>

See Also

estimate, summary

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw, p = 1)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new(p = 1) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_fitted_values() -> fitted
```

compute_fitted_values.PosteriorBSVAREXH *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 'PosteriorBSVAREXH'
compute_fitted_values(posterior)
```

Arguments

posterior	posterior estimation outcome - an object of class PosteriorBSVAREXH 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>

See Also

estimate, summary

Examples

```
specification  = specify_bsvar_exh$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_exh$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_fitted_values() -> csd
```

`compute_fitted_values.PosteriorBSVARHMSH` *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 'PosteriorBSVARHMSH'
compute_fitted_values(posterior)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVARHMSH` 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>

See Also

`estimate`, `summary`

Examples

```
specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_hmsh$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_fitted_values() -> csd
```

compute_fitted_values.PosteriorBSVARMIX *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 'PosteriorBSVARMIX'
compute_fitted_values(posterior)
```

Arguments

posterior	posterior estimation outcome - an object of class PosteriorBSVARMIX 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>

See Also

estimate, summary

Examples

```
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, M = 2)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_mix$new(M = 2) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_fitted_values() -> csd
```

compute_fitted_values.PosteriorBSVARMSH *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 'PosteriorBSVARMSH'
compute_fitted_values(posterior)
```

Arguments

posterior	posterior estimation outcome - an object of class PosteriorBSVARMSH 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>

See Also

estimate, summary

Examples

```
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, M = 2)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new(M = 2) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_fitted_values() -> csd
```

`compute_fitted_values.PosteriorBSVARSV` *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 'PosteriorBSVARSV'
compute_fitted_values(posterior)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVARSV` 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>

See Also

`estimate`, `summary`

Examples

```
specification  = specify_bsvar_sv$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_sv$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_fitted_values() -> csd
```

compute_fitted_values.PosteriorBSVART *Computes posterior draws from data predictive density*

Description

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

Usage

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

Arguments

posterior	posterior estimation outcome - an object of class PosteriorBSVART 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>

See Also

estimate, summary

Examples

```
specification  = specify_bsvar_t$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_t$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_fitted_values() -> fitted
```

`compute_historical_decompositions` *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

```
compute_historical_decompositions(posterior, show_progress = TRUE)
```

Arguments

`posterior`	posterior estimation outcome obtained by running the `estimate` function. The interpretation depends on the normalisation of the shocks using function `normalise()`. Verify if the default settings are appropriate.
`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)

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

References

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

See Also

`estimate`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar$new(diff(us_fiscal_lsuw))
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute historical decompositions
hd             = compute_historical_decompositions(posterior)
```

```
# workflow with the pipe |>
############################################################
diff(us_fiscal_lsuw) |>
  specify_bsvar$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_historical_decompositions() -> hd
```

`compute_historical_decompositions.PosteriorBSVAR`
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 'PosteriorBSVAR'
compute_historical_decompositions(posterior, show_progress = TRUE)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVAR` 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)

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`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar$new(diff(us_fiscal_lsuw))
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute historical decompositions
hd             = compute_historical_decompositions(posterior)

# workflow with the pipe |>
############################################################
diff(us_fiscal_lsuw) |>
  specify_bsvar$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_historical_decompositions() -> hd
```

`compute_historical_decompositions.PosteriorBSVAREXH`
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 'PosteriorBSVAREXH'
compute_historical_decompositions(posterior, show_progress = TRUE)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVAREXH` 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)

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, normalise, summary

Examples

```
specification  = specify_bsvar_exh$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute historical decompositions
hd             = compute_historical_decompositions(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_exh$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_historical_decompositions() -> hds
```

---

compute_historical_decompositions.PosteriorBSVARHMSH

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 'PosteriorBSVARHMSH'
compute_historical_decompositions(posterior, show_progress = TRUE)
```

Arguments

posterior	posterior estimation outcome - an object of class PosteriorBSVARHMSH 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)

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`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute historical decompositions
hd             = compute_historical_decompositions(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_hmsh$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_historical_decompositions() -> hds
```

---

compute_historical_decompositions.PosteriorBSVARMIX

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 'PosteriorBSVARMIX'
compute_historical_decompositions(posterior, show_progress = TRUE)
```

Arguments

posterior	posterior estimation outcome - an object of class PosteriorBSVARMIX 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)

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, normalise, summary

Examples

```
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, M = 2)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute historical decompositions
hd             = compute_historical_decompositions(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_mix$new(M = 2) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_historical_decompositions() -> hds
```

`compute_historical_decompositions.PosteriorBSVARMSH` *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 'PosteriorBSVARMSH'
compute_historical_decompositions(posterior, show_progress = TRUE)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVARMSH` 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)

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`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, M = 2)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute historical decompositions
hd             = compute_historical_decompositions(posterior)
```

```
# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new(M = 2) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_historical_decompositions() -> hds
```

`compute_historical_decompositions.PosteriorBSVARSV`
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 'PosteriorBSVARSV'
compute_historical_decompositions(posterior, show_progress = TRUE)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVARSV` 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)

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`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar_sv$new(us_fiscal_lsuw, p = 1)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute historical decompositions
hd             = compute_historical_decompositions(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_sv$new(p = 1) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_historical_decompositions() -> hds
```

---

compute_historical_decompositions.PosteriorBSVART

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 'PosteriorBSVART'
compute_historical_decompositions(posterior, show_progress = TRUE)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVART` 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)

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, normalise, summary

Examples

```
specification  = specify_bsvar_t$new(diff(us_fiscal_lsuw))
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute historical decompositions
hd            = compute_historical_decompositions(posterior)

# workflow with the pipe |>
############################################################
diff(us_fiscal_lsuw) |>
  specify_bsvar_t$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_historical_decompositions() -> hd
```

`compute_impulse_responses`

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

```
compute_impulse_responses(posterior, horizon, standardise = FALSE)
```

Arguments

`posterior`	posterior estimation outcome obtained by running the `estimate` function. The interpretation depends on the normalisation of the shocks using function `normalise()`. Verify if the default settings are appropriate.
`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)

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, normalise, summary

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new(p = 1) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_impulse_responses(horizon = 8) -> ir
```

`compute_impulse_responses.PosteriorBSVAR`
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 'PosteriorBSVAR'
compute_impulse_responses(posterior, horizon, standardise = FALSE)
```

Arguments

posterior	posterior estimation outcome - an object of class PosteriorBSVAR 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)

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, normalise, summary

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new(p = 1) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_impulse_responses(horizon = 8) -> ir
```

`compute_impulse_responses.PosteriorBSVAREXH` *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 'PosteriorBSVAREXH'
compute_impulse_responses(posterior, horizon, standardise = FALSE)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVAREXH` 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)

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`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar_exh$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_exh$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_impulse_responses(horizon = 4) -> irfs
```

---

compute_impulse_responses.PosteriorBSVARHMSH

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 'PosteriorBSVARHMSH'
compute_impulse_responses(posterior, horizon, standardise = FALSE)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVARHMSH` 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)

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, normalise, summary

Examples

```
specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_impulse_responses(horizon = 4) -> irfs
```

---

compute_impulse_responses.PosteriorBSVARMIX

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 'PosteriorBSVARMIX'
compute_impulse_responses(posterior, horizon, standardise = FALSE)
```

Arguments

posterior	posterior estimation outcome - an object of class PosteriorBSVARMIX 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)

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, normalise, summary

Examples

```
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, p = 1, M = 2)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_mix$new(p = 1, M = 2) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_impulse_responses(horizon = 4) -> irfs
```

`compute_impulse_responses.PosteriorBSVARMSH` *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 'PosteriorBSVARMSH'
compute_impulse_responses(posterior, horizon, standardise = FALSE)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVARMSH` 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)

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`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, p = 1, M = 2)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new(p = 1, M = 2) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_impulse_responses(horizon = 4) -> irfs
```

`compute_impulse_responses.PosteriorBSVARSV`
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 'PosteriorBSVARSV'
compute_impulse_responses(posterior, horizon, standardise = FALSE)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVARSV` 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)

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, normalise, summary

Examples

```
specification  = specify_bsvar_sv$new(us_fiscal_lsuw, p = 1)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_sv$new(p = 1) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_impulse_responses(horizon = 4) -> irfs
```

---

compute_impulse_responses.PosteriorBSVART

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 'PosteriorBSVART'
compute_impulse_responses(posterior, horizon, standardise = FALSE)
```

Arguments

posterior	posterior estimation outcome - an object of class PosteriorBSVART 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)

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, normalise, summary

Examples

```
specification  = specify_bsvar_t$new(us_fiscal_lsuw, p = 1)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_t$new(p = 1) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_impulse_responses(horizon = 4) -> irfs
```

`compute_regime_probabilities` *Computes posterior draws of regime probabilities*

Description

Each of the draws from the posterior estimation of a model is transformed into a draw from the posterior distribution of the regime probabilities. These represent either the realisations of the regime indicators, when `type = "realized"`, filtered probabilities, when `type = "filtered"`, forecasted regime probabilities, when `type = "forecasted"`, or the smoothed probabilities, when `type = "smoothed"`, .

Usage

```
compute_regime_probabilities(
  posterior,
  type = c("realized", "filtered", "forecasted", "smoothed")
)
```

Arguments

`posterior`	posterior estimation outcome of regime-dependent heteroskedastic models - an object of either of the classes: PosteriorBSVARMSH, or PosteriorBSVARMIX obtained by running the `estimate` function.
`type`	one of the values `"realized"`, `"filtered"`, `"forecasted"`, or `"smoothed"` denoting the type of probabilities to be computed.

Value

An object of class PosteriorRegimePr. Realized regimes are returned as a compact zero-based `1xTxS` array (or `1xTxNxS` for HMSH); other types are returned as regime-probability arrays with first dimension `M`.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Song, Y., and Woźniak, T., (2021) Markov Switching. *Oxford Research Encyclopedia of Economics and Finance*, Oxford University Press, doi:10.1093/acrefore/9780190625979.013.174.

See Also

`estimate`, `summary`

Examples

```
specification  = specify_bsvar_msh$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute the posterior draws of realized regime indicators
regimes        = compute_regime_probabilities(posterior)

# compute the posterior draws of filtered probabilities
filtered       = compute_regime_probabilities(posterior, "filtered")

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new() |>
  estimate(S = 5) |>
  estimate(S = 5) -> posterior
regimes        = compute_regime_probabilities(posterior)
filtered       = compute_regime_probabilities(posterior, "filtered")
```

`compute_regime_probabilities.PosteriorBSVAREXH`
Computes posterior draws of regime probabilities

Description

Each of the draws from the posterior estimation of a model is transformed into a draw from the posterior distribution of the regime probabilities. These represent either the realisations of the regime indicators, when `type = "realized"`, filtered probabilities, when `type = "filtered"`, forecasted regime probabilities, when `type = "forecasted"`, or the smoothed probabilities, when `type = "smoothed"`, .

Usage

```
## S3 method for class 'PosteriorBSVAREXH'
compute_regime_probabilities(posterior, type = "realized")
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVAREXH` obtained by running the `estimate` function.
`type`	one of the values `"realized"`, `"filtered"`, `"forecasted"`, or `"smoothed"` denoting the type of probabilities to be computed.

Value

An object of class PosteriorRegimePr. Realized regimes are returned as a compact zero-based `1xTxS` array (or `1xTxNxS` for HMSH); other types are returned as regime-probability arrays with first dimension `M`.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Song, Y., and Woźniak, T., (2021) Markov Switching. *Oxford Research Encyclopedia of Economics and Finance*, Oxford University Press, doi:10.1093/acrefore/9780190625979.013.174.

See Also

estimate, summary

Examples

```
specification  = specify_bsvar_msh$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute the posterior draws of realized regime indicators
regimes        = compute_regime_probabilities(posterior)

# compute the posterior draws of filtered probabilities
filtered       = compute_regime_probabilities(posterior, "filtered")

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new() |>
  estimate(S = 5) |>
  estimate(S = 5) -> posterior
regimes        = compute_regime_probabilities(posterior)
filtered       = compute_regime_probabilities(posterior, "filtered")
```

---

compute_regime_probabilities.PosteriorBSVARHMSH

Computes posterior draws of regime probabilities

Description

Each of the draws from the posterior estimation of a model is transformed into a draw from the posterior distribution of the regime probabilities. These represent either the realisations of the regime indicators, when `type = "realized"`, filtered probabilities, when `type = "filtered"`, forecasted regime probabilities, when `type = "forecasted"`, or the smoothed probabilities, when `type = "smoothed"`, .

Usage

```
## S3 method for class 'PosteriorBSVARHMSH'
compute_regime_probabilities(
  posterior,
  type = c("realized", "filtered", "forecasted", "smoothed")
)
```

Arguments

posterior	posterior estimation outcome - an object of class PosteriorBSVARHMSH obtained by running the estimate function.
type	one of the values "realized", "filtered", "forecasted", or "smoothed" denoting the type of probabilities to be computed.

Value

An object of class PosteriorRegimePr. Realized regimes are returned as a compact zero-based 1xTxS array (or 1xTxNxS for HMSH); other types are returned as regime-probability arrays with first dimension M.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Song, Y., and Woźniak, T., (2021) Markov Switching. *Oxford Research Encyclopedia of Economics and Finance*, Oxford University Press, doi:10.1093/acrefore/9780190625979.013.174.

See Also

estimate, summary

Examples

```
specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute the posterior draws of realized regime indicators
regimes        = compute_regime_probabilities(posterior)

# compute the posterior draws of filtered probabilities
filtered       = compute_regime_probabilities(posterior, "filtered")

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_hmsh$new() |>
  estimate(S = 5) |>
```

```
  estimate(S = 5) -> posterior
regimes        = compute_regime_probabilities(posterior)
filtered       = compute_regime_probabilities(posterior, "filtered")
```

`compute_regime_probabilities.PosteriorBSVARMIX`
Computes posterior draws of regime probabilities

Description

Each of the draws from the posterior estimation of a model is transformed into a draw from the posterior distribution of the regime probabilities. These represent either the realisations of the regime indicators, when `type = "realized"`, filtered probabilities, when `type = "filtered"`, forecasted regime probabilities, when `type = "forecasted"`, or the smoothed probabilities, when `type = "smoothed"`, .

Usage

```
## S3 method for class 'PosteriorBSVARMIX'
compute_regime_probabilities(
  posterior,
  type = c("realized", "filtered", "forecasted", "smoothed")
)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVARMIX` obtained by running the `estimate` function.
`type`	one of the values `"realized"`, `"filtered"`, `"forecasted"`, or `"smoothed"` denoting the type of probabilities to be computed.

Value

An object of class PosteriorRegimePr. Realized regimes are returned as a compact zero-based `1xTxS` array (or `1xTxNxS` for HMSH); other types are returned as regime-probability arrays with first dimension `M`.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Song, Y., and Woźniak, T., (2021) Markov Switching. *Oxford Research Encyclopedia of Economics and Finance*, Oxford University Press, doi:10.1093/acrefore/9780190625979.013.174.

See Also

estimate, summary

Examples

```
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, p = 2, M = 2)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute the posterior draws of realized regime indicators
regimes        = compute_regime_probabilities(posterior)

# compute the posterior draws of filtered probabilities
filtered       = compute_regime_probabilities(posterior, "filtered")

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_mix$new(p = 1, M = 2) |>
  estimate(S = 5) |>
  estimate(S = 5) -> posterior
regimes        = compute_regime_probabilities(posterior)
filtered       = compute_regime_probabilities(posterior, "filtered")
```

---

compute_regime_probabilities.PosteriorBSVARMSH

Computes posterior draws of regime probabilities

Description

Each of the draws from the posterior estimation of a model is transformed into a draw from the posterior distribution of the regime probabilities. These represent either the realisations of the regime indicators, when `type = "realized"`, filtered probabilities, when `type = "filtered"`, forecasted regime probabilities, when `type = "forecasted"`, or the smoothed probabilities, when `type = "smoothed"`, .

Usage

```
## S3 method for class 'PosteriorBSVARMSH'
compute_regime_probabilities(
  posterior,
  type = c("realized", "filtered", "forecasted", "smoothed")
)
```

Arguments

posterior	posterior estimation outcome - an object of class PosteriorBSVARMSH obtained by running the estimate function.
type	one of the values "realized", "filtered", "forecasted", or "smoothed" denoting the type of probabilities to be computed.

Value

An object of class PosteriorRegimePr. Realized regimes are returned as a compact zero-based 1xTxS array (or 1xTxNxS for HMSH); other types are returned as regime-probability arrays with first dimension M.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Song, Y., and Woźniak, T., (2021) Markov Switching. *Oxford Research Encyclopedia of Economics and Finance*, Oxford University Press, doi:10.1093/acrefore/9780190625979.013.174.

See Also

estimate, summary

Examples

```
specification  = specify_bsvar_msh$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute the posterior draws of realized regime indicators
regimes        = compute_regime_probabilities(posterior)

# compute the posterior draws of filtered probabilities
filtered       = compute_regime_probabilities(posterior, "filtered")

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new() |>
  estimate(S = 5) |>
  estimate(S = 5) -> posterior
regimes        = compute_regime_probabilities(posterior)
filtered       = compute_regime_probabilities(posterior, "filtered")
```

`compute_structural_shocks` *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

```
compute_structural_shocks(posterior)
```

Arguments

`posterior`	posterior estimation outcome obtained by running the `estimate` function. The interpretation depends on the normalisation of the shocks using function `normalise()`. Verify if the default settings are appropriate.

Value

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

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

`estimate`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw, p = 1)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute structural shocks
shocks         = compute_structural_shocks(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new(p = 1) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_structural_shocks() -> ss
```

`compute_structural_shocks.PosteriorBSVAR` *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 'PosteriorBSVAR'
compute_structural_shocks(posterior)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVAR` 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)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

`estimate`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw, p = 1)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute structural shocks
shocks         = compute_structural_shocks(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new(p = 1) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_structural_shocks() -> ss
```

`compute_structural_shocks.PosteriorBSVAREXH` *Computes posterior draws of structural shocks*

Description

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

Usage

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

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVAREXH` 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)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

`estimate`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar_exh$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute structural shocks
shocks         = compute_structural_shocks(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_exh$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_structural_shocks() -> ss
```

`compute_structural_shocks.PosteriorBSVARHMSH` *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 'PosteriorBSVARHMSH'
compute_structural_shocks(posterior)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVARHMSH` 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)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

`estimate`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute structural shocks
shocks         = compute_structural_shocks(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_hmsh$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_structural_shocks() -> ss
```

`compute_structural_shocks.PosteriorBSVARMIX` *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 'PosteriorBSVARMIX'
compute_structural_shocks(posterior)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVARMIX` 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)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

`estimate`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, p = 1, M = 2)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute structural shocks
shocks         = compute_structural_shocks(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_mix$new(p = 1, M = 2) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_structural_shocks() -> ss
```

`compute_structural_shocks.PosteriorBSVARMSH` *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 'PosteriorBSVARMSH'
compute_structural_shocks(posterior)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVARMSH` 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)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

`estimate`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, p = 1, M = 2)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute structural shocks
shocks         = compute_structural_shocks(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new(p = 1, M = 2) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_structural_shocks() -> ss
```

`compute_structural_shocks.PosteriorBSVARSV` *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 'PosteriorBSVARSV'
compute_structural_shocks(posterior)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVARSV` 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)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

`estimate`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar_sv$new(us_fiscal_lsuw, p = 1)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute structural shocks
shocks         = compute_structural_shocks(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_sv$new(p = 1) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_structural_shocks() -> ss
```

`compute_structural_shocks.PosteriorBSVART` *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 'PosteriorBSVART'
compute_structural_shocks(posterior)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVART` 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)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

`estimate`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar_t$new(us_fiscal_lsuw, p = 1)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute structural shocks
shocks         = compute_structural_shocks(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_t$new(p = 1) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_structural_shocks() -> ss
```

`compute_variance_decompositions` *Computes posterior draws of the forecast error variance decomposition*

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 forecast error variance decomposition.

Usage

```
compute_variance_decompositions(posterior, horizon)
```

Arguments

`posterior`	posterior estimation outcome obtained by running the `estimate` function. The interpretation depends on the normalisation of the shocks using function `normalise()`. Verify if the default settings are appropriate.
`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.

See Also

`compute_impulse_responses`, `estimate`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw, p = 1)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

```
# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new(p = 1) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_variance_decompositions(horizon = 8) -> fevd
```

`compute_variance_decompositions.PosteriorBSVAR` *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 'PosteriorBSVAR'
compute_variance_decompositions(posterior, horizon)
```

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVAR` 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.

See Also

`compute_impulse_responses`, `estimate`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw, p = 1)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new(p = 1) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_variance_decompositions(horizon = 8) -> fevd
```

`compute_variance_decompositions.PosteriorBSVAREXH`

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. In this heteroskedastic model the forecast error variance decompositions are computed for the forecasts with the origin at the last observation in sample data and using the conditional variance forecasts.

Usage

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

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVAREXH` 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.

See Also

compute_impulse_responses, estimate, normalise, summary

Examples

```
specification  = specify_bsvar_exh$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_exh$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_variance_decompositions(horizon = 8) -> fevd
```

---

compute_variance_decompositions.PosteriorBSVARHMSH

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. In this heteroskedastic model the forecast error variance decompositions are computed for the forecasts with the origin at the last observation in sample data and using the conditional variance forecasts.

Usage

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

Arguments

posterior	posterior estimation outcome - an object of class PosteriorBSVARHMSH 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.

See Also

compute_impulse_responses, estimate, normalise, summary

Examples

```
specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_hmsh$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_variance_decompositions(horizon = 8) -> fevd
```

---

compute_variance_decompositions.PosteriorBSVARMIX

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. In this mixture model the forecast error variance decompositions are computed for the forecasts with the origin at the last observation in sample data and using the conditional variance forecasts.

Usage

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

Arguments

posterior	posterior estimation outcome - an object of class PosteriorBSVARMIX 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.

See Also

compute_impulse_responses, estimate, normalise, summary

Examples

```
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, p = 1, M = 2)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_mix$new(p = 1, M = 2) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_variance_decompositions(horizon = 8) -> fevd
```

`compute_variance_decompositions.PosteriorBSVARMSH` *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. In this heteroskedastic model the forecast error variance decompositions are computed for the forecasts with the origin at the last observation in sample data and using the conditional variance forecasts.

Usage

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

Arguments

`posterior`	posterior estimation outcome - an object of class `PosteriorBSVARMSH` 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.

See Also

`compute_impulse_responses`, `estimate`, `normalise`, `summary`

Examples

```
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, p = 1, M = 2)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new(p = 1, M = 2) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_variance_decompositions(horizon = 8) -> fevd
```

---

compute_variance_decompositions.PosteriorBSVARSV

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. In this heteroskedastic model the forecast error variance decompositions are computed for the forecasts with the origin at the last observation in sample data and using the conditional variance forecasts.

Usage

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

Arguments

posterior	posterior estimation outcome - an object of class PosteriorBSVARSV 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.

See Also

compute_impulse_responses, estimate, normalise, summary

Examples

```
specification  = specify_bsvar_sv$new(us_fiscal_lsuw, p = 1)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_sv$new(p = 1) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_variance_decompositions(horizon = 8) -> fevd
```

---

compute_variance_decompositions.PosteriorBSVART

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 'PosteriorBSVART'
compute_variance_decompositions(posterior, horizon)
```

Arguments

posterior	posterior estimation outcome - an object of class PosteriorBSVART 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.

See Also

compute_impulse_responses, estimate, normalise, summary

Examples

```
specification  = specify_bsvar_t$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_t$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_variance_decompositions(horizon = 8) -> fevd
```

---

estimate *Bayesian estimation of Structural Vector Autoregressions via Gibbs sampler*

Description

Estimates homo- or heteroskedastic SVAR models for packages **bsvars** and **bsvarSIGNs**. The packages apply the Gibbs sampler proposed by Waggoner & Zha (2003) for the structural matrix B and the equation-by-equation sampler by Chan, Koop, & Yu (2024) for the autoregressive slope parameters A. Additionally, the parameter matrices A and B follow a Minnesota prior and generalised-normal prior distributions respectively with the matrix-specific 3-level equation-specific local-global hierarchical prior for the shrinkage parameters. A variety of models for conditional variances are possible including versions of Stochastic Volatility and Markov-switching

heteroskedasticity. Non-normal specifications include finite and sparse normal mixture model for the structural shocks. The estimation algorithms for particular models are scrutinised in Lütkepohl, Shang, Uzeda, & Woźniak (2024) and Woźniak & Droumaguet (2024) and some other inferential and identification problems are considered in Lütkepohl & Woźniak (2020) and Song & Woźniak (2021). Models from package **bsvars** implement identification via exclusion restrictions, heteroskedasticity and non-normality. Models from package **bsvarSIGNs** implement identification via sign and narrative restrictions. See section **Details** and package **bsvarSIGNs** documentation for more information.

Usage

```
estimate(specification, S, thin = 1, show_progress = TRUE)
```

Arguments

`specification`	an object generated using one of the `specify_bsvar*` functions or an object generated using the function `estimate`. The latter type of input 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

Details

The homoskedastic SVAR 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.

The structural shocks, `U`, are temporally and contemporaneously independent and jointly distributed with zero mean and unit variances. The shock density can be either normal or Student-t with shock-specific degrees-of-freedom parameters that are estimated.

The various SVAR models estimated differ by the specification of structural shocks variances. Their specification depends on the `specify_bsvar*` function used. The different models include:

- homoskedastic model with unit variances
- heteroskedastic model with stationary Markov switching in the variances
- heteroskedastic model with Stochastic Volatility process for variances
- non-normal model with a finite mixture of normal components and component-specific variances

- heteroskedastic model with sparse Markov switching in the variances where the number of heteroskedastic components is estimated
- non-normal model with a sparse mixture of normal components and component-specific variances where the number of heteroskedastic components is estimated

Value

An object of class PosteriorBSVAR* 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 many arrays and vectors whose selection depends on the model specification. `last_draw` an object generated by one of the `specify_bsvar*` functions 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>

References

Chan, J.C.C., Koop, G, and Yu, X. (2024) Large Order-Invariant Bayesian VARs with Stochastic Volatility. *Journal of Business & Economic Statistics*, **42**, doi:10.1080/07350015.2023.2252039.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Song, Y., and Woźniak, T. (2021) Markov Switching Heteroskedasticity in Time Series Analysis. In: *Oxford Research Encyclopedia of Economics and Finance*. Oxford University Press, doi:10.1093/acrefore/9780190625979.013.174.

Waggoner, D.F., and Zha, T., (2003) A Gibbs sampler for structural vector autoregressions. *Journal of Economic Dynamics and Control*, **28**, 349–366, doi:10.1016/S01651889(02)001689.

Woźniak, T., and Droumaguet, M., (2024) Bayesian Assessment of Identifying Restrictions for Heteroskedastic Structural VARs.

See Also

`specify_bsvar`, `specify_bsvar_msh`, `specify_bsvar_mix`, `specify_bsvar_sv`, `normalise`

Examples

```
# simple workflow
############################################################
# upload data
data(us_fiscal_lsuw)
```

```
# specify the model and set seed
specification  = specify_bsvar$new(us_fiscal_lsuw, p = 4)
set.seed(123)

# run the burn-in
burn_in        = estimate(specification, 5)

# estimate the model
posterior      = estimate(burn_in, 10, thin = 2)

# workflow with the pipe |>
############################################################
set.seed(123)
us_fiscal_lsuw |>
  specify_bsvar$new(p = 1) |>
  estimate(S = 5) |>
  estimate(S = 10, thin = 2) -> posterior
```

`estimate.BSVAR` *Bayesian estimation of a homoskedastic Structural Vector Autoregression via Gibbs sampler*

Description

Estimates the homoskedastic SVAR using the Gibbs sampler proposed by Waggoner & Zha (2003) for the structural matrix B and the equation-by-equation sampler by Chan, Koop, & Yu (2024) for the autoregressive slope parameters A. 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 as in Lütkepohl, Shang, Uzeda, and Woźniak (2025). See section **Details** for the model equations.

Usage

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

Arguments

`specification`	an object of class BSVAR generated using the `specify_bsvar$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 homoskedastic SVAR 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.

Finally, the structural shocks, `U`, are temporally and contemporaneously independent and jointly distributed with zero mean and unit variances. The structural shocks can be either normally or Student-t distributed, where in the latter case the shock-specific degrees of freedom parameters are estimated.

Value

An object of class PosteriorBSVAR 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

`last_draw` an object of class BSVAR 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>`

References

Chan, J.C.C., Koop, G, and Yu, X. (2024) Large Order-Invariant Bayesian VARs with Stochastic Volatility. *Journal of Business & Economic Statistics*, **42**, doi:10.1080/07350015.2023.2252039.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2025) Partial identification of structural vector autoregressions with non-centred stochastic volatility. *Journal of Econometrics*, 1–18, doi:10.1016/j.jeconom.2025.106107.

Waggoner, D.F., and Zha, T., (2003) A Gibbs sampler for structural vector autoregressions. *Journal of Economic Dynamics and Control*, **28**, 349–366, doi:10.1016/S01651889(02)001689.

See Also

specify_bsvar, specify_posterior_bsvar, normalise

Examples

```
# simple workflow
############################################################
specification  = specify_bsvar$new(us_fiscal_lsuw, p = 4)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new(p = 1) |>
  estimate(S = 5) |>
  estimate(S = 5) -> post
```

`estimate.BSVAREXH` *Bayesian estimation of a Structural Vector Autoregression with exogenous heteroskedastic regime changes via Gibbs sampler*

Description

Estimates the SVAR with exogenous heteroskedastic regime changes with `M` regimes (MS(M)) proposed by Woźniak & Droumaguet (2022). Implements the Gibbs sampler proposed by Waggoner & Zha (2003) for the structural matrix B and the equation-by-equation sampler by Chan, Koop, & Yu (2024) for the autoregressive slope parameters A. 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 thanks to a hierarchical prior distribution. The MS model is estimated using the prior distributions and algorithms proposed by Woźniak & Droumaguet (2024), Lütkepohl & Woźniak (2020), and Song & Woźniak (2021). See section **Details** for the model equations.

Usage

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

Arguments

`specification`	an object of class BSVAREXH generated using the `specify_bsvar_exh$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 heteroskedastic SVAR 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.

Finally, the structural shocks, U, are temporally and contemporaneously independent and jointly distributed with zero mean. The structural shocks can be either normally or Student-t distributed, where in the latter case the shock-specific degrees of freedom parameters are estimated. The conditional variance of the `n`th shock at time `t` is given by:

$$Var_{t-1}[u_{n.t}] = s^2_{n.s_t}$$

where s_t is an exogenous process driving the time-variability of the regime-specific conditional variances of structural shocks $s^2_{n.s_t}$. In this model, the variances of each of the structural shocks sum to `M`. The model selection also with this respect is made using function `specify_bsvar_exh`.

Value

An object of class PosteriorBSVAREXH 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

sigma2 an `NxMxS` array with the posterior draws for the structural shocks conditional variances

xi a `1xTxS` array with the zero-based exogenous regime allocation.

sigma an `NxTxS` array with the posterior draws for the structural shocks conditional standard deviations' series over the sample period

`last_draw` an object of class BSVAREXH 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>

References

Chan, J.C.C., Koop, G, and Yu, X. (2024) Large Order-Invariant Bayesian VARs with Stochastic Volatility. *Journal of Business & Economic Statistics*, **42**, doi:10.1080/07350015.2023.2252039.

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Song, Y., and Woźniak, T., (2021) Markov Switching. *Oxford Research Encyclopedia of Economics and Finance*, Oxford University Press, doi:10.1093/acrefore/9780190625979.013.174.

Waggoner, D.F., and Zha, T., (2003) A Gibbs sampler for structural vector autoregressions. *Journal of Economic Dynamics and Control*, **28**, 349–366, doi:10.1016/S01651889(02)001689.

Woźniak, T., and Droumaguet, M., (2024) Bayesian Assessment of Identifying Restrictions for Heteroskedastic Structural VARs

See Also

`specify_bsvar_exh`, `specify_posterior_bsvar_exh`, `normalise`

Examples

```
# simple workflow
############################################################
spec  = specify_bsvar_exh$new(us_fiscal_lsuw)
burn  = estimate(spec, 5)
post  = estimate(burn, 5)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_exh$new() |>
  estimate(S = 5) |>
  estimate(S = 5) -> post
```

`estimate.BSVARHMSH` *Bayesian estimation of a Structural Vector Autoregression with Heterogeneous Markov-switching heteroskedasticity via Gibbs sampler*

Description

Estimates the SVAR with Heterogeneous Markov-switching heteroskedasticity with `M` regimes (HMS(M)) proposed by Shang & Woźniak (2025). Implements the Gibbs sampler proposed by Waggoner & Zha (2003) for the structural matrix B and the equation-by-equation sampler by Chan, Koop, & Yu (2024) for the autoregressive slope parameters A. 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 thanks to a hierarchical prior distribution. The MS model is estimated using the prior distributions and algorithms proposed Shang & Woźniak (2025), Lütkepohl & Woźniak (2020), and Song & Woźniak (2021). See section **Details** for the model equations.

Usage

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

Arguments

specification	an object of class BSVARHMSH generated using the specify_bsvar_hmsh$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 heteroskedastic SVAR 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.

Finally, the structural shocks, U, are temporally and contemporaneously independent and jointly distributed with zero mean. The structural shocks can be either normally or Student-t distributed, where in the latter case the shock-specific degrees of freedom parameters are estimated. The conditional variance of the nth shock at time t is given by:

$$Var_{t-1}[u_{n.t}] = s^2_{n.s_{n.t}}$$

where $s_{n.t}$ is an equation-specific Markov process driving the time-variability of the regime-specific conditional variances of the nth structural shock $s^2_{n.s_{n.t}}$. In this model, the variances of each of the structural shocks sum to M.

Each of the Markov processes $s_{n.t}$ is either:

- stationary, irreducible, and aperiodic which requires all regimes to have a positive number occurrences over the sample period, or
- sparse with potentially many regimes with zero occurrences over the sample period and in which the number of regimes is estimated.

Value

An object of class PosteriorBSVARHMSH 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

sigma2 an NxMxS array with the posterior draws for the structural shocks conditional variances

PR_TR an MxMxNxS array with the posterior draws for the transition matrix.

xi a 1xTxNxS array with zero-based posterior draws for the regime allocation.

pi_0 an MxNxS matrix with the posterior draws for the initial state probabilities

sigma an NxTxS array with the posterior draws for the structural shocks conditional standard deviations' series over the sample period

last_draw an object of class BSVARHMSH 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>

References

Shang, F., and Woźniak, T. (2025) Identification Verification using Structural Vector Autoregressions with Heterogeneous Markov Switching Heteroskedasticity.

Chan, J.C.C., Koop, G, and Yu, X. (2024) Large Order-Invariant Bayesian VARs with Stochastic Volatility. *Journal of Business & Economic Statistics*, **42**, doi:10.1080/07350015.2023.2252039.

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Song, Y., and Woźniak, T., (2021) Markov Switching. *Oxford Research Encyclopedia of Economics and Finance*, Oxford University Press, doi:10.1093/acrefore/9780190625979.013.174.

Waggoner, D.F., and Zha, T., (2003) A Gibbs sampler for structural vector autoregressions. *Journal of Economic Dynamics and Control*, **28**, 349–366, doi:10.1016/S01651889(02)001689.

Woźniak, T., and Droumaguet, M., (2024) Bayesian Assessment of Identifying Restrictions for Heteroskedastic Structural VARs

See Also

specify_bsvar_hmsh, specify_posterior_bsvar_hmsh, normalise

Examples

```
# simple workflow
############################################################
specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw, M = 2)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)
```

```
# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_hmsh$new(M = 2) |>
  estimate(S = 5) |>
  estimate(S = 5) -> posterior
```

`estimate.BSVARMIX` *Bayesian estimation of a Structural Vector Autoregression with shocks following a finite mixture of normal components via Gibbs sampler*

Description

Estimates the SVAR with non-normal residuals following a finite `M` mixture of normal distributions proposed by Woźniak & Droumaguet (2022). Implements the Gibbs sampler proposed by Waggoner & Zha (2003) for the structural matrix B and the equation-by-equation sampler by Chan, Koop, & Yu (2024) for the autoregressive slope parameters A. 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 thanks to a hierarchical prior distribution. The finite mixture of normals model is estimated using the prior distributions and algorithms proposed by Woźniak & Droumaguet (2024), Lütkepohl & Woźniak (2020), and Song & Woźniak (2021). See section **Details** for the model equations.

Usage

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

Arguments

`specification`	an object of class BSVARMIX generated using the `specify_bsvar_mix$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 heteroskedastic SVAR 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.

Finally, the structural shocks, U, are temporally and contemporaneously independent and finite-mixture of normals distributed with zero mean. Alternatively, the structural shocks can be Student-t distributed, where the shock-specific degrees of freedom parameters are estimated. The conditional variance of the nth shock at time t is given by:

$$Var_{t-1}[u_{n.t}] = s^2_{n.s_t}$$

where s_t is a the regime indicator of the regime-specific conditional variances of structural shocks $s^2_{n.s_t}$. In this model, the variances of each of the structural shocks sum to M.

The regime indicator s_t is either such that:

- the regime probabilities are non-zero which requires all regimes to have a positive number occurrences over the sample period, or
- sparse with potentially many regimes with zero occurrences over the sample period and in which the number of regimes is estimated.

These model selection also with this respect is made using function specify_bsvar_mix.

Value

An object of class PosteriorBSVARMIX 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

sigma2 an NxMxS array with the posterior draws for the structural shocks conditional variances

PR_TR an MxMxS array with the posterior draws for the transition matrix.

xi a 1xTxS array with zero-based posterior draws for the regime allocation.

pi_0 an MxS matrix with the posterior draws for the ergodic probabilities

sigma an NxTxS array with the posterior draws for the structural shocks conditional standard deviations' series over the sample period

last_draw an object of class BSVARMIX 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>

References

Chan, J.C.C., Koop, G, and Yu, X. (2024) Large Order-Invariant Bayesian VARs with Stochastic Volatility. *Journal of Business & Economic Statistics*, **42**, doi:10.1080/07350015.2023.2252039.

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Song, Y., and Woźniak, T., (2021) Markov Switching. *Oxford Research Encyclopedia of Economics and Finance*, Oxford University Press, doi:10.1093/acrefore/9780190625979.013.174.

Waggoner, D.F., and Zha, T., (2003) A Gibbs sampler for structural vector autoregressions. *Journal of Economic Dynamics and Control*, **28**, 349–366, doi:10.1016/S01651889(02)001689.

Woźniak, T., and Droumaguet, M., (2024) Bayesian Assessment of Identifying Restrictions for Heteroskedastic Structural VARs

See Also

`specify_bsvar_mix`, `specify_posterior_bsvar_mix`, `normalise`

Examples

```
# simple workflow
############################################################
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, M = 2)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_mix$new(M = 2) |>
  estimate(S = 5) |>
  estimate(S = 5) -> post
```

`estimate.BSVARMSH` *Bayesian estimation of a Structural Vector Autoregression with Markov-switching heteroskedasticity via Gibbs sampler*

Description

Estimates the SVAR with Markov-switching heteroskedasticity with `M` regimes (MS(M)) proposed by Woźniak & Droumaguet (2022). Implements the Gibbs sampler proposed by Waggoner & Zha (2003) for the structural matrix B and the equation-by-equation sampler by Chan, Koop, & Yu (2024) for the autoregressive slope parameters A. 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 thanks to a hierarchical prior distribution. The MS model is estimated using the prior distributions and algorithms proposed by Woźniak & Droumaguet (2024), Lütkepohl & Woźniak (2020), and Song & Woźniak (2021). See section **Details** for the model equations.

Usage

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

Arguments

specification	an object of class BSVARMSH generated using the specify_bsvar_msh$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 heteroskedastic SVAR 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.

Finally, the structural shocks, U, are temporally and contemporaneously independent and jointly distributed with zero mean. The structural shocks can be either normally or Student-t distributed, where in the latter case the shock-specific degrees of freedom parameters are estimated. The conditional variance of the nth shock at time t is given by:

$$Var_{t-1}[u_{n.t}] = s^2_{n.s_t}$$

where s_t is a Markov process driving the time-variability of the regime-specific conditional variances of structural shocks $s^2_{n.s_t}$. In this model, the variances of each of the structural shocks sum to M.

The Markov process s_t is either:

- stationary, irreducible, and aperiodic which requires all regimes to have a positive number occurrences over the sample period, or
- sparse with potentially many regimes with zero occurrences over the sample period and in which the number of regimes is estimated.

These model selection also with this respect is made using function specify_bsvar_msh.

Value

An object of class PosteriorBSVARMSH 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

sigma2 an `NxMxS` array with the posterior draws for the structural shocks conditional variances

PR_TR an `MxMxS` array with the posterior draws for the transition matrix.

xi a `1xTxS` array with zero-based posterior draws for the regime allocation.

pi_0 an `MxS` matrix with the posterior draws for the initial state probabilities

sigma an `NxTxS` array with the posterior draws for the structural shocks conditional standard deviations' series over the sample period

`last_draw` an object of class BSVARMSH 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>

References

Chan, J.C.C., Koop, G, and Yu, X. (2024) Large Order-Invariant Bayesian VARs with Stochastic Volatility. *Journal of Business & Economic Statistics*, **42**, doi:10.1080/07350015.2023.2252039.

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Song, Y., and Woźniak, T., (2021) Markov Switching. *Oxford Research Encyclopedia of Economics and Finance*, Oxford University Press, doi:10.1093/acrefore/9780190625979.013.174.

Waggoner, D.F., and Zha, T., (2003) A Gibbs sampler for structural vector autoregressions. *Journal of Economic Dynamics and Control*, **28**, 349–366, doi:10.1016/S01651889(02)001689.

Woźniak, T., and Droumaguet, M., (2024) Bayesian Assessment of Identifying Restrictions for Heteroskedastic Structural VARs

See Also

`specify_bsvar_msh`, `specify_posterior_bsvar_msh`, `normalise`

Examples

```
# simple workflow
############################################################
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, M = 2)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new(M = 2) |>
  estimate(S = 5) |>
  estimate(S = 5) -> post
```

`estimate.BSVARSV` *Bayesian estimation of a Structural Vector Autoregression with Stochastic Volatility heteroskedasticity via Gibbs sampler*

Description

Estimates the SVAR with Stochastic Volatility (SV) heteroskedasticity proposed by Lütkepohl, Shang, Uzeda, and Woźniak (2025). Implements the Gibbs sampler proposed by Waggoner & Zha (2003) for the structural matrix B and the equation-by-equation sampler by Chan, Koop, & Yu (2024) for the autoregressive slope parameters A. 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 thanks to a hierarchical prior distribution. The SV model is estimated using a range of techniques including: simulation smoother, auxiliary mixture, ancillarity-sufficiency interweaving strategy, and generalised inverse Gaussian distribution summarised by Kastner & Frühwirth-Schnatter (2014). See section **Details** for the model equations.

Usage

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

Arguments

`specification`	an object of class BSVARSV generated using the `specify_bsvar_sv$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 heteroskedastic SVAR 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. Finally, the structural shocks, U, are temporally and contemporaneously independent and jointly distributed with zero mean. The structural shocks can be either normally or Student-t distributed, where in the latter case the shock-specific degrees of freedom parameters are estimated.

Two alternative specifications of the conditional variance of the nth shock at time t can be estimated: non-centred Stochastic Volatility by Lütkepohl, Shang, Uzeda, and Woźniak (2022) or centred Stochastic Volatility by Chan, Koop, & Yu (2021).

The non-centred Stochastic Volatility by Lütkepohl, Shang, Uzeda, and Woźniak (2022) is selected by setting argument centred_sv of function specify_bsvar_sv$new() to value FALSE. It has the conditional variances given by:

$$Var_{t-1}[u_{n.t}] = exp(w_n h_{n.t})$$

where w_n is the estimated conditional standard deviation of the log-conditional variance and the log-volatility process $h_{n.t}$ follows an autoregressive process:

$$h_{n.t} = g_n h_{n.t-1} + v_{n.t}$$

where $h_{n.0} = 0$, g_n is an autoregressive parameter and $v_{n.t}$ is a standard normal error term.

The centred Stochastic Volatility by Chan, Koop, & Yu (2021) is selected by setting argument centred_sv of function specify_bsvar_sv$new() to value TRUE. Its conditional variances are given by:

$$Var_{t-1}[u_{n.t}] = exp(h_{n.t})$$

where the log-conditional variances $h_{n.t}$ follow an autoregressive process:

$$h_{n.t} = g_n h_{n.t-1} + v_{n.t}$$

where $h_{n.0} = 0$, g_n is an autoregressive parameter and $v_{n.t}$ is a zero-mean normal error term with variance $s^2_{v.n}$.

Value

An object of class PosteriorBSVARSV 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

h an NxTxS array with the posterior draws of the log-volatility processes

rho an NxS matrix with the posterior draws of SV autoregressive parameters

omega an NxS matrix with the posterior draws of SV process conditional standard deviations

S an NxTxS array with the posterior draws of the auxiliary mixture component indicators

sigma2_omega an NxS matrix with the posterior draws of the variances of the zero-mean normal prior for omega

s_ an S-vector with the posterior draws of the scale of the gamma prior of the hierarchical prior for sigma2_omega

last_draw an object of class BSVARSV 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>

References

Chan, J.C.C., Koop, G, and Yu, X. (2024) Large Order-Invariant Bayesian VARs with Stochastic Volatility. *Journal of Business & Economic Statistics*, **42**, doi:10.1080/07350015.2023.2252039.

Kastner, G. and Frühwirth-Schnatter, S. (2014) Ancillarity-Sufficiency Interweaving Strategy (ASIS) for Boosting MCMC Estimation of Stochastic Volatility Models. *Computational Statistics & Data Analysis*, **76**, 408–423, doi:10.1016/j.csda.2013.01.002.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2025) Partial identification of structural vector autoregressions with non-centred stochastic volatility. *Journal of Econometrics*, 1–18, doi:10.1016/j.jeconom.2025.106107.

Waggoner, D.F., and Zha, T., (2003) A Gibbs sampler for structural vector autoregressions. *Journal of Economic Dynamics and Control*, **28**, 349–366, doi:10.1016/S01651889(02)001689.

See Also

specify_bsvar_sv, specify_posterior_bsvar_sv, normalise

Examples

```
# simple workflow
############################################################
specification  = specify_bsvar_sv$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

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

`estimate.BSVART` *Bayesian estimation of a homoskedastic Structural Vector Autoregression with t-distributed structural shocks via Gibbs sampler*

Description

Estimates the homoskedastic SVAR using the Gibbs sampler proposed by Waggoner & Zha (2003) for the structural matrix B and the equation-by-equation sampler by Chan, Koop, & Yu (2024) for the autoregressive slope parameters A. The Robust Adaptive Metropolis algorithm by Vihola (2012) is used to the `df` parameter of the Student-t distribution. 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 as in Lütkepohl, Shang, Uzeda, and Woźniak (2025). See section **Details** for the model equations.

Usage

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

Arguments

`specification`	an object of class `BSVART` generated using the `specify_bsvar_t$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 homoskedastic SVAR model with t-distributed structural shocks 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.

Finally, the structural shocks, `U`, are temporally and contemporaneously independent and jointly Student-t distributed with zero mean, unit variances, and estimated equation-specific degrees-of-freedom parameter.

Value

An object of class PosteriorBSVART 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

df an NxS matrix with the posterior draws for the equation-specific degrees-of-freedom parameter of the Student-t distribution

lambda an NxTxS array with the posterior draws for the latent variable

last_draw an object of class BSVART 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>

References

Chan, J.C.C., Koop, G, and Yu, X. (2024) Large Order-Invariant Bayesian VARs with Stochastic Volatility. *Journal of Business & Economic Statistics*, **42**, doi:10.1080/07350015.2023.2252039.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2025) Partial identification of structural vector autoregressions with non-centred stochastic volatility. *Journal of Econometrics*, 1–18, doi:10.1016/j.jeconom.2025.106107.

Waggoner, D.F., and Zha, T., (2003) A Gibbs sampler for structural vector autoregressions. *Journal of Economic Dynamics and Control*, **28**, 349–366, doi:10.1016/S01651889(02)001689.

Vihola, M. (2012) Robust adaptive Metropolis algorithm with coerced acceptance rate. *Statistics & Computing*, 22, 997–1008, doi:10.1007/s1122201192695.

See Also

specify_bsvar_t, specify_posterior_bsvar_t, normalise

Examples

```
# simple workflow
############################################################
specification  = specify_bsvar_t$new(us_fiscal_lsuw, p = 4)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

```
specify_bsvar_t$new(p = 4) |>
estimate(S = 5) |>
estimate(S = 5) -> posterior
```

`estimate.PosteriorBSVAR` *Bayesian estimation of a homoskedastic Structural Vector Autoregression via Gibbs sampler*

Description

Estimates the homoskedastic SVAR using the Gibbs sampler proposed by Waggoner & Zha (2003) for the structural matrix B and the equation-by-equation sampler by Chan, Koop, & Yu (2024) for the autoregressive slope parameters A. 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 as in Lütkepohl, Shang, Uzeda, and Woźniak (2025). See section **Details** for the model equations.

Usage

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

Arguments

`specification`	an object of class PosteriorBSVAR generated using the `estimate.BSVAR()` 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

Details

The homoskedastic SVAR 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.

Finally, the structural shocks, `U`, are temporally and contemporaneously independent and jointly distributed with zero mean and unit variances. The structural shocks can be either normally or Student-t distributed, where in the latter case the shock-specific degrees of freedom parameters are estimated.

Value

An object of class PosteriorBSVAR 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

`last_draw` an object of class BSVAR 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>

References

Chan, J.C.C., Koop, G, and Yu, X. (2024) Large Order-Invariant Bayesian VARs with Stochastic Volatility. *Journal of Business & Economic Statistics*, **42**, doi:10.1080/07350015.2023.2252039.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2025) Partial identification of structural vector autoregressions with non-centred stochastic volatility. *Journal of Econometrics*, 1–18, doi:10.1016/j.jeconom.2025.106107.

Waggoner, D.F., and Zha, T., (2003) A Gibbs sampler for structural vector autoregressions. *Journal of Economic Dynamics and Control*, **28**, 349–366, doi:10.1016/S01651889(02)001689.

See Also

`specify_bsvar`, `specify_posterior_bsvar`, `normalise`

Examples

```
# simple workflow
############################################################
specification  = specify_bsvar$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

```
us_fiscal_lsuw |>
  specify_bsvar$new(p = 1) |>
  estimate(S = 5) |>
  estimate(S = 5) -> post
```

`estimate.PosteriorBSVAREXH` *Bayesian estimation of a Structural Vector Autoregression with exogenous heteroskedastic regime changes via Gibbs sampler*

Description

Estimates the SVAR with exogenous heteroskedastic regime changes with `M` regimes (MS(M)) proposed by Woźniak & Droumaguet (2022). Implements the Gibbs sampler proposed by Waggoner & Zha (2003) for the structural matrix B and the equation-by-equation sampler by Chan, Koop, & Yu (2024) for the autoregressive slope parameters A. 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 thanks to a hierarchical prior distribution. The MS model is estimated using the prior distributions and algorithms proposed by Woźniak & Droumaguet (2024), Lütkepohl & Woźniak (2020), and Song & Woźniak (2021). See section **Details** for the model equations.

Usage

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

Arguments

`specification`	an object of class PosteriorBSVAREXH generated using the `estimate()` 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

Details

The heteroskedastic SVAR 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.

Finally, the structural shocks, U, are temporally and contemporaneously independent and jointly distributed with zero mean. The structural shocks can be either normally or Student-t distributed, where in the latter case the shock-specific degrees of freedom parameters are estimated. The conditional variance of the nth shock at time t is given by:

$$Var_{t-1}[u_{n.t}] = s^2_{n.s_t}$$

where s_t is an exogenous process driving the time-variability of the regime-specific conditional variances of structural shocks $s^2_{n.s_t}$. In this model, the variances of each of the structural shocks sum to M. The model selection also with this respect is made using function specify_bsvar_exh.

Value

An object of class PosteriorBSVAREXH 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

sigma2 an NxMxS array with the posterior draws for the structural shocks conditional variances

xi a 1xTxS array with the zero-based exogenous regime allocation.

sigma an NxTxS array with the posterior draws for the structural shocks conditional standard deviations' series over the sample period

last_draw an object of class BSVAREXH 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>

References

Chan, J.C.C., Koop, G, and Yu, X. (2024) Large Order-Invariant Bayesian VARs with Stochastic Volatility. *Journal of Business & Economic Statistics*, **42**, doi:10.1080/07350015.2023.2252039.

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Song, Y., and Woźniak, T., (2021) Markov Switching. *Oxford Research Encyclopedia of Economics and Finance*, Oxford University Press, doi:10.1093/acrefore/9780190625979.013.174.

Waggoner, D.F., and Zha, T., (2003) A Gibbs sampler for structural vector autoregressions. *Journal of Economic Dynamics and Control*, **28**, 349–366, doi:10.1016/S01651889(02)001689.

Woźniak, T., and Droumaguet, M., (2024) Bayesian Assessment of Identifying Restrictions for Heteroskedastic Structural VARs

See Also

specify_bsvar_exh, specify_posterior_bsvar_exh, normalise

Examples

```
# simple workflow
############################################################
spec  = specify_bsvar_exh$new(us_fiscal_lsuw)
burn  = estimate(spec, 5)
post  = estimate(burn, 5)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_exh$new() |>
  estimate(S = 5) |>
  estimate(S = 5)  -> post
```

`estimate.PosteriorBSVARHMSH` *Bayesian estimation of a Structural Vector Autoregression with Heterogeneous Markov-switching heteroskedasticity via Gibbs sampler*

Description

Estimates the SVAR with Heterogeneous Markov-switching heteroskedasticity with `M` regimes (HMS(M)) proposed by Shang & Woźniak (2025). Implements the Gibbs sampler proposed by Waggoner & Zha (2003) for the structural matrix B and the equation-by-equation sampler by Chan, Koop, & Yu (2024) for the autoregressive slope parameters A. 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 thanks to a hierarchical prior distribution. The MS model is estimated using the prior distributions and algorithms proposed Shang & Woźniak (2025), Lütkepohl & Woźniak (2020), and Song & Woźniak (2021). See section **Details** for the model equations.

Usage

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

Arguments

specification	an object of class PosteriorBSVARHMSH generated using the estimate() 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

Details

The heteroskedastic SVAR 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.

Finally, the structural shocks, U, are temporally and contemporaneously independent and jointly distributed with zero mean. The structural shocks can be either normally or Student-t distributed, where in the latter case the shock-specific degrees of freedom parameters are estimated. The conditional variance of the nth shock at time t is given by:

$$Var_{t-1}[u_{n.t}] = s^2_{n.s_{n.t}}$$

where $s_{n.t}$ is an equation-specific Markov process driving the time-variability of the regime-specific conditional variances of the nth structural shock $s^2_{n.s_{n.t}}$. In this model, the variances of each of the structural shocks sum to M.

Each of the Markov processes $s_{n.t}$ is either:

- stationary, irreducible, and aperiodic which requires all regimes to have a positive number occurrences over the sample period, or
- sparse with potentially many regimes with zero occurrences over the sample period and in which the number of regimes is estimated.

Value

An object of class PosteriorBSVARHMSH 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

sigma2 an NxMxS array with the posterior draws for the structural shocks conditional variances

PR_TR an MxMxNxS array with the posterior draws for the transition matrix.

xi a 1xTxNxS array with zero-based posterior draws for the regime allocation.

pi_0 an MxNxS matrix with the posterior draws for the initial state probabilities

sigma an NxTxS array with the posterior draws for the structural shocks conditional standard deviations' series over the sample period

last_draw an object of class BSVARHMSH 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>

References

Shang, F., and Woźniak, T. (2025) Identification Verification using Structural Vector Autoregressions with Heterogeneous Markov Switching Heteroskedasticity.

Chan, J.C.C., Koop, G, and Yu, X. (2024) Large Order-Invariant Bayesian VARs with Stochastic Volatility. *Journal of Business & Economic Statistics*, **42**, doi:10.1080/07350015.2023.2252039.

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Song, Y., and Woźniak, T., (2021) Markov Switching. *Oxford Research Encyclopedia of Economics and Finance*, Oxford University Press, doi:10.1093/acrefore/9780190625979.013.174.

Waggoner, D.F., and Zha, T., (2003) A Gibbs sampler for structural vector autoregressions. *Journal of Economic Dynamics and Control*, **28**, 349–366, doi:10.1016/S01651889(02)001689.

Woźniak, T., and Droumaguet, M., (2024) Bayesian Assessment of Identifying Restrictions for Heteroskedastic Structural VARs

See Also

specify_bsvar_hmsh, specify_posterior_bsvar_hmsh, normalise

Examples

```
# simple workflow
############################################################
specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw, M = 2)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

```
specify_bsvar_hmsh$new(M = 2) |>
estimate(S = 5) |>
estimate(S = 5) -> posterior
```

`estimate.PosteriorBSVARMIX` *Bayesian estimation of a Structural Vector Autoregression with shocks following a finite mixture of normal components via Gibbs sampler*

Description

Estimates the SVAR with non-normal residuals following a finite `M` mixture of normal distributions proposed by Woźniak & Droumaguet (2022). Implements the Gibbs sampler proposed by Waggoner & Zha (2003) for the structural matrix B and the equation-by-equation sampler by Chan, Koop, & Yu (2024) for the autoregressive slope parameters A. 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 thanks to a hierarchical prior distribution. The finite mixture of normals model is estimated using the prior distributions and algorithms proposed by Woźniak & Droumaguet (2024), Lütkepohl & Woźniak (2020), and Song & Woźniak (2021). See section **Details** for the model equations.

Usage

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

Arguments

`specification`	an object of class PosteriorBSVARMIX generated using the `estimate.BSVAR()` 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

Details

The heteroskedastic SVAR 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.

Finally, the structural shocks, U, are temporally and contemporaneously independent and finite-mixture of normals distributed with zero mean. Alternatively, the structural shocks can be Student-t distributed, where the shock-specific degrees of freedom parameters are estimated. The conditional variance of the nth shock at time t is given by:

$$Var_{t-1}[u_{n.t}] = s^2_{n.s_t}$$

where s_t is a the regime indicator of the regime-specific conditional variances of structural shocks $s^2_{n.s_t}$. In this model, the variances of each of the structural shocks sum to M.

The regime indicator s_t is either such that:

- the regime probabilities are non-zero which requires all regimes to have a positive number occurrences over the sample period, or
- sparse with potentially many regimes with zero occurrences over the sample period and in which the number of regimes is estimated.

These model selection also with this respect is made using function specify_bsvar_mix.

Value

An object of class PosteriorBSVARMIX 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

sigma2 an NxMxS array with the posterior draws for the structural shocks conditional variances

PR_TR an MxMxS array with the posterior draws for the transition matrix.

xi a 1xTxS array with zero-based posterior draws for the regime allocation.

pi_0 an MxS matrix with the posterior draws for the ergodic probabilities

sigma an NxTxS array with the posterior draws for the structural shocks conditional standard deviations' series over the sample period

last_draw an object of class BSVARMIX 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>

References

Chan, J.C.C., Koop, G, and Yu, X. (2024) Large Order-Invariant Bayesian VARs with Stochastic Volatility. *Journal of Business & Economic Statistics*, **42**, doi:10.1080/07350015.2023.2252039.

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Song, Y., and Woźniak, T., (2021) Markov Switching. *Oxford Research Encyclopedia of Economics and Finance*, Oxford University Press, doi:10.1093/acrefore/9780190625979.013.174.

Waggoner, D.F., and Zha, T., (2003) A Gibbs sampler for structural vector autoregressions. *Journal of Economic Dynamics and Control*, **28**, 349–366, doi:10.1016/S01651889(02)001689.

Woźniak, T., and Droumaguet, M., (2024) Bayesian Assessment of Identifying Restrictions for Heteroskedastic Structural VARs

See Also

`specify_bsvar_mix`, `specify_posterior_bsvar_mix`, `normalise`

Examples

```
# simple workflow
############################################################
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, M = 2)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_mix$new(M = 2) |>
  estimate(S = 5) |>
  estimate(S = 5) -> post
```

`estimate.PosteriorBSVARMSH`

Bayesian estimation of a Structural Vector Autoregression with Markov-switching heteroskedasticity via Gibbs sampler

---

Description

Estimates the SVAR with Markov-switching heteroskedasticity with `M` regimes (MS(M)) proposed by Woźniak & Droumaguet (2022). Implements the Gibbs sampler proposed by Waggoner & Zha (2003) for the structural matrix B and the equation-by-equation sampler by Chan, Koop, & Yu (2024) for the autoregressive slope parameters A. 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 thanks to a hierarchical prior distribution. The MS

model is estimated using the prior distributions and algorithms proposed by Woźniak & Droumaguet (2024), Lütkepohl & Woźniak (2020), and Song & Woźniak (2021). See section **Details** for the model equations.

Usage

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

Arguments

`specification`	an object of class PosteriorBSVARMSH generated using the `estimate.BSVAR()` 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

Details

The heteroskedastic SVAR 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.

Finally, the structural shocks, U, are temporally and contemporaneously independent and jointly distributed with zero mean. The structural shocks can be either normally or Student-t distributed, where in the latter case the shock-specific degrees of freedom parameters are estimated. The conditional variance of the `n`th shock at time `t` is given by:

$$Var_{t-1}[u_{n.t}] = s^2_{n.s_t}$$

where s_t is a Markov process driving the time-variability of the regime-specific conditional variances of structural shocks $s^2_{n.s_t}$. In this model, the variances of each of the structural shocks sum to `M`.

The Markov process s_t is either:

- stationary, irreducible, and aperiodic which requires all regimes to have a positive number occurrences over the sample period, or
- sparse with potentially many regimes with zero occurrences over the sample period and in which the number of regimes is estimated.

These model selection also with this respect is made using function `specify_bsvar_msh`.

Value

An object of class PosteriorBSVARMSH 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

sigma2 an `NxMxS` array with the posterior draws for the structural shocks conditional variances

PR_TR an `MxMxS` array with the posterior draws for the transition matrix.

xi a `1xTxS` array with zero-based posterior draws for the regime allocation.

pi_0 an `MxS` matrix with the posterior draws for the initial state probabilities

sigma an `NxTxS` array with the posterior draws for the structural shocks conditional standard deviations' series over the sample period

`last_draw` an object of class BSVARMSH 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>

References

Chan, J.C.C., Koop, G, and Yu, X. (2024) Large Order-Invariant Bayesian VARs with Stochastic Volatility. *Journal of Business & Economic Statistics*, **42**, doi:10.1080/07350015.2023.2252039.

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Song, Y., and Woźniak, T., (2021) Markov Switching. *Oxford Research Encyclopedia of Economics and Finance*, Oxford University Press, doi:10.1093/acrefore/9780190625979.013.174.

Waggoner, D.F., and Zha, T., (2003) A Gibbs sampler for structural vector autoregressions. *Journal of Economic Dynamics and Control*, **28**, 349–366, doi:10.1016/S01651889(02)001689.

Woźniak, T., and Droumaguet, M., (2024) Bayesian Assessment of Identifying Restrictions for Heteroskedastic Structural VARs

See Also

`specify_bsvar_msh`, `specify_posterior_bsvar_msh`, `normalise`

Examples

```
# simple workflow
############################################################
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, M = 2)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new(M = 2) |>
  estimate(S = 5) |>
  estimate(S = 5) -> post
```

`estimate.PosteriorBSVARSV`

Bayesian estimation of a Structural Vector Autoregression with Stochastic Volatility heteroskedasticity via Gibbs sampler

---

Description

Estimates the SVAR with Stochastic Volatility (SV) heteroskedasticity proposed by Lütkepohl, Shang, Uzeda, and Woźniak (2025). Implements the Gibbs sampler proposed by Waggoner & Zha (2003) for the structural matrix B and the equation-by-equation sampler by Chan, Koop, & Yu (2024) for the autoregressive slope parameters A. 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 thanks to a hierarchical prior distribution. The SV model is estimated using a range of techniques including: simulation smoother, auxiliary mixture, ancillarity-sufficiency interweaving strategy, and generalised inverse Gaussian distribution summarised by Kastner & Frühwirth-Schnatter (2014). See section **Details** for the model equations.

Usage

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

Arguments

`specification`	an object of class PosteriorBSVARSV generated using the `estimate.BSVAR()` 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

Details

The heteroskedastic SVAR 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. Finally, the structural shocks, U, are temporally and contemporaneously independent and jointly distributed with zero mean. The structural shocks can be either normally or Student-t distributed, where in the latter case the shock-specific degrees of freedom parameters are estimated.

Two alternative specifications of the conditional variance of the nth shock at time t can be estimated: non-centred Stochastic Volatility by Lütkepohl, Shang, Uzeda, and Woźniak (2022) or centred Stochastic Volatility by Chan, Koop, & Yu (2021).

The non-centred Stochastic Volatility by Lütkepohl, Shang, Uzeda, and Woźniak (2022) is selected by setting argument centred_sv of function specify_bsvar_sv$new() to value FALSE. It has the conditional variances given by:

$$Var_{t-1}[u_{n.t}] = exp(w_n h_{n.t})$$

where w_n is the estimated conditional standard deviation of the log-conditional variance and the log-volatility process $h_{n.t}$ follows an autoregressive process:

$$h_{n.t} = g_n h_{n.t-1} + v_{n.t}$$

where $h_{n.0} = 0$, g_n is an autoregressive parameter and $v_{n.t}$ is a standard normal error term.

The centred Stochastic Volatility by Chan, Koop, & Yu (2021) is selected by setting argument centred_sv of function specify_bsvar_sv$new() to value TRUE. Its conditional variances are given by:

$$Var_{t-1}[u_{n.t}] = exp(h_{n.t})$$

where the log-conditional variances $h_{n.t}$ follow an autoregressive process:

$$h_{n.t} = g_n h_{n.t-1} + v_{n.t}$$

where $h_{n.0} = 0$, g_n is an autoregressive parameter and $v_{n.t}$ is a zero-mean normal error term with variance $s^2_{v.n}$.

Value

An object of class PosteriorBSVARSV 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

h an NxTxS array with the posterior draws of the log-volatility processes

rho an NxS matrix with the posterior draws of SV autoregressive parameters

omega an NxS matrix with the posterior draws of SV process conditional standard deviations

S an NxTxS array with the posterior draws of the auxiliary mixture component indicators

sigma2_omega an NxS matrix with the posterior draws of the variances of the zero-mean normal prior for omega

s_ an S-vector with the posterior draws of the scale of the gamma prior of the hierarchical prior for sigma2_omega

last_draw an object of class BSVARSV 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>

References

Chan, J.C.C., Koop, G, and Yu, X. (2024) Large Order-Invariant Bayesian VARs with Stochastic Volatility. *Journal of Business & Economic Statistics*, **42**, doi:10.1080/07350015.2023.2252039.

Kastner, G. and Frühwirth-Schnatter, S. (2014) Ancillarity-Sufficiency Interweaving Strategy (ASIS) for Boosting MCMC Estimation of Stochastic Volatility Models. *Computational Statistics & Data Analysis*, **76**, 408–423, doi:10.1016/j.csda.2013.01.002.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2025) Partial identification of structural vector autoregressions with non-centred stochastic volatility. *Journal of Econometrics*, 1–18, doi:10.1016/j.jeconom.2025.106107.

Waggoner, D.F., and Zha, T., (2003) A Gibbs sampler for structural vector autoregressions. *Journal of Economic Dynamics and Control*, **28**, 349–366, doi:10.1016/S01651889(02)001689.

See Also

specify_bsvar_sv, specify_posterior_bsvar_sv, normalise

Examples

```
# simple workflow
############################################################
specification  = specify_bsvar_sv$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

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

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

`estimate.PosteriorBSVART` *Bayesian estimation of a homoskedastic Structural Vector Autoregression with t-distributed structural shocks via Gibbs sampler*

Description

Estimates the homoskedastic SVAR using the Gibbs sampler proposed by Waggoner & Zha (2003) for the structural matrix B and the equation-by-equation sampler by Chan, Koop, & Yu (2024) for the autoregressive slope parameters A. The Robust Adaptive Metropolis algorithm by Vihola (2012) is used to the `df` parameter of the Student-t distribution. 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 as in Lütkepohl, Shang, Uzeda, and Woźniak (2025). See section **Details** for the model equations.

Usage

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

Arguments

`specification`	an object of class PosteriorBSVART generated using the `estimate.BSVART()` 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

Details

The homoskedastic SVAR model with t-distributed structural shocks 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.

Finally, the structural shocks, U, are temporally and contemporaneously independent and jointly Student-t distributed with zero mean, unit variances, and estimated equation-specific degrees-of-freedom parameter.

Value

An object of class PosteriorBSVART 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

df an NxS matrix with the posterior draws for the equation-specific degrees-of-freedom parameter of the Student-t distribution

lambda an NxTxS array with the posterior draws for the latent variable

last_draw an object of class BSVART 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>

References

Chan, J.C.C., Koop, G, and Yu, X. (2024) Large Order-Invariant Bayesian VARs with Stochastic Volatility. *Journal of Business & Economic Statistics*, **42**, doi:10.1080/07350015.2023.2252039.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2025) Partial identification of structural vector autoregressions with non-centred stochastic volatility. *Journal of Econometrics*, 1–18, doi:10.1016/j.jeconom.2025.106107.

Waggoner, D.F., and Zha, T., (2003) A Gibbs sampler for structural vector autoregressions. *Journal of Economic Dynamics and Control*, **28**, 349–366, doi:10.1016/S01651889(02)001689.

Vihola, M. (2012) Robust adaptive Metropolis algorithm with coerced acceptance rate. *Statistics & Computing*, 22, 997–1008, doi:10.1007/s1122201192695.

See Also

specify_bsvar_t, specify_posterior_bsvar_t, normalise

Examples

```
# simple workflow
############################################################
specification  = specify_bsvar_t$new(us_fiscal_lsuw, p = 4)
burn_in        = estimate(specification, 5)
```

```
posterior      = estimate(burn_in, 5)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_t$new(p = 4) |>
  estimate(S = 5) |>
  estimate(S = 5) -> posterior
```

`forecast.PosteriorBSVAR` *Forecasting using Bayesian Structural Vector Autoregression*

Description

Samples from the joint predictive density of all of the dependent variables for models at forecast horizons from 1 to `horizon` specified as an argument of the function.

Usage

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

Arguments

`object`	posterior estimation outcome - an object of class `PosteriorBSVAR` 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 for heteroskedastic models the draws from the predictive density of structural shocks conditional standard deviations and data. The output elements include:

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

Y an NxT matrix with the data on dependent variables

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

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

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)
predictive     = forecast(posterior, 4)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  forecast(horizon = 4) -> predictive

# conditional forecasting using a model with exogenous variables
############################################################
specification  = specify_bsvar$new(us_fiscal_lsuw, exogenous = us_fiscal_ex)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# forecast 2 years ahead
predictive     = forecast(
                    posterior,
                    horizon = 8,
                    exogenous_forecast = us_fiscal_ex_forecasts,
                    conditional_forecast = us_fiscal_cond_forecasts
                  )
summary(predictive)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new( exogenous = us_fiscal_ex) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  forecast(
```

```
  horizon = 8,
  exogenous_forecast = us_fiscal_ex_forecasts,
  conditional_forecast = us_fiscal_cond_forecasts
) |> plot()
```

`forecast.PosteriorBSVAREXH` *Forecasting using Bayesian Structural Vector Autoregression*

Description

Samples from the joint predictive density of all of the dependent variables for models at forecast horizons from 1 to `horizon` specified as an argument of the function.

Usage

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

Arguments

`object`	posterior estimation outcome - an object of class `PosteriorBSVAREXH` 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 for heteroskedastic models the draws from the predictive density of structural shocks conditional standard deviations and data. The output elements include:

forecasts an NxTxS array with the draws from predictive density

Y an NxT matrix with the data on dependent variables

forecast_mean an NxTxS array with the mean of the predictive density

forecast_covariance an NxTxS array with the covariance of the predictive density

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

Examples

```
specification  = specify_bsvar_exh$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)
predictive     = forecast(posterior, 4)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_exh$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  forecast(horizon = 4) -> predictive

# conditional forecasting using a model with exogenous variables
############################################################
specification  = specify_bsvar_exh$new(us_fiscal_lsuw, exogenous = us_fiscal_ex)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# forecast 2 years ahead
predictive     = forecast(
                     posterior,
                     horizon = 8,
                     exogenous_forecast = us_fiscal_ex_forecasts,
                     conditional_forecast = us_fiscal_cond_forecasts
                   )
summary(predictive)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_exh$new(exogenous = us_fiscal_ex) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  forecast(
    horizon = 8,
    exogenous_forecast = us_fiscal_ex_forecasts,
    conditional_forecast = us_fiscal_cond_forecasts
  ) |> plot()
```

`forecast.PosteriorBSVARHMSH` *Forecasting using Bayesian Structural Vector Autoregression*

Description

Samples from the joint predictive density of all of the dependent variables for models at forecast horizons from 1 to `horizon` specified as an argument of the function.

Usage

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

Arguments

`object`	posterior estimation outcome - an object of class `PosteriorBSVARHMSH` 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 for heteroskedastic models the draws from the predictive density of structural shocks conditional standard deviations and data. The output elements include:

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

Y an NxT matrix with the data on dependent variables

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

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

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

Examples

```
specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw, M = 2)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)
predictive     = forecast(posterior, 4)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_hmsh$new(M = 2) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  forecast(horizon = 4) -> predictive

# forecasting using a model with exogenous variables
############################################################
specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw, M = 2, exogenous = us_fiscal_ex)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# forecast 2 years ahead
predictive     = forecast(
                     posterior,
                     horizon = 8,
                     exogenous_forecast = us_fiscal_ex_forecasts
                   )
summary(predictive)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_hmsh$new(M = 2, exogenous = us_fiscal_ex) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  forecast(
    horizon = 8,
    exogenous_forecast = us_fiscal_ex_forecasts
  ) |> plot()
```

`forecast.PosteriorBSVARMIX` *Forecasting using Bayesian Structural Vector Autoregression*

Description

Samples from the joint predictive density of all of the dependent variables for models at forecast horizons from 1 to `horizon` specified as an argument of the function.

Usage

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

Arguments

`object`	posterior estimation outcome - an object of class `PosteriorBSVARMIX` 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 for heteroskedastic models the draws from the predictive density of structural shocks conditional standard deviations and data. The output elements include:

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

Y an NxT matrix with the data on dependent variables

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

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

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

Examples

```
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, M = 2)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)
predictive     = forecast(posterior, 4)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_mix$new(M = 2) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  forecast(horizon = 4) -> predictive

# conditional forecasting using a model with exogenous variables
############################################################
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, M = 2, exogenous = us_fiscal_ex)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# forecast 2 years ahead
predictive     = forecast(
                     posterior,
                     horizon = 8,
                     exogenous_forecast = us_fiscal_ex_forecasts,
                     conditional_forecast = us_fiscal_cond_forecasts
                   )
summary(predictive)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_mix$new(M = 2, exogenous = us_fiscal_ex) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  forecast(
    horizon = 8,
    exogenous_forecast = us_fiscal_ex_forecasts,
    conditional_forecast = us_fiscal_cond_forecasts
  ) |> plot()
```

`forecast.PosteriorBSVARMSH`

Forecasting using Bayesian Structural Vector Autoregression

Description

Samples from the joint predictive density of all of the dependent variables for models at forecast horizons from 1 to `horizon` specified as an argument of the function.

Usage

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

Arguments

`object`	posterior estimation outcome - an object of class `PosteriorBSVARMSH` 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 for heteroskedastic models the draws from the predictive density of structural shocks conditional standard deviations and data. The output elements include:

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

Y an NxT matrix with the data on dependent variables

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

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

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

Examples

```
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, M = 2)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)
predictive     = forecast(posterior, 4)

# workflow with the pipe |>
```

```
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new(M = 2) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  forecast(horizon = 4) -> predictive

# conditional forecasting using a model with exogenous variables
############################################################
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, M = 2, exogenous = us_fiscal_ex)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# forecast 2 years ahead
predictive     = forecast(
                     posterior,
                     horizon = 8,
                     exogenous_forecast = us_fiscal_ex_forecasts,
                     conditional_forecast = us_fiscal_cond_forecasts
                   )
summary(predictive)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new(M = 2, exogenous = us_fiscal_ex) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  forecast(
    horizon = 8,
    exogenous_forecast = us_fiscal_ex_forecasts,
    conditional_forecast = us_fiscal_cond_forecasts
  ) |> plot()
```

`forecast.PosteriorBSVARSV` *Forecasting using Bayesian Structural Vector Autoregression*

Description

Samples from the joint predictive density of all of the dependent variables for models at forecast horizons from 1 to `horizon` specified as an argument of the function.

Usage

```
## S3 method for class 'PosteriorBSVARSV'
forecast(
  object,
  horizon = 1,
```

```
  exogenous_forecast = NULL,
  conditional_forecast = NULL,
  ...
)
```

Arguments

`object`	posterior estimation outcome - an object of class `PosteriorBSVARSV` 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 for heteroskedastic models the draws from the predictive density of structural shocks conditional standard deviations and data. The output elements include:

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

Y an NxT matrix with the data on dependent variables

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

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

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

Examples

```
specification  = specify_bsvar_sv$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)
predictive     = forecast(posterior, 2)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_sv$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  forecast(horizon = 2) -> predictive
```

```
# conditional forecasting using a model with exogenous variables
############################################################
specification  = specify_bsvar_sv$new(us_fiscal_lsuw, exogenous = us_fiscal_ex)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# forecast 2 years ahead
predictive     = forecast(
                     posterior,
                     horizon = 8,
                     exogenous_forecast = us_fiscal_ex_forecasts,
                     conditional_forecast = us_fiscal_cond_forecasts
                   )
summary(predictive)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_sv$new(exogenous = us_fiscal_ex) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  forecast(
    horizon = 8,
    exogenous_forecast = us_fiscal_ex_forecasts,
    conditional_forecast = us_fiscal_cond_forecasts
  ) |> plot()
```

`forecast.PosteriorBSVART` *Forecasting using Bayesian Structural Vector Autoregression*

Description

Samples from the joint predictive density of all of the dependent variables for models at forecast horizons from 1 to `horizon` specified as an argument of the function.

Usage

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

Arguments

`object`	posterior estimation outcome - an object of class `PosteriorBSVART` 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 for heteroskedastic models the draws from the predictive density of structural shocks conditional standard deviations and data. The output elements include:

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

Y an NxT matrix with the data on dependent variables

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

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

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

Examples

```
specification  = specify_bsvar_t$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)
predictive     = forecast(posterior, 4)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_t$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  forecast(horizon = 4) -> predictive

# conditional forecasting using a model with exogenous variables
############################################################
specification  = specify_bsvar_t$new(us_fiscal_lsuw, exogenous = us_fiscal_ex)
burn_in        = estimate(specification, 5)
```

```
posterior      = estimate(burn_in, 5)

# forecast 2 years ahead
predictive     = forecast(
                     posterior,
                     horizon = 8,
                     exogenous_forecast = us_fiscal_ex_forecasts,
                     conditional_forecast = us_fiscal_cond_forecasts
                   )
summary(predictive)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_t$new(exogenous = us_fiscal_ex) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  forecast(
    horizon = 8,
    exogenous_forecast = us_fiscal_ex_forecasts,
    conditional_forecast = us_fiscal_cond_forecasts
  ) |> plot()
```

`normalise` *Waggoner & Zha (2003) row signs normalisation of the posterior draws for the structural matrix B*

Description

Normalises the sign of rows of matrix B MCMC draws, relative to matrix `B_benchmark`, provided as the second argument. The implemented procedure proposed by Waggoner, Zha (2003) normalises the MCMC output in an optimal way leading to the unimodal posterior. Only normalised MCMC output is suitable for the computations of the posterior characteristics of the B matrix elements and their functions such as the impulse response functions and other economically interpretable values.

Usage

```
normalise(posterior, B_benchmark = NULL)
```

Arguments

`posterior`	posterior estimation outcome obtained using function `estimate()` containing, amongst other draws, the `S` draws from the posterior distribution of the `NxN` structural matrix of contemporaneous relationships B. These draws are to be normalised with respect to the matrix `B_benchmark`.
`B_benchmark`	the benchmark `NxN` structural matrix specified by the user to have the desired row signs

Value

An object of the same class as that provided as the input argument `posterior` containing the posterior draws including the draws of the normalised structural matrix.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Waggoner, D.F., and Zha, T., (2003) Likelihood Preserving Normalization in Multiple Equation Models. *Journal of Econometrics*, **114**(2), 329–47, doi:10.1016/S03044076(03)000873.

See Also

`estimate`

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw)        # specify the model
burn_in        = estimate(specification, 5)               # run the burn-in
posterior      = estimate(burn_in, 5)                     # estimate the model

# normalise the posterior
BB             = posterior$last_draw$starting_values$B    # get the last draw of B
B_benchmark       = diag((-1) * sign(diag(BB))) %*% BB     # set negative diagonal elements
posterior    = normalise(posterior, B_benchmark)         # draws in posterior are normalised
```

`normalise.PosteriorBSVAR`

Waggoner & Zha (2003) row signs normalisation of the posterior draws for the structural matrix B

Description

Normalises the sign of rows of matrix B MCMC draws, relative to matrix `B_benchmark`, provided as the second argument. The implemented procedure proposed by Waggoner, Zha (2003) normalises the MCMC output in an optimal way leading to the unimodal posterior. Only normalised MCMC output is suitable for the computations of the posterior characteristics of the B matrix elements and their functions such as the impulse response functions and other economically interpretable values.

Usage

```
## S3 method for class 'PosteriorBSVAR'
normalise(posterior, B_benchmark = NULL)
```

Arguments

posterior	posterior estimation outcome of class PosteriorBSVAR generated using the estimate() function, amongst other draws, the S draws from the posterior distribution of the NxN structural matrix of contemporaneous relationships B. These draws are to be normalised with respect to the matrix B_benchmark.
B_benchmark	the benchmark NxN structural matrix specified by the user to have the desired row signs

Value

An object of the same class as that provided as the input argument posterior containing the posterior draws including the draws of the normalised structural matrix.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Waggoner, D.F., and Zha, T., (2003) Likelihood Preserving Normalization in Multiple Equation Models. *Journal of Econometrics*, **114**(2), 329–47, doi:10.1016/S03044076(03)000873.

See Also

estimate

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw)        # specify the model
burn_in        = estimate(specification, 5)               # run the burn-in
posterior      = estimate(burn_in, 5)                     # estimate the model

# normalise the posterior
BB             = posterior$last_draw$starting_values$B    # get the last draw of B
B_benchmark       = diag((-1) * sign(diag(BB))) %*% BB     # set negative diagonal elements
posterior    = normalise(posterior, B_benchmark)         # draws in posterior are normalised
```

---

normalise.PosteriorBSVAREXH

Waggoner & Zha (2003) row signs normalisation of the posterior draws for the structural matrix B

Description

Normalises the sign of rows of matrix B MCMC draws, relative to matrix B_benchmark, provided as the second argument. The implemented procedure proposed by Waggoner, Zha (2003) normalises the MCMC output in an optimal way leading to the unimodal posterior. Only normalised MCMC output is suitable for the computations of the posterior characteristics of the B matrix elements and their functions such as the impulse response functions and other economically interpretable values.

Usage

```
## S3 method for class 'PosteriorBSVAREXH'
normalise(posterior, B_benchmark = NULL)
```

Arguments

posterior	posterior estimation outcome of class PosteriorBSVAREXH generated using the estimate() function, amongst other draws, the S draws from the posterior distribution of the NxN structural matrix of contemporaneous relationships B. These draws are to be normalised with respect to the matrix B_benchmark.
B_benchmark	the benchmark NxN structural matrix specified by the user to have the desired row signs

Value

An object of the same class as that provided as the input argument posterior containing the posterior draws including the draws of the normalised structural matrix.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Waggoner, D.F., and Zha, T., (2003) Likelihood Preserving Normalization in Multiple Equation Models. *Journal of Econometrics*, **114**(2), 329–47, doi:10.1016/S03044076(03)000873.

See Also

estimate

Examples

```
specification  = specify_bsvar_exh$new(us_fiscal_lsuw)    # specify the model
burn_in        = estimate(specification, 5)               # run the burn-in
posterior      = estimate(burn_in, 5)                     # estimate the model

# normalise the posterior
BB             = posterior$last_draw$starting_values$B    # get the last draw of B
B_benchmark       = diag((-1) * sign(diag(BB))) %*% BB     # set negative diagonal elements
posterior    = normalise(posterior, B_benchmark)         # draws in posterior are normalised
```

`normalise.PosteriorBSVARHMSH` *Waggoner & Zha (2003) row signs normalisation of the posterior draws for the structural matrix B*

Description

Normalises the sign of rows of matrix B MCMC draws, relative to matrix `B_benchmark`, provided as the second argument. The implemented procedure proposed by Waggoner, Zha (2003) normalises the MCMC output in an optimal way leading to the unimodal posterior. Only normalised MCMC output is suitable for the computations of the posterior characteristics of the B matrix elements and their functions such as the impulse response functions and other economically interpretable values.

Usage

```
## S3 method for class 'PosteriorBSVARHMSH'
normalise(posterior, B_benchmark = NULL)
```

Arguments

`posterior`	posterior estimation outcome of class `PosteriorBSVARHMSH` generated using the `estimate()` function, amongst other draws, the `S` draws from the posterior distribution of the `NxN` structural matrix of contemporaneous relationships B. These draws are to be normalised with respect to the matrix `B_benchmark`.
`B_benchmark`	the benchmark `NxN` structural matrix specified by the user to have the desired row signs

Value

An object of the same class as that provided as the input argument `posterior` containing the posterior draws including the draws of the normalised structural matrix.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Waggoner, D.F., and Zha, T., (2003) Likelihood Preserving Normalization in Multiple Equation Models. *Journal of Econometrics*, **114**(2), 329–47, doi:10.1016/S03044076(03)000873.

See Also

`estimate`

Examples

```
specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw)    # specify the model
burn_in        = estimate(specification, 5)               # run the burn-in
posterior      = estimate(burn_in, 5)                     # estimate the model

# normalise the posterior
BB             = posterior$last_draw$starting_values$B    # get the last draw of B
B_benchmark       = diag((-1) * sign(diag(BB))) %*% BB     # set negative diagonal elements
posterior    = normalise(posterior, B_benchmark)         # draws in posterior are normalised
```

`normalise.PosteriorBSVARMIX` *Waggoner & Zha (2003) row signs normalisation of the posterior draws for the structural matrix B*

Description

Normalises the sign of rows of matrix B MCMC draws, relative to matrix `B_benchmark`, provided as the second argument. The implemented procedure proposed by Waggoner, Zha (2003) normalises the MCMC output in an optimal way leading to the unimodal posterior. Only normalised MCMC output is suitable for the computations of the posterior characteristics of the B matrix elements and their functions such as the impulse response functions and other economically interpretable values.

Usage

```
## S3 method for class 'PosteriorBSVARMIX'
normalise(posterior, B_benchmark = NULL)
```

Arguments

`posterior`	posterior estimation outcome of class `PosteriorBSVARMIX` generated using the `estimate()` function, amongst other draws, the `S` draws from the posterior distribution of the `NxN` structural matrix of contemporaneous relationships B. These draws are to be normalised with respect to the matrix `B_benchmark`.
`B_benchmark`	the benchmark `NxN` structural matrix specified by the user to have the desired row signs

Value

An object of the same class as that provided as the input argument `posterior` containing the posterior draws including the draws of the normalised structural matrix.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Waggoner, D.F., and Zha, T., (2003) Likelihood Preserving Normalization in Multiple Equation Models. *Journal of Econometrics*, **114**(2), 329–47, doi:10.1016/S03044076(03)000873.

See Also

estimate

Examples

```
specification  = specify_bsvar_mix$new(us_fiscal_lsuw)    # specify the model
burn_in        = estimate(specification, 5)               # run the burn-in
posterior      = estimate(burn_in, 5)                     # estimate the model

# normalise the posterior
BB             = posterior$last_draw$starting_values$B    # get the last draw of B
B_benchmark       = diag((-1) * sign(diag(BB))) %*% BB     # set negative diagonal elements
posterior    = normalise(posterior, B_benchmark)         # draws in posterior are normalised
```

`normalise.PosteriorBSVARMSH`

Waggoner & Zha (2003) row signs normalisation of the posterior draws for the structural matrix B

Description

Normalises the sign of rows of matrix B MCMC draws, relative to matrix `B_benchmark`, provided as the second argument. The implemented procedure proposed by Waggoner, Zha (2003) normalises the MCMC output in an optimal way leading to the unimodal posterior. Only normalised MCMC output is suitable for the computations of the posterior characteristics of the B matrix elements and their functions such as the impulse response functions and other economically interpretable values.

Usage

```
## S3 method for class 'PosteriorBSVARMSH'
normalise(posterior, B_benchmark = NULL)
```

Arguments

`posterior`	posterior estimation outcome of class `PosteriorBSVARMSH` generated using the `estimate()` function, amongst other draws, the `S` draws from the posterior distribution of the `NxN` structural matrix of contemporaneous relationships B. These draws are to be normalised with respect to the matrix `B_benchmark`.
`B_benchmark`	the benchmark `NxN` structural matrix specified by the user to have the desired row signs

Value

An object of the same class as that provided as the input argument `posterior` containing the posterior draws including the draws of the normalised structural matrix.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Waggoner, D.F., and Zha, T., (2003) Likelihood Preserving Normalization in Multiple Equation Models. *Journal of Econometrics*, **114**(2), 329–47, doi:10.1016/S03044076(03)000873.

See Also

`estimate`

Examples

```
specification  = specify_bsvar_msh$new(us_fiscal_lsuw)    # specify the model
burn_in        = estimate(specification, 5)               # run the burn-in
posterior      = estimate(burn_in, 5)                     # estimate the model

# normalise the posterior
BB             = posterior$last_draw$starting_values$B    # get the last draw of B
B_benchmark       = diag((-1) * sign(diag(BB))) %*% BB     # set negative diagonal elements
posterior    = normalise(posterior, B_benchmark)         # draws in posterior are normalised
```

`normalise.PosteriorBSVARSV`

Waggoner & Zha (2003) row signs normalisation of the posterior draws for the structural matrix B

Description

Normalises the sign of rows of matrix B MCMC draws, relative to matrix `B_benchmark`, provided as the second argument. The implemented procedure proposed by Waggoner, Zha (2003) normalises the MCMC output in an optimal way leading to the unimodal posterior. Only normalised MCMC output is suitable for the computations of the posterior characteristics of the B matrix elements and their functions such as the impulse response functions and other economically interpretable values.

Usage

```
## S3 method for class 'PosteriorBSVARSV'
normalise(posterior, B_benchmark = NULL)
```

Arguments

posterior	posterior estimation outcome of class PosteriorBSVARSV generated using the estimate() function, amongst other draws, the S draws from the posterior distribution of the NxN structural matrix of contemporaneous relationships B. These draws are to be normalised with respect to the matrix B_benchmark.
B_benchmark	the benchmark NxN structural matrix specified by the user to have the desired row signs

Value

An object of the same class as that provided as the input argument posterior containing the posterior draws including the draws of the normalised structural matrix.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Waggoner, D.F., and Zha, T., (2003) Likelihood Preserving Normalization in Multiple Equation Models. *Journal of Econometrics*, **114**(2), 329–47, doi:10.1016/S03044076(03)000873.

See Also

estimate

Examples

```
specification  = specify_bsvar_sv$new(us_fiscal_lsuw)     # specify the model
burn_in        = estimate(specification, 5)               # run the burn-in
posterior      = estimate(burn_in, 5)                     # estimate the model

# normalise the posterior
BB             = posterior$last_draw$starting_values$B    # get the last draw of B
B_benchmark       = diag((-1) * sign(diag(BB))) %*% BB     # set negative diagonal elements
posterior    = normalise(posterior, B_benchmark)         # draws in posterior are normalised
```

---

normalise.PosteriorBSVART

Waggoner & Zha (2003) row signs normalisation of the posterior draws for the structural matrix B

Description

Normalises the sign of rows of matrix B MCMC draws, relative to matrix B_benchmark, provided as the second argument. The implemented procedure proposed by Waggoner, Zha (2003) normalises the MCMC output in an optimal way leading to the unimodal posterior. Only normalised MCMC output is suitable for the computations of the posterior characteristics of the B matrix elements and their functions such as the impulse response functions and other economically interpretable values.

Usage

```
## S3 method for class 'PosteriorBSVART'
normalise(posterior, B_benchmark = NULL)
```

Arguments

posterior	posterior estimation outcome of class PosteriorBSVART generated using the estimate() function, amongst other draws, the S draws from the posterior distribution of the NxN structural matrix of contemporaneous relationships B. These draws are to be normalised with respect to the matrix B_benchmark.
B_benchmark	the benchmark NxN structural matrix specified by the user to have the desired row signs

Value

An object of the same class as that provided as the input argument posterior containing the posterior draws including the draws of the normalised structural matrix.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Waggoner, D.F., and Zha, T., (2003) Likelihood Preserving Normalization in Multiple Equation Models. *Journal of Econometrics*, **114**(2), 329–47, doi:10.1016/S03044076(03)000873.

See Also

estimate

Examples

```
specification  = specify_bsvar_t$new(us_fiscal_lsuw)     # specify the model
burn_in        = estimate(specification, 5)               # run the burn-in
posterior      = estimate(burn_in, 5)                     # estimate the model

# normalise the posterior
BB             = posterior$last_draw$starting_values$B    # get the last draw of B
B_benchmark       = diag((-1) * sign(diag(BB))) %*% BB     # set negative diagonal elements
posterior    = normalise(posterior, B_benchmark)         # draws in posterior are normalised
```

`plot.Forecasts` *Plots fitted values of dependent variables*

Description

Plots of fitted values of dependent variables including their median and percentiles.

Usage

```
## S3 method for class 'Forecasts'
plot(
  x,
  probability = 0.9,
  data_in_plot = 1,
  col = "#ff69b4",
  main,
  xlab,
  mar.multi = c(1, 4.6, 0, 2.1),
  oma.multi = c(6, 0, 5, 0),
  ...
)
```

Arguments

`x`	an object of class Forecasts obtained using the `forecast()` function containing posterior draws of fitted values of dependent variables.
`probability`	a parameter determining the interval to be plotted. The interval stretches from the `0.5 * (1 - probability)` to `1 - 0.5 * (1 - probability)` percentile of the posterior distribution.
`data_in_plot`	a fraction value in the range (0, 1) determining how many of the last observations in the data should be plotted with the forecasts.
`col`	a colour of the plot line and the ribbon
`main`	an alternative main title for the plot
`xlab`	an alternative x-axis label for the plot
`mar.multi`	the default `mar` argument setting in `graphics::par`. Modify with care!
`oma.multi`	the default `oma` argument setting in `graphics::par`. Modify with care!
`...`	additional arguments affecting the summary produced.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

forecast

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw)    # specify model
burn_in        = estimate(specification, 5)           # run the burn-in
posterior      = estimate(burn_in, 5)                # estimate the model

# compute forecasts
fore            = forecast(posterior, horizon = 4)
plot(fore)                                            # plot forecasts

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  forecast(horizon = 4) |>
  plot()
```

---

plot.PosteriorFEVD *Plots forecast error variance decompositions*

Description

Plots of the posterior means of the forecast error variance decompositions.

Usage

```
## S3 method for class 'PosteriorFEVD'
plot(
  x,
  shock_names,
  cols,
  main,
  xlab,
  mar.multi = c(1, 4.6, 0, 4.6),
  oma.multi = c(6, 0, 5, 0),
  ...
)
```

Arguments

x	an object of class PosteriorFEVD obtained using the compute_variance_decompositions() function containing posterior draws of forecast error variance decompositions.

shock_names	a vector of length N containing names of the structural shocks.
cols	an N-vector with colours of the plot
main	an alternative main title for the plot
xlab	an alternative x-axis label for the plot
mar.multi	the default mar argument setting in graphics::par. Modify with care!
oma.multi	the default oma argument setting in graphics::par. Modify with care!
...	additional arguments affecting the summary produced.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

compute_variance_decompositions

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw)    # specify model
burn_in        = estimate(specification, 5)           # run the burn-in
posterior      = estimate(burn_in, 5)                 # estimate the model

# compute forecast error variance decompositions
fevd           = compute_variance_decompositions(posterior, horizon = 4)
plot(fevd)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_variance_decompositions(horizon = 4) |>
  plot()
```

---

plot.PosteriorFitted *Plots fitted values of dependent variables*

Description

Plots of fitted values of dependent variables including their median and percentiles.

Usage

```
## S3 method for class 'PosteriorFitted'
plot(
  x,
  probability = 0.9,
  col = "#ff69b4",
  main,
  xlab,
  mar.multi = c(1, 4.6, 0, 2.1),
  oma.multi = c(6, 0, 5, 0),
  ...
)
```

Arguments

x	an object of class PosteriorFitted obtained using the compute_fitted_values() function containing posterior draws of fitted values of dependent variables.
probability	a parameter determining the interval to be plotted. The interval stretches from the 0.5 * (1 - probability) to 1 - 0.5 * (1 - probability) percentile of the posterior distribution.
col	a colour of the plot line and the ribbon
main	an alternative main title for the plot
xlab	an alternative x-axis label for the plot
mar.multi	the default mar argument setting in graphics::par. Modify with care!
oma.multi	the default oma argument setting in graphics::par. Modify with care!
...	additional arguments affecting the summary produced.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

compute_fitted_values

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw)    # specify model
burn_in        = estimate(specification, 5)          # run the burn-in
posterior      = estimate(burn_in, 5)                # estimate the model

# compute fitted values
fitted         = compute_fitted_values(posterior)
plot(fitted)                                          # plot fitted values

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

```
specify_bsvar$new() |>
estimate(S = 5) |>
estimate(S = 5) |>
compute_fitted_values() |>
plot()
```

`plot.PosteriorHD` *Plots historical decompositions*

Description

Plots of the posterior means of the historical decompositions.

Usage

```
## S3 method for class 'PosteriorHD'
plot(
  x,
  shock_names,
  cols,
  main,
  xlab,
  mar.multi = c(1, 4.6, 0, 4.6),
  oma.multi = c(6, 0, 5, 0),
  ...
)
```

Arguments

`x`	an object of class PosteriorHD obtained using the `compute_historical_decompositions()` function containing posterior draws of historical decompositions.
`shock_names`	a vector of length `N` containing names of the structural shocks.
`cols`	an `N`-vector with colours of the plot
`main`	an alternative main title for the plot
`xlab`	an alternative x-axis label for the plot
`mar.multi`	the default `mar` argument setting in `graphics::par`. Modify with care!
`oma.multi`	the default `oma` argument setting in `graphics::par`. Modify with care!
`...`	additional arguments affecting the summary produced.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

`compute_historical_decompositions`

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw)    # specify model
burn_in        = estimate(specification, 5)           # run the burn-in
posterior      = estimate(burn_in, 5)                 # estimate the model

# compute historical decompositions
fevd           = compute_historical_decompositions(posterior)
plot(fevd)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_historical_decompositions() |>
  plot()
```

`plot.PosteriorIR` *Plots impulse responses*

Description

Plots of of all variables to all shocks including their median and percentiles.

Usage

```
## S3 method for class 'PosteriorIR'
plot(
  x,
  probability = 0.9,
  shock_names,
  col = "#ff69b4",
  main,
  xlab,
  mar.multi = c(1, 4.1, 0, 1.1),
  oma.multi = c(6, 0, 5, 0),
  ...
)
```

Arguments

`x`	an object of class PosteriorIR obtained using the `compute_impulse_responses()` function containing posterior draws of impulse responses.
`probability`	a parameter determining the interval to be plotted. The interval stretches from the `0.5 * (1 - probability)` to `1 - 0.5 * (1 - probability)` percentile of the posterior distribution.

shock_names	a vector of length N containing names of the structural shocks.
col	a colour of the plot line and the ribbon
main	an alternative main title for the plot
xlab	an alternative x-axis label for the plot
mar.multi	the default mar argument setting in graphics::par. Modify with care!
oma.multi	the default oma argument setting in graphics::par. Modify with care!
...	additional arguments affecting the summary produced.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

compute_impulse_responses

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw)   # specify model
burn_in        = estimate(specification, 5)          # run the burn-in
posterior      = estimate(burn_in, 5)                # estimate the model

# compute impulse responses``
fitted         = compute_impulse_responses(posterior, horizon = 4)
plot(fitted)                                          # plot

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_impulse_responses(horizon = 4) |>
  plot()
```

---

plot.PosteriorRegimePr

Plots estimated regime probabilities

Description

Plots of estimated regime probabilities of Markov-switching heteroskedasticity or allocations of normal-mixture components including their median and percentiles. Compact zero-based realized regime indices are displayed using the equivalent regime indicators.

Usage

```
## S3 method for class 'PosteriorRegimePr'
plot(
  x,
  probability = 0.9,
  col = "#ff69b4",
  main,
  xlab,
  mar.multi = c(1, 4.6, 0, 2.1),
  oma.multi = c(6, 0, 5, 0),
  ...
)
```

Arguments

x	an object of class PosteriorRegimePr obtained using the compute_regime_probabilities() function containing posterior draws of compact realized regime indices or regime probabilities.
probability	a parameter determining the interval to be plotted. The interval stretches from the 0.5 * (1 - probability) to 1 - 0.5 * (1 - probability) percentile of the posterior distribution.
col	a colour of the plot line and the ribbon
main	an alternative main title for the plot
xlab	an alternative x-axis label for the plot
mar.multi	the default mar argument setting in graphics::par. Modify with care!
oma.multi	the default oma argument setting in graphics::par. Modify with care!
...	additional arguments affecting the summary produced.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

compute_regime_probabilities

Examples

```
specification  = specify_bsvar_msh$new(us_fiscal_lsuw)# specify model
burn_in        = estimate(specification, 5)           # run the burn-in
posterior      = estimate(burn_in, 5)                 # estimate the model

# compute regime probabilities
rp             = compute_regime_probabilities(posterior)
plot(rp)                                              # plot

# workflow with the pipe |>
```

```
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_regime_probabilities() |>
  plot()
```

`plot.PosteriorShocks` *Plots structural shocks*

Description

Plots of structural shocks including their median and percentiles.

Usage

```
## S3 method for class 'PosteriorShocks'
plot(
  x,
  probability = 0.9,
  shock_names,
  col = "#ff69b4",
  main,
  xlab,
  mar.multi = c(1, 4.6, 0, 2.1),
  oma.multi = c(6, 0, 5, 0),
  ...
)
```

Arguments

`x`	an object of class PosteriorShocks obtained using the `compute_structural_shocks()` function containing posterior draws of structural shocks.
`probability`	a parameter determining the interval to be plotted. The interval stretches from the `0.5 * (1 - probability)` to `1 - 0.5 * (1 - probability)` percentile of the posterior distribution.
`shock_names`	a vector of length `N` containing names of the structural shocks.
`col`	a colour of the plot line and the ribbon
`main`	an alternative main title for the plot
`xlab`	an alternative x-axis label for the plot
`mar.multi`	the default `mar` argument setting in `graphics::par`. Modify with care!
`oma.multi`	the default `oma` argument setting in `graphics::par`. Modify with care!
`...`	additional arguments affecting the summary produced.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

compute_structural_shocks

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw)    # specify model
burn_in        = estimate(specification, 5)           # run the burn-in
posterior      = estimate(burn_in, 5)                 # estimate the model

# compute structural shocks
shocks         = compute_structural_shocks(posterior)
plot(shocks)                                          # plot

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_structural_shocks() |>
  plot()
```

---

plot.PosteriorSigma *Plots structural shocks' conditional standard deviations*

Description

Plots of structural shocks' conditional standard deviations including their median and percentiles.

Usage

```
## S3 method for class 'PosteriorSigma'
plot(
  x,
  probability = 0.9,
  shock_names,
  col = "#ff69b4",
  main,
  xlab,
  mar.multi = c(1, 4.6, 0, 2.1),
  oma.multi = c(6, 0, 5, 0),
  ...
)
```

Arguments

x	an object of class PosteriorSigma obtained using the compute_conditional_sd() function containing posterior draws of conditional standard deviations of structural shocks.
probability	a parameter determining the interval to be plotted. The interval stretches from the 0.5 * (1 - probability) to 1 - 0.5 * (1 - probability) percentile of the posterior distribution.
shock_names	a vector of length N containing names of the structural shocks.
col	a colour of the plot line and the ribbon
main	an alternative main title for the plot
xlab	an alternative x-axis label for the plot
mar.multi	the default mar argument setting in graphics::par. Modify with care!
oma.multi	the default oma argument setting in graphics::par. Modify with care!
...	additional arguments affecting the summary produced.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

compute_conditional_sd

Examples

```
specification  = specify_bsvar_sv$new(us_fiscal_lsuw) # specify model
burn_in        = estimate(specification, 5)           # run the burn-in
posterior      = estimate(burn_in, 5)                 # estimate the model

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

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_sv$new(p = 1) |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_conditional_sd() |>
  plot()
```

plot_ribbon *Plots the median and an interval between two specified percentiles for a sequence of* K *random variables*

Description

Plots the median and an interval between two specified percentiles for a sequence of K random variables based on the S posterior draws provided for each of them.

Usage

```
plot_ribbon(
  draws,
  probability = 0.9,
  col = "#ff69b4",
  ylim,
  ylab,
  xlab,
  start_at = 0,
  add = FALSE,
  ...
)
```

Arguments

draws	a K x S matrix with S posterior draws of K random variables, or a K x S x N array with N such matrices
probability	a number from interval (0,1) denoting the probability content of the plotted interval. The interval stretches from the 0.5 * (1 - probability) to 1 - 0.5 * (1 - probability) percentile of the posterior distribution.
col	a colour of the plot
ylim	the range of the y axis
ylab	the label of the y axis
xlab	the label of the x axis
start_at	an integer to denote the beginning of the x axis range
add	a logical value. If TRUE the current ribbon plot is added to an existing plot
...	other graphical parameters to be passed to base::plot

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw)                # specify model
burn_in        = estimate(specification, 10)                      # run the burn-in
posterior      = estimate(burn_in, 20, thin = 1)                  # estimate the model
irf            = compute_impulse_responses(posterior, horizon = 4) # impulse responses
plot_ribbon(irf[1,1,,])
```

`specify_bsvar` *R6 Class representing the specification of the homoskedastic BSVAR model*

Description

The class BSVAR presents complete specification for the homoskedastic bsvar model.

Public fields

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

`identification` an object IdentificationBSVAR with the identifying restrictions.

`prior` an object PriorBSVAR with the prior specification.

`data_matrices` an object DataMatricesBSVAR with the data matrices.

`starting_values` an object StartingValuesBSVAR with the starting values.

Methods

Public methods:

- `BSVAR$new()`
- `BSVAR$get_normal()`
- `BSVAR$get_data_matrices()`
- `BSVAR$get_identification()`
- `BSVAR$get_prior()`
- `BSVAR$get_starting_values()`
- `BSVAR$clone()`

`BSVAR$new()`: Create a new specification of the homoskedastic bsvar model BSVAR.

Usage:

```
BSVAR$new(
  data,
  p = 1L,
  B,
  A,
  distribution = c("norm", "t"),
  exogenous = NULL,
  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.

`B` a logical `NxN` matrix containing value `TRUE` for the elements of the structural matrix B to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

`A` a logical `NxK` matrix containing value `TRUE` for the elements of the autoregressive matrix A to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

`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.

`exogenous` a `(T+p)xd` 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 random walk process, otherwise to white noise.

Returns: A new complete specification for the homoskedastic bsvar model BSVAR.

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

Usage:

```
BSVAR$get_normal()
```

Examples:

```
spec = specify_bsvar$new(us_fiscal_lsuw)
spec$get_normal()
```

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

Usage:

```
BSVAR$get_data_matrices()
```

Examples:

```
data(us_fiscal_lsuw)
spec = specify_bsvar$new(
   data = us_fiscal_lsuw,
   p = 4
)
spec$get_data_matrices()
```

`BSVAR$get_identification()`**:** Returns the identifying restrictions as the IdentificationBSVARs object.

Usage:

```
BSVAR$get_identification()
```

Examples:

```
data(us_fiscal_lsuw)
spec = specify_bsvar$new(
   data = us_fiscal_lsuw,
   p = 4
)
spec$get_identification()
```

`BSVAR$get_prior()`: Returns the prior specification as the PriorBSVAR object.

Usage:

```
BSVAR$get_prior()
```

Examples:

```
data(us_fiscal_lsuw)
spec = specify_bsvar$new(
   data = us_fiscal_lsuw,
   p = 4
)
spec$get_prior()
```

`BSVAR$get_starting_values()`: Returns the starting values as the StartingValuesBSVAR object.

Usage:

```
BSVAR$get_starting_values()
```

Examples:

```
data(us_fiscal_lsuw)
spec = specify_bsvar$new(
   data = us_fiscal_lsuw,
   p = 4
)
spec$get_starting_values()
```

`BSVAR$clone()`: The objects of this class are cloneable with this method.

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

See Also

`estimate`, `specify_posterior_bsvar`

Examples

```
data(us_fiscal_lsuw)
spec = specify_bsvar$new(
   data = us_fiscal_lsuw,
   p = 4
)


## ------------------------------------------------
## Method `BSVAR$get_normal()`
## ------------------------------------------------

spec = specify_bsvar$new(us_fiscal_lsuw)
```

```
spec$get_normal()


## ------------------------------------------------
## Method `BSVAR$get_data_matrices()`
## ------------------------------------------------

data(us_fiscal_lsuw)
spec = specify_bsvar$new(
   data = us_fiscal_lsuw,
   p = 4
)
spec$get_data_matrices()


## ------------------------------------------------
## Method `BSVAR$get_identification()`
## ------------------------------------------------

data(us_fiscal_lsuw)
spec = specify_bsvar$new(
   data = us_fiscal_lsuw,
   p = 4
)
spec$get_identification()


## ------------------------------------------------
## Method `BSVAR$get_prior()`
## ------------------------------------------------

data(us_fiscal_lsuw)
spec = specify_bsvar$new(
   data = us_fiscal_lsuw,
   p = 4
)
spec$get_prior()


## ------------------------------------------------
## Method `BSVAR$get_starting_values()`
## ------------------------------------------------

data(us_fiscal_lsuw)
spec = specify_bsvar$new(
   data = us_fiscal_lsuw,
   p = 4
)
spec$get_starting_values()
```

`specify_bsvar_exh` *R6 Class representing the specification of the BSVAREXH model with exogenous heteroskedastic regime change.*

Description

The class BSVAREXH presents complete specification for the BSVAR model with exogenous heteroskedastic regime change.

Public fields

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

`identification` an object IdentificationBSVARs with the identifying restrictions.

`prior` an object PriorBSVAREXH with the prior specification.

`data_matrices` an object DataMatricesBSVAR with the data matrices.

`starting_values` an object StartingValuesBSVAREXH with the starting values.

`variance_regimes` a `T`-vector with exogenous regime indicators that are integer numbers associating the time observation with heteroskedastic regime.

Methods

Public methods:

- `BSVAREXH$new()`
- `BSVAREXH$get_normal()`
- `BSVAREXH$get_data_matrices()`
- `BSVAREXH$get_identification()`
- `BSVAREXH$get_prior()`
- `BSVAREXH$get_starting_values()`
- `BSVAREXH$clone()`

`BSVAREXH$new()`**:** Create a new specification of the BSVAR model with Markov Switching Heteroskedasticity, BSVAREXH.

Usage:

```
BSVAREXH$new(
  data,
  p = 1L,
  B,
  A,
  distribution = c("norm", "t"),
  exogenous = NULL,
  stationary = rep(FALSE, ncol(data)),
  variance_regimes = NULL
)
```

Arguments:

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

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

`B` a logical `NxN` matrix containing value `TRUE` for the elements of the structural matrix B to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

`A` a logical `NxK` matrix containing value `TRUE` for the elements of the autoregressive matrix A to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

`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.

`exogenous` a `(T+p)xd` 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 random walk process, otherwise to white noise.

`variance_regimes` a `T`-vector with exogenous regime indicators that are integer numbers associating the time observation with heteroskedastic regime.

Returns: A new complete specification for the bsvar model with exogenous heteroskedastic regime change, BSVAREXH.

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

Usage:

```
BSVAREXH$get_normal()
```

Examples:

```
spec = specify_bsvar_exh$new(us_fiscal_lsuw)
spec$get_normal()
```

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

Usage:

```
BSVAREXH$get_data_matrices()
```

Examples:

```
spec = specify_bsvar_exh$new(us_fiscal_lsuw)
spec$get_data_matrices()
```

`BSVAREXH$get_identification()`**:** Returns the identifying restrictions as the IdentificationBSVARs object.

Usage:

```
BSVAREXH$get_identification()
```

Examples:

```
spec = specify_bsvar_exh$new(us_fiscal_lsuw)
spec$get_identification()
```

`BSVAREXH$get_prior()`**:** Returns the prior specification as the PriorBSVAREXH object.

Usage:

```
BSVAREXH$get_prior()
```

Examples:

```
spec = specify_bsvar_exh$new(us_fiscal_lsuw)
spec$get_prior()
```

`BSVAREXH$get_starting_values()`: Returns the starting values as the StartingValuesBSVAREXH object.

Usage:

```
BSVAREXH$get_starting_values()
```

Examples:

```
spec = specify_bsvar_exh$new(us_fiscal_lsuw)
spec$get_starting_values()
```

`BSVAREXH$clone()`: The objects of this class are cloneable with this method.

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

See Also

`estimate`, `specify_posterior_bsvar_exh`

Examples

```
spec = specify_bsvar_exh$new(us_fiscal_lsuw)


## ------------------------------------------------
## Method `BSVAREXH$get_normal()`
## ------------------------------------------------

spec = specify_bsvar_exh$new(us_fiscal_lsuw)
spec$get_normal()


## ------------------------------------------------
## Method `BSVAREXH$get_data_matrices()`
## ------------------------------------------------

spec = specify_bsvar_exh$new(us_fiscal_lsuw)
spec$get_data_matrices()


## ------------------------------------------------
## Method `BSVAREXH$get_identification()`
## ------------------------------------------------
```

```
spec = specify_bsvar_exh$new(us_fiscal_lsuw)
spec$get_identification()


## ------------------------------------------------
## Method `BSVAREXH$get_prior()`
## ------------------------------------------------

spec = specify_bsvar_exh$new(us_fiscal_lsuw)
spec$get_prior()


## ------------------------------------------------
## Method `BSVAREXH$get_starting_values()`
## ------------------------------------------------

spec = specify_bsvar_exh$new(us_fiscal_lsuw)
spec$get_starting_values()
```

`specify_bsvar_hmsh` *R6 Class representing the specification of the BSVARHMSH model with Heterogeneous Markov Switching Heteroskedasticity.*

Description

The class BSVARHMSH presents complete specification for the BSVAR model with Heterogeneous Markov Switching Heteroskedasticity.

Public fields

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

`identification` an object IdentificationBSVARs with the identifying restrictions.

`prior` an object PriorBSVARMSH with the prior specification.

`data_matrices` an object DataMatricesBSVAR with the data matrices.

`starting_values` an object StartingValuesBSVARHMSH with the starting values.

`finiteM` a logical value - if true a stationary Markov switching model is estimated. Otherwise, a sparse Markov switching model is estimated in which `M=20` and the number of visited states is estimated.

Methods

Public methods:

- `BSVARHMSH$new()`
- `BSVARHMSH$get_normal()`
- `BSVARHMSH$get_data_matrices()`

- BSVARHMSH$get_identification()
- BSVARHMSH$get_prior()
- BSVARHMSH$get_starting_values()
- BSVARHMSH$clone()

`BSVARHMSH$new()`: Create a new specification of the BSVAR model with Heterogeneous Markov Switching Heteroskedasticity, BSVARHMSH.

Usage:

```
BSVARHMSH$new(
  data,
  p = 1L,
  M = 2L,
  B,
  A,
  distribution = c("norm", "t"),
  exogenous = NULL,
  stationary = rep(FALSE, ncol(data)),
  finiteM = TRUE
)
```

Arguments:

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

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

`M` an integer greater than 1 - the number of Markov process' heteroskedastic regimes.

`B` a logical `NxN` matrix containing value `TRUE` for the elements of the structural matrix B to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

`A` a logical `NxK` matrix containing value `TRUE` for the elements of the autoregressive matrix A to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

`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.

`exogenous` a `(T+p)xd` 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 random walk process, otherwise to white noise.

`finiteM` a logical value - if true a stationary Markov switching model is estimated. Otherwise, a sparse Markov switching model is estimated in which `M=20` and the number of visited states is estimated.

Returns: A new complete specification for the bsvar model with Heterogeneous Markov Switching Heteroskedasticity, BSVARHMSH.

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

Usage:

```
BSVARHMSH$get_normal()
```

Examples:

```
spec = specify_bsvar_hmsh$new(us_fiscal_lsuw)
spec$get_normal()
```

`BSVARHMSH$get_data_matrices()`: Returns the data matrices as the DataMatricesBSVAR object.

Usage:

```
BSVARHMSH$get_data_matrices()
```

Examples:

```
spec = specify_bsvar_hmsh$new(
   data = us_fiscal_lsuw,
   p = 4,
   M = 2
)
spec$get_data_matrices()
```

`BSVARHMSH$get_identification()`: Returns the identifying restrictions as the IdentificationBSVARs object.

Usage:

```
BSVARHMSH$get_identification()
```

Examples:

```
spec = specify_bsvar_hmsh$new(
   data = us_fiscal_lsuw,
   p = 4,
   M = 2
)
spec$get_identification()
```

`BSVARHMSH$get_prior()`: Returns the prior specification as the PriorBSVARMSH object.

Usage:

```
BSVARHMSH$get_prior()
```

Examples:

```
spec = specify_bsvar_hmsh$new(
   data = us_fiscal_lsuw,
   p = 4,
   M = 2
)
spec$get_prior()
```

`BSVARHMSH$get_starting_values()`: Returns the starting values as the StartingValuesBSVARHMSH object.

Usage:

```
BSVARHMSH$get_starting_values()
```

Examples:

```
spec = specify_bsvar_hmsh$new(
   data = us_fiscal_lsuw,
   p = 4,
   M = 2
)
spec$get_starting_values()
```

`BSVARHMSH$clone()`: The objects of this class are cloneable with this method.

Usage:

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

Arguments:

deep Whether to make a deep clone.

See Also

estimate, specify_posterior_bsvar_hmsh

Examples

```
spec = specify_bsvar_hmsh$new(
   data = us_fiscal_lsuw,
   p = 4,
   M = 2
)


## ------------------------------------------------
## Method `BSVARHMSH$get_normal()`
## ------------------------------------------------

spec = specify_bsvar_hmsh$new(us_fiscal_lsuw)
spec$get_normal()


## ------------------------------------------------
## Method `BSVARHMSH$get_data_matrices()`
## ------------------------------------------------

spec = specify_bsvar_hmsh$new(
   data = us_fiscal_lsuw,
   p = 4,
   M = 2
)
spec$get_data_matrices()


## ------------------------------------------------
## Method `BSVARHMSH$get_identification()`
## ------------------------------------------------

spec = specify_bsvar_hmsh$new(
```

```
   data = us_fiscal_lsuw,
   p = 4,
   M = 2
)
spec$get_identification()


## ------------------------------------------------
## Method `BSVARHMSH$get_prior()`
## ------------------------------------------------

spec = specify_bsvar_hmsh$new(
   data = us_fiscal_lsuw,
   p = 4,
   M = 2
)
spec$get_prior()


## ------------------------------------------------
## Method `BSVARHMSH$get_starting_values()`
## ------------------------------------------------

spec = specify_bsvar_hmsh$new(
   data = us_fiscal_lsuw,
   p = 4,
   M = 2
)
spec$get_starting_values()
```

`specify_bsvar_mix` *R6 Class representing the specification of the BSVAR model with a zero-mean mixture of normals model for structural shocks.*

Description

The class BSVARMIX presents complete specification for the BSVAR model with a zero-mean mixture of normals model for structural shocks.

Super class

`BSVARMSH` -> `BSVARMIX`

Public fields

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

`identification` an object IdentificationBSVARs with the identifying restrictions.

`prior` an object PriorBSVARMIX with the prior specification.

`data_matrices` an object DataMatricesBSVAR with the data matrices.

`starting_values` an object StartingValuesBSVARMIX with the starting values.

`finiteM` a logical value - if true a finite mixture model is estimated. Otherwise, a sparse mixture model is estimated in which `M=20` and the number of visited states is estimated.

Methods

Public methods:

- `BSVARMIX$new()`
- `BSVARMIX$clone()`

`BSVARMIX$new()`: Create a new specification of the BSVAR model with a zero-mean mixture of normals model for structural shocks, BSVARMIX.

Usage:

```
BSVARMIX$new(
  data,
  p = 1L,
  M = 2L,
  B,
  A,
  distribution = c("norm", "t"),
  exogenous = NULL,
  stationary = rep(FALSE, ncol(data)),
  finiteM = TRUE
)
```

Arguments:

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

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

`M` an integer greater than 1 - the number of components of the mixture of normals.

`B` a logical `NxN` matrix containing value `TRUE` for the elements of the structural matrix B to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

`A` a logical `NxK` matrix containing value `TRUE` for the elements of the autoregressive matrix A to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

`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.

`exogenous` a `(T+p)xd` 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 random walk process, otherwise to white noise.

`finiteM` a logical value - if true a finite mixture model is estimated. Otherwise, a sparse mixture model is estimated in which `M=20` and the number of visited states is estimated.

Returns: A new complete specification for the bsvar model with a zero-mean mixture of normals model for structural shocks, BSVARMIX.

`BSVARMIX$clone()`: The objects of this class are cloneable with this method.

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

See Also

`estimate`, `specify_posterior_bsvar_mix`

Examples

```
data(us_fiscal_lsuw)
spec = specify_bsvar_mix$new(
   data = us_fiscal_lsuw,
   p = 4,
   M = 2
)
```

`specify_bsvar_msh` *R6 Class representing the specification of the BSVAR model with Markov Switching Heteroskedasticity.*

Description

The class BSVARMSH presents complete specification for the BSVAR model with Markov Switching Heteroskedasticity.

Public fields

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

`identification` an object IdentificationBSVARs with the identifying restrictions.

`prior` an object PriorBSVARMSH with the prior specification.

`data_matrices` an object DataMatricesBSVAR with the data matrices.

`starting_values` an object StartingValuesBSVARMSH with the starting values.

`finiteM` a logical value - if true a stationary Markov switching model is estimated. Otherwise, a sparse Markov switching model is estimated in which `M=20` and the number of visited states is estimated.

Methods

Public methods:

- `BSVARMSH$new()`
- `BSVARMSH$get_normal()`
- `BSVARMSH$get_data_matrices()`

- BSVARMSH$get_identification()
- BSVARMSH$get_prior()
- BSVARMSH$get_starting_values()
- BSVARMSH$clone()

`BSVARMSH$new()`: Create a new specification of the BSVAR model with Markov Switching Heteroskedasticity, BSVARMSH.

Usage:

```
BSVARMSH$new(
  data,
  p = 1L,
  M = 2L,
  B,
  A,
  distribution = c("norm", "t"),
  exogenous = NULL,
  stationary = rep(FALSE, ncol(data)),
  finiteM = TRUE
)
```

Arguments:

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

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

`M` an integer greater than 1 - the number of Markov process' heteroskedastic regimes.

`B` a logical `NxN` matrix containing value `TRUE` for the elements of the structural matrix B to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

`A` a logical `NxK` matrix containing value `TRUE` for the elements of the autoregressive matrix A to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

`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.

`exogenous` a `(T+p)xd` 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 random walk process, otherwise to white noise.

`finiteM` a logical value - if true a stationary Markov switching model is estimated. Otherwise, a sparse Markov switching model is estimated in which `M=20` and the number of visited states is estimated.

Returns: A new complete specification for the bsvar model with Markov Switching Heteroskedasticity, BSVARMSH.

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

Usage:

```
BSVARMSH$get_normal()
```

Examples:

```
spec = specify_bsvar_msh$new(us_fiscal_lsuw)
spec$get_normal()
```

`BSVARMSH$get_data_matrices()`: Returns the data matrices as the DataMatricesBSVAR object.

Usage:

```
BSVARMSH$get_data_matrices()
```

Examples:

```
data(us_fiscal_lsuw)
spec = specify_bsvar_msh$new(
   data = us_fiscal_lsuw,
   p = 4,
   M = 2
)
spec$get_data_matrices()
```

`BSVARMSH$get_identification()`: Returns the identifying restrictions as the IdentificationBSVARs object.

Usage:

```
BSVARMSH$get_identification()
```

Examples:

```
data(us_fiscal_lsuw)
spec = specify_bsvar_msh$new(
   data = us_fiscal_lsuw,
   p = 4,
   M = 2
)
spec$get_identification()
```

`BSVARMSH$get_prior()`: Returns the prior specification as the PriorBSVARMSH object.

Usage:

```
BSVARMSH$get_prior()
```

Examples:

```
data(us_fiscal_lsuw)
spec = specify_bsvar_msh$new(
   data = us_fiscal_lsuw,
   p = 4,
   M = 2
)
spec$get_prior()
```

`BSVARMSH$get_starting_values()`: Returns the starting values as the StartingValuesBSVARMSH object.

Usage:

```
BSVARMSH$get_starting_values()
```

Examples:

```
data(us_fiscal_lsuw)
spec = specify_bsvar_msh$new(
   data = us_fiscal_lsuw,
   p = 4,
   M = 2
)
spec$get_starting_values()
```

`BSVARMSH$clone()`: The objects of this class are cloneable with this method.

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

See Also

`estimate`, `specify_posterior_bsvar_msh`

Examples

```
data(us_fiscal_lsuw)
spec = specify_bsvar_msh$new(
   data = us_fiscal_lsuw,
   p = 4,
   M = 2
)


## ------------------------------------------------
## Method `BSVARMSH$get_normal()`
## ------------------------------------------------

spec = specify_bsvar_msh$new(us_fiscal_lsuw)
spec$get_normal()


## ------------------------------------------------
## Method `BSVARMSH$get_data_matrices()`
## ------------------------------------------------

data(us_fiscal_lsuw)
spec = specify_bsvar_msh$new(
   data = us_fiscal_lsuw,
   p = 4,
   M = 2
)
spec$get_data_matrices()
```

```
## ------------------------------------------------
## Method `BSVARMSH$get_identification()`
## ------------------------------------------------

data(us_fiscal_lsuw)
spec = specify_bsvar_msh$new(
   data = us_fiscal_lsuw,
   p = 4,
   M = 2
)
spec$get_identification()


## ------------------------------------------------
## Method `BSVARMSH$get_prior()`
## ------------------------------------------------

data(us_fiscal_lsuw)
spec = specify_bsvar_msh$new(
   data = us_fiscal_lsuw,
   p = 4,
   M = 2
)
spec$get_prior()


## ------------------------------------------------
## Method `BSVARMSH$get_starting_values()`
## ------------------------------------------------

data(us_fiscal_lsuw)
spec = specify_bsvar_msh$new(
   data = us_fiscal_lsuw,
   p = 4,
   M = 2
)
spec$get_starting_values()
```

`specify_bsvar_sv` *R6 Class representing the specification of the BSVAR model with Stochastic Volatility heteroskedasticity.*

Description

The class BSVARSV presents complete specification for the BSVAR model with Stochastic Volatility heteroskedasticity.

Public fields

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

`identification` an object IdentificationBSVARs with the identifying restrictions.

`prior` an object PriorBSVARSV with the prior specification.

`data_matrices` an object DataMatricesBSVAR with the data matrices.

`starting_values` an object StartingValuesBSVARSV with the starting values.

`centred_sv` a logical value - if true a centred parameterisation of the Stochastic Volatility process is estimated. Otherwise, its non-centred parameterisation is estimated. See Lütkepohl, Shang, Uzeda, Woźniak (2022) for more info.

Methods

Public methods:

- `BSVARSV$new()`
- `BSVARSV$get_normal()`
- `BSVARSV$get_data_matrices()`
- `BSVARSV$get_identification()`
- `BSVARSV$get_prior()`
- `BSVARSV$get_starting_values()`
- `BSVARSV$clone()`

`BSVARSV$new()`: Create a new specification of the BSVAR model with Stochastic Volatility heteroskedasticity, BSVARSV.

Usage:

```
BSVARSV$new(
  data,
  p = 1L,
  B,
  A,
  distribution = c("norm", "t"),
  exogenous = NULL,
  centred_sv = FALSE,
  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.

`B` a logical `NxN` matrix containing value `TRUE` for the elements of the structural matrix B to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

`A` a logical `NxK` matrix containing value `TRUE` for the elements of the autoregressive matrix A to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

`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.

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

`centred_sv` a logical value. If `FALSE` a non-centred Stochastic Volatility processes for conditional variances are estimated. Otherwise, a centred process is estimated.

`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 random walk process, otherwise to white noise.

Returns: A new complete specification for the bsvar model with Stochastic Volatility heteroskedasticity, BSVARSV.

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

Usage:

```
BSVARSV$get_normal()
```

Examples:

```
spec = specify_bsvar_sv$new(us_fiscal_lsuw)
spec$get_normal()
```

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

Usage:

```
BSVARSV$get_data_matrices()
```

Examples:

```
data(us_fiscal_lsuw)
spec = specify_bsvar_sv$new(
   data = us_fiscal_lsuw,
   p = 4
)
spec$get_data_matrices()
```

`BSVARSV$get_identification()`**:** Returns the identifying restrictions as the IdentificationBSVARs object.

Usage:

```
BSVARSV$get_identification()
```

Examples:

```
data(us_fiscal_lsuw)
spec = specify_bsvar_sv$new(
   data = us_fiscal_lsuw,
   p = 4
)
spec$get_identification()
```

`BSVARSV$get_prior()`**:** Returns the prior specification as the PriorBSVARSV object.

Usage:

```
BSVARSV$get_prior()
```

Examples:

```
data(us_fiscal_lsuw)
spec = specify_bsvar_sv$new(
   data = us_fiscal_lsuw,
   p = 4
)
spec$get_prior()
```

`BSVARSV$get_starting_values()`**:** Returns the starting values as the StartingValuesBSVARSV object.

Usage:

```
BSVARSV$get_starting_values()
```

Examples:

```
data(us_fiscal_lsuw)
spec = specify_bsvar_sv$new(
   data = us_fiscal_lsuw,
   p = 4
)
spec$get_starting_values()
```

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

Usage:

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

Arguments:

deep Whether to make a deep clone.

See Also

estimate, specify_posterior_bsvar_sv

Examples

```
data(us_fiscal_lsuw)
spec = specify_bsvar_sv$new(
   data = us_fiscal_lsuw,
   p = 4
)


## ------------------------------------------------
## Method `BSVARSV$get_normal()`
## ------------------------------------------------

spec = specify_bsvar_sv$new(us_fiscal_lsuw)
spec$get_normal()


## ------------------------------------------------
## Method `BSVARSV$get_data_matrices()`
## ------------------------------------------------
```

```
data(us_fiscal_lsuw)
spec = specify_bsvar_sv$new(
   data = us_fiscal_lsuw,
   p = 4
)
spec$get_data_matrices()


## ------------------------------------------------
## Method `BSVARSV$get_identification()`
## ------------------------------------------------

data(us_fiscal_lsuw)
spec = specify_bsvar_sv$new(
   data = us_fiscal_lsuw,
   p = 4
)
spec$get_identification()


## ------------------------------------------------
## Method `BSVARSV$get_prior()`
## ------------------------------------------------

data(us_fiscal_lsuw)
spec = specify_bsvar_sv$new(
   data = us_fiscal_lsuw,
   p = 4
)
spec$get_prior()


## ------------------------------------------------
## Method `BSVARSV$get_starting_values()`
## ------------------------------------------------

data(us_fiscal_lsuw)
spec = specify_bsvar_sv$new(
   data = us_fiscal_lsuw,
   p = 4
)
spec$get_starting_values()
```

`specify_bsvar_t` *R6 Class representing the specification of the BSVAR model with t-distributed structural shocks.*

Description

The class BSVART presents complete specification for the BSVAR model with t-distributed structural shocks.

Super class

`BSVAR -> BSVART`

Public fields

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

`identification` an object IdentificationBSVARs with the identifying restrictions.

`prior` an object PriorBSVART with the prior specification.

`data_matrices` an object DataMatricesBSVAR with the data matrices.

`starting_values` an object StartingValuesBSVART with the starting values.

`adaptiveMH` a vector of two values setting the Robust Adaptive Metropolis sampler for df: target acceptance rate and adaptive rate.

Methods

Public methods:

- `BSVART$new()`
- `BSVART$clone()`

`BSVART$new()`: Create a new specification of the BSVAR model with t-distributed structural shocks, BSVART.

Usage:

```
BSVART$new(
  data,
  p = 1L,
  B,
  A,
  exogenous = NULL,
  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.

`B` a logical `NxN` matrix containing value `TRUE` for the elements of the structural matrix B to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

`A` a logical `NxK` matrix containing value `TRUE` for the elements of the autoregressive matrix A to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

`exogenous` a `(T+p)xd` 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 random walk process, otherwise to white noise.

Returns: A new complete specification for the bsvar model with t-distributed structural shocks, BSVART.

`BSVART$clone()`: The objects of this class are cloneable with this method.

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

See Also

`estimate`, `specify_posterior_bsvar_t`

Examples

```
data(us_fiscal_lsuw)
spec = specify_bsvar_t$new(
   data = us_fiscal_lsuw,
   p = 4
)
```

`specify_data_matrices` *R6 Class Representing DataMatricesBSVAR*

Description

The class DataMatricesBSVAR presents the data matrices of dependent variables, Y, and regressors, X, for the homoskedastic bsvar model.

Public fields

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

`X` an `KxT` matrix of regressors, X.

Methods

Public methods:

- `DataMatricesBSVAR$new()`
- `DataMatricesBSVAR$get_data_matrices()`
- `DataMatricesBSVAR$clone()`

`DataMatricesBSVAR$new()`: Create new data matrices DataMatricesBSVAR.

Usage:

```
DataMatricesBSVAR$new(data, p = 1L, exogenous = NULL)
```

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. This matrix should not include a constant term.

Returns: New data matrices DataMatricesBSVAR.

`DataMatricesBSVAR$get_data_matrices()`**:** Returns the data matrices DataMatricesBSVAR as a `list`.

Usage:

```
DataMatricesBSVAR$get_data_matrices()
```

Examples:

```
data(us_fiscal_lsuw)
YX = specify_data_matrices$new(data = us_fiscal_lsuw, p = 4)
YX$get_data_matrices()
```

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Examples

```
data(us_fiscal_lsuw)
YX = specify_data_matrices$new(data = us_fiscal_lsuw, p = 4)
dim(YX$Y); dim(YX$X)


## ------------------------------------------------
## Method `DataMatricesBSVAR$get_data_matrices()`
## ------------------------------------------------

data(us_fiscal_lsuw)
YX = specify_data_matrices$new(data = us_fiscal_lsuw, p = 4)
YX$get_data_matrices()
```

`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:

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

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

Usage:

```
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`.

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

Usage:

```
Forecasts$get_forecasts()
```

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

`Forecasts$clone()`: The objects of this class are cloneable with this method.

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Examples

```
spec = specify_bsvar$new(us_fiscal_lsuw)
burn = estimate(spec, 5)
post = estimate(burn, 5)
fore = forecast(post, 4)
apply(fore$forecasts, 1:2, mean) # compute mean forecasts
```

`specify_identification_bsvars`

R6 Class Representing IdentificationBSVARs

Description

The class IdentificationBSVARs presents the identifying restrictions for the bsvar models.

Public fields

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

`VA` a list of `N` matrices determining the unrestricted elements of matrix A.

Methods

Public methods:

- `IdentificationBSVARs$new()`
- `IdentificationBSVARs$get_identification()`
- `IdentificationBSVARs$set_identification()`
- `IdentificationBSVARs$clone()`

`IdentificationBSVARs$new()`: Create new identifying restrictions IdentificationBSVARs.

Usage:

```
IdentificationBSVARs$new(B, A, N, K)
```

Arguments:

`B` a logical `NxN` matrix containing value `TRUE` for the elements of the structural matrix B to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

A a logical NxK matrix containing value TRUE for the elements of the autoregressive matrix A to be estimated and value FALSE for exclusion restrictions to be set to zero.

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

K a positive integer - the number of parameters in a row of autoregressive matrix.

Returns: Identifying restrictions IdentificationBSVARs.

`IdentificationBSVARs$get_identification()`: Returns the elements of the identification pattern IdentificationBSVARs as a `list`.

Usage:

```
IdentificationBSVARs$get_identification()
```

Examples:

```
B    = matrix(c(TRUE,TRUE,TRUE,FALSE,FALSE,TRUE,FALSE,TRUE,TRUE), 3, 3); B
spec = specify_identification_bsvars$new(B = B, N = 3, K = 4)
spec$get_identification()
```

`IdentificationBSVARs$set_identification()`: Set new starting values StartingValuesBSVAR.

Usage:

```
IdentificationBSVARs$set_identification(B, A, N, K)
```

Arguments:

B a logical NxN matrix containing value TRUE for the elements of the structural matrix B to be estimated and value FALSE for exclusion restrictions to be set to zero.

A a logical NxK matrix containing value TRUE for the elements of the autoregressive matrix A to be estimated and value FALSE for exclusion restrictions to be set to zero.

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

K a positive integer - the number of parameters in a row of autoregressive matrix.

Examples:

```
spec = specify_identification_bsvars$new(N = 3, K = 4) # specify a model with the default option
B    = matrix(c(TRUE,TRUE,TRUE,FALSE,FALSE,TRUE,FALSE,TRUE,TRUE), 3, 3); B
spec$set_identification(B = B, N = 3, K = 4)  # modify an existing specification
spec$get_identification()              # check the outcome
```

`IdentificationBSVARs$clone()`: The objects of this class are cloneable with this method.

Usage:

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

Arguments:

deep Whether to make a deep clone.

Examples

```
specify_identification_bsvars$new(N = 3, K = 4) # recursive specification for a 3-variable system

B = matrix(c(TRUE,TRUE,TRUE,FALSE,FALSE,TRUE,FALSE,TRUE,TRUE), 3, 3); B
specify_identification_bsvars$new(B = B, N = 3, K = 4) # an alternative identification pattern
```

```
## ------------------------------------------------
## Method `IdentificationBSVARs$get_identification()`
## ------------------------------------------------

B    = matrix(c(TRUE,TRUE,TRUE,FALSE,FALSE,TRUE,FALSE,TRUE,TRUE), 3, 3); B
spec = specify_identification_bsvars$new(B = B, N = 3, K = 4)
spec$get_identification()


## ------------------------------------------------
## Method `IdentificationBSVARs$set_identification()`
## ------------------------------------------------

spec = specify_identification_bsvars$new(N = 3, K = 4) # specify a model with the default option
B    = matrix(c(TRUE,TRUE,TRUE,FALSE,FALSE,TRUE,FALSE,TRUE,TRUE), 3, 3); B
spec$set_identification(B = B, N = 3, K = 4)  # modify an existing specification
spec$get_identification()              # check the outcome
```

`specify_posterior_bsvar` *R6 Class Representing PosteriorBSVAR*

Description

The class PosteriorBSVAR contains posterior output and the specification including the last MCMC draw for the homoskedastic bsvar 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 BSVAR 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 an `NxNxS` array `B`, an `NxKxS` array `A`, and a `5xS` matrix `hyper`.

Methods

Public methods:

- `PosteriorBSVAR$new()`
- `PosteriorBSVAR$get_posterior()`
- `PosteriorBSVAR$get_last_draw()`
- `PosteriorBSVAR$is_normalised()`
- `PosteriorBSVAR$set_normalised()`
- `PosteriorBSVAR$clone()`

`PosteriorBSVAR$new()`: Create a new posterior output PosteriorBSVAR.

Usage:

```
PosteriorBSVAR$new(specification_bsvar, posterior_bsvar)
```

Arguments:

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

`posterior_bsvar` 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 PosteriorBSVAR.

`PosteriorBSVAR$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:

```
PosteriorBSVAR$get_posterior()
```

Examples:

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

`PosteriorBSVAR$get_last_draw()`**:** Returns an object of class BSVAR 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:

```
PosteriorBSVAR$get_last_draw()
```

Examples:

```
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar$new(us_fiscal_lsuw, p = 4)
set.seed(123)

# run the burn-in
burn_in        = estimate(specification, 10)

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

`PosteriorBSVAR$is_normalised()`**:** Returns `TRUE` if the posterior has been normalised using `normalise()` and `FALSE` otherwise.

Usage:

```
PosteriorBSVAR$is_normalised()
```

Examples:

```
# upload data
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar$new(us_fiscal_lsuw, p = 4)
set.seed(123)

# estimate the model
posterior      = estimate(specification, 10, thin = 1)

# check normalisation status beforehand
posterior$is_normalised()

# normalise the posterior
BB         = posterior$last_draw$starting_values$B    # get the last draw of B
B_hat      = diag((-1) * sign(diag(BB))) %*% BB      # set negative diagonal elements
posterior    = normalise(posterior, B_hat)           # draws in posterior are normalised

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

`PosteriorBSVAR$set_normalised()`**:** Sets the private indicator `normalised` to TRUE.

Usage:

```
PosteriorBSVAR$set_normalised(value)
```

Arguments:

`value` (optional) a logical value to be passed to indicator `normalised`.

Examples:

```
# This is an internal function that is run while executing normalise()
# Observe its working by analysing the workflow:

# upload data
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar$new(us_fiscal_lsuw, p = 4)
set.seed(123)

# estimate the model
posterior      = estimate(specification, 10, thin = 1)

# check normalisation status beforehand
posterior$is_normalised()

# normalise the posterior
BB         = posterior$last_draw$starting_values$B    # get the last draw of B
B_hat      = diag(sign(diag(BB))) %*% BB           # set positive diagonal elements
posterior    = normalise(posterior, B_hat)           # draws in posterior are normalised
```

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

`PosteriorBSVAR$clone()`: The objects of this class are cloneable with this method.

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

See Also

`estimate`, `specify_bsvar`

Examples

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


## ------------------------------------------------
## Method `PosteriorBSVAR$get_posterior()`
## ------------------------------------------------

data(us_fiscal_lsuw)
specification  = specify_bsvar$new(us_fiscal_lsuw)
set.seed(123)
estimate       = estimate(specification, 50)
estimate$get_posterior()


## ------------------------------------------------
## Method `PosteriorBSVAR$get_last_draw()`
## ------------------------------------------------

data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar$new(us_fiscal_lsuw, p = 4)
set.seed(123)

# run the burn-in
burn_in        = estimate(specification, 10)

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

```
## ------------------------------------------------
## Method `PosteriorBSVAR$is_normalised()`
## ------------------------------------------------

# upload data
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar$new(us_fiscal_lsuw, p = 4)
set.seed(123)

# estimate the model
posterior      = estimate(specification, 10, thin = 1)

# check normalisation status beforehand
posterior$is_normalised()

# normalise the posterior
BB             = posterior$last_draw$starting_values$B      # get the last draw of B
B_hat        = diag((-1) * sign(diag(BB))) %*% BB        # set negative diagonal elements
posterior    = normalise(posterior, B_hat)             # draws in posterior are normalised

# check normalisation status afterwards
posterior$is_normalised()


## ------------------------------------------------
## Method `PosteriorBSVAR$set_normalised()`
## ------------------------------------------------

# This is an internal function that is run while executing normalise()
# Observe its working by analysing the workflow:

# upload data
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar$new(us_fiscal_lsuw, p = 4)
set.seed(123)

# estimate the model
posterior      = estimate(specification, 10, thin = 1)

# check normalisation status beforehand
posterior$is_normalised()

# normalise the posterior
BB             = posterior$last_draw$starting_values$B      # get the last draw of B
B_hat        = diag(sign(diag(BB))) %*% BB               # set positive diagonal elements
posterior    = normalise(posterior, B_hat)             # draws in posterior are normalised

# check normalisation status afterwards
```

```
posterior$is_normalised()
```

`specify_posterior_bsvar_exh` *R6 Class Representing PosteriorBSVAREXH*

Description

The class PosteriorBSVAREXH contains posterior output and the specification including the last MCMC draw for the bsvar model with exogenous heteroskedastic regime changes. 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 BSVAREXH 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.

Methods

Public methods:

- `PosteriorBSVAREXH$new()`
- `PosteriorBSVAREXH$get_posterior()`
- `PosteriorBSVAREXH$get_last_draw()`
- `PosteriorBSVAREXH$is_normalised()`
- `PosteriorBSVAREXH$set_normalised()`
- `PosteriorBSVAREXH$clone()`

`PosteriorBSVAREXH$new()`**:** Create a new posterior output PosteriorBSVAREXH

Usage:

```
PosteriorBSVAREXH$new(specification_bsvar, posterior_bsvar)
```

Arguments:

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

`posterior_bsvar` a list containing Bayesian estimation output.

Returns: A posterior output PosteriorBSVAREXH

`PosteriorBSVAREXH$get_posterior()`**:** Returns a list containing Bayesian estimation output.

Usage:

```
PosteriorBSVAREXH$get_posterior()
```

Examples:

```
spec   = specify_bsvar_exh$new(us_fiscal_lsuw)
post   = estimate(spec, 10, thin = 1)
post$get_posterior()
```

`PosteriorBSVAREXH$get_last_draw()`**:** Returns an object of class BSVAREXH 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:

```
PosteriorBSVAREXH$get_last_draw()
```

Examples:

```
# specify the model
spec   = specify_bsvar_exh$new(us_fiscal_lsuw)

# run the burn-in
burn   = estimate(spec, 10, thin = 2)

# estimate the model
post   = estimate(burn, 10, thin = 2)
```

`PosteriorBSVAREXH$is_normalised()`**:** Returns `TRUE` if the posterior has been normalised using `normalise()` and `FALSE` otherwise.

Usage:

```
PosteriorBSVAREXH$is_normalised()
```

Examples:

```
# specify the model
spec   = specify_bsvar_exh$new(us_fiscal_lsuw)

# estimate the model
post   = estimate(spec, 10, thin = 1)

# check normalisation status beforehand
post$is_normalised()

# normalise the posterior
BB           = post$last_draw$starting_values$B      # get the last draw of B
B_hat      = diag((-1) * sign(diag(BB))) %*% BB   # set negative diagonal elements
post       = normalise(post, B_hat)           # draws in posterior are normalised

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

`PosteriorBSVAREXH$set_normalised()`**:** Sets the private indicator `normalised` to TRUE.

Usage:

```
PosteriorBSVAREXH$set_normalised(value)
```

Arguments:

`value` (optional) a logical value to be passed to indicator `normalised`.

Examples:

```
# This is an internal function that is run while executing normalise()
# Observe its working by analysing the workflow:

spec   = specify_bsvar_exh$new(us_fiscal_lsuw)
post   = estimate(spec, 10, thin = 1)

# check normalisation status beforehand
post$is_normalised()

# normalise the posterior
BB           = post$last_draw$starting_values$B      # get the last draw of B
B_hat      = diag(sign(diag(BB))) %*% BB       # set positive diagonal elements
post       = normalise(post, B_hat)           # draws in posterior are normalised

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

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

Usage:

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

Arguments:

deep Whether to make a deep clone.

See Also

estimate, specify_bsvar_exh

Examples

```
# This is a function that is used within estimate()
spec   = specify_bsvar_exh$new(us_fiscal_lsuw)
post   = estimate(spec, 10, thin = 1)
class(post)


## ------------------------------------------------
## Method `PosteriorBSVAREXH$get_posterior()`
## ------------------------------------------------

spec   = specify_bsvar_exh$new(us_fiscal_lsuw)
post   = estimate(spec, 10, thin = 1)
post$get_posterior()


## ------------------------------------------------
## Method `PosteriorBSVAREXH$get_last_draw()`
## ------------------------------------------------
```

```
# specify the model
spec   = specify_bsvar_exh$new(us_fiscal_lsuw)

# run the burn-in
burn   = estimate(spec, 10, thin = 2)

# estimate the model
post   = estimate(burn, 10, thin = 2)


## ------------------------------------------------
## Method `PosteriorBSVAREXH$is_normalised()`
## ------------------------------------------------

# specify the model
spec   = specify_bsvar_exh$new(us_fiscal_lsuw)

# estimate the model
post   = estimate(spec, 10, thin = 1)

# check normalisation status beforehand
post$is_normalised()

# normalise the posterior
BB            = post$last_draw$starting_values$B      # get the last draw of B
B_hat         = diag((-1) * sign(diag(BB))) %*% BB    # set negative diagonal elements
post          = normalise(post, B_hat)               # draws in posterior are normalised

# check normalisation status afterwards
post$is_normalised()


## ------------------------------------------------
## Method `PosteriorBSVAREXH$set_normalised()`
## ------------------------------------------------

# This is an internal function that is run while executing normalise()
# Observe its working by analysing the workflow:

spec   = specify_bsvar_exh$new(us_fiscal_lsuw)
post   = estimate(spec, 10, thin = 1)

# check normalisation status beforehand
post$is_normalised()

# normalise the posterior
BB            = post$last_draw$starting_values$B      # get the last draw of B
B_hat         = diag(sign(diag(BB))) %*% BB           # set positive diagonal elements
post          = normalise(post, B_hat)               # draws in posterior are normalised

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

specify_posterior_bsvar_hmsh
R6 Class Representing PosteriorBSVARHMSH

Description

The class PosteriorBSVARHMSH contains posterior output and the specification including the last MCMC draw for the bsvar model with Hetrogeneous Markov Switching Heteroskedasticity. 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 BSVARHMSH 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.

Methods

Public methods:

- `PosteriorBSVARHMSH$new()`
- `PosteriorBSVARHMSH$get_posterior()`
- `PosteriorBSVARHMSH$get_last_draw()`
- `PosteriorBSVARHMSH$is_normalised()`
- `PosteriorBSVARHMSH$set_normalised()`
- `PosteriorBSVARHMSH$clone()`

`PosteriorBSVARHMSH$new()`**:** Create a new posterior output PosteriorBSVARHMSH.

Usage:

```
PosteriorBSVARHMSH$new(specification_bsvar, posterior_bsvar)
```

Arguments:

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

`posterior_bsvar` a list containing Bayesian estimation output.

Returns: A posterior output PosteriorBSVARHMSH.

`PosteriorBSVARHMSH$get_posterior()`**:** Returns a list containing Bayesian estimation output.

Usage:

```
PosteriorBSVARHMSH$get_posterior()
```

Examples:

```
specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw, M = 2)
set.seed(123)
estimate       = estimate(specification, 10)
estimate$get_posterior()
```

`PosteriorBSVARHMSH$get_last_draw()`: Returns an object of class BSVARHMSH 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:

```
PosteriorBSVARHMSH$get_last_draw()
```

Examples:

```
# specify the model and set seed
specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw, p = 4, M = 2)

# run the burn-in
set.seed(123)
burn_in        = estimate(specification, 5)

# estimate the model
posterior      = estimate(burn_in, 5)
```

`PosteriorBSVARHMSH$is_normalised()`: Returns `TRUE` if the posterior has been normalised using `normalise()` and `FALSE` otherwise.

Usage:

```
PosteriorBSVARHMSH$is_normalised()
```

Examples:

```
# specify the model and set seed
specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw, p = 4, M = 2)

# estimate the model
posterior      = estimate(specification, 5)

# check normalisation status beforehand
posterior$is_normalised()

# normalise the posterior
BB         = posterior$last_draw$starting_values$B    # get the last draw of B
B_hat      = diag((-1) * sign(diag(BB))) %*% BB      # set negative diagonal elements
posterior    = normalise(posterior, B_hat)           # draws in posterior are normalised

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

`PosteriorBSVARHMSH$set_normalised()`: Sets the private indicator `normalised` to TRUE.

Usage:

```
PosteriorBSVARHMSH$set_normalised(value)
```

Arguments:

`value` (optional) a logical value to be passed to indicator `normalised`.

Examples:

```
# This is an internal function that is run while executing normalise()
# Observe its working by analysing the workflow:

# specify the model and set seed
specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw, p = 4, M = 2)
set.seed(123)

# estimate the model
posterior      = estimate(specification, 5)

# check normalisation status beforehand
posterior$is_normalised()

# normalise the posterior
BB         = posterior$last_draw$starting_values$B     # get the last draw of B
B_hat      = diag(sign(diag(BB))) %*% BB           # set positive diagonal elements
posterior    = normalise(posterior, B_hat)           # draws in posterior are normalised

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

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

Usage:

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

Arguments:

deep Whether to make a deep clone.

See Also

estimate, specify_bsvar_hmsh

Examples

```
# This is a function that is used within estimate()
specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw, p = 4, M = 2)
set.seed(123)
estimate       = estimate(specification, 10, thin = 1)
class(estimate)


## ------------------------------------------------
## Method `PosteriorBSVARHMSH$get_posterior()`
## ------------------------------------------------

specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw, M = 2)
set.seed(123)
estimate       = estimate(specification, 10)
estimate$get_posterior()
```

```
## ------------------------------------------------
## Method `PosteriorBSVARHMSH$get_last_draw()`
## ------------------------------------------------

# specify the model and set seed
specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw, p = 4, M = 2)

# run the burn-in
set.seed(123)
burn_in        = estimate(specification, 5)

# estimate the model
posterior      = estimate(burn_in, 5)


## ------------------------------------------------
## Method `PosteriorBSVARHMSH$is_normalised()`
## ------------------------------------------------

# specify the model and set seed
specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw, p = 4, M = 2)

# estimate the model
posterior      = estimate(specification, 5)

# check normalisation status beforehand
posterior$is_normalised()

# normalise the posterior
BB            = posterior$last_draw$starting_values$B      # get the last draw of B
B_hat        = diag((-1) * sign(diag(BB))) %*% BB        # set negative diagonal elements
posterior    = normalise(posterior, B_hat)             # draws in posterior are normalised

# check normalisation status afterwards
posterior$is_normalised()


## ------------------------------------------------
## Method `PosteriorBSVARHMSH$set_normalised()`
## ------------------------------------------------

# This is an internal function that is run while executing normalise()
# Observe its working by analysing the workflow:

# specify the model and set seed
specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw, p = 4, M = 2)
set.seed(123)

# estimate the model
posterior      = estimate(specification, 5)

# check normalisation status beforehand
```

```
posterior$is_normalised()

# normalise the posterior
BB            = posterior$last_draw$starting_values$B      # get the last draw of B
B_hat         = diag(sign(diag(BB))) %*% BB              # set positive diagonal elements
posterior     = normalise(posterior, B_hat)            # draws in posterior are normalised

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

`specify_posterior_bsvar_mix`

R6 Class Representing PosteriorBSVARMIX

Description

The class PosteriorBSVARMIX contains posterior output and the specification including the last MCMC draw for the bsvar model with a zero-mean mixture of normals model for structural shocks. 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 BSVARMIX 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.

Methods

Public methods:

- `PosteriorBSVARMIX$new()`
- `PosteriorBSVARMIX$get_posterior()`
- `PosteriorBSVARMIX$get_last_draw()`
- `PosteriorBSVARMIX$is_normalised()`
- `PosteriorBSVARMIX$set_normalised()`
- `PosteriorBSVARMIX$clone()`

`PosteriorBSVARMIX$new()`**:** Create a new posterior output PosteriorBSVARMIX.

Usage:

```
PosteriorBSVARMIX$new(specification_bsvar, posterior_bsvar)
```

Arguments:

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

`posterior_bsvar` a list containing Bayesian estimation output.

Returns: A posterior output PosteriorBSVARMIX.

`PosteriorBSVARMIX$get_posterior()`**:** Returns a list containing Bayesian estimation output.

Usage:

```
PosteriorBSVARMIX$get_posterior()
```

Examples:

```
data(us_fiscal_lsuw)
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, M = 2)
set.seed(123)
estimate       = estimate(specification, 10, thin = 1)
estimate$get_posterior()
```

`PosteriorBSVARMIX$get_last_draw()`**:** Returns an object of class BSVARMIX 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:

```
PosteriorBSVARMIX$get_last_draw()
```

Examples:

```
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, p = 4, M = 2)

# run the burn-in
set.seed(123)
burn_in        = estimate(specification, 10, thin = 2)

# estimate the model
posterior      = estimate(burn_in, 10, thin = 2)
```

`PosteriorBSVARMIX$is_normalised()`**:** Returns `TRUE` if the posterior has been normalised using `normalise()` and `FALSE` otherwise.

Usage:

```
PosteriorBSVARMIX$is_normalised()
```

Examples:

```
# upload data
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, p = 4, M = 2)

# estimate the model
set.seed(123)
posterior      = estimate(specification, 10, thin = 1)

# check normalisation status beforehand
```

```
posterior$is_normalised()

# normalise the posterior
BB         = posterior$last_draw$starting_values$B    # get the last draw of B
B_hat      = diag((-1) * sign(diag(BB))) %*% BB      # set negative diagonal elements
posterior    = normalise(posterior, B_hat)           # draws in posterior are normalised

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

`PosteriorBSVARMIX$set_normalised()`**:** Sets the private indicator `normalised` to TRUE.

Usage:

```
PosteriorBSVARMIX$set_normalised(value)
```

Arguments:

`value` (optional) a logical value to be passed to indicator `normalised`.

Examples:

```
# This is an internal function that is run while executing normalise()
# Observe its working by analysing the workflow:

# upload data
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, p = 4, M = 2)
set.seed(123)

# estimate the model
posterior      = estimate(specification, 10, thin = 1)

# check normalisation status beforehand
posterior$is_normalised()

# normalise the posterior
BB         = posterior$last_draw$starting_values$B    # get the last draw of B
B_hat      = diag(sign(diag(BB))) %*% BB           # set positive diagonal elements
posterior    = normalise(posterior, B_hat)           # draws in posterior are normalised

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

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

See Also

estimate, specify_bsvar_mix

Examples

```
# This is a function that is used within estimate()
data(us_fiscal_lsuw)
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, p = 4, M = 2)
set.seed(123)
estimate       = estimate(specification, 10, thin = 1)
class(estimate)


## ------------------------------------------------
## Method `PosteriorBSVARMIX$get_posterior()`
## ------------------------------------------------

data(us_fiscal_lsuw)
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, M = 2)
set.seed(123)
estimate       = estimate(specification, 10, thin = 1)
estimate$get_posterior()


## ------------------------------------------------
## Method `PosteriorBSVARMIX$get_last_draw()`
## ------------------------------------------------

data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, p = 4, M = 2)

# run the burn-in
set.seed(123)
burn_in        = estimate(specification, 10, thin = 2)

# estimate the model
posterior      = estimate(burn_in, 10, thin = 2)


## ------------------------------------------------
## Method `PosteriorBSVARMIX$is_normalised()`
## ------------------------------------------------

# upload data
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, p = 4, M = 2)

# estimate the model
```

```
set.seed(123)
posterior      = estimate(specification, 10, thin = 1)

# check normalisation status beforehand
posterior$is_normalised()

# normalise the posterior
BB             = posterior$last_draw$starting_values$B      # get the last draw of B
B_hat         = diag((-1) * sign(diag(BB))) %*% BB        # set negative diagonal elements
posterior     = normalise(posterior, B_hat)             # draws in posterior are normalised

# check normalisation status afterwards
posterior$is_normalised()


## ------------------------------------------------
## Method `PosteriorBSVARMIX$set_normalised()`
## ------------------------------------------------

# This is an internal function that is run while executing normalise()
# Observe its working by analysing the workflow:

# upload data
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, p = 4, M = 2)
set.seed(123)

# estimate the model
posterior      = estimate(specification, 10, thin = 1)

# check normalisation status beforehand
posterior$is_normalised()

# normalise the posterior
BB             = posterior$last_draw$starting_values$B      # get the last draw of B
B_hat         = diag(sign(diag(BB))) %*% BB               # set positive diagonal elements
posterior     = normalise(posterior, B_hat)             # draws in posterior are normalised

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

---

specify_posterior_bsvar_msh

R6 Class Representing PosteriorBSVARMSH

Description

The class PosteriorBSVARMSH contains posterior output and the specification including the last MCMC draw for the bsvar model with Markov Switching Heteroskedasticity. 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 BSVARMSH 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.

Methods

Public methods:

- `PosteriorBSVARMSH$new()`
- `PosteriorBSVARMSH$get_posterior()`
- `PosteriorBSVARMSH$get_last_draw()`
- `PosteriorBSVARMSH$is_normalised()`
- `PosteriorBSVARMSH$set_normalised()`
- `PosteriorBSVARMSH$clone()`

`PosteriorBSVARMSH$new()`**:** Create a new posterior output PosteriorBSVARMSH.

Usage:

```
PosteriorBSVARMSH$new(specification_bsvar, posterior_bsvar)
```

Arguments:

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

`posterior_bsvar` a list containing Bayesian estimation output.

Returns: A posterior output PosteriorBSVARMSH.

`PosteriorBSVARMSH$get_posterior()`**:** Returns a list containing Bayesian estimation output.

Usage:

```
PosteriorBSVARMSH$get_posterior()
```

Examples:

```
data(us_fiscal_lsuw)
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, M = 2)
set.seed(123)
estimate       = estimate(specification, 10, thin = 1)
estimate$get_posterior()
```

`PosteriorBSVARMSH$get_last_draw()`**:** Returns an object of class BSVARMSH 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:

```
PosteriorBSVARMSH$get_last_draw()
```

Examples:

```
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, p = 4, M = 2)

# run the burn-in
set.seed(123)
burn_in        = estimate(specification, 10, thin = 2)

# estimate the model
posterior      = estimate(burn_in, 10, thin = 2)
```

`PosteriorBSVARMSH$is_normalised()`**:** Returns `TRUE` if the posterior has been normalised using `normalise()` and `FALSE` otherwise.

Usage:

```
PosteriorBSVARMSH$is_normalised()
```

Examples:

```
# upload data
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, p = 4, M = 2)

# estimate the model
set.seed(123)
posterior      = estimate(specification, 10, thin = 1)

# check normalisation status beforehand
posterior$is_normalised()

# normalise the posterior
BB         = posterior$last_draw$starting_values$B    # get the last draw of B
B_hat      = diag((-1) * sign(diag(BB))) %*% BB      # set negative diagonal elements
posterior    = normalise(posterior, B_hat)           # draws in posterior are normalised

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

`PosteriorBSVARMSH$set_normalised()`**:** Sets the private indicator `normalised` to TRUE.

Usage:

```
PosteriorBSVARMSH$set_normalised(value)
```

Arguments:

`value` (optional) a logical value to be passed to indicator `normalised`.

Examples:

```
# This is an internal function that is run while executing normalise()
# Observe its working by analysing the workflow:

# upload data
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, p = 4, M = 2)
set.seed(123)

# estimate the model
posterior      = estimate(specification, 10, thin = 1)

# check normalisation status beforehand
posterior$is_normalised()

# normalise the posterior
BB         = posterior$last_draw$starting_values$B    # get the last draw of B
B_hat      = diag(sign(diag(BB))) %*% BB           # set positive diagonal elements
posterior    = normalise(posterior, B_hat)           # draws in posterior are normalised

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

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

Usage:

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

Arguments:

deep Whether to make a deep clone.

See Also

estimate, specify_bsvar_msh

Examples

```
# This is a function that is used within estimate()
data(us_fiscal_lsuw)
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, p = 4, M = 2)
set.seed(123)
estimate       = estimate(specification, 10, thin = 1)
class(estimate)


## ------------------------------------------------
## Method `PosteriorBSVARMSH$get_posterior()`
## ------------------------------------------------
```

```
data(us_fiscal_lsuw)
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, M = 2)
set.seed(123)
estimate       = estimate(specification, 10, thin = 1)
estimate$get_posterior()


## ------------------------------------------------
## Method `PosteriorBSVARMSH$get_last_draw()`
## ------------------------------------------------

data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, p = 4, M = 2)

# run the burn-in
set.seed(123)
burn_in        = estimate(specification, 10, thin = 2)

# estimate the model
posterior      = estimate(burn_in, 10, thin = 2)


## ------------------------------------------------
## Method `PosteriorBSVARMSH$is_normalised()`
## ------------------------------------------------

# upload data
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, p = 4, M = 2)

# estimate the model
set.seed(123)
posterior      = estimate(specification, 10, thin = 1)

# check normalisation status beforehand
posterior$is_normalised()

# normalise the posterior
BB             = posterior$last_draw$starting_values$B      # get the last draw of B
B_hat        = diag((-1) * sign(diag(BB))) %*% BB        # set negative diagonal elements
posterior    = normalise(posterior, B_hat)            # draws in posterior are normalised

# check normalisation status afterwards
posterior$is_normalised()


## ------------------------------------------------
## Method `PosteriorBSVARMSH$set_normalised()`
## ------------------------------------------------
```

```
# This is an internal function that is run while executing normalise()
# Observe its working by analysing the workflow:

# upload data
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, p = 4, M = 2)
set.seed(123)

# estimate the model
posterior      = estimate(specification, 10, thin = 1)

# check normalisation status beforehand
posterior$is_normalised()

# normalise the posterior
BB             = posterior$last_draw$starting_values$B      # get the last draw of B
B_hat         = diag(sign(diag(BB))) %*% BB               # set positive diagonal elements
posterior     = normalise(posterior, B_hat)             # draws in posterior are normalised

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

`specify_posterior_bsvar_sv`

R6 Class Representing PosteriorBSVARSV

Description

The class PosteriorBSVARSV contains posterior output and the specification including the last MCMC draw for the bsvar model with Stochastic Volatility heteroskedasticity. 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 BSVARSV 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.

Methods

Public methods:

- `PosteriorBSVARSV$new()`
- `PosteriorBSVARSV$get_posterior()`

- PosteriorBSVARSV$get_last_draw()
- PosteriorBSVARSV$is_normalised()
- PosteriorBSVARSV$set_normalised()
- PosteriorBSVARSV$clone()

`PosteriorBSVARSV$new()`: Create a new posterior output PosteriorBSVARSV.

Usage:

```
PosteriorBSVARSV$new(specification_bsvar, posterior_bsvar)
```

Arguments:

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

`posterior_bsvar` a list containing Bayesian estimation output.

Returns: A posterior output PosteriorBSVARSV.

`PosteriorBSVARSV$get_posterior()`: Returns a list containing Bayesian estimation.

Usage:

```
PosteriorBSVARSV$get_posterior()
```

Examples:

```
data(us_fiscal_lsuw)
specification  = specify_bsvar_sv$new(us_fiscal_lsuw)
set.seed(123)
estimate       = estimate(specification, 5, thin = 1)
estimate$get_posterior()
```

`PosteriorBSVARSV$get_last_draw()`: Returns an object of class BSVARSV 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:

```
PosteriorBSVARSV$get_last_draw()
```

Examples:

```
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_sv$new(us_fiscal_lsuw, p = 4)
set.seed(123)

# run the burn-in
burn_in        = estimate(specification, 5, thin = 1)

# estimate the model
posterior      = estimate(burn_in, 5, thin = 1)
```

`PosteriorBSVARSV$is_normalised()`: Returns `TRUE` if the posterior has been normalised using `normalise()` and `FALSE` otherwise.

Usage:

```
PosteriorBSVARSV$is_normalised()
```

Examples:

```
# upload data
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_sv$new(us_fiscal_lsuw, p = 4)

# estimate the model
set.seed(123)
posterior      = estimate(specification, 5, thin = 1)

# check normalisation status beforehand
posterior$is_normalised()

# normalise the posterior
BB         = posterior$last_draw$starting_values$B    # get the last draw of B
B_hat      = diag((-1) * sign(diag(BB))) %*% BB      # set negative diagonal elements
posterior    = normalise(posterior, B_hat)           # draws in posterior are normalised

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

`PosteriorBSVARSV$set_normalised()`**:** Sets the private indicator `normalised` to TRUE.

Usage:

```
PosteriorBSVARSV$set_normalised(value)
```

Arguments:

`value` (optional) a logical value to be passed to indicator `normalised`.

Examples:

```
# This is an internal function that is run while executing normalise()
# Observe its working by analysing the workflow:

# upload data
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_sv$new(us_fiscal_lsuw, p = 4)

# estimate the model
set.seed(123)
posterior      = estimate(specification, 5, thin = 1)

# check normalisation status beforehand
posterior$is_normalised()

# normalise the posterior
BB         = posterior$last_draw$starting_values$B    # get the last draw of B
```

```
B_hat      = diag(sign(diag(BB))) %*% BB          # set positive diagonal elements
posterior    = normalise(posterior, B_hat)          # draws in posterior are normalised

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

`PosteriorBSVARSV$clone()`: The objects of this class are cloneable with this method.

Usage:

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

Arguments:

deep Whether to make a deep clone.

See Also

estimate, specify_bsvar_sv

Examples

```
# This is a function that is used within estimate()
data(us_fiscal_lsuw)
specification  = specify_bsvar_sv$new(us_fiscal_lsuw, p = 4)
set.seed(123)
estimate       = estimate(specification, 5, thin = 1)
class(estimate)


## ------------------------------------------------
## Method `PosteriorBSVARSV$get_posterior()`
## ------------------------------------------------

data(us_fiscal_lsuw)
specification  = specify_bsvar_sv$new(us_fiscal_lsuw)
set.seed(123)
estimate       = estimate(specification, 5, thin = 1)
estimate$get_posterior()


## ------------------------------------------------
## Method `PosteriorBSVARSV$get_last_draw()`
## ------------------------------------------------

data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_sv$new(us_fiscal_lsuw, p = 4)
set.seed(123)

# run the burn-in
burn_in        = estimate(specification, 5, thin = 1)

# estimate the model
```

```
posterior      = estimate(burn_in, 5, thin = 1)


## ------------------------------------------------
## Method `PosteriorBSVARSV$is_normalised()`
## ------------------------------------------------

# upload data
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_sv$new(us_fiscal_lsuw, p = 4)

# estimate the model
set.seed(123)
posterior      = estimate(specification, 5, thin = 1)

# check normalisation status beforehand
posterior$is_normalised()

# normalise the posterior
BB             = posterior$last_draw$starting_values$B      # get the last draw of B
B_hat        = diag((-1) * sign(diag(BB))) %*% BB        # set negative diagonal elements
posterior    = normalise(posterior, B_hat)             # draws in posterior are normalised

# check normalisation status afterwards
posterior$is_normalised()


## ------------------------------------------------
## Method `PosteriorBSVARSV$set_normalised()`
## ------------------------------------------------

# This is an internal function that is run while executing normalise()
# Observe its working by analysing the workflow:

# upload data
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_sv$new(us_fiscal_lsuw, p = 4)

# estimate the model
set.seed(123)
posterior      = estimate(specification, 5, thin = 1)

# check normalisation status beforehand
posterior$is_normalised()

# normalise the posterior
BB             = posterior$last_draw$starting_values$B      # get the last draw of B
B_hat        = diag(sign(diag(BB))) %*% BB               # set positive diagonal elements
posterior    = normalise(posterior, B_hat)             # draws in posterior are normalised
```

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

`specify_posterior_bsvar_t` *R6 Class Representing PosteriorBSVART*

Description

The class PosteriorBSVART contains posterior output and the specification including the last MCMC draw for the bsvar model with t-distributed structural shocks. 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 BSVART 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.

Methods

Public methods:

- `PosteriorBSVART$new()`
- `PosteriorBSVART$get_posterior()`
- `PosteriorBSVART$get_last_draw()`
- `PosteriorBSVART$is_normalised()`
- `PosteriorBSVART$set_normalised()`
- `PosteriorBSVART$clone()`

`PosteriorBSVART$new()`**:** Create a new posterior output PosteriorBSVART.

Usage:

```
PosteriorBSVART$new(specification_bsvar, posterior_bsvar)
```

Arguments:

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

`posterior_bsvar` a list containing Bayesian estimation output.

Returns: A posterior output PosteriorBSVART.

`PosteriorBSVART$get_posterior()`**:** Returns a list containing Bayesian estimation output.

Usage:

```
PosteriorBSVART$get_posterior()
```

Examples:

```
data(us_fiscal_lsuw)
specification  = specify_bsvar_t$new(us_fiscal_lsuw)
set.seed(123)
estimate       = estimate(specification, 10)
estimate$get_posterior()
```

`PosteriorBSVART$get_last_draw()`**:** Returns an object of class BSVART 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:

```
PosteriorBSVART$get_last_draw()
```

Examples:

```
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_t$new(us_fiscal_lsuw, p = 4)

# run the burn-in
set.seed(123)
burn_in        = estimate(specification, 10)

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

`PosteriorBSVART$is_normalised()`**:** Returns `TRUE` if the posterior has been normalised using `normalise()` and `FALSE` otherwise.

Usage:

```
PosteriorBSVART$is_normalised()
```

Examples:

```
# upload data
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_t$new(us_fiscal_lsuw, p = 4)

# estimate the model
set.seed(123)
posterior      = estimate(specification, 10)

# check normalisation status beforehand
posterior$is_normalised()

# normalise the posterior
BB         = posterior$last_draw$starting_values$B     # get the last draw of B
B_hat      = diag((-1) * sign(diag(BB))) %*% BB      # set negative diagonal elements
```

```
posterior    = normalise(posterior, B_hat)           # draws in posterior are normalised

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

`PosteriorBSVART$set_normalised()`: Sets the private indicator `normalised` to TRUE.

Usage:

```
PosteriorBSVART$set_normalised(value)
```

Arguments:

`value` (optional) a logical value to be passed to indicator `normalised`.

Examples:

```
# This is an internal function that is run while executing normalise()
# Observe its working by analysing the workflow:

# upload data
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_t$new(us_fiscal_lsuw, p = 4)
set.seed(123)

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

# check normalisation status beforehand
posterior$is_normalised()

# normalise the posterior
BB         = posterior$last_draw$starting_values$B     # get the last draw of B
B_hat      = diag(sign(diag(BB))) %*% BB           # set positive diagonal elements
posterior    = normalise(posterior, B_hat)           # draws in posterior are normalised

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

`PosteriorBSVART$clone()`: The objects of this class are cloneable with this method.

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

See Also

`estimate`, `specify_bsvar_t`

Examples

```
# This is a function that is used within estimate()
data(us_fiscal_lsuw)
specification  = specify_bsvar_t$new(us_fiscal_lsuw, p = 4)
set.seed(123)
estimate       = estimate(specification, 10)
class(estimate)


## ------------------------------------------------
## Method `PosteriorBSVART$get_posterior()`
## ------------------------------------------------

data(us_fiscal_lsuw)
specification  = specify_bsvar_t$new(us_fiscal_lsuw)
set.seed(123)
estimate       = estimate(specification, 10)
estimate$get_posterior()


## ------------------------------------------------
## Method `PosteriorBSVART$get_last_draw()`
## ------------------------------------------------

data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_t$new(us_fiscal_lsuw, p = 4)

# run the burn-in
set.seed(123)
burn_in        = estimate(specification, 10)

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


## ------------------------------------------------
## Method `PosteriorBSVART$is_normalised()`
## ------------------------------------------------

# upload data
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_t$new(us_fiscal_lsuw, p = 4)

# estimate the model
set.seed(123)
posterior      = estimate(specification, 10)

# check normalisation status beforehand
```

```
posterior$is_normalised()

# normalise the posterior
BB            = posterior$last_draw$starting_values$B      # get the last draw of B
B_hat        = diag((-1) * sign(diag(BB))) %*% BB        # set negative diagonal elements
posterior    = normalise(posterior, B_hat)            # draws in posterior are normalised

# check normalisation status afterwards
posterior$is_normalised()


## ------------------------------------------------
## Method `PosteriorBSVART$set_normalised()`
## ------------------------------------------------

# This is an internal function that is run while executing normalise()
# Observe its working by analysing the workflow:

# upload data
data(us_fiscal_lsuw)

# specify the model and set seed
specification  = specify_bsvar_t$new(us_fiscal_lsuw, p = 4)
set.seed(123)

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

# check normalisation status beforehand
posterior$is_normalised()

# normalise the posterior
BB            = posterior$last_draw$starting_values$B      # get the last draw of B
B_hat        = diag(sign(diag(BB))) %*% BB              # set positive diagonal elements
posterior    = normalise(posterior, B_hat)            # draws in posterior are normalised

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

`specify_prior_bsvar` *R6 Class Representing PriorBSVAR*

Description

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

Public fields

`A` an `NxK` matrix, the mean of the normal prior distribution for the parameter matrix A.

A_V_inv a KxK precision matrix of the normal prior distribution for each of the row of the parameter matrix A. This precision matrix is equation invariant.

B_V_inv an NxN precision matrix of the generalised-normal prior distribution for the structural matrix B. This precision matrix is equation invariant.

B_nu a positive integer greater of equal than N, a shape parameter of the generalised-normal prior distribution for the structural matrix B.

hyper_nu_B a positive scalar, the shape parameter of the inverted-gamma 2 prior for the overall shrinkage parameter for matrix B.

hyper_a_B a positive scalar, the shape parameter of the gamma prior for the second-level hierarchy for the overall shrinkage parameter for matrix B.

hyper_s_BB a positive scalar, the scale parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix B.

hyper_nu_BB a positive scalar, the shape parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix B.

hyper_nu_A a positive scalar, the shape parameter of the inverted-gamma 2 prior for the overall shrinkage parameter for matrix A.

hyper_a_A a positive scalar, the shape parameter of the gamma prior for the second-level hierarchy for the overall shrinkage parameter for matrix A.

hyper_s_AA a positive scalar, the scale parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix A.

hyper_nu_AA a positive scalar, the shape parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix A.

Methods

Public methods:

- PriorBSVAR$new()
- PriorBSVAR$get_prior()
- PriorBSVAR$clone()

PriorBSVAR$new(): Create a new prior specification PriorBSVAR.

Usage:

```
PriorBSVAR$new(N, p, d = 0, stationary = rep(FALSE, N))
```

Arguments:

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

p a positive integer - the autoregressive lag order of the SVAR 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 Nth equation to the random walk process, otherwise to white noise.

Returns: A new prior specification PriorBSVAR.

Examples:

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

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

Usage:

```
PriorBSVAR$get_prior()
```

Examples:

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

`PriorBSVAR$clone()`: The objects of this class are cloneable with this method.

Usage:

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

Arguments:

deep Whether to make a deep clone.

Examples

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


## ------------------------------------------------
## Method `PriorBSVAR$new()`
## ------------------------------------------------

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


## ------------------------------------------------
## Method `PriorBSVAR$get_prior()`
## ------------------------------------------------

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

`specify_prior_bsvar_exh` *R6 Class Representing PriorBSVAREXH*

Description

The class PriorBSVAREXH presents a prior specification for the bsvar model with Exogenous regime change Heteroskedasticity.

Super class

`PriorBSVAR` -> `PriorBSVAREXH`

Public fields

`A` an `NxK` matrix, the mean of the normal prior distribution for the parameter matrix A.

`A_V_inv` a `KxK` precision matrix of the normal prior distribution for each of the row of the parameter matrix A. This precision matrix is equation invariant.

`B_V_inv` an `NxN` precision matrix of the generalised-normal prior distribution for the structural matrix B. This precision matrix is equation invariant.

`B_nu` a positive integer greater of equal than `N`, a shape parameter of the generalised-normal prior distribution for the structural matrix B.

`hyper_nu_B` a positive scalar, the shape parameter of the inverted-gamma 2 prior for the overall shrinkage parameter for matrix B.

`hyper_a_B` a positive scalar, the shape parameter of the gamma prior for the second-level hierarchy for the overall shrinkage parameter for matrix B.

`hyper_s_BB` a positive scalar, the scale parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix B.

`hyper_nu_BB` a positive scalar, the shape parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix B.

`hyper_nu_A` a positive scalar, the shape parameter of the inverted-gamma 2 prior for the overall shrinkage parameter for matrix A.

`hyper_a_A` a positive scalar, the shape parameter of the gamma prior for the second-level hierarchy for the overall shrinkage parameter for matrix A.

`hyper_s_AA` a positive scalar, the scale parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix A.

`hyper_nu_AA` a positive scalar, the shape parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix A.

`sigma_nu` a positive scalar, the shape parameter of the inverted-gamma 2 for state-dependent variances of the structural shocks, $\sigma^2_{n.s_t}$.

`sigma_s` a positive scalar, the scale parameter of the inverted-gamma 2 for state-dependent variances of the structural shocks, $\sigma^2_{n.s_t}$.

Methods

Public methods:

- PriorBSVAREXH$new()
- PriorBSVAREXH$get_prior()
- PriorBSVAREXH$clone()

`PriorBSVAREXH$new()`: Create a new prior specification PriorBSVAREXH

Usage:

```
PriorBSVAREXH$new(N, p, d = 0, stationary = rep(FALSE, N))
```

Arguments:

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

`p` a positive integer - the autoregressive lag order of the SVAR 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 random walk process, otherwise to white noise.

Returns: A new prior specification PriorBSVAREXH

`PriorBSVAREXH$get_prior()`: Returns the elements of the prior specification PriorBSVAREXH as a `list`.

Usage:

```
PriorBSVAREXH$get_prior()
```

Examples:

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

`PriorBSVAREXH$clone()`: The objects of this class are cloneable with this method.

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Examples

```
prior = specify_prior_bsvar_exh$new(N = 3, p = 1)  # specify the prior
prior$A                                            # show autoregressive prior mean


## ------------------------------------------------
## Method `PriorBSVAREXH$get_prior()`
## ------------------------------------------------

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

`specify_prior_bsvar_mix`

R6 Class Representing PriorBSVARMIX

Description

The class PriorBSVARMIX presents a prior specification for the bsvar model with a zero-mean mixture of normals model for structural shocks.

Super classes

`PriorBSVAR` -> `PriorBSVARMSH` -> `PriorBSVARMIX`

Public fields

`A` an `NxK` matrix, the mean of the normal prior distribution for the parameter matrix A.

`A_V_inv` a `KxK` precision matrix of the normal prior distribution for each of the row of the parameter matrix A. This precision matrix is equation invariant.

`B_V_inv` an `NxN` precision matrix of the generalised-normal prior distribution for the structural matrix B. This precision matrix is equation invariant.

`B_nu` a positive integer greater of equal than `N`, a shape parameter of the generalised-normal prior distribution for the structural matrix B.

`hyper_nu_B` a positive scalar, the shape parameter of the inverted-gamma 2 prior for the overall shrinkage parameter for matrix B.

`hyper_a_B` a positive scalar, the shape parameter of the gamma prior for the second-level hierarchy for the overall shrinkage parameter for matrix B.

`hyper_s_BB` a positive scalar, the scale parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix B.

`hyper_nu_BB` a positive scalar, the shape parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix B.

`hyper_nu_A` a positive scalar, the shape parameter of the inverted-gamma 2 prior for the overall shrinkage parameter for matrix A.

`hyper_a_A` a positive scalar, the shape parameter of the gamma prior for the second-level hierarchy for the overall shrinkage parameter for matrix A.

`hyper_s_AA` a positive scalar, the scale parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix A.

`hyper_nu_AA` a positive scalar, the shape parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix A.

`sigma_nu` a positive scalar, the shape parameter of the inverted-gamma 2 for mixture component-dependent variances of the structural shocks, $\sigma^2_{n.s_t}$.

`sigma_s` a positive scalar, the scale parameter of the inverted-gamma 2 for mixture component-dependent variances of the structural shocks, $\sigma^2_{n.s_t}$.

`PR_TR` an `MxM` matrix, the matrix of hyper-parameters of the row-specific Dirichlet prior distribution for the state probabilities the Markov process s_t. Its rows must be identical.

Methods

Public methods:

- PriorBSVARMIX$clone()

PriorBSVARMIX$clone(): The objects of this class are cloneable with this method.

Usage:

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

Arguments:

deep Whether to make a deep clone.

Examples

```
prior = specify_prior_bsvar_mix$new(N = 3, p = 1, M = 2)  # specify the prior
prior$A                                                   # show autoregressive prior mean
```

---

specify_prior_bsvar_msh

R6 Class Representing PriorBSVARMSH

Description

The class PriorBSVARMSH presents a prior specification for the bsvar model with Markov Switching Heteroskedasticity.

Super class

PriorBSVAR -> PriorBSVARMSH

Public fields

A an NxK matrix, the mean of the normal prior distribution for the parameter matrix A.

A_V_inv a KxK precision matrix of the normal prior distribution for each of the row of the parameter matrix A. This precision matrix is equation invariant.

B_V_inv an NxN precision matrix of the generalised-normal prior distribution for the structural matrix B. This precision matrix is equation invariant.

B_nu a positive integer greater of equal than N, a shape parameter of the generalised-normal prior distribution for the structural matrix B.

hyper_nu_B a positive scalar, the shape parameter of the inverted-gamma 2 prior for the overall shrinkage parameter for matrix B.

hyper_a_B a positive scalar, the shape parameter of the gamma prior for the second-level hierarchy for the overall shrinkage parameter for matrix B.

hyper_s_BB a positive scalar, the scale parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix B.

`hyper_nu_BB` a positive scalar, the shape parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix B.

`hyper_nu_A` a positive scalar, the shape parameter of the inverted-gamma 2 prior for the overall shrinkage parameter for matrix A.

`hyper_a_A` a positive scalar, the shape parameter of the gamma prior for the second-level hierarchy for the overall shrinkage parameter for matrix A.

`hyper_s_AA` a positive scalar, the scale parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix A.

`hyper_nu_AA` a positive scalar, the shape parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix A.

`sigma_nu` a positive scalar, the shape parameter of the inverted-gamma 2 for MS state-dependent variances of the structural shocks, $\sigma^2_{n.s_t}$.

`sigma_s` a positive scalar, the scale parameter of the inverted-gamma 2 for MS state-dependent variances of the structural shocks, $\sigma^2_{n.s_t}$.

`PR_TR` an `MxM` matrix, the matrix of hyper-parameters of the row-specific Dirichlet prior distribution for transition probabilities matrix P of the Markov process s_t.

Methods

Public methods:

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

`PriorBSVARMSH$new()`**:** Create a new prior specification PriorBSVARMSH.

Usage:

```
PriorBSVARMSH$new(N, p, d = 0, M, stationary = rep(FALSE, N))
```

Arguments:

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

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

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

`M` an integer greater than 1 - the number of Markov process' heteroskedastic regimes.

`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 random walk process, otherwise to white noise.

Returns: A new prior specification PriorBSVARMSH.

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

Usage:

```
PriorBSVARMSH$get_prior()
```

Examples:

```
# a prior for 3-variable example with four lags and two regimes
prior = specify_prior_bsvar_msh$new(N = 3, p = 4, M = 2)
prior$get_prior() # show the prior as list
```

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Examples

```
prior = specify_prior_bsvar_msh$new(N = 3, p = 1, M = 2)  # specify the prior
prior$A                                          # show autoregressive prior mean


## ------------------------------------------------
## Method `PriorBSVARMSH$get_prior()`
## ------------------------------------------------

# a prior for 3-variable example with four lags and two regimes
prior = specify_prior_bsvar_msh$new(N = 3, p = 4, M = 2)
prior$get_prior() # show the prior as list
```

`specify_prior_bsvar_sv`

R6 Class Representing PriorBSVARSV

Description

The class PriorBSVARSV presents a prior specification for the bsvar model with Stochastic Volatility heteroskedasticity.

Super class

`PriorBSVAR` -> `PriorBSVARSV`

Public fields

`A` an `NxK` matrix, the mean of the normal prior distribution for the parameter matrix A.

`A_V_inv` a `KxK` precision matrix of the normal prior distribution for each of the row of the parameter matrix A. This precision matrix is equation invariant.

`B_V_inv` an `NxN` precision matrix of the generalised-normal prior distribution for the structural matrix B. This precision matrix is equation invariant.

`B_nu` a positive integer greater of equal than `N`, a shape parameter of the generalised-normal prior distribution for the structural matrix B.

`hyper_nu_B` a positive scalar, the shape parameter of the inverted-gamma 2 prior for the overall shrinkage parameter for matrix B.

`hyper_a_B` a positive scalar, the shape parameter of the gamma prior for the second-level hierarchy for the overall shrinkage parameter for matrix B.

`hyper_s_BB` a positive scalar, the scale parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix B.

`hyper_nu_BB` a positive scalar, the shape parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix B.

`hyper_nu_A` a positive scalar, the shape parameter of the inverted-gamma 2 prior for the overall shrinkage parameter for matrix A.

`hyper_a_A` a positive scalar, the shape parameter of the gamma prior for the second-level hierarchy for the overall shrinkage parameter for matrix A.

`hyper_s_AA` a positive scalar, the scale parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix A.

`hyper_nu_AA` a positive scalar, the shape parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix A.

`sv_a_` a positive scalar, the shape parameter of the gamma prior in the hierarchical prior for σ_ω^2.

`sv_s_` a positive scalar, the scale parameter of the gamma prior in the hierarchical prior for σ_ω^2.

Methods

Public methods:

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

`PriorBSVARSV$new()`: Create a new prior specification PriorBSVARSV.

Usage:

```
PriorBSVARSV$new(N, p, d = 0, stationary = rep(FALSE, N))
```

Arguments:

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

`p` a positive integer - the autoregressive lag order of the SVAR 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 random walk process, otherwise to white noise.

Returns: A new prior specification PriorBSVARSV.

`PriorBSVARSV$get_prior()`: Returns the elements of the prior specification PriorBSVARSV as a `list`.

Usage:

```
PriorBSVARSV$get_prior()
```

Examples:

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

`PriorBSVARSV$clone()`: The objects of this class are cloneable with this method.

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Examples

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


## ------------------------------------------------
## Method `PriorBSVARSV$get_prior()`
## ------------------------------------------------

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

---

specify_prior_bsvar_t *R6 Class Representing PriorBSVART*

Description

The class PriorBSVART presents a prior specification for the bsvar model with t-distributed structural shocks.

Super class

`PriorBSVAR` -> `PriorBSVART`

Public fields

`A` an `NxK` matrix, the mean of the normal prior distribution for the parameter matrix A.

`A_V_inv` a `KxK` precision matrix of the normal prior distribution for each of the row of the parameter matrix A. This precision matrix is equation invariant.

`B_V_inv` an `NxN` precision matrix of the generalised-normal prior distribution for the structural matrix B. This precision matrix is equation invariant.

`B_nu` a positive integer greater of equal than `N`, a shape parameter of the generalised-normal prior distribution for the structural matrix B.

`hyper_nu_B` a positive scalar, the shape parameter of the inverted-gamma 2 prior for the overall shrinkage parameter for matrix B.

`hyper_a_B` a positive scalar, the shape parameter of the gamma prior for the second-level hierarchy for the overall shrinkage parameter for matrix B.

`hyper_s_BB` a positive scalar, the scale parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix B.

`hyper_nu_BB` a positive scalar, the shape parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix B.

`hyper_nu_A` a positive scalar, the shape parameter of the inverted-gamma 2 prior for the overall shrinkage parameter for matrix A.

`hyper_a_A` a positive scalar, the shape parameter of the gamma prior for the second-level hierarchy for the overall shrinkage parameter for matrix A.

`hyper_s_AA` a positive scalar, the scale parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix A.

`hyper_nu_AA` a positive scalar, the shape parameter of the inverted-gamma 2 prior for the third-level of hierarchy for overall shrinkage parameter for matrix A.

Methods

Public methods:

- `PriorBSVART$clone()`

`PriorBSVART$clone()`: The objects of this class are cloneable with this method.

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Examples

```
prior = specify_prior_bsvar_t$new(N = 3, p = 1)  # specify the prior
prior$A                                          # show autoregressive prior mean
```

`specify_starting_values_bsvar`

R6 Class Representing StartingValuesBSVAR

Description

The class StartingValuesBSVAR presents starting values for the homoskedastic bsvar model.

Public fields

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

`B` an `NxN` matrix of starting values for the parameter B.

`hyper` a `(2*N+1)x2` matrix of starting values for the shrinkage hyper-parameters of the hierarchical prior distribution.

`lambda` a `NxT` matrix of starting values for latent variables.

`df` an `Nx1` vector of positive numbers with starting values for the equation-specific degrees of freedom parameters of the Student-t conditional distribution of structural shocks.

Methods

Public methods:

- `StartingValuesBSVAR$new()`
- `StartingValuesBSVAR$get_starting_values()`
- `StartingValuesBSVAR$set_starting_values()`
- `StartingValuesBSVAR$clone()`

`StartingValuesBSVAR$new()`**:** Create new starting values StartingValuesBSVAR.

Usage:

```
StartingValuesBSVAR$new(A, B, N, T, p, d = 0)
```

Arguments:

`A` a logical `NxK` matrix containing value `TRUE` for the elements of the autoregressive matrix A to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

`B` a logical `NxN` matrix containing value `TRUE` for the elements of the staructural matrix B to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

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

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

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

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

Returns: Starting values StartingValuesBSVAR.

Examples:

```
# starting values for a homoskedastic bsvar with 4 lags for a 3-variable system
A = matrix(TRUE, 3, 13)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar$new(A = A, B = B, N = 3, T = 120, p = 4)
```

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

Usage:

```
StartingValuesBSVAR$get_starting_values()
```

Examples:

```
# starting values for a homoskedastic bsvar with 1 lag for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar$new(A = A, B = B, N = 3, T = 120, p = 1)
sv$get_starting_values()   # show starting values as list
```

`StartingValuesBSVAR$set_starting_values()`**:** Returns the elements of the starting values StartingValuesBSVAR as a `list`.

Usage:

```
StartingValuesBSVAR$set_starting_values(last_draw)
```

Arguments:

`last_draw` a list containing the last draw of elements `B` - an `NxN` matrix, `A` - an `NxK` matrix, and `hyper` - a vector of 5 positive real numbers.

Returns: An object of class StartingValuesBSVAR 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 homoskedastic bsvar with 1 lag for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar$new(A = A, B = B, N = 3, T = 120, p = 1)

# 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
```

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Examples

```
# starting values for a homoskedastic bsvar for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar$new(A = A, B = B, N = 3, T = 120, p = 1)


## ------------------------------------------------
## Method `StartingValuesBSVAR$new()`
## ------------------------------------------------

# starting values for a homoskedastic bsvar with 4 lags for a 3-variable system
A = matrix(TRUE, 3, 13)
```

```
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar$new(A = A, B = B, N = 3, T = 120, p = 4)


## ------------------------------------------------
## Method `StartingValuesBSVAR$get_starting_values()`
## ------------------------------------------------

# starting values for a homoskedastic bsvar with 1 lag for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar$new(A = A, B = B, N = 3, T = 120, p = 1)
sv$get_starting_values()   # show starting values as list


## ------------------------------------------------
## Method `StartingValuesBSVAR$set_starting_values()`
## ------------------------------------------------

# starting values for a homoskedastic bsvar with 1 lag for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar$new(A = A, B = B, N = 3, T = 120, p = 1)

# 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
```

`specify_starting_values_bsvar_exh`

R6 Class Representing StartingValuesBSVAREXH

Description

The class StartingValuesBSVAREXH presents starting values for the bsvar model with exogenous regime change Heteroskedasticity.

Super class

`StartingValuesBSVAR` -> `StartingValuesBSVAREXH`

Public fields

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

`B` an `NxN` matrix of starting values for the parameter B.

`hyper` a `(2*N+1)x2` matrix of starting values for the shrinkage hyper-parameters of the hierarchical prior distribution.

`sigma2` an `NxM` matrix of starting values for the regime-specific variances of the structural shocks. Its elements sum to value `M` over the rows.

`xi` a `1xT` matrix of starting values for the Markov process indicator with integer values from `0` to `M-1`.

`lambda` a `NxT` matrix of starting values for latent variables.

`df` an `Nx1` vector of positive numbers with starting values for the equation-specific degrees of freedom parameters of the Student-t conditional distribution of structural shocks.

Methods

Public methods:

- `StartingValuesBSVAREXH$new()`
- `StartingValuesBSVAREXH$get_starting_values()`
- `StartingValuesBSVAREXH$set_starting_values()`
- `StartingValuesBSVAREXH$clone()`

`StartingValuesBSVAREXH$new()`**:** Create new starting values StartingValuesBSVAREXH.

Usage:

```
StartingValuesBSVAREXH$new(A, B, N, p, T, d = 0, variance_regimes = rep(1, T))
```

Arguments:

`A` a logical `NxK` matrix containing value `TRUE` for the elements of the autoregressive matrix A to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

`B` a logical `NxN` matrix containing value `TRUE` for the elements of the structural matrix B to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

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

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

`T` a positive integer - the the time series dimension of the dependent variable matrix Y.

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

`variance_regimes` a `T`-vector with exogenous regime indicators that are integer numbers associating the time observation with heteroskedastic regime.

Returns: Starting values StartingValuesBSVAREXH.

`StartingValuesBSVAREXH$get_starting_values()`**:** Returns the elements of the starting values StartingValuesBSVAR-MS as a `list`.

Usage:

```
StartingValuesBSVAREXH$get_starting_values()
```

Examples:

```
# starting values for a homoskedastic bsvar with 1 lag for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar_exh$new(A = A, B = B, N = 3, p = 1, T = 100)
sv$get_starting_values()   # show starting values as list
```

`StartingValuesBSVAREXH$set_starting_values()`**:** Returns the elements of the starting values StartingValuesBSVAREXH as a `list`.

Usage:

```
StartingValuesBSVAREXH$set_starting_values(last_draw)
```

Arguments:

`last_draw` a list containing the last draw.

Returns: An object of class StartingValuesBSVAREXH 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 bsvar model with 1 lag for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar_exh$new(A = A, B = B, N = 3, p = 1, 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
```

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

Usage:

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

Arguments:

deep Whether to make a deep clone.

Examples

```
# starting values for a bsvar model for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar_exh$new(A = A, B = B, N = 3, p = 1, T = 100)


## ------------------------------------------------
## Method `StartingValuesBSVAREXH$get_starting_values()`
## ------------------------------------------------

# starting values for a homoskedastic bsvar with 1 lag for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar_exh$new(A = A, B = B, N = 3, p = 1, T = 100)
sv$get_starting_values()   # show starting values as list


## ------------------------------------------------
## Method `StartingValuesBSVAREXH$set_starting_values()`
## ------------------------------------------------

# starting values for a bsvar model with 1 lag for a 3-variable system
```

```
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar_exh$new(A = A, B = B, N = 3, p = 1, 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
```

`specify_starting_values_bsvar_hmsh`

R6 Class Representing StartingValuesBSVAHMSH

Description

The class StartingValuesBSVARHMSH presents starting values for the bsvar model with Heterogeneous Markov Switching Heteroskedasticity.

Super class

`StartingValuesBSVAR` -> `StartingValuesBSVARHMSH`

Public fields

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

`B` an `NxN` matrix of starting values for the parameter B.

`hyper` a `(2*N+1)x2` matrix of starting values for the shrinkage hyper-parameters of the hierarchical prior distribution.

`sigma2` an `NxM` matrix of starting values for the MS state-specific variances of the structural shocks. Its elements sum to value `M` over the rows.

`PR_TR` an `MxMxN` array of starting values for the transition probability matrix of the Markov process. Its elements sum to 1 over the rows.

`xi` a `1xTxN` array of starting values for the Markov process indicator with integer values from `0` to `M-1`.

`pi_0` an `MxN` matrix of starting values for state probability at time `t=0`. Its elements sum to 1 in columns.

`lambda` a `NxT` matrix of starting values for latent variables.

`df` an `Nx1` vector of positive numbers with starting values for the equation-specific degrees of freedom parameters of the Student-t conditional distribution of structural shocks.

Methods

Public methods:

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

StartingValuesBSVARHMSH$new(): Create new starting values StartingValuesBSVARHMSH.

Usage:

```
StartingValuesBSVARHMSH$new(A, B, N, p, M, T, d = 0, finiteM = TRUE)
```

Arguments:

A a logical NxK matrix containing value TRUE for the elements of the autoregressive matrix A to be estimated and value FALSE for exclusion restrictions to be set to zero.

B a logical NxN matrix containing value TRUE for the elements of the structural matrix B to be estimated and value FALSE for exclusion restrictions to be set to zero.

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

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

M an integer greater than 1 - the number of Markov process' heteroskedastic regimes.

T a positive integer - the the time series dimension of the dependent variable matrix Y.

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

finiteM a logical value - if true a stationary Markov switching model is estimated. Otherwise, a sparse Markov switching model is estimated in which M=20 and the number of visited states is estimated.

Returns: Starting values StartingValuesBSVARHMSH.

StartingValuesBSVARHMSH$get_starting_values(): Returns the elements of the starting values StartingValuesBSVARHMSH as a list.

Usage:

```
StartingValuesBSVARHMSH$get_starting_values()
```

Examples:

```
# starting values for a homoskedastic bsvar with 1 lag for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar_hmsh$new(A = A, B = B, N = 3, p = 1, M = 2, T = 100)
sv$get_starting_values()   # show starting values as list
```

StartingValuesBSVARHMSH$set_starting_values(): Returns the elements of the starting values StartingValuesBSVARHMSH as a list.

Usage:

```
StartingValuesBSVARHMSH$set_starting_values(last_draw)
```

Arguments:

last_draw a list containing the last draw.

Returns: An object of class StartingValuesBSVARHMSH 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 bsvar model with 1 lag for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar_hmsh$new(A = A, B = B, N = 3, p = 1, M = 2, 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
```

`StartingValuesBSVARHMSH$clone()`: The objects of this class are cloneable with this method.

Usage:

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

Arguments:

deep Whether to make a deep clone.

Examples

```
# starting values for a bsvar model for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar_hmsh$new(A = A, B = B, N = 3, p = 1, M = 2, T = 100)


## ------------------------------------------------
## Method `StartingValuesBSVARHMSH$get_starting_values()`
## ------------------------------------------------

# starting values for a homoskedastic bsvar with 1 lag for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar_hmsh$new(A = A, B = B, N = 3, p = 1, M = 2, T = 100)
sv$get_starting_values()   # show starting values as list


## ------------------------------------------------
## Method `StartingValuesBSVARHMSH$set_starting_values()`
## ------------------------------------------------

# starting values for a bsvar model with 1 lag for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar_hmsh$new(A = A, B = B, N = 3, p = 1, M = 2, 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
```

---

specify_starting_values_bsvar_mix
R6 Class Representing StartingValuesBSVARMIX

Description

The class StartingValuesBSVARMIX presents starting values for the bsvar model with a zero-mean mixture of normals model for structural shocks.

Super classes

`StartingValuesBSVAR` -> `StartingValuesBSVARMSH` -> `StartingValuesBSVARMIX`

Public fields

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

`B` an `NxN` matrix of starting values for the parameter B.

`hyper` a `(2*N+1)x2` matrix of starting values for the shrinkage hyper-parameters of the hierarchical prior distribution.

`sigma2` an `NxM` matrix of starting values for the MS state-specific variances of the structural shocks. Its elements sum to value `M` over the rows.

`PR_TR` an `MxM` matrix of starting values for the probability matrix of the Markov process. Its rows must be identical and the elements of each row sum to 1 over the rows.

`xi` a `1xT` matrix of starting values for the Markov process indicator with integer values from `0` to `M-1`.

`pi_0` an `M`-vector of starting values for mixture components state probabilities. Its elements sum to 1.

`lambda` a `NxT` matrix of starting values for latent variables.

`df` an `Nx1` vector of positive numbers with starting values for the equation-specific degrees of freedom parameters of the Student-t conditional distribution of structural shocks.

Methods

Public methods:

- `StartingValuesBSVARMIX$new()`
- `StartingValuesBSVARMIX$clone()`

`StartingValuesBSVARMIX$new()`**:** Create new starting values StartingValuesBSVARMIX.

Usage:

```
StartingValuesBSVARMIX$new(A, B, N, p, M, T, d = 0, finiteM = TRUE)
```

Arguments:

`A` a logical `NxK` matrix containing value `TRUE` for the elements of the autoregressive matrix A to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

`B` a logical `NxN` matrix containing value `TRUE` for the elements of the staructural matrix B to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

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

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

`M` an integer greater than 1 - the number of components of the mixture of normals.

`T` a positive integer - the the time series dimension of the dependent variable matrix Y.

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

`finiteM` a logical value - if true a finite mixture model is estimated. Otherwise, a sparse mixture model is estimated in which `M=20` and the number of visited states is estimated.

Returns: Starting values StartingValuesBSVARMIX.

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Examples

```
# starting values for a bsvar model for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar_mix$new(A = A, B = B, N = 3, p = 1, M = 2, T = 100)
```

`specify_starting_values_bsvar_msh`

R6 Class Representing StartingValuesBSVARMSH

Description

The class StartingValuesBSVARMSH presents starting values for the bsvar model with Markov Switching Heteroskedasticity.

Super class

`StartingValuesBSVAR` -> `StartingValuesBSVARMSH`

Public fields

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

`B` an `NxN` matrix of starting values for the parameter B.

`hyper` a `(2*N+1)x2` matrix of starting values for the shrinkage hyper-parameters of the hierarchical prior distribution.

`sigma2` an `NxM` matrix of starting values for the MS state-specific variances of the structural shocks. Its elements sum to value `M` over the rows.

`PR_TR` an `MxM` matrix of starting values for the transition probability matrix of the Markov process. Its elements sum to 1 over the rows.

`xi` a `1xT` matrix of starting values for the Markov process indicator with integer values from `0` to `M-1`.

`pi_0` an `M`-vector of starting values for state probability at time `t=0`. Its elements sum to 1.

`lambda` a `NxT` matrix of starting values for latent variables.

`df` an `Nx1` vector of positive numbers with starting values for the equation-specific degrees of freedom parameters of the Student-t conditional distribution of structural shocks.

Methods

Public methods:

- `StartingValuesBSVARMSH$new()`
- `StartingValuesBSVARMSH$get_starting_values()`
- `StartingValuesBSVARMSH$set_starting_values()`
- `StartingValuesBSVARMSH$clone()`

`StartingValuesBSVARMSH$new()`: Create new starting values StartingValuesBSVAR-MS.

Usage:

```
StartingValuesBSVARMSH$new(A, B, N, p, M, T, d = 0, finiteM = TRUE)
```

Arguments:

`A` a logical `NxK` matrix containing value `TRUE` for the elements of the autoregressive matrix A to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

`B` a logical `NxN` matrix containing value `TRUE` for the elements of the staructural matrix B to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

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

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

`M` an integer greater than 1 - the number of Markov process' heteroskedastic regimes.

`T` a positive integer - the the time series dimension of the dependent variable matrix Y.

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

`finiteM` a logical value - if true a stationary Markov switching model is estimated. Otherwise, a sparse Markov switching model is estimated in which `M=20` and the number of visited states is estimated.

Returns: Starting values StartingValuesBSVAR-MS.

`StartingValuesBSVARMSH$get_starting_values()`**:** Returns the elements of the starting values StartingValuesBSVAR-MS as a `list`.

Usage:

```
StartingValuesBSVARMSH$get_starting_values()
```

Examples:

```
# starting values for a homoskedastic bsvar with 1 lag for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar_msh$new(A = A, B = B, N = 3, p = 1, M = 2, T = 100)
sv$get_starting_values()   # show starting values as list
```

`StartingValuesBSVARMSH$set_starting_values()`**:** Returns the elements of the starting values StartingValuesBSVARMSH as a `list`.

Usage:

```
StartingValuesBSVARMSH$set_starting_values(last_draw)
```

Arguments:

`last_draw` a list containing the last draw.

Returns: An object of class StartingValuesBSVAR-MS 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 bsvar model with 1 lag for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar_msh$new(A = A, B = B, N = 3, p = 1, M = 2, 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
```

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Examples

```
# starting values for a bsvar model for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar_msh$new(A = A, B = B, N = 3, p = 1, M = 2, T = 100)
```

```
## ------------------------------------------------
## Method `StartingValuesBSVARMSH$get_starting_values()`
## ------------------------------------------------

# starting values for a homoskedastic bsvar with 1 lag for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar_msh$new(A = A, B = B, N = 3, p = 1, M = 2, T = 100)
sv$get_starting_values()   # show starting values as list


## ------------------------------------------------
## Method `StartingValuesBSVARMSH$set_starting_values()`
## ------------------------------------------------

# starting values for a bsvar model with 1 lag for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar_msh$new(A = A, B = B, N = 3, p = 1, M = 2, 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
```

`specify_starting_values_bsvar_sv`

R6 Class Representing StartingValuesBSVARSV

Description

The class StartingValuesBSVARSV presents starting values for the bsvar model with Stochastic Volatility heteroskedasticity.

Super class

`StartingValuesBSVAR` -> `StartingValuesBSVARSV`

Public fields

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

`B` an `NxN` matrix of starting values for the parameter B.

`hyper` a `(2*N+1)x2` matrix of starting values for the shrinkage hyper-parameters of the hierarchical prior distribution.

`h` an `NxT` matrix with the starting values of the log-volatility processes.

`rho` an `N`-vector with values of SV autoregressive parameters.

`omega` an `N`-vector with values of SV process conditional standard deviations.

`sigma2v` an N-vector with values of SV process conditional variances.

`S` an NxT integer matrix with the auxiliary mixture component indicators.

`sigma2_omega` an N-vector with variances of the zero-mean normal prior for ω_n.

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

`lambda` a NxT matrix of starting values for latent variables.

`df` an Nx1 vector of positive numbers with starting values for the equation-specific degrees of freedom parameters of the Student-t conditional distribution of structural shocks.

Methods

Public methods:

- `StartingValuesBSVARSV$new()`
- `StartingValuesBSVARSV$get_starting_values()`
- `StartingValuesBSVARSV$set_starting_values()`
- `StartingValuesBSVARSV$clone()`

`StartingValuesBSVARSV$new()`**:** Create new starting values StartingValuesBSVARSV.

Usage:

```
StartingValuesBSVARSV$new(A, B, N, p, T, d = 0)
```

Arguments:

`A` a logical NxK matrix containing value `TRUE` for the elements of the autoregressive matrix A to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

`B` a logical NxN matrix containing value `TRUE` for the elements of the staructural matrix B to be estimated and value `FALSE` for exclusion restrictions to be set to zero.

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

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

`T` a positive integer - the the time series dimension of the dependent variable matrix Y.

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

Returns: Starting values StartingValuesBSVARSV.

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

Usage:

```
StartingValuesBSVARSV$get_starting_values()
```

Examples:

```
# starting values for a bsvar model with 1 lag for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar_sv$new(A = A, B = B, N = 3, p = 1, T = 100)
sv$get_starting_values()   # show starting values as list
```

`StartingValuesBSVARSV$set_starting_values()`**:** Returns the elements of the starting values StartingValuesBSVAR_SV as a `list`.

Usage:

```
StartingValuesBSVARSV$set_starting_values(last_draw)
```

Arguments:

`last_draw` a list containing the last draw of the current MCMC run.

Returns: An object of class StartingValuesBSVAR 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 bsvar model with 1 lag for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar_sv$new(A = A, B = B, N = 3, p = 1, 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
```

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

Examples

```
# starting values for a bsvar model for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar_sv$new(A = A, B = B, N = 3, p = 1, T = 100)


## ------------------------------------------------
## Method `StartingValuesBSVARSV$get_starting_values()`
## ------------------------------------------------

# starting values for a bsvar model with 1 lag for a 3-variable system
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar_sv$new(A = A, B = B, N = 3, p = 1, T = 100)
sv$get_starting_values()   # show starting values as list


## ------------------------------------------------
## Method `StartingValuesBSVARSV$set_starting_values()`
## ------------------------------------------------

# starting values for a bsvar model with 1 lag for a 3-variable system
```

```
A = matrix(TRUE, 3, 4)
B = matrix(TRUE, 3, 3)
sv = specify_starting_values_bsvar_sv$new(A = A, B = B, N = 3, p = 1, 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.Forecasts` *Provides posterior summary of Forecasts*

Description

Provides posterior summary of the forecasts including their mean, standard deviations, as well as 5 and 95 percentiles.

Usage

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

Arguments

`object`	an object of class Forecasts obtained using the `forecast()` function containing draws the predictive density.
`...`	additional arguments affecting the summary produced.

Value

A list reporting the posterior mean, standard deviations, as well as 5 and 95 percentiles of the forecasts for each of the variables and forecast horizons.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

`forecast`

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# forecast
fore           = forecast(posterior, horizon = 2)
fore_summary   = summary(fore)
fore_summary$variable1

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  forecast(horizon = 2) |>
  summary() -> fore_summary
fore_summary$variable1
```

`summary.PosteriorBSVAR` *Provides posterior summary of homoskedastic Structural VAR estimation*

Description

Provides posterior mean, standard deviations, as well as 5 and 95 percentiles of the parameters: the structural matrix B, autoregressive parameters A, and hyper parameters.

Usage

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

Arguments

`object`	an object of class PosteriorBSVAR obtained using the `estimate()` function applied to homoskedastic Bayesian Structural VAR model specification set by function `specify_bsvar$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: the structural matrix B, autoregressive parameters A, and hyper-parameters.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

estimate, specify_bsvar

Examples

```
specification = specify_bsvar$new(us_fiscal_lsuw)
burn_in       = estimate(specification, 5)
posterior     = estimate(burn_in, 5)
summ          = summary(posterior)
summ

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  summary() -> summ
summ
```

---

summary.PosteriorBSVAREXH

Provides posterior summary of heteroskedastic Structural VAR estimation

---

Description

Provides posterior mean, standard deviations, as well as 5 and 95 percentiles of the parameters: the structural matrix B, autoregressive parameters A, and hyper parameters.

Usage

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

Arguments

object	an object of class PosteriorBSVAREXH obtained using the estimate() function applied to heteroskedastic Bayesian Structural VAR model specification set by function specify_bsvar_exh$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: the structural matrix B, autoregressive parameters A, and hyper-parameters.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

estimate, specify_bsvar_exh

Examples

```
spec  = specify_bsvar_exh$new(us_fiscal_lsuw)
burn  = estimate(spec, 5)
post  = estimate(burn, 5)
summ  = summary(post)
summ

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_exh$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  summary() -> summ
summ
```

---

summary.PosteriorBSVARHMSH

Provides posterior summary of heteroskedastic Structural VAR estimation

---

Description

Provides posterior mean, standard deviations, as well as 5 and 95 percentiles of the parameters: the structural matrix B, autoregressive parameters A, and hyper parameters.

Usage

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

Arguments

`object`	an object of class PosteriorBSVARHMSH obtained using the `estimate()` function applied to heteroskedastic Bayesian Structural VAR model specification set by function `specify_bsvar_hmsh$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: the structural matrix B, autoregressive parameters A, and hyper-parameters.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

estimate, specify_bsvar_hmsh

Examples

```
specification  = specify_bsvar_msh$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)
summ           = summary(posterior)
summ$B$equation1[,1] # access posterior mean

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  summary() -> summ
summ$B$equation1[,1] # access posterior mean
```

---

summary.PosteriorBSVARMIX

Provides posterior summary of non-normal Structural VAR estimation

---

Description

Provides posterior mean, standard deviations, as well as 5 and 95 percentiles of the parameters: the structural matrix B, autoregressive parameters A, and hyper parameters.

Usage

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

Arguments

object	an object of class PosteriorBSVARMIX obtained using the estimate() function applied to non-normal Bayesian Structural VAR model specification set by function specify_bsvar_mix$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: the structural matrix B, autoregressive parameters A, and hyper-parameters.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

estimate, specify_bsvar_mix

Examples

```
specification  = specify_bsvar_mix$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)
summ           = summary(posterior)
summ$A$equation1[,1] # access posterior means

# workflow with the pipe |>
############################################################
set.seed(123)
us_fiscal_lsuw |>
  specify_bsvar_mix$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  summary() -> summ
summ$A$equation1[,1] # access posterior means
```

summary.PosteriorBSVARMSH *Provides posterior summary of heteroskedastic Structural VAR estimation*

Description

Provides posterior mean, standard deviations, as well as 5 and 95 percentiles of the parameters: the structural matrix B, autoregressive parameters A, and hyper parameters.

Usage

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

Arguments

object	an object of class PosteriorBSVARMSH obtained using the `estimate()` function applied to heteroskedastic Bayesian Structural VAR model specification set by function `specify_bsvar_msh$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: the structural matrix B, autoregressive parameters A, and hyper-parameters.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

estimate, specify_bsvar_msh

Examples

```
specification  = specify_bsvar_msh$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)
summ           = summary(posterior)
summ

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new() |>
```

```
  estimate(S = 5) |>
  estimate(S = 5) |>
  summary() -> summ
summ
```

`summary.PosteriorBSVARSV` *Provides posterior summary of heteroskedastic Structural VAR estimation*

Description

Provides posterior mean, standard deviations, as well as 5 and 95 percentiles of the parameters: the structural matrix B, autoregressive parameters A, and hyper parameters.

Usage

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

Arguments

`object`	an object of class PosteriorBSVARSV obtained using the `estimate()` function applied to heteroskedastic Bayesian Structural VAR model specification set by function `specify_bsvar_sv$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: the structural matrix B, autoregressive parameters A, and hyper-parameters.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

`estimate`, `specify_bsvar_sv`

Examples

```
specification  = specify_bsvar_sv$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)
summ           = summary(posterior)
summ

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_sv$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  summary() -> summ
summ
```

`summary.PosteriorBSVART`

Provides posterior summary of Structural VAR with t-distributed shocks estimation

---

Description

Provides posterior mean, standard deviations, as well as 5 and 95 percentiles of the parameters: the structural matrix B, autoregressive parameters A, hyper-parameters, and Student-t degrees-of-freedom parameter ν.

Usage

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

Arguments

`object`	an object of class PosteriorBSVART obtained using the `estimate()` function applied to homoskedastic Bayesian Structural VAR model specification set by function `specify_bsvar$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: the structural matrix B, autoregressive parameters A, hyper-parameters, and Student-t degrees-of-freedom parameter ν.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

estimate, specify_bsvar_t

Examples

```
specification  = specify_bsvar_t$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)
summ           = summary(posterior)
summ$A$equation1[,1] # access posterior means

# workflow with the pipe |>
############################################################
set.seed(123)
us_fiscal_lsuw |>
  specify_bsvar_t$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  summary() -> summ
summ$A$equation1[,1] # access posterior means
```

---

summary.PosteriorFEVD *Provides posterior summary of forecast error variance decompositions*

Description

Provides posterior means of the forecast error variance decompositions of each variable at all horizons.

Usage

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

Arguments

object	an object of class PosteriorFEVD obtained using the compute_variance_decompositions() function containing draws from the posterior distribution of the forecast error variance decompositions.
...	additional arguments affecting the summary produced.

Value

A list reporting the posterior mean of the forecast error variance decompositions of each variable at all horizons.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

compute_variance_decompositions

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute forecast error variance decompositions
fevd           = compute_variance_decompositions(posterior, horizon = 4)
fevd_summary   = summary(fevd)
fevd_summary

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_variance_decompositions(horizon = 4) |>
  summary() -> fevd_summary
fevd_summary
```

---

summary.PosteriorFitted

Provides posterior summary of variables' fitted values

Description

Provides posterior summary of the fitted values including their mean, standard deviations, as well as 5 and 95 percentiles.

Usage

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

Arguments

`object`	an object of class PosteriorFitted obtained using the `compute_fitted_values()` function containing draws the predictive density of the sample data.
`...`	additional arguments affecting the summary produced.

Value

A list reporting the posterior mean, standard deviations, as well as 5 and 95 percentiles of the fitted values for each of the shocks and periods.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

compute_fitted_values

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute fitted values
fitted         = compute_fitted_values(posterior)
fitted_summary = summary(fitted)
fitted_summary$ttr[,1] # access posterior mean of ttr

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_fitted_values() |>
  summary() -> fitted_summary
fitted_summary$ttr[,1] # access posterior mean of ttr
```

---

summary.PosteriorHD *Provides posterior summary of historical decompositions*

Description

Provides posterior means of the historical decompositions variable by variable.

Usage

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

Arguments

object	an object of class PosteriorHD obtained using the compute_historical_decompositions() function containing posterior draws of historical decompositions.
...	additional arguments affecting the summary produced.

Value

A list reporting the posterior means of historical decompositions for each of the variables.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

compute_historical_decompositions

Examples

```
specification  = specify_bsvar$new(diff(us_fiscal_lsuw))
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute historical decompositions
hds            = compute_historical_decompositions(posterior)
hds_summary    = summary(hds)
head(hds_summary$gdp) # browse the contributions

# workflow with the pipe |>
############################################################
diff(us_fiscal_lsuw) |>
  specify_bsvar$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_historical_decompositions() |>
  summary() -> hds_summary
hds_summary$gdp # browse the contributions
```

summary.PosteriorIR *Provides posterior summary of impulse responses*

Description

Provides posterior summary of the impulse responses of each variable to each of the shocks at all horizons. Includes their posterior means, standard deviations, as well as 5 and 95 percentiles.

Usage

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

Arguments

object	an object of class PosteriorIR obtained using the compute_impulse_responses() function containing draws from the posterior distribution of the impulse responses.
...	additional arguments affecting the summary produced.

Value

A list reporting the posterior mean, standard deviations, as well as 5 and 95 percentiles of the impulse responses of each variable to each of the shocks at all horizons.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

compute_impulse_responses

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute impulse responses
irf            = compute_impulse_responses(posterior, horizon = 4)
irf_summary    = summary(irf)
irf_summary$shock1 # inspect IRFs of the first shock

# workflow with the pipe |>
############################################################
set.seed(123)
us_fiscal_lsuw |>
  specify_bsvar$new() |>
```

```
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_impulse_responses(horizon = 4) |>
  summary() -> irf_summary
irf_summary$shock1 # inspect IRFs of the first shock
```

---

summary.PosteriorRegimePr *Provides posterior summary of regimes or regime probabilities*

Description

Provides posterior summary of realized regime indices or regime probabilities, including their mean and standard deviations.

Usage

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

Arguments

object	an object of class PosteriorRegimePr obtained using the compute_regime_probabilities() function containing posterior draws of regime allocations.
...	additional arguments affecting the summary produced.

Value

A list reporting the posterior mean and standard deviations of the realized regime indices or regime probabilities.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

compute_regime_probabilities

Examples

```
specification  = specify_bsvar_msh$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute regime probabilities
rp             = compute_regime_probabilities(posterior)
rp_summary     = summary(rp)
```

```
head(rp_summary$MarkovProcess1$regime) # browse the results

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_regime_probabilities() |>
  summary() -> rp_summary
head(rp_summary$MarkovProcess1$regime) # browse the results
```

---

summary.PosteriorShocks

Provides posterior summary of structural shocks

Description

Provides posterior summary of the structural shocks including their mean, standard deviations, as well as 5 and 95 percentiles.

Usage

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

Arguments

object	an object of class PosteriorShocks obtained using the compute_structural_shocks() function containing draws the posterior distribution of the structural shocks.
...	additional arguments affecting the summary produced.

Value

A list reporting the posterior mean, standard deviations, as well as 5 and 95 percentiles of the structural shocks for each of the equations and periods.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

compute_structural_shocks

Examples

```
specification  = specify_bsvar$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute structural shocks
shocks         = compute_structural_shocks(posterior)
shocks_summary = summary(shocks)
head(shocks_summary$shock1)

# workflow with the pipe |>
############################################################
set.seed(123)
us_fiscal_lsuw |>
  specify_bsvar$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_structural_shocks() |>
  summary() -> shocks_summary
head(shocks_summary$shock1)
```

---

summary.PosteriorSigma

Provides posterior summary of structural shocks' conditional standard deviations

---

Description

Provides posterior summary of structural shocks' conditional standard deviations including their mean, standard deviations, as well as 5 and 95 percentiles.

Usage

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

Arguments

`object`	an object of class PosteriorSigma obtained using the `compute_conditional_sd()` function containing posterior draws of conditional standard deviations of structural shocks.
`...`	additional arguments affecting the summary produced.

Value

A list reporting the posterior mean, standard deviations, as well as 5 and 95 percentiles of the structural shocks' conditional standard deviations for each of the shocks and periods.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

compute_conditional_sd

Examples

```
specification  = specify_bsvar_sv$new(us_fiscal_lsuw)
burn_in        = estimate(specification, 5)
posterior      = estimate(burn_in, 5)

# compute structural shocks' conditional standard deviations
sigma          = compute_conditional_sd(posterior)
sigma_summary  = summary(sigma)
sigma_summary$shock1[,1] # access posterior mean of shock1

# workflow with the pipe |>
############################################################
set.seed(123)
us_fiscal_lsuw |>
  specify_bsvar_sv$new() |>
  estimate(S = 5) |>
  estimate(S = 5) |>
  compute_conditional_sd() |>
  summary() -> sigma_summary
sigma_summary$shock1[,1]
```

---

summary.SDDRautoregression

Provides summary of verifying hypotheses about autoregressive parameters

---

Description

Provides summary of the Savage-Dickey density ratios for verification of hypotheses about autoregressive parameters.

Usage

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

Arguments

object	an object of class SDDRautoregression obtained using the verify_autoregression() function.
...	additional arguments affecting the summary produced.

Value

A table reporting the logarithm of Bayes factors of the restriction against no restriction posterior odds in "log(SDDR)", its numerical standard error "NSE", and the implied posterior probability of the restriction holding or not hypothesis, "Pr[H0|data]" and "Pr[H1|data]" respectively.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

verify_autoregression

Examples

```
specification  = specify_bsvar_sv$new(us_fiscal_lsuw, p = 1)
posterior      = estimate(specification, 10)

# verify autoregression
H0             = matrix(NA, ncol(us_fiscal_lsuw), ncol(us_fiscal_lsuw) + 1)
H0[1,3]        = 0        # a hypothesis of no Granger causality from gdp to ttr
sddr           = verify_autoregression(posterior, H0)
summary(sddr)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_sv$new(p = 1) |>
  estimate(S = 10) |>
  verify_autoregression(hypothesis = H0) |>
  summary() -> sddr_summary
```

---

summary.SDDRidMIX *Provides summary of verifying shocks' normality*

Description

Provides summary of the Savage-Dickey density ratios for verification of structural shocks normality. The outcomes can be used to make probabilistic statements about identification through non-normality.

Usage

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

Arguments

object	an object of class SDDRidMIX obtained using the verify_identification.PosteriorBSVARMIX function.
...	additional arguments affecting the summary produced.

Value

A table reporting the logarithm of Bayes factors of normal to non-normal shocks posterior odds "log(SDDR)" for each structural shock, their numerical standard errors "NSE", and the implied posterior probability of the normality and non-normality hypothesis, "Pr[normal|data]" and "Pr[non-normal|data]" respectively.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

verify_identification.PosteriorBSVARMIX

Examples

```
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, M = 2)
posterior      = estimate(specification, 10)

# verify heteroskedasticity
sddr           = verify_identification(posterior)
summary(sddr)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_mix$new(M = 2) |>
  estimate(S = 10) |>
  verify_identification() |>
  summary() -> sddr_summary
```

---

summary.SDDRidMSH *Provides summary of verifying homoskedasticity*

Description

Provides summary of the Savage-Dickey density ratios for verification of structural shocks homoskedasticity. The outcomes can be used to make probabilistic statements about identification through heteroskedasticity closely following ideas by Lütkepohl& Woźniak (2020).

Usage

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

Arguments

object	an object of class SDDRidMSH obtained using the verify_identification.PosteriorBSVARMSH function.
...	additional arguments affecting the summary produced.

Value

A table reporting the logarithm of Bayes factors of homoskedastic to heteroskedastic posterior odds "log(SDDR)" for each structural shock, their numerical standard errors "NSE", and the implied posterior probability of the homoskedasticity and heteroskedasticity hypothesis, "Pr[homoskedasticity|data]" and "Pr[heteroskedasticity|data]" respectively.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

See Also

verify_identification.PosteriorBSVARMSH

Examples

```
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, M = 2)
posterior      = estimate(specification, 10)

# verify heteroskedasticity
sddr           = verify_identification(posterior)
summary(sddr)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new(M = 2) |>
  estimate(S = 10) |>
  verify_identification() |>
  summary() -> sddr_summary
```

summary.SDDRidSV *Provides summary of verifying homoskedasticity*

Description

Provides summary of the Savage-Dickey density ratios for verification of structural shocks homoskedasticity. The outcomes can be used to make probabilistic statements about identification through heteroskedasticity following Lütkepohl, Shang, Uzeda & Woźniak (2024).

Usage

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

Arguments

object	an object of class SDDRidSV obtained using the verify_identification.PosteriorBSVARSV function.
...	additional arguments affecting the summary produced.

Value

A table reporting the logarithm of Bayes factors of homoskedastic to heteroskedastic posterior odds "log(SDDR)" for each structural shock, their numerical standard errors "NSE", and the implied posterior probability of the homoskedasticity and heteroskedasticity hypothesis, "Pr[homoskedasticity|data]" and "Pr[heteroskedasticity|data]" respectively.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

See Also

verify_identification.PosteriorBSVARSV

Examples

```
specification  = specify_bsvar_sv$new(us_fiscal_lsuw)
posterior      = estimate(specification, 10)

# verify heteroskedasticity
sddr           = verify_identification(posterior)
```

```
summary(sddr)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_sv$new() |>
  estimate(S = 10) |>
  verify_identification() |>
  summary() -> sddr_summary
```

---

summary.SDDRidT *Provides summary of verifying shocks' normality*

Description

Provides summary of the Savage-Dickey density ratios for verification of structural shocks normality. The outcomes can be used to make probabilistic statements about identification through non-normality.

Usage

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

Arguments

object	an object of class SDDRidT obtained using the verify_identification.PosteriorBSVART function.
...	additional arguments affecting the summary produced.

Value

A table reporting the Bayes factor of normal to Student-t shocks posterior odds "SDDR" as well as its logarithm "log(SDDR)"for each structural shock, and the implied posterior probability of the normality and Student-t hypothesis, "Pr[normal|data]" and "Pr[Student-t|data]" respectively.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

verify_identification.PosteriorBSVART

Examples

```
specification  = specify_bsvar_t$new(us_fiscal_lsuw)
posterior      = estimate(specification, 10)

# verify heteroskedasticity
sddr           = verify_identification(posterior)
summary(sddr)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_t$new() |>
  estimate(S = 10) |>
  verify_identification() |>
  summary() -> sddr_summary
```

`summary.SDDRvolatility`

Provides summary of verifying homoskedasticity

Description

Provides summary of the Savage-Dickey density ratios for verification of structural shocks homoskedasticity.

Usage

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

Arguments

`object`	an object of class `SDDRvolatility` obtained using the `verify_volatility()` function.
`...`	additional arguments affecting the summary produced.

Value

A table reporting the logarithm of Bayes factors of homoskedastic to heteroskedastic posterior odds `"log(SDDR)"` for each structural shock, their numerical standard errors `"NSE"`, and the implied posterior probability of the homoskedasticity and heteroskedasticity hypothesis, `"Pr[homoskedasticity|data]"` and `"Pr[heteroskedasticity|data]"` respectively.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

See Also

verify_volatility

Examples

```
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, p = 1, M = 2)
posterior      = estimate(specification, 10)

# verify heteroskedasticity
sddr           = verify_volatility(posterior)
summary(sddr)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new(p = 1, M = 2) |>
  estimate(S = 10) |>
  verify_volatility() |>
  summary() -> sddr_summary
```

---

us_fiscal_cond_forecasts

A matrix to be used in a conditional forecasting example including the projected values of total tax revenue that are projected to increase at an average quarterly sample growth rate. The other two columns are filled with NA *values, which implies that the future values of the corresponding endogenous variables, namely government spending and GDP, will be forecasted given the provided projected values of total tax revenue. The matrix includes future values for the forecast horizon of two years for the US fiscal model for the period 2024 Q3 – 2026 Q1.*

Description

Conditional projections variables to be used in conditional forecasting of government spending and GDP given the provided projected values of total tax revenue. Last data update was implemented on 2026-06-09.

Usage

```
data(us_fiscal_cond_forecasts)
```

Format

A matrix and a `ts` object with time series of eight values on 3 variables:

ttr the values are provided. This variable will not be forecasted.

gs not provided. This variable will be forecasted conditionally on the provided values for ttr.

gdp not provided. This variable will be forecasted conditionally on the provided values for ttr

The series are as described by Mertens & Ravn (2014). The data was used by Lütkepohl, Shang, Uzeda, Woźniak (2024).

References

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

Mertens, K., and Ravn, M.O. (2014) A Reconciliation of SVAR and Narrative Estimates of Tax Multipliers, *Journal of Monetary Economics*, 68(S), S1–S19. DOI: doi:10.1016/j.jmoneco.2013.04.004.

Examples

```
data(us_fiscal_cond_forecasts)   # upload the data
```

`us_fiscal_ex` *A 3-variable system of exogenous variables for the US fiscal model for the period 1948 Q1 – 2026 Q1*

Description

Exogenous variables used to identify the US fiscal policy shocks. Last data update was implemented on 2026-06-09.

Usage

```
data(us_fiscal_ex)
```

Format

A matrix and a `ts` object with time series of over three hundred observations on 3 variables:

t a time trend

t^2 a quadratic trend

1975Q2 a dummy variable taking the value of 1 for quarter 2 1975 and zero elsewhere

The series are as described by Mertens & Ravn (2014). The data was used by Lütkepohl, Shang, Uzeda, Woźniak (2024).

References

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

Mertens, K., and Ravn, M.O. (2014) A Reconciliation of SVAR and Narrative Estimates of Tax Multipliers, *Journal of Monetary Economics*, 68(S), S1–S19. DOI: doi:10.1016/j.jmoneco.2013.04.004.

Examples

```
data(us_fiscal_ex)   # upload the data
plot(us_fiscal_ex)   # plot the data
```

`us_fiscal_ex_forecasts`

A 3-variable system of exogenous variables' future values for the forecast horizon of two years for the US fiscal model for the period 2026 Q2 – 2028 Q1

Description

Exogenous variables to be used in forecasting of the US fiscal policy shocks. Last data update was implemented on 2026-06-09.

Usage

```
data(us_fiscal_ex_forecasts)
```

Format

A matrix and a `ts` object with time series of eight values on 3 variables:

t a time trend

t^2 a quadratic trend

1975Q2 a dummy variable taking the value of 1 for quarter 2 1975 and zero elsewhere

The series are as described by Mertens & Ravn (2014). The data was used by Lütkepohl, Shang, Uzeda, Woźniak (2024).

References

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

Mertens, K., and Ravn, M.O. (2014) A Reconciliation of SVAR and Narrative Estimates of Tax Multipliers, *Journal of Monetary Economics*, 68(S), S1–S19. DOI: doi:10.1016/j.jmoneco.2013.04.004.

Examples

```
data(us_fiscal_ex_forecasts)   # upload the data
```

us_fiscal_lsuw *A 3-variable US fiscal system for the period 1948 Q1 – 2026 Q1*

Description

A system used to identify the US fiscal policy shocks used by Lütkepohl, Shang, Uzeda, Woźniak (2026). Last data update was implemented on 2026-06-09.

Usage

```
data(us_fiscal_lsuw)
```

Format

A matrix and a `ts` object with time series of over three hundred observations on 3 variables:

ttr quarterly US total tax revenue expressed in log, real, per person terms

gs quarterly US total government spending expressed in log, real, per person terms

gdp quarterly US gross domestic product expressed in log, real, per person terms

The series are as described by Mertens & Ravn (2014) in footnote 3 and main body on page S3 of the paper. Differences with respect to Mertens & Ravn's data :

- The sample period is from quarter 1 of 1948 to the last available observation,
- The population variable is not from Francis & Ramey (2009) but from the FRED (with the same definition),
- The original monthly population data is transformed to quarterly by taking monthly averages.

Source

U.S. Bureau of Economic Analysis, National Income and Product Accounts, https://www.bea.gov/

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

References

Francis, N., and Ramey, V.A. (2009) Measures of per capita Hours and Their Implications for the Technology-hours Debate. *Journal of Money, Credit and Banking*, 41(6), 1071-1097, DOI: doi:10.1111/j.15384616.2009.00247.x.

Mertens, K., and Ravn, M.O. (2014) A Reconciliation of SVAR and Narrative Estimates of Tax Multipliers, *Journal of Monetary Economics*, 68(S), S1–S19. DOI: doi:10.1016/j.jmoneco.2013.04.004.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2025) Partial identification of structural vector autoregressions with non-centred stochastic volatility. *Journal of Econometrics* **256**, 106107, doi:10.1016/j.jeconom.2025.106107.

Examples

```
data(us_fiscal_lsuw)   # upload the data
plot(us_fiscal_lsuw)   # plot the data
```

`us_fiscal_sww` *A 10-variable US fiscal system for the period 1948 Q1 – 2025 Q3*

Description

A system used to identify the US fiscal policy shocks used by Shang, Wang, Woźniak (2026). Last data update was implemented on 2026-08-18.

Usage

```
data(us_fiscal_sww)
```

Format

A matrix and a `ts` object with time series of over three hundred observations on 10 variables:

ttr quarterly US total tax revenue expressed in log, real, per person terms

gs quarterly US total government spending expressed in log, real, per person terms

gdp quarterly US gross domestic product expressed in log, real, per person terms

FFR quarterly Federal Funds Effective Rate

cons quarterly private consumption expressed in log, real, per person terms

rw quarterly real wages expressed in log, real, per person terms

inv quarterly private non-residential investment expressed in log, real, per person terms

m2 quarterly Monetary Base M2SL expressed in log, real, per person terms

ppiic quarterly Producer Price Index by Commodity: Industrial Commodities expressed in log, real, per person terms

pi quarterly inflation rate expressed in log, real, per person terms

The system was defined by Mountford, Uhlig (2009) and used by Shang, Wang, Woźniak (2026).

Source

U.S. Bureau of Economic Analysis, National Income and Product Accounts, https://www.bea.gov/

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

References

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2025) Partial identification of structural vector autoregressions with non-centred stochastic volatility. *Journal of Econometrics* **256**, 106107, doi:10.1016/j.jeconom.2025.106107.

Mountford, A. and H. Uhlig (2009) What are the effects of fiscal policy shocks? *Journal of Applied Econometrics* **24**, 960–992., doi:10.1002/jae.1079.

Examples

```
data(us_fiscal_sww)   # upload the data
plot(us_fiscal_sww)   # plot the data
```

`verify_autoregression` *Verifies hypotheses involving autoregressive parameters*

Description

Computes the logarithm of Bayes factor for the joint hypothesis, H_0, possibly for many autoregressive parameters represented by argument `hypothesis` via Savage-Dickey Density Ration (SDDR). The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(H_0|data) - logp(H_0)$$

Therefore, a negative value of the difference is the evidence against hypothesis. The estimation of both elements of the difference requires numerical integration.

Usage

```
verify_autoregression(posterior, hypothesis)
```

Arguments

`posterior`	the `posterior` element of the list from the estimation outcome
`hypothesis`	an `NxK` matrix of the same dimension as the autoregressive matrix A with numeric values for the parameters to be verified, in which case the values represent the joint hypothesis, and missing value `NA` for these parameters that are not tested

Value

An object of class `SDDRautoregression` that is a list of three components:

`logSDDR` a scalar with values of the logarithm of the Bayes factors for the autoregressive hypothesis for each of the shocks

`log_SDDR_se` an `N`-vector with estimation standard errors of the logarithm of the Bayes factors reported in output element `logSDDR` that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

`components` a list of three components for the computation of the Bayes factor

log_denominator an `N`-vector with values of the logarithm of the Bayes factor denominators

log_numerator an `N`-vector with values of the logarithm of the Bayes factor numerators

log_numerator_s an `NxS` matrix of the log-full conditional posterior density ordinates computed to estimate the numerator

log_denominator_s an NxS matrix of the log-full conditional posterior density ordinates computed to estimate the denominator

se_components a 30-vector containing the log-Bayes factors on the basis of which the standard errors are computed

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Woźniak, T., and Droumaguet, M., (2024) Bayesian Assessment of Identifying Restrictions for Heteroskedastic Structural VARs

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar$new(us_fiscal_lsuw, p = 1)

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

# verify autoregression
H0             = matrix(NA, ncol(us_fiscal_lsuw), ncol(us_fiscal_lsuw) + 1)
H0[1,3]        = 0        # a hypothesis of no Granger causality from gdp to ttr
sddr           = verify_autoregression(posterior, H0)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new(p = 1) |>
  estimate(S = 10) |>
  verify_autoregression(hypothesis = H0) -> sddr
```

---

verify_autoregression.PosteriorBSVAR

Verifies hypotheses involving autoregressive parameters

Description

Computes the logarithm of Bayes factor for the joint hypothesis, H_0, possibly for many autoregressive parameters represented by argument `hypothesis` via Savage-Dickey Density Ration (SDDR). The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(H_0|data) - logp(H_0)$$

Therefore, a negative value of the difference is the evidence against hypothesis. The estimation of both elements of the difference requires numerical integration.

Usage

```
## S3 method for class 'PosteriorBSVAR'
verify_autoregression(posterior, hypothesis)
```

Arguments

`posterior`	the `posterior` element of the list from the estimation outcome
`hypothesis`	an `NxK` matrix of the same dimension as the autoregressive matrix A with numeric values for the parameters to be verified, in which case the values represent the joint hypothesis, and missing value `NA` for these parameters that are not tested

Value

An object of class `SDDRautoregression` that is a list of three components:

`logSDDR` a scalar with values of the logarithm of the Bayes factors for the autoregressive hypothesis for each of the shocks

`log_SDDR_se` an `N`-vector with estimation standard errors of the logarithm of the Bayes factors reported in output element `logSDDR` that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

`components` a list of three components for the computation of the Bayes factor

log_denominator an `N`-vector with values of the logarithm of the Bayes factor denominators

log_numerator an `N`-vector with values of the logarithm of the Bayes factor numerators

log_numerator_s an `NxS` matrix of the log-full conditional posterior density ordinates computed to estimate the numerator

log_denominator_s an `NxS` matrix of the log-full conditional posterior density ordinates computed to estimate the denominator

se_components a `30`-vector containing the log-Bayes factors on the basis of which the standard errors are computed

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Woźniak, T., and Droumaguet, M., (2024) Bayesian Assessment of Identifying Restrictions for Heteroskedastic Structural VARs

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar$new(us_fiscal_lsuw, p = 1)

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

# verify autoregression
H0             = matrix(NA, ncol(us_fiscal_lsuw), ncol(us_fiscal_lsuw) + 1)
H0[1,3]        = 0        # a hypothesis of no Granger causality from gdp to ttr
sddr           = verify_autoregression(posterior, H0)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new(p = 1) |>
  estimate(S = 10) |>
  verify_autoregression(hypothesis = H0) -> sddr
```

`verify_autoregression.PosteriorBSVAREXH`
Verifies hypotheses involving autoregressive parameters

Description

Computes the logarithm of Bayes factor for the joint hypothesis, H_0, possibly for many autoregressive parameters represented by argument `hypothesis` via Savage-Dickey Density Ration (SDDR). The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(H_0|data) - logp(H_0)$$

Therefore, a negative value of the difference is the evidence against hypothesis. The estimation of both elements of the difference requires numerical integration.

Usage

```
## S3 method for class 'PosteriorBSVAREXH'
verify_autoregression(posterior, hypothesis)
```

Arguments

`posterior`	the `posterior` element of the list from the estimation outcome
`hypothesis`	an `NxK` matrix of the same dimension as the autoregressive matrix A with numeric values for the parameters to be verified, in which case the values represent the joint hypothesis, and missing value `NA` for these parameters that are not tested

Value

An object of class SDDRautoregression that is a list of three components:

logSDDR a scalar with values of the logarithm of the Bayes factors for the autoregressive hypothesis for each of the shocks

log_SDDR_se an N-vector with estimation standard errors of the logarithm of the Bayes factors reported in output element logSDDR that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

components a list of three components for the computation of the Bayes factor

log_denominator an N-vector with values of the logarithm of the Bayes factor denominators

log_numerator an N-vector with values of the logarithm of the Bayes factor numerators

log_numerator_s an NxS matrix of the log-full conditional posterior density ordinates computed to estimate the numerator

log_denominator_s an NxS matrix of the log-full conditional posterior density ordinates computed to estimate the denominator

se_components a 30-vector containing the log-Bayes factors on the basis of which the standard errors are computed

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Woźniak, T., and Droumaguet, M., (2024) Bayesian Assessment of Identifying Restrictions for Heteroskedastic Structural VARs

Examples

```
# simple workflow
############################################################
# specify the model
spec  = specify_bsvar_exh$new(us_fiscal_lsuw)

# estimate the model
post  = estimate(spec, 10)

# verify autoregression
H0             = matrix(NA, ncol(us_fiscal_lsuw), ncol(us_fiscal_lsuw) + 1)
H0[1,3]        = 0        # a hypothesis of no Granger causality from gdp to ttr
sddr           = verify_autoregression(post, H0)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_exh$new() |>
  estimate(S = 10) |>
  verify_autoregression(hypothesis = H0) -> sddr
```

`verify_autoregression.PosteriorBSVARHMSH` *Verifies hypotheses involving autoregressive parameters*

Description

Computes the logarithm of Bayes factor for the joint hypothesis, H_0, possibly for many autoregressive parameters represented by argument `hypothesis` via Savage-Dickey Density Ration (SDDR). The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(H_0|data) - logp(H_0)$$

Therefore, a negative value of the difference is the evidence against hypothesis. The estimation of both elements of the difference requires numerical integration.

Usage

```
## S3 method for class 'PosteriorBSVARHMSH'
verify_autoregression(posterior, hypothesis)
```

Arguments

`posterior`	the `posterior` element of the list from the estimation outcome
`hypothesis`	an `NxK` matrix of the same dimension as the autoregressive matrix A with numeric values for the parameters to be verified, in which case the values represent the joint hypothesis, and missing value `NA` for these parameters that are not tested

Value

An object of class `SDDRautoregression` that is a list of three components:

`logSDDR` a scalar with values of the logarithm of the Bayes factors for the autoregressive hypothesis for each of the shocks

`log_SDDR_se` an `N`-vector with estimation standard errors of the logarithm of the Bayes factors reported in output element `logSDDR` that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

`components` a list of three components for the computation of the Bayes factor

log_denominator an `N`-vector with values of the logarithm of the Bayes factor denominators

log_numerator an `N`-vector with values of the logarithm of the Bayes factor numerators

log_numerator_s an `NxS` matrix of the log-full conditional posterior density ordinates computed to estimate the numerator

log_denominator_s an `NxS` matrix of the log-full conditional posterior density ordinates computed to estimate the denominator

se_components a `30`-vector containing the log-Bayes factors on the basis of which the standard errors are computed

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Woźniak, T., and Droumaguet, M., (2024) Bayesian Assessment of Identifying Restrictions for Heteroskedastic Structural VARs

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw)

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

# verify autoregression
H0             = matrix(NA, ncol(us_fiscal_lsuw), ncol(us_fiscal_lsuw) + 1)
H0[1,3]        = 0        # a hypothesis of no Granger causality from gdp to ttr
sddr           = verify_autoregression(posterior, H0)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_hmsh$new() |>
  estimate(S = 10) |>
  verify_autoregression(hypothesis = H0) -> sddr
```

---

verify_autoregression.PosteriorBSVARMIX

Verifies hypotheses involving autoregressive parameters

Description

Computes the logarithm of Bayes factor for the joint hypothesis, H_0, possibly for many autoregressive parameters represented by argument `hypothesis` via Savage-Dickey Density Ration (SDDR). The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(H_0|data) - logp(H_0)$$

Therefore, a negative value of the difference is the evidence against hypothesis. The estimation of both elements of the difference requires numerical integration.

Usage

```
## S3 method for class 'PosteriorBSVARMIX'
verify_autoregression(posterior, hypothesis)
```

Arguments

posterior	the posterior element of the list from the estimation outcome
hypothesis	an NxK matrix of the same dimension as the autoregressive matrix A with numeric values for the parameters to be verified, in which case the values represent the joint hypothesis, and missing value NA for these parameters that are not tested

Value

An object of class SDDRautoregression that is a list of three components:

logSDDR a scalar with values of the logarithm of the Bayes factors for the autoregressive hypothesis for each of the shocks

log_SDDR_se an N-vector with estimation standard errors of the logarithm of the Bayes factors reported in output element logSDDR that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

components a list of three components for the computation of the Bayes factor

log_denominator an N-vector with values of the logarithm of the Bayes factor denominators

log_numerator an N-vector with values of the logarithm of the Bayes factor numerators

log_numerator_s an NxS matrix of the log-full conditional posterior density ordinates computed to estimate the numerator

log_denominator_s an NxS matrix of the log-full conditional posterior density ordinates computed to estimate the denominator

se_components a 30-vector containing the log-Bayes factors on the basis of which the standard errors are computed

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Woźniak, T., and Droumaguet, M., (2024) Bayesian Assessment of Identifying Restrictions for Heteroskedastic Structural VARs

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, p = 1, M = 2)

# estimate the model
```

```
posterior      = estimate(specification, 10)

# verify autoregression
H0             = matrix(NA, ncol(us_fiscal_lsuw), ncol(us_fiscal_lsuw) + 1)
H0[1,3]        = 0        # a hypothesis of no Granger causality from gdp to ttr
sddr           = verify_autoregression(posterior, H0)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_mix$new(p = 1, M = 2) |>
  estimate(S = 10) |>
  verify_autoregression(hypothesis = H0) -> sddr
```

`verify_autoregression.PosteriorBSVARMSH`
Verifies hypotheses involving autoregressive parameters

Description

Computes the logarithm of Bayes factor for the joint hypothesis, H_0, possibly for many autoregressive parameters represented by argument `hypothesis` via Savage-Dickey Density Ration (SDDR). The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(H_0|data) - logp(H_0)$$

Therefore, a negative value of the difference is the evidence against hypothesis. The estimation of both elements of the difference requires numerical integration.

Usage

```
## S3 method for class 'PosteriorBSVARMSH'
verify_autoregression(posterior, hypothesis)
```

Arguments

`posterior`	the `posterior` element of the list from the estimation outcome
`hypothesis`	an `NxK` matrix of the same dimension as the autoregressive matrix A with numeric values for the parameters to be verified, in which case the values represent the joint hypothesis, and missing value `NA` for these parameters that are not tested

Value

An object of class SDDRautoregression that is a list of three components:

logSDDR a scalar with values of the logarithm of the Bayes factors for the autoregressive hypothesis for each of the shocks

log_SDDR_se an N-vector with estimation standard errors of the logarithm of the Bayes factors reported in output element logSDDR that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

components a list of three components for the computation of the Bayes factor

log_denominator an N-vector with values of the logarithm of the Bayes factor denominators

log_numerator an N-vector with values of the logarithm of the Bayes factor numerators

log_numerator_s an NxS matrix of the log-full conditional posterior density ordinates computed to estimate the numerator

log_denominator_s an NxS matrix of the log-full conditional posterior density ordinates computed to estimate the denominator

se_components a 30-vector containing the log-Bayes factors on the basis of which the standard errors are computed

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Woźniak, T., and Droumaguet, M., (2024) Bayesian Assessment of Identifying Restrictions for Heteroskedastic Structural VARs

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, p = 1, M = 2)
set.seed(123)

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

# verify autoregression
H0             = matrix(NA, ncol(us_fiscal_lsuw), ncol(us_fiscal_lsuw) + 1)
H0[1,3]        = 0        # a hypothesis of no Granger causality from gdp to ttr
sddr           = verify_autoregression(posterior, H0)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new(p = 1, M = 2) |>
  estimate(S = 10) |>
```

```
verify_autoregression(hypothesis = H0) -> sddr
```

`verify_autoregression.PosteriorBSVARSV` *Verifies hypotheses involving autoregressive parameters*

Description

Computes the logarithm of Bayes factor for the joint hypothesis, H_0, possibly for many autoregressive parameters represented by argument `hypothesis` via Savage-Dickey Density Ration (SDDR). The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(H_0|data) - logp(H_0)$$

Therefore, a negative value of the difference is the evidence against hypothesis. The estimation of both elements of the difference requires numerical integration.

Usage

```
## S3 method for class 'PosteriorBSVARSV'
verify_autoregression(posterior, hypothesis)
```

Arguments

`posterior`	the `posterior` element of the list from the estimation outcome
`hypothesis`	an `NxK` matrix of the same dimension as the autoregressive matrix A with numeric values for the parameters to be verified, in which case the values represent the joint hypothesis, and missing value `NA` for these parameters that are not tested

Value

An object of class `SDDRautoregression` that is a list of three components:

`logSDDR` a scalar with values of the logarithm of the Bayes factors for the autoregressive hypothesis for each of the shocks

`log_SDDR_se` an `N`-vector with estimation standard errors of the logarithm of the Bayes factors reported in output element `logSDDR` that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

`components` a list of three components for the computation of the Bayes factor

log_denominator an `N`-vector with values of the logarithm of the Bayes factor denominators

log_numerator an `N`-vector with values of the logarithm of the Bayes factor numerators

log_numerator_s an `NxS` matrix of the log-full conditional posterior density ordinates computed to estimate the numerator

log_denominator_s an NxS matrix of the log-full conditional posterior density ordinates computed to estimate the denominator

se_components a 30-vector containing the log-Bayes factors on the basis of which the standard errors are computed

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Woźniak, T., and Droumaguet, M., (2024) Bayesian Assessment of Identifying Restrictions for Heteroskedastic Structural VARs

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar_sv$new(us_fiscal_lsuw, p = 1)

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

# verify autoregression
H0             = matrix(NA, ncol(us_fiscal_lsuw), ncol(us_fiscal_lsuw) + 1)
H0[1,3]        = 0        # a hypothesis of no Granger causality from gdp to ttr
sddr           = verify_autoregression(posterior, H0)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_sv$new(p = 1) |>
  estimate(S = 10) |>
  verify_autoregression(hypothesis = H0) -> sddr
```

---

verify_autoregression.PosteriorBSVART

Verifies hypotheses involving autoregressive parameters

Description

Computes the logarithm of Bayes factor for the joint hypothesis, H_0, possibly for many autoregressive parameters represented by argument `hypothesis` via Savage-Dickey Density Ration (SDDR). The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(H_0|data) - logp(H_0)$$

Therefore, a negative value of the difference is the evidence against hypothesis. The estimation of both elements of the difference requires numerical integration.

Usage

```
## S3 method for class 'PosteriorBSVART'
verify_autoregression(posterior, hypothesis)
```

Arguments

`posterior`	the `posterior` element of the list from the estimation outcome
`hypothesis`	an `NxK` matrix of the same dimension as the autoregressive matrix A with numeric values for the parameters to be verified, in which case the values represent the joint hypothesis, and missing value `NA` for these parameters that are not tested

Value

An object of class `SDDRautoregression` that is a list of three components:

`logSDDR` a scalar with values of the logarithm of the Bayes factors for the autoregressive hypothesis for each of the shocks

`log_SDDR_se` an `N`-vector with estimation standard errors of the logarithm of the Bayes factors reported in output element `logSDDR` that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

`components` a list of three components for the computation of the Bayes factor

log_denominator an `N`-vector with values of the logarithm of the Bayes factor denominators

log_numerator an `N`-vector with values of the logarithm of the Bayes factor numerators

log_numerator_s an `NxS` matrix of the log-full conditional posterior density ordinates computed to estimate the numerator

log_denominator_s an `NxS` matrix of the log-full conditional posterior density ordinates computed to estimate the denominator

se_components a `30`-vector containing the log-Bayes factors on the basis of which the standard errors are computed

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Woźniak, T., and Droumaguet, M., (2024) Bayesian Assessment of Identifying Restrictions for Heteroskedastic Structural VARs

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar_t$new(us_fiscal_lsuw)
set.seed(123)

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

# verify autoregression
H0             = matrix(NA, ncol(us_fiscal_lsuw), ncol(us_fiscal_lsuw) + 1)
H0[1,3]        = 0        # a hypothesis of no Granger causality from gdp to ttr
sddr           = verify_autoregression(posterior, H0)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_t$new() |>
  estimate(S = 10) |>
  verify_autoregression(hypothesis = H0) -> sddr
```

`verify_identification` *Verifies identification through heteroskedasticity or non-normality of of structural shocks*

Description

Computes the logarithm of Bayes factor(s) for the hypothesis in which the model is not identified through heteroskedasticity of non-normality using Savage-Dickey Density Ration (SDDR). The hypothesis of no such identification, H_0, is represented by model-specific restrictions.Consult help files for individual classes of models for details. The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of the logarithm of the marginal posterior distribution ordinate at the restriction less the log-marginal prior distribution ordinate at the same point:

$$logp(H_0|data) - logp(H_0)$$

Therefore, a negative value of the difference is the evidence against the lack of identification of the structural shock through heteroskedasticity or non-normality.

Usage

```
verify_identification(posterior)
```

Arguments

`posterior`	the estimation outcome obtained using `estimate` function

Value

An object of class `SDDRid*` that is a list with components:

`logSDDR` a vector with values of the logarithm of the Bayes factors

`log_SDDR_se` a vector with numerical standard errors of the logarithm of the Bayes factors reported in output element `logSDDR` that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

Author(s)

Tomasz Woźniak `<wozniak.tom@pm.me>`

References

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

See Also

`verify_identification.PosteriorBSVAR`, `verify_identification.PosteriorBSVARSV`, `verify_identification.Po`
`verify_identification.PosteriorBSVARMSH`, `verify_identification.PosteriorBSVART`

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar_sv$new(us_fiscal_lsuw, p = 1)

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

# verify heteroskedasticity
sddr           = verify_identification(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_sv$new(p = 1) |>
  estimate(S = 10) |>
  verify_identification() -> sddr
```

`verify_identification.PosteriorBSVAR` *Verifies identification through heteroskedasticity or non-normality of of structural shocks*

Description

Displays information that the model is homoskedastic and with normal shocks.

Usage

```
## S3 method for class 'PosteriorBSVAR'
verify_identification(posterior)
```

Arguments

`posterior`	the estimation outcome obtained using `estimate` function

Value

Nothing. Just displays a message.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

See Also

`verify_identification.PosteriorBSVAR`, `verify_identification.PosteriorBSVARSV`, `verify_identification.Po`
`verify_identification.PosteriorBSVARMSH`, `verify_identification.PosteriorBSVART`

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar$new(us_fiscal_lsuw, p = 1)

# estimate the model
```

```
posterior      = estimate(specification, 10)

# verify heteroskedasticity
sddr           = verify_identification(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new(p = 1) |>
  estimate(S = 10) |>
  verify_identification() -> sddr
```

`verify_identification.PosteriorBSVAREXH`

Verifies identification through heteroskedasticity or non-normality of of structural shocks

---

Description

Computes the logarithm of Bayes factor for the homoskedasticity hypothesis for each of the structural shocks via Savage-Dickey Density Ration (SDDR). The hypothesis of homoskedasticity is represented by restriction:

$$H_0 : \sigma^2_{n.1} = ... = \sigma^2_{n.M} = 1$$

The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(H_0|data) - logp(H_0)$$

Therefore, a negative value of the difference is the evidence against homoskedasticity of the structural shock. The estimation of both elements of the difference requires numerical integration.

Usage

```
## S3 method for class 'PosteriorBSVAREXH'
verify_identification(posterior)
```

Arguments

`posterior`	the estimation outcome obtained using `estimate` function

Value

An object of class `SDDRid*` that is a list with components:

`logSDDR` a vector with values of the logarithm of the Bayes factors

`log_SDDR_se` a vector with numerical standard errors of the logarithm of the Bayes factors reported in output element `logSDDR` that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

See Also

specify_bsvar_exh, estimate

Examples

```
# simple workflow
############################################################
# specify the model
spec  = specify_bsvar_exh$new(us_fiscal_lsuw)

# estimate the model
post  = estimate(spec, 10)

# verify heteroskedasticity
sddr  = verify_identification(post)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_exh$new() |>
  estimate(S = 10) |>
  verify_identification() -> sddr
```

---

verify_identification.PosteriorBSVARHMSH

Verifies identification through heteroskedasticity or non-normality of of structural shocks

---

Description

Computes the logarithm of Bayes factor for the homoskedasticity hypothesis for each of the structural shocks via Savage-Dickey Density Ration (SDDR). The hypothesis of homoskedasticity is represented by restriction:

$$H_0 : \sigma^2_{n.1} = ... = \sigma^2_{n.M} = 1$$

The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(H_0|data) - logp(H_0)$$

Therefore, a negative value of the difference is the evidence against homoskedasticity of the structural shock. The estimation of both elements of the difference requires numerical integration.

Usage

```
## S3 method for class 'PosteriorBSVARHMSH'
verify_identification(posterior)
```

Arguments

`posterior`	the estimation outcome obtained using `estimate` function

Value

An object of class `SDDRid*` that is a list with components:

`logSDDR` a vector with values of the logarithm of the Bayes factors

`log_SDDR_se` a vector with numerical standard errors of the logarithm of the Bayes factors reported in output element `logSDDR` that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

See Also

`specify_bsvar_hmsh`, `estimate`

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar_hmsh$new(us_fiscal_lsuw)
```

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

# verify heteroskedasticity
sddr           = verify_identification(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new() |>
  estimate(S = 10) |>
  verify_identification() -> sddr
```

`verify_identification.PosteriorBSVARMIX` *Verifies identification through heteroskedasticity or non-normality of of structural shocks*

Description

Computes the logarithm of Bayes factor for the hypothesis of normality for each of the structural shocks via Savage-Dickey Density Ration (SDDR). The hypothesis of normality in this mixture of normals model is represented by restriction:

$$H_0 : \sigma^2_{n.1} = ... = \sigma^2_{n.M} = 1$$

The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(H_0|data) - logp(H_0)$$

Therefore, a negative value of the difference is the evidence against homoskedasticity of the structural shock. The estimation of both elements of the difference requires numerical integration.

Usage

```
## S3 method for class 'PosteriorBSVARMIX'
verify_identification(posterior)
```

Arguments

`posterior`	the estimation outcome obtained using `estimate` function

Value

An object of class `SDDRid*` that is a list with components:

`logSDDR` a vector with values of the logarithm of the Bayes factors

`log_SDDR_se` a vector with numerical standard errors of the logarithm of the Bayes factors reported in output element `logSDDR` that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

See Also

specify_bsvar_mix, estimate

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, p = 1, M = 2)

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

# verify heteroskedasticity
sddr           = verify_identification(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_mix$new(p = 1, M = 2) |>
  estimate(S = 10) |>
  verify_identification() -> sddr
```

---

verify_identification.PosteriorBSVARMSH

Verifies identification through heteroskedasticity or non-normality of of structural shocks

---

Description

Computes the logarithm of Bayes factor for the homoskedasticity hypothesis for each of the structural shocks via Savage-Dickey Density Ration (SDDR). The hypothesis of homoskedasticity is represented by restriction:

$$H_0 : \sigma^2_{n.1} = ... = \sigma^2_{n.M} = 1$$

The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(H_0|data) - logp(H_0)$$

Therefore, a negative value of the difference is the evidence against homoskedasticity of the structural shock. The estimation of both elements of the difference requires numerical integration.

Usage

```
## S3 method for class 'PosteriorBSVARMSH'
verify_identification(posterior)
```

Arguments

`posterior`	the estimation outcome obtained using `estimate` function

Value

An object of class `SDDRid*` that is a list with components:

`logSDDR` a vector with values of the logarithm of the Bayes factors

`log_SDDR_se` a vector with numerical standard errors of the logarithm of the Bayes factors reported in output element `logSDDR` that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

See Also

`specify_bsvar_msh`, `estimate`

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, p = 1, M = 2)
```

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

# verify heteroskedasticity
sddr           = verify_identification(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new(p = 1, M = 2) |>
  estimate(S = 10) |>
  verify_identification() -> sddr
```

`verify_identification.PosteriorBSVARSV` *Verifies identification through heteroskedasticity or non-normality of of structural shocks*

Description

Computes the logarithm of Bayes factor for the homoskedasticity hypothesis for each of the structural shocks via Savage-Dickey Density Ratio (SDDR). The hypothesis of homoskedasticity for the structural shock `n` is represented by restriction:

$$H_0 : \omega_n = 0$$

The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of the logarithm of the marginal posterior distribution ordinate at the restriction less the log-marginal prior distribution ordinate at the same point:

$$logp(\omega_n = 0|data) - logp(\omega_n = 0)$$

Therefore, a negative value of the difference is the evidence against homoskedasticity of the structural shock. The estimation of both elements of the difference requires numerical integration.

Usage

```
## S3 method for class 'PosteriorBSVARSV'
verify_identification(posterior)
```

Arguments

`posterior`	the estimation outcome obtained using `estimate` function

Value

An object of class `SDDRid*` that is a list with components:

`logSDDR` a vector with values of the logarithm of the Bayes factors

`log_SDDR_se` a vector with numerical standard errors of the logarithm of the Bayes factors reported in output element `logSDDR` that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

See Also

specify_bsvar_sv, estimate

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar_sv$new(us_fiscal_lsuw, p = 1)

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

# verify heteroskedasticity
sddr           = verify_identification(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_sv$new(p = 1) |>
  estimate(S = 10) |>
  verify_identification() -> sddr
```

---

verify_identification.PosteriorBSVART

Verifies identification through heteroskedasticity or non-normality of of structural shocks

---

Description

Computes the logarithm of Bayes factor for the hypothesis of normality of the joint conditional distribution of the structural shocks via Savage-Dickey Density Ration (SDDR). The hypothesis

of normality in this t-distributed shocks model is represented by restriction setting the degrees-of-freedom parameter ν to infinity:

$$H_0 : \nu = \infty$$

The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(H_0|data) - logp(H_0)$$

Therefore, a negative value of the difference is the evidence against homoskedasticity of the structural shock. The estimation of the marginal posterior ordinate is done using truncated Gaussian kernel smoothing.

Usage

```
## S3 method for class 'PosteriorBSVART'
verify_identification(posterior)
```

Arguments

`posterior`	the estimation outcome obtained using `estimate` function

Value

An object of class `SDDRidT` that is a list with components:

`logSDDR` the value of the logarithm of the Bayes factor

`SDDR` the value of the Bayes factor

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

See Also

`specify_bsvar_t`, `estimate`

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar_t$new(us_fiscal_lsuw)

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

# verify heteroskedasticity
sddr           = verify_identification(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_t$new() |>
  estimate(S = 10) |>
  verify_identification() -> sddr
```

`verify_normality` *Verifies normality of structural shocks equation by equation*

Description

Computes the logarithm of Bayes factor for the normality hypothesis for each of the structural shocks via Savage-Dickey Density Ration (SDDR). The hypothesis of normality, H_0, is represented by restriction that the equation-specific degrees of freedom parameter is equal to infinity, $\nu_n \to \infty$. The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(H_0|data) - logp(H_0)$$

Therefore, a negative value of the difference is the evidence against normality of the structural shock. The estimation of th first element relies on kernel density estimation of the marginal posterior density, whereas the second element is equal to the log of value 1.

Usage

```
verify_normality(posterior)
```

Arguments

`posterior`	the `posterior` element of the list from the estimation outcome

Value

An object of class SDDRnormality that is a list of three components:

logSDDR an N-vector with values of the logarithm of the Bayes factors for the normality hypothesis for each of the shocks

log_SDDR_se an N-vector with estimation standard errors of the logarithm of the Bayes factors reported in output element logSDDR that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar_sv$new(us_fiscal_lsuw, distribution = "t")

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

# verify heteroskedasticity
sddr           = verify_normality(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_sv$new(distribution = "t") |>
  estimate(S = 10) |>
  verify_normality() -> sddr
```

---

verify_normality.PosteriorBSVAREXH

Verifies normality of structural shocks equation by equation

Description

Computes the logarithm of Bayes factor for the normality hypothesis for each of the structural shocks via Savage-Dickey Density Ration (SDDR). The hypothesis of normality, H_0, is represented by restriction that the equation-specific degrees of freedom parameter is equal to infinity, $\nu_n \to \infty$. The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(H_0|data) - logp(H_0)$$

Therefore, a negative value of the difference is the evidence against normality of the structural shock. The estimation of th first element relies on kernel density estimation of the marginal posterior density, whereas the second element is equal to the log of value 1.

Usage

```
## S3 method for class 'PosteriorBSVAREXH'
verify_normality(posterior)
```

Arguments

`posterior`	the `posterior` element of the list from the estimation outcome

Value

An object of class `SDDRnormality` that is a list of three components:

`logSDDR` an `N`-vector with values of the logarithm of the Bayes factors for the normality hypothesis for each of the shocks

`log_SDDR_se` an `N`-vector with estimation standard errors of the logarithm of the Bayes factors reported in output element `logSDDR` that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

See Also

`specify_bsvar_exh`, `estimate`

Examples

```
# simple workflow
############################################################
# specify the model
spec  = specify_bsvar_exh$new(us_fiscal_lsuw, distribution = "t")

# estimate the model
post  = estimate(spec, 10)

# verify heteroskedasticity
sddr  = verify_normality(post)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_exh$new(distribution = "t") |>
  estimate(S = 10) |>
  verify_normality() -> sddr
```

---

verify_normality.PosteriorBSVARHMSH

Verifies normality of structural shocks equation by equation

Description

Computes the logarithm of Bayes factor for the normality hypothesis for each of the structural shocks via Savage-Dickey Density Ration (SDDR). The hypothesis of normality, H_0, is represented by restriction that the equation-specific degrees of freedom parameter is equal to infinity, $\nu_n \to \infty$. The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(H_0|data) - logp(H_0)$$

Therefore, a negative value of the difference is the evidence against normality of the structural shock. The estimation of th first element relies on kernel density estimation of the marginal posterior density, whereas the second element is equal to the log of value 1.

Usage

```
## S3 method for class 'PosteriorBSVARHMSH'
verify_normality(posterior)
```

Arguments

`posterior`	the `posterior` element of the list from the estimation outcome

Value

An object of class SDDRnormality that is a list of three components:

logSDDR an N-vector with values of the logarithm of the Bayes factors for the normality hypothesis for each of the shocks

log_SDDR_se an N-vector with estimation standard errors of the logarithm of the Bayes factors reported in output element logSDDR that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

See Also

specify_bsvar_hmsh, estimate

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, M = 2, distribution = "t")

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

# verify heteroskedasticity
sddr           = verify_normality(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new(M = 2, distribution = "t") |>
  estimate(S = 10) |>
  verify_normality() -> sddr
```

`verify_normality.PosteriorBSVARMIX` *Verifies normality of structural shocks equation by equation*

Description

Computes the logarithm of Bayes factor for the normality hypothesis for each of the structural shocks via Savage-Dickey Density Ration (SDDR). The hypothesis of normality, H_0, is represented by restriction that the equation-specific degrees of freedom parameter is equal to infinity, $\nu_n \to \infty$. The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(H_0|data) - logp(H_0)$$

Therefore, a negative value of the difference is the evidence against normality of the structural shock. The estimation of th first element relies on kernel density estimation of the marginal posterior density, whereas the second element is equal to the log of value 1.

Usage

```
## S3 method for class 'PosteriorBSVARMIX'
verify_normality(posterior)
```

Arguments

`posterior`	the `posterior` element of the list from the estimation outcome

Value

An object of class `SDDRnormality` that is a list of three components:

`logSDDR` an `N`-vector with values of the logarithm of the Bayes factors for the normality hypothesis for each of the shocks

`log_SDDR_se` an `N`-vector with estimation standard errors of the logarithm of the Bayes factors reported in output element `logSDDR` that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

Author(s)

Tomasz Woźniak `<wozniak.tom@pm.me>`

References

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

See Also

specify_bsvar_mix, estimate

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, M = 2, distribution = "t")

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

# verify heteroskedasticity
sddr           = verify_normality(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_mix$new(M = 2, distribution = "t") |>
  estimate(S = 10) |>
  verify_normality() -> sddr
```

---

verify_normality.PosteriorBSVARMSH

Verifies normality of structural shocks equation by equation

Description

Computes the logarithm of Bayes factor for the normality hypothesis for each of the structural shocks via Savage-Dickey Density Ration (SDDR). The hypothesis of normality, H_0, is represented by restriction that the equation-specific degrees of freedom parameter is equal to infinity, $\nu_n \to \infty$. The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(H_0|data) - logp(H_0)$$

Therefore, a negative value of the difference is the evidence against normality of the structural shock. The estimation of th first element relies on kernel density estimation of the marginal posterior density, whereas the second element is equal to the log of value 1.

Usage

```
## S3 method for class 'PosteriorBSVARMSH'
verify_normality(posterior)
```

Arguments

posterior	the posterior element of the list from the estimation outcome

Value

An object of class SDDRnormality that is a list of three components:

logSDDR an N-vector with values of the logarithm of the Bayes factors for the normality hypothesis for each of the shocks

log_SDDR_se an N-vector with estimation standard errors of the logarithm of the Bayes factors reported in output element logSDDR that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

See Also

specify_bsvar_msh, estimate

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, M = 2, distribution = "t")

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

# verify heteroskedasticity
sddr           = verify_normality(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new(M = 2, distribution = "t") |>
  estimate(S = 10) |>
  verify_normality() -> sddr
```

`verify_normality.PosteriorBSVARSV` *Verifies normality of structural shocks equation by equation*

Description

Computes the logarithm of Bayes factor for the normality hypothesis for each of the structural shocks via Savage-Dickey Density Ration (SDDR). The hypothesis of normality, H_0, is represented by restriction that the equation-specific degrees of freedom parameter is equal to infinity, $\nu_n \to \infty$. The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(H_0|data) - logp(H_0)$$

Therefore, a negative value of the difference is the evidence against normality of the structural shock. The estimation of th first element relies on kernel density estimation of the marginal posterior density, whereas the second element is equal to the log of value 1.

Usage

```
## S3 method for class 'PosteriorBSVARSV'
verify_normality(posterior)
```

Arguments

`posterior`	the `posterior` element of the list from the estimation outcome

Value

An object of class `SDDRnormality` that is a list of three components:

`logSDDR` an `N`-vector with values of the logarithm of the Bayes factors for the normality hypothesis for each of the shocks

`log_SDDR_se` an `N`-vector with estimation standard errors of the logarithm of the Bayes factors reported in output element `logSDDR` that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

See Also

specify_bsvar_sv, estimate

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar_sv$new(us_fiscal_lsuw, distribution = "t")

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

# verify heteroskedasticity
sddr           = verify_normality(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_sv$new(distribution = "t") |>
  estimate(S = 10) |>
  verify_normality() -> sddr
```

---

verify_volatility *Verifies heteroskedasticity of structural shocks equation by equation*

Description

This function will be deprecated starting from version 4.0. It is replaced by verify_identification function.

Computes the logarithm of Bayes factor for the homoskedasticity hypothesis for each of the structural shocks via Savage-Dickey Density Ration (SDDR). The hypothesis of homoskedasticity, H_0, is represented by model-specific restrictions. Consult help files for individual classes of models for details. The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(H_0|data) - logp(H_0)$$

Therefore, a negative value of the difference is the evidence against homoskedasticity of the structural shock. The estimation of both elements of the difference requires numerical integration.

Usage

```
verify_volatility(posterior)
```

Arguments

`posterior`	the `posterior` element of the list from the estimation outcome

Value

An object of class `SDDRvolatility` that is a list of three components:

`logSDDR` an `N`-vector with values of the logarithm of the Bayes factors for the homoskedasticity hypothesis for each of the shocks

`log_SDDR_se` an `N`-vector with estimation standard errors of the logarithm of the Bayes factors reported in output element `logSDDR` that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

`components` a list of three components for the computation of the Bayes factor

log_denominator an `N`-vector with values of the logarithm of the Bayes factor denominators

log_numerator an `N`-vector with values of the logarithm of the Bayes factor numerators

log_numerator_s an `NxS` matrix of the log-full conditional posterior density ordinates computed to estimate the numerator

se_components an `Nx30` matrix containing the log-Bayes factors on the basis of which the standard errors are computed

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar_sv$new(us_fiscal_lsuw, p = 1)

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

# verify heteroskedasticity
sddr           = verify_volatility(posterior)

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

```
us_fiscal_lsuw |>
  specify_bsvar_sv$new(p = 1) |>
  estimate(S = 10) |>
  verify_volatility() -> sddr
```

`verify_volatility.PosteriorBSVAR` *Verifies heteroskedasticity of structural shocks equation by equation*

Description

This function will be deprecated starting from version 4.0. It is replaced by `verify_identification` function.

Displays information that the model is homoskedastic.

Usage

```
## S3 method for class 'PosteriorBSVAR'
verify_volatility(posterior)
```

Arguments

`posterior`	the `posterior` element of the list from the estimation outcome

Value

Nothing. Just displays a message: The model is homoskedastic.

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar$new(us_fiscal_lsuw, p = 1)

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

# verify heteroskedasticity
sddr           = verify_volatility(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar$new(p = 1) |>
  estimate(S = 10) |>
  verify_volatility() -> sddr
```

---

verify_volatility.PosteriorBSVAREXH

Verifies heteroskedasticity of structural shocks equation by equation

---

Description

This function will be deprecated starting from version 4.0. It is replaced by `verify_identification` function.

Computes the logarithm of Bayes factor for the homoskedasticity hypothesis for each of the structural shocks via Savage-Dickey Density Ration (SDDR). The hypothesis of homoskedasticity is represented by restriction:

$$H_0 : \sigma^2_{n.1} = ... = \sigma^2_{n.M} = 1$$

The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(\omega_n = 0|data) - logp(\omega_n = 0)$$

Therefore, a negative value of the difference is the evidence against homoskedasticity of the structural shock. The estimation of both elements of the difference requires numerical integration.

Usage

```
## S3 method for class 'PosteriorBSVAREXH'
verify_volatility(posterior)
```

Arguments

`posterior`	the `posterior` element of the list from the estimation outcome

Value

An object of class `SDDRvolatility` that is a list of three components:

`logSDDR` an `N`-vector with values of the logarithm of the Bayes factors for the homoskedasticity hypothesis for each of the shocks

`log_SDDR_se` an `N`-vector with estimation standard errors of the logarithm of the Bayes factors reported in output element `logSDDR` that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

`components` a list of three components for the computation of the Bayes factor

log_denominator an `N`-vector with values of the logarithm of the Bayes factor denominators

log_numerator an `N`-vector with values of the logarithm of the Bayes factor numerators

log_numerator_s an `NxS` matrix of the log-full conditional posterior density ordinates computed to estimate the numerator

se_components an `Nx30` matrix containing the log-Bayes factors on the basis of which the standard errors are computed

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

See Also

`specify_bsvar_exh`, `estimate`

Examples

```
# simple workflow
############################################################
# specify the model
spec  = specify_bsvar_exh$new(us_fiscal_lsuw)

# estimate the model
post  = estimate(spec, 10)

# verify heteroskedasticity
sddr           = verify_volatility(post)

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

```
us_fiscal_lsuw |>
  specify_bsvar_exh$new() |>
  estimate(S = 10) |>
  verify_volatility() -> sddr
```

`verify_volatility.PosteriorBSVARHMSH` *Verifies heteroskedasticity of structural shocks equation by equation*

Description

This function will be deprecated starting from version 4.0. It is replaced by `verify_identification` function.

Computes the logarithm of Bayes factor for the homoskedasticity hypothesis for each of the structural shocks via Savage-Dickey Density Ration (SDDR). The hypothesis of homoskedasticity is represented by restriction:

$$H_0 : \sigma^2_{n.1} = ... = \sigma^2_{n.M} = 1$$

The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(\omega_n = 0|data) - logp(\omega_n = 0)$$

Therefore, a negative value of the difference is the evidence against homoskedasticity of the structural shock. The estimation of both elements of the difference requires numerical integration.

Usage

```
## S3 method for class 'PosteriorBSVARHMSH'
verify_volatility(posterior)
```

Arguments

`posterior`	the `posterior` element of the list from the estimation outcome

Value

An object of class `SDDRvolatility` that is a list of three components:

`logSDDR` an `N`-vector with values of the logarithm of the Bayes factors for the homoskedasticity hypothesis for each of the shocks

`log_SDDR_se` an `N`-vector with estimation standard errors of the logarithm of the Bayes factors reported in output element `logSDDR` that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

`components` a list of three components for the computation of the Bayes factor

log_denominator an `N`-vector with values of the logarithm of the Bayes factor denominators

log_numerator an N-vector with values of the logarithm of the Bayes factor numerators

log_numerator_s an NxS matrix of the log-full conditional posterior density ordinates computed to estimate the numerator

se_components an Nx30 matrix containing the log-Bayes factors on the basis of which the standard errors are computed

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

See Also

specify_bsvar_hmsh, estimate

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, p = 1, M = 2)

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

# verify heteroskedasticity
sddr           = verify_volatility(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new(p = 1, M = 2) |>
  estimate(S = 10) |>
  verify_volatility() -> sddr
```

`verify_volatility.PosteriorBSVARMIX` *Verifies heteroskedasticity of structural shocks equation by equation*

Description

This function will be deprecated starting from version 4.0. It is replaced by `verify_identification` function.

Computes the logarithm of Bayes factor for the homoskedasticity hypothesis for each of the structural shocks via Savage-Dickey Density Ration (SDDR). The hypothesis of homoskedasticity is represented by restriction:

$$H_0 : \sigma^2_{n.1} = ... = \sigma^2_{n.M} = 1$$

The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(\omega_n = 0|data) - logp(\omega_n = 0)$$

Therefore, a negative value of the difference is the evidence against homoskedasticity of the structural shock. The estimation of both elements of the difference requires numerical integration.

Usage

```
## S3 method for class 'PosteriorBSVARMIX'
verify_volatility(posterior)
```

Arguments

`posterior`	the `posterior` element of the list from the estimation outcome

Value

An object of class `SDDRvolatility` that is a list of three components:

`logSDDR` an `N`-vector with values of the logarithm of the Bayes factors for the homoskedasticity hypothesis for each of the shocks

`log_SDDR_se` an `N`-vector with estimation standard errors of the logarithm of the Bayes factors reported in output element `logSDDR` that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

`components` a list of three components for the computation of the Bayes factor

log_denominator an `N`-vector with values of the logarithm of the Bayes factor denominators

log_numerator an `N`-vector with values of the logarithm of the Bayes factor numerators

log_numerator_s an `NxS` matrix of the log-full conditional posterior density ordinates computed to estimate the numerator

se_components an `Nx30` matrix containing the log-Bayes factors on the basis of which the standard errors are computed

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

See Also

specify_bsvar_mix, estimate

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar_mix$new(us_fiscal_lsuw, p = 1, M = 2)

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

# verify heteroskedasticity
sddr           = verify_volatility(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_mix$new(p = 1, M = 2) |>
  estimate(S = 10) |>
  verify_volatility() -> sddr
```

---

verify_volatility.PosteriorBSVARMSH

Verifies heteroskedasticity of structural shocks equation by equation

---

Description

This function will be deprecated starting from version 4.0. It is replaced by verify_identification function.

Computes the logarithm of Bayes factor for the homoskedasticity hypothesis for each of the structural shocks via Savage-Dickey Density Ration (SDDR). The hypothesis of homoskedasticity is represented by restriction:

$$H_0 : \sigma^2_{n.1} = ... = \sigma^2_{n.M} = 1$$

The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(\omega_n = 0|data) - logp(\omega_n = 0)$$

Therefore, a negative value of the difference is the evidence against homoskedasticity of the structural shock. The estimation of both elements of the difference requires numerical integration.

Usage

```
## S3 method for class 'PosteriorBSVARMSH'
verify_volatility(posterior)
```

Arguments

`posterior`	the `posterior` element of the list from the estimation outcome

Value

An object of class `SDDRvolatility` that is a list of three components:

`logSDDR` an `N`-vector with values of the logarithm of the Bayes factors for the homoskedasticity hypothesis for each of the shocks

`log_SDDR_se` an `N`-vector with estimation standard errors of the logarithm of the Bayes factors reported in output element `logSDDR` that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

`components` a list of three components for the computation of the Bayes factor

log_denominator an `N`-vector with values of the logarithm of the Bayes factor denominators

log_numerator an `N`-vector with values of the logarithm of the Bayes factor numerators

log_numerator_s an `NxS` matrix of the log-full conditional posterior density ordinates computed to estimate the numerator

se_components an `Nx30` matrix containing the log-Bayes factors on the basis of which the standard errors are computed

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

See Also

specify_bsvar_msh, estimate

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar_msh$new(us_fiscal_lsuw, p = 1, M = 2)

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

# verify heteroskedasticity
sddr           = verify_volatility(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_msh$new(p = 1, M = 2) |>
  estimate(S = 10) |>
  verify_volatility() -> sddr
```

---

verify_volatility.PosteriorBSVARSV

Verifies heteroskedasticity of structural shocks equation by equation

---

Description

This function will be deprecated starting from version 4.0. It is replaced by verify_identification function.

Computes the logarithm of Bayes factor for the homoskedasticity hypothesis for each of the structural shocks via Savage-Dickey Density Ration (SDDR). The hypothesis of homoskedasticity is represented by restriction:

$$H_0 : \omega_n = 0$$

The logarithm of Bayes factor for this hypothesis can be computed using the SDDR as the difference of logarithms of the marginal posterior distribution ordinate at the restriction less the marginal prior distribution ordinate at the same point:

$$logp(\omega_n = 0|data) - logp(\omega_n = 0)$$

Therefore, a negative value of the difference is the evidence against homoskedasticity of the structural shock. The estimation of both elements of the difference requires numerical integration.

Usage

```
## S3 method for class 'PosteriorBSVARSV'
verify_volatility(posterior)
```

Arguments

posterior	the posterior element of the list from the estimation outcome

Value

An object of class SDDRvolatility that is a list of three components:

logSDDR an N-vector with values of the logarithm of the Bayes factors for the homoskedasticity hypothesis for each of the shocks

log_SDDR_se an N-vector with estimation standard errors of the logarithm of the Bayes factors reported in output element logSDDR that are computed based on 30 random sub-samples of the log-ordinates of the marginal posterior and prior distributions.

components a list of three components for the computation of the Bayes factor

log_denominator an N-vector with values of the logarithm of the Bayes factor denominators

log_numerator an N-vector with values of the logarithm of the Bayes factor numerators

log_numerator_s an NxS matrix of the log-full conditional posterior density ordinates computed to estimate the numerator

se_components an Nx30 matrix containing the log-Bayes factors on the basis of which the standard errors are computed

Author(s)

Tomasz Woźniak <wozniak.tom@pm.me>

References

Lütkepohl, H., and Woźniak, T., (2020) Bayesian Inference for Structural Vector Autoregressions Identified by Markov-Switching Heteroskedasticity. *Journal of Economic Dynamics and Control* **113**, 103862, doi:10.1016/j.jedc.2020.103862.

Lütkepohl, H., Shang, F., Uzeda, L., and Woźniak, T. (2024) Partial Identification of Heteroskedastic Structural VARs: Theory and Bayesian Inference. *University of Melbourne Working Paper*, 1–57, doi:10.48550/arXiv.2404.11057.

See Also

specify_bsvar_sv, estimate

Examples

```
# simple workflow
############################################################
# specify the model
specification  = specify_bsvar_sv$new(us_fiscal_lsuw, p = 1)

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

# verify heteroskedasticity
```

```
sddr           = verify_volatility(posterior)

# workflow with the pipe |>
############################################################
us_fiscal_lsuw |>
  specify_bsvar_sv$new(p = 1) |>
  estimate(S = 10) |>
  verify_volatility() -> sddr
```

Index

∗ **datasets**
    us_fiscal_cond_forecasts, 252
    us_fiscal_ex, 253
    us_fiscal_ex_forecasts, 254
    us_fiscal_lsuw, 255
    us_fiscal_sww, 256
∗ **models**
    bsvars-package, 6
∗ **package**
    bsvars-package, 6
∗ **ts**
    bsvars-package, 6

bsvars *(*bsvars-package*)*, 6
bsvars-package, 6

compute_conditional_sd, 10, *140*, *245*
compute_conditional_sd.PosteriorBSVAR, 11
compute_conditional_sd.PosteriorBSVAREXH, 12
compute_conditional_sd.PosteriorBSVARHMSH, 13
compute_conditional_sd.PosteriorBSVARMIX, 14
compute_conditional_sd.PosteriorBSVARMSH, 15
compute_conditional_sd.PosteriorBSVARSV, 16
compute_conditional_sd.PosteriorBSVART, 17
compute_fitted_values, 18, *133*, *239*
compute_fitted_values.PosteriorBSVAR, 19
compute_fitted_values.PosteriorBSVAREXH, 20
compute_fitted_values.PosteriorBSVARHMSH, 21
compute_fitted_values.PosteriorBSVARMIX, 22
compute_fitted_values.PosteriorBSVARMSH, 23
compute_fitted_values.PosteriorBSVARSV, 24
compute_fitted_values.PosteriorBSVART, 25
compute_historical_decompositions, 26, *134*, *240*
compute_historical_decompositions.PosteriorBSVAR, 27
compute_historical_decompositions.PosteriorBSVAREXH, 28
compute_historical_decompositions.PosteriorBSVARHMSH, 29
compute_historical_decompositions.PosteriorBSVARMIX, 30
compute_historical_decompositions.PosteriorBSVARMSH, 32
compute_historical_decompositions.PosteriorBSVARSV, 33
compute_historical_decompositions.PosteriorBSVART, 34
compute_impulse_responses, 35, *61*, *62*, *64–67*, *69*, *70*, *136*, *241*
compute_impulse_responses.PosteriorBSVAR, 36
compute_impulse_responses.PosteriorBSVAREXH, 38
compute_impulse_responses.PosteriorBSVARHMSH, 39
compute_impulse_responses.PosteriorBSVARMIX, 40
compute_impulse_responses.PosteriorBSVARMSH, 42
compute_impulse_responses.PosteriorBSVARSV, 43
compute_impulse_responses.PosteriorBSVART, 44
compute_regime_probabilities, 46, *137*,

242
compute_regime_probabilities.PosteriorBSVAREXH, 47
compute_regime_probabilities.PosteriorBSVARHMSH, 48
compute_regime_probabilities.PosteriorBSVARMIX, 50
compute_regime_probabilities.PosteriorBSVARMSH, 51
compute_structural_shocks, 53, *139*, *243*
compute_structural_shocks.PosteriorBSVAR, 54
compute_structural_shocks.PosteriorBSVAREXH, 55
compute_structural_shocks.PosteriorBSVARHMSH, 56
compute_structural_shocks.PosteriorBSVARMIX, 57
compute_structural_shocks.PosteriorBSVARMSH, 58
compute_structural_shocks.PosteriorBSVARSV, 59
compute_structural_shocks.PosteriorBSVART, 60
compute_variance_decompositions, 61, *132*, *238*
compute_variance_decompositions.PosteriorBSVAR, 62
compute_variance_decompositions.PosteriorBSVAREXH, 63
compute_variance_decompositions.PosteriorBSVARHMSH, 64
compute_variance_decompositions.PosteriorBSVARMIX, 65
compute_variance_decompositions.PosteriorBSVARMSH, 67
compute_variance_decompositions.PosteriorBSVARSV, 68
compute_variance_decompositions.PosteriorBSVART, 69

estimate, *10–27*, *29–33*, *35–38*, *40–42*, *44–46*, *48, 49*, *51–62*, *64–67*, *69*, 70, *70*, *121–124*, *126–129*, *144*, *148*, *152*, *155*, *158*, *162*, *165*, *173*, *177*, *181*, *186*, *190*, *195*, *199*, *230–235*, *237*, *270*, *272–281*, *284*, *286*, *288, 289*, *291*, *295*, *297*, *299*, *301, 302*
estimate.BSVAR, 73
estimate.BSVAREXH, 75
estimate.BSVARHMSH, 77
estimate.BSVARMIX, 80
estimate.BSVARMSH, 82
estimate.BSVARSV, 85
estimate.BSVART, 88
estimate.PosteriorBSVAR, 90
estimate.PosteriorBSVAREXH, 92
estimate.PosteriorBSVARHMSH, 94
estimate.PosteriorBSVARMIX, 97
estimate.PosteriorBSVARMSH, 99
estimate.PosteriorBSVARSV, 102
estimate.PosteriorBSVART, 105

forecast, *131*, *228*
forecast.PosteriorBSVAR, 107
forecast.PosteriorBSVAREXH, 109
forecast.PosteriorBSVARHMSH, 111
forecast.PosteriorBSVARMIX, 112
forecast.PosteriorBSVARMSH, 114
forecast.PosteriorBSVARSV, 116
forecast.PosteriorBSVART, 118

normalise, *10–17*, *26, 27*, *29–33*, *35–38*, *40–42*, *44, 45*, *53–62*, *64–67*, *69, 70*, *72*, *75*, *77*, *79*, *82*, *84*, *87*, *89*, *91*, *94*, *96*, *99*, *101*, *104*, *106*, 120
normalise.PosteriorBSVAR, 121
normalise.PosteriorBSVAREXH, 122
normalise.PosteriorBSVARHMSH, 124
normalise.PosteriorBSVARMIX, 125
normalise.PosteriorBSVARMSH, 126
normalise.PosteriorBSVARSV, 127
normalise.PosteriorBSVART, 128

plot.Forecasts, 130
plot.PosteriorFEVD, 131
plot.PosteriorFitted, 132
plot.PosteriorHD, 134
plot.PosteriorIR, 135
plot.PosteriorRegimePr, 136
plot.PosteriorShocks, 138
plot.PosteriorSigma, 139
plot_ribbon, 141

specify_bsvar, *72*, *75*, *91*, 142, *173*, *230*
specify_bsvar_exh, *76, 77*, *93, 94*, 146, *177*, *231*, *274*, *284*, *295*
specify_bsvar_hmsh, *79*, *96*, 149, *181*, *232*, *275*, *286*, *297*

specify_bsvar_mix, *72*, *81, 82*, *98, 99*, 153, *186*, *233*, *277*, *288*, *299*
specify_bsvar_msh, *72*, *83, 84*, *100, 101*, 155, *190*, *234*, *278*, *289*, *301*
specify_bsvar_sv, *72*, *87*, *104*, 159, *195*, *235*, *280*, *291*, *302*
specify_bsvar_t, *89*, *106*, 163, *199*, *237*, *281*
specify_data_matrices, 165
specify_forecasts, 166
specify_identification_bsvars, 168
specify_posterior_bsvar, *75*, *91*, *144*, 170
specify_posterior_bsvar_exh, *77*, *94*, *148*, 175
specify_posterior_bsvar_hmsh, *79*, *96*, *152*, 179
specify_posterior_bsvar_mix, *82*, *99*, *155*, 183
specify_posterior_bsvar_msh, *84*, *101*, *158*, 187
specify_posterior_bsvar_sv, *87*, *104*, *162*, 192
specify_posterior_bsvar_t, *89*, *106*, *165*, 197
specify_prior_bsvar, 201
specify_prior_bsvar_exh, 204
specify_prior_bsvar_mix, 206
specify_prior_bsvar_msh, 207
specify_prior_bsvar_sv, 209
specify_prior_bsvar_t, 211
specify_starting_values_bsvar, 212
specify_starting_values_bsvar_exh, 215
specify_starting_values_bsvar_hmsh, 218
specify_starting_values_bsvar_mix, 221
specify_starting_values_bsvar_msh, 222
specify_starting_values_bsvar_sv, 225
summary, *10–27*, *29–33*, *35–38*, *40–42*, *44–46*, *48, 49*, *51–62*, *64–67*, *69, 70*
summary.Forecasts, 228
summary.PosteriorBSVAR, 229
summary.PosteriorBSVAREXH, 230
summary.PosteriorBSVARHMSH, 231
summary.PosteriorBSVARMIX, 232
summary.PosteriorBSVARMSH, 234
summary.PosteriorBSVARSV, 235
summary.PosteriorBSVART, 236
summary.PosteriorFEVD, 237
summary.PosteriorFitted, 238
summary.PosteriorHD, 239
summary.PosteriorIR, 241
summary.PosteriorRegimePr, 242
summary.PosteriorShocks, 243
summary.PosteriorSigma, 244
summary.SDDRautoregression, 245
summary.SDDRidMIX, 246
summary.SDDRidMSH, 247
summary.SDDRidSV, 249
summary.SDDRidT, 250
summary.SDDRvolatility, 251

us_fiscal_cond_forecasts, 252
us_fiscal_ex, 253
us_fiscal_ex_forecasts, 254
us_fiscal_lsuw, 255
us_fiscal_sww, 256

verify_autoregression, *246*, 257
verify_autoregression.PosteriorBSVAR, 258
verify_autoregression.PosteriorBSVAREXH, 260
verify_autoregression.PosteriorBSVARHMSH, 262
verify_autoregression.PosteriorBSVARMIX, 263
verify_autoregression.PosteriorBSVARMSH, 265
verify_autoregression.PosteriorBSVARSV, 267
verify_autoregression.PosteriorBSVART, 268
verify_identification, 270, *291*, *293, 294*, *296*, *298, 299*, *301*
verify_identification.PosteriorBSVAR, *271*, 272, *272*
verify_identification.PosteriorBSVAREXH, 273
verify_identification.PosteriorBSVARHMSH, 274
verify_identification.PosteriorBSVARMIX, *247*, *271, 272*, 276
verify_identification.PosteriorBSVARMSH, *248*, *271, 272*, 277
verify_identification.PosteriorBSVARSV, *249*, *271, 272*, 279
verify_identification.PosteriorBSVART, *250*, *271, 272*, 280

verify_normality, 282
verify_normality.PosteriorBSVAREXH, 283
verify_normality.PosteriorBSVARHMSH, 285
verify_normality.PosteriorBSVARMIX, 287
verify_normality.PosteriorBSVARMSH, 288
verify_normality.PosteriorBSVARSV, 290
verify_volatility, *252*, 291
verify_volatility.PosteriorBSVAR, 293
verify_volatility.PosteriorBSVAREXH, 294
verify_volatility.PosteriorBSVARHMSH, 296
verify_volatility.PosteriorBSVARMIX, 298
verify_volatility.PosteriorBSVARMSH, 299
verify_volatility.PosteriorBSVARSV, 301