# Package: bayesianVARs (via r-universe)

October 23, 2026

**Title** MCMC Estimation of Bayesian Vectorautoregressions

**Version** 0.1.8

**Description** Efficient Markov Chain Monte Carlo (MCMC) algorithms for the fully Bayesian estimation of vectorautoregressions (VARs) featuring stochastic volatility (SV). Implements state-of-the-art shrinkage priors following Gruber & Kastner (2025) <doi:10.1016/j.ijforecast.2025.02.001>. Efficient equation-per-equation estimation following Kastner & Huber (2020) <doi:10.1002/for.2680> and Carrerio et al. (2021) <doi:10.1016/j.jeconom.2021.11.010>.

**License** GPL (>= 3)

**URL** https://github.com/luisgruber/bayesianVARs, https://luisgruber.github.io/bayesianVARs/

**BugReports** https://github.com/luisgruber/bayesianVARs/issues

**Depends** R (>= 3.5.0)

**Imports** colorspace, factorstochvol (>= 1.1.0), GIGrvg (>= 0.7), graphics, MASS, mvtnorm, Rcpp (>= 1.0.0), scales, stats, stochvol (>= 3.0.3), utils

**Suggests** coda, knitr, quarto, rmarkdown, testthat (>= 3.0.0), lpSolveAPI, bsvarSIGNs

**LinkingTo** factorstochvol, Rcpp, RcppArmadillo, RcppProgress, stochvol, lpSolveAPI

**VignetteBuilder** knitr, quarto

**Config/testthat/edition** 3

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.3.3

**NeedsCompilation** yes

**Author** Luis Gruber [cph, aut, cre] (ORCID:
<https://orcid.org/0000-0002-2399-738X>), Stefan Haan [aut],
Gregor Kastner [aut, ths] (ORCID:
<https://orcid.org/0000-0002-8237-8271>)

**Maintainer** Luis Gruber <Luis.Gruber@aau.at>

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

**Date/Publication** 2026-02-19 14:20:08 UTC

**RemoteUrl** https://github.com/cran/bayesianVARs

**RemoteRef** HEAD

**RemoteSha** 64d81868e89460c736e2194fe01857c619a3fefb

# Contents

[.bayesianVARs_coef . . . 3
[.bayesianVARs_draws . . . 3
bvar . . . 4
coef . . . 8
extractB0 . . . 9
fitted.bayesianVARs_bvar . . . 10
irf . . . 11
my_gig . . . 13
pairs_predict . . . 13
plot.bayesianVARs_bvar . . . 14
plot.bayesianVARs_fitted . . . 15
plot.bayesianVARs_irf . . . 17
plot.bayesianVARs_predict . . . 18
plot.bayesianVARs_residuals . . . 19
posterior_heatmap . . . 20
predict.bayesianVARs_bvar . . . 23
print.bayesianVARs_bvar . . . 25
print.bayesianVARs_predict . . . 25
print.summary.bayesianVARs_bvar . . . 26
print.summary.bayesianVARs_predict . . . 27
residuals.bayesianVARs_bvar . . . 28
specify_prior_phi . . . 29
specify_prior_sigma . . . 33
specify_structural_restrictions . . . 39
stable_bvar . . . 41
summary.bayesianVARs_bvar . . . 42
summary.bayesianVARs_draws . . . 43
summary.bayesianVARs_predict . . . 44
usmacro_growth . . . 45
vcov.bayesianVARs_bvar . . . 45

**Index** **47**

## [.bayesianVARs_coef *Extract or Replace Parts of a bayesianVARs_coef object*

### Description

Extract or replace parts of a bayesianVARs_coef object.

### Usage

```
## S3 method for class 'bayesianVARs_coef'
x[i, j, ...]
```

### Arguments

| | |
|---|---|
| x | An object of type bayesianVARs_coef. |
| i | indices |
| j | indices |
| ... | further indices |

### Value

An object of type bayesianVARs_coef.

### Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]

# Estimate a model
mod <- bvar(data, sv_keep = "all", quiet = TRUE)

# Extract coefficients, which are of class bayesianVARs_coef
phi <- coef(mod)
phi[1,1,1]
```

## [.bayesianVARs_draws *Extract or Replace Parts of a bayesianVARs_draws object*

### Description

Extract or replace parts of a bayesianVARs_draws object.

### Usage

```
## S3 method for class 'bayesianVARs_draws'
x[i, j, ...]
```

### Arguments

| | |
|---|---|
| x | An object of type bayesianVARs_draws. |
| i | indices |
| j | indices |
| ... | further indices |

### Value

An object of type bayesianVARs_draws.

### Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]

# Estimate a model
mod <- bvar(data, sv_keep = "all", quiet = TRUE)

# Extract coefficients, which are of class bayesianVARs_draws
phi <- coef(mod)
phi[1,1,1]
```

---

## bvar *Markov Chain Monte Carlo Sampling for Bayesian Vectorautoregressions*

---

### Description

bvar simulates from the joint posterior distribution of the parameters and latent variables and returns the posterior draws.

### Usage

```
bvar(
  data,
  lags = 1L,
  draws = 1000L,
  burnin = 1000L,
  thin = 1L,
  prior_intercept = 10,
  prior_phi = specify_prior_phi(data = data, lags = lags, prior = "HS"),
  prior_sigma = specify_prior_sigma(data = data, type = "factor", quiet = TRUE),
  sv_keep = "last",
  expert_huge = FALSE,
  quiet = FALSE,
  startvals = list()
)
```

## Arguments

| | |
|---|---|
| `data` | Data matrix (can be a time series object). Each of $M$ columns is assumed to contain a single time-series of length $T$. |
| `lags` | Integer indicating the order of the VAR, i.e. the number of lags of the dependent variables included as predictors. |
| `draws` | single integer indicating the number of draws after the burnin |
| `burnin` | single integer indicating the number of draws discarded as burnin |
| `thin` | single integer. Every $thin$th draw will be stored. Default is `thin=1L`. |
| `prior_intercept` | Either `prior_intercept=FALSE` and no constant term (intercept) will be included. Or a numeric vector of length $M$ indicating the (fixed) prior standard deviations on the constant term. A single number will be recycled accordingly. Default is `prior_intercept=10`. |
| `prior_phi` | `bayesianVARs_prior_phi` object specifying prior for the reduced form VAR coefficients. Best use constructor `specify_prior_phi`. |
| `prior_sigma` | `bayesianVARs_prior_sigma` object specifying prior for the variance-covariance matrix of the VAR. Best use constructor `specify_prior_sigma`. |
| `sv_keep` | String equal to `"all"` or `"last"`. In case of `sv_keep = "last"`, the default, only draws for the very last log-variance $h_T$ are stored. |
| `expert_huge` | logical value indicating whether to use a specific algorithm for huge VARs. Only possible if a factor structure for the errors is specified. See section MCMC algorithm below for more details. |
| `quiet` | logical value indicating whether information about the progress during sampling should be displayed during sampling (default is `TRUE`). |
| `startvals` | optional list with starting values. |

## Details

The VAR(p) model is of the following form: $\boldsymbol{y}_t' = \boldsymbol{\iota}' + \boldsymbol{x}_t'\boldsymbol{\Phi} + \boldsymbol{\epsilon}_t'$, where $\boldsymbol{y}_t$ is a $M$-dimensional vector of dependent variables and $\boldsymbol{\epsilon}_t$ is the error term of the same dimension. $\boldsymbol{x}_t$ is a $K = pM$-dimensional vector containing lagged/past values of the dependent variables $\boldsymbol{y}_{t-l}$ for $l = 1, \ldots, p$ and $\boldsymbol{\iota}$ is a constant term (intercept) of dimension $M \times 1$. The reduced-form coefficient matrix $\boldsymbol{\Phi}$ is of dimension $K \times M$.

`bvar` offers two different specifications for the errors: The user can choose between a factor stochastic volatility structure or a cholesky stochastic volatility structure. In both cases the disturbances $\boldsymbol{\epsilon}_t$ are assumed to follow a $M$-dimensional multivariate normal distribution with zero mean and variance-covariance matrix $\boldsymbol{\Sigma}_t$. In case of the cholesky specification $\boldsymbol{\Sigma}_t = \boldsymbol{U}'^{-1}\boldsymbol{D}_t\boldsymbol{U}^{-1}$, where $\boldsymbol{U}^{-1}$ is upper unitriangular (with ones on the diagonal). The diagonal matrix $\boldsymbol{D}_t$ depends upon latent log-variances, i.e. $\boldsymbol{D}_t = diag(exp(h_{1t}), \ldots, exp(h_{Mt}))$. The log-variances follow a priori independent autoregressive processes $h_{it} \sim N(\mu_i + \phi_i(h_{i,t-1} - \mu_i), \sigma_i^2)$ for $i = 1, \ldots, M$. In case of the factor structure, $\boldsymbol{\Sigma}_t = \boldsymbol{\Lambda}\boldsymbol{V}_t\boldsymbol{\Lambda}' + \boldsymbol{G}_t$. The diagonal matrices $\boldsymbol{V}_t$ and $\boldsymbol{G}_t$ depend upon latent log-variances, i.e. $\boldsymbol{G}_t = diag(exp(h_{1t}), \ldots, exp(h_{Mt}))$ and $\boldsymbol{V}_t = diag(exp(h_{M+1,t}), \ldots, exp(h_{M+r,t}))$. The log-variances follow a priori independent autoregressive processes $h_{it} \sim N(\mu_i + \phi_i(h_{i,t-1} - \mu_i), \sigma_i^2)$ for $i = 1, \ldots, M$ and $h_{M+j,t} \sim N(\phi_i h_{M+j,t-1}, \sigma_{M+j}^2)$ for $j = 1, \ldots, r$.

## Value

An object of type `bayesianVARs_bvar`, a list containing the following objects:

- `PHI`: A `bayesianVARs_coef` object, an array, containing the posterior draws of the VAR coefficients (including the intercept).
- `U`: A `bayesianVARs_draws` object, a matrix, containing the posterior draws of the contemporaneous coefficients (if cholesky decomposition for sigma is specified).
- `logvar`: A `bayesianVARs_draws` object containing the log-variance draws.
- `sv_para`: A `baysesianVARs_draws` object containing the posterior draws of the stochastic volatility related parameters.
- `phi_hyperparameter`: A matrix containing the posterior draws of the hyperparameters of the conditional normal prior on the VAR coefficients.
- `u_hyperparameter`: A matrix containing the posterior draws of the hyperparameters of the conditional normal prior on U (if cholesky decomposition for sigma is specified).
- `bench`: `proc_time` object containing the run time of the sampler.
- `V_prior`: An array containing the posterior draws of the variances of the conditional normal prior on the VAR coefficients.
- `facload`: A `bayesianVARs_draws` object, an array, containing draws from the posterior distribution of the factor loadings matrix (if factor decomposition for sigma is specified).
- `fac`: A `bayesianVARs_draws` object, an array, containing factor draws from the posterior distribution (if factor decomposition for sigma is specified).
- `Y`: Matrix containing the dependent variables used for estimation.
- `X` matrix containing the lagged values of the dependent variables, i.e. the covariates.
- `lags`: Integer indicating the lag order of the VAR.
- `intercept`: Logical indicating whether a constant term is included.
- `heteroscedastic` logical indicating whether heteroscedasticity is assumed.
- `Yraw`: Matrix containing the dependent variables, including the initial 'lags' observations.
- `Traw`: Integer indicating the total number of observations.
- `sigma_type`: Character specifying the decomposition of the variance-covariance matrix.
- `datamat`: Matrix containing both 'Y' and 'X'.
- `config`: List containing information on configuration parameters.

## MCMC algorithm

To sample efficiently the reduced-form VAR coefficients assuming a **factor structure for the errors**, the equation per equation algorithm in Kastner & Huber (2020) is implemented. The factor structure has the advantage that an algorithm for sampling from high-dimensional Gaussian distributions can be exploited by setting `expert_huge = TRUE`. However, this speeds up computations only if $K > T$, i.e. the number of coefficients per equations exceeds the number of observations. All parameters and latent variables associated with the factor-structure are sampled using package `factorstochvol-package`'s function `update_fsv` callable on the C-level only.

To sample efficiently the reduced-form VAR coefficients, assuming a **cholesky-structure for the errors**, the corrected triangular algorithm in Carriero et al. (2021) is implemented. The SV parameters and latent variables are sampled using package `stochvol`'s `update_fast_sv` function. The precision parameters, i.e. the free off-diagonal elements in $\boldsymbol{U}$, can be sampled using Bayesian linear regression methods, see Cogley and Sargent (2005).

## References


Gruber, L. and Kastner, G. (2025). Forecasting macroeconomic data with Bayesian VARs: Sparse or dense? It depends! *International Journal of Forecasting*. doi:10.1016/j.ijforecast.2025.02.001.

Kastner, G. and Huber, F. Sparse (2020). Bayesian vector autoregressions in huge dimensions. *Journal of Forecasting*. **39**, 1142–1165, doi:10.1002/for.2680.

Kastner, G. (2019). Sparse Bayesian Time-Varying Covariance Estimation in Many Dimensions *Journal of Econometrics*, **210**(1), 98–115, doi:10.1016/j.jeconom.2018.11.007.

Carriero, A. and Chan, J. and Clark, T. E. and Marcellino, M. (2021). Corrigendum to "Large Bayesian vector autoregressions with stochastic volatility and non-conjugate priors" [J. Econometrics 212 (1) (2019) 137–154]. *Journal of Econometrics*, doi:10.1016/j.jeconom.2021.11.010.

Cogley, S. and Sargent, T. (2005). Drifts and volatilities: monetary policies and outcomes in the post WWII US. *Review of Economic Dynamics*, **8**, 262–302, doi:10.1016/j.red.2004.10.009.

Hosszejni, D. and Kastner, G. (2021). Modeling Univariate and Multivariate Stochastic Volatility in R with stochvol and factorstochvol. *Journal of Statistical Software*, *100*, 1—34. doi:10.18637/jss.v100.i12.


## See Also

- Helpers for prior configuration: `specify_prior_phi()`, `specify_prior_sigma()`.
- Plotting: `plot.bayesianVARs_bvar()`.
- Extractors: `coef.bayesianVARs_bvar()`, `vcov.bayesianVARs_bvar()`.
- 'stable' bvar: `stable_bvar()`.
- summary method: `summary.bayesianVARs_bvar()`.
- predict method: `predict.bayesianVARs_bvar()`.
- fitted method: `fitted.bayesianVARs_bvar()`.

## Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]

# Estimate a model
mod <- bvar(data, sv_keep = "all", quiet = TRUE)

# Plot
plot(mod)

# Summary
summary(mod)
```

coef *Extract VAR coefficients*

## Description

Extracts posterior draws of the VAR coefficients from a VAR model estimated with `bvar()`.

## Usage

```
## S3 method for class 'bayesianVARs_bvar'
coef(object, ...)
```

## Arguments

| | |
|---|---|
| `object` | A `bayesianVARs_bvar` object obtained from `bvar()`. |
| `...` | Currently ignored. |

## Value

Returns a numeric array of dimension $M \times K \times draws$, where M is the number of time-series, K is the number of covariates per equation (including the intercept) and draws is the number of stored posterior draws.

## See Also

`summary.bayesianVARs_draws()`, `vcov.bayesianVARs_bvar()`.

## Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]

# Estimate a model
mod <- bvar(data, sv_keep = "all", quiet = TRUE)

# Extract posterior draws of VAR coefficients
bvar_coefs <- coef(mod)
```

extractB0 *Retrieve the structural parameter $\boldsymbol{B}_0$ samples from an IRF object.*

## Description

Retrieve the structural parameter $\boldsymbol{B}_0$ samples from an IRF object.

## Usage

```
extractB0(x)
```

## Arguments

| | |
|---|---|
| x | a bayesianVARs_irf object |

## Author(s)

Stefan Haan <sthaan@edu.aau.at>

## See Also

specify_structural_restrictions

## Examples

```
train_data <- 100 * usmacro_growth[,c("GDPC1", "GDPCTPI", "GS1", "M2REAL", "CPIAUCSL")]
prior_sigma <- specify_prior_sigma(train_data, type="cholesky", cholesky_heteroscedastic=FALSE)
mod <- bvar(train_data, lags=5L, prior_sigma=prior_sigma, quiet=TRUE)

structural_restrictions <- specify_structural_restrictions(
 mod,
 restrictions_B0=rbind(
   c(1 ,NA,0 ,NA,NA),
   c(0 ,1 ,0 ,NA,NA),
   c(0 ,NA,1 ,NA,NA),
   c(0 ,0 ,NA,1 ,NA),
   c(0 ,0 ,0 ,0 ,1 )
 )
)
irf_structural <- irf(
 mod, ahead=8,
 structural_restrictions=structural_restrictions
)

B0 <- extractB0(irf_structural)

# Visually check that the restriction B0[1,1] >= 0 has been satisfied
hist(
 B0[1,1,],
 xlim=range(0, B0),
```

```
 main = paste0("Posterior B0[", 1, ",", 1,"]")
)
abline(v=0, col=2, lwd=2)
```

---

`fitted.bayesianVARs_bvar` *Simulate fitted/predicted historical values for an estimated VAR model*

---

## Description

Simulates the fitted/predicted (in-sample) values for an estimated VAR model.

## Usage

```
## S3 method for class 'bayesianVARs_bvar'
fitted(object, error_term = TRUE, ...)
```

## Arguments

| | |
|---|---|
| `object` | A `bayesianVARs_bvar` object estimated via `bvar()`. |
| `error_term` | logical indicating whether to include the error term or not. |
| `...` | Currently ignored. |

## Value

An object of class `bayesianVARs_fitted`.

## Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]

# Estimate a model
mod <- bvar(data, sv_keep = "all", quiet = TRUE)

# Simulate predicted historical values including the error term.
pred <- fitted(mod, error_term = TRUE)

# Simulate fitted historical values not including the error term.
fit <- fitted(mod, error_term = FALSE)

# Visualize
plot(pred)
plot(fit)
```

irf *Impulse response functions*

## Description

Effect of the structural (factor) shocks over time.

## Usage

```
irf(
  x,
  ahead = 8,
  structural_restrictions = NULL,
  shocks = NULL,
  hairy = FALSE,
  ...
)
```

## Arguments

| | |
|---|---|
| x | An object of type bayesianVARs_bvar. |
| ahead | maximum number of time horizons. |
| structural_restrictions | an object generated by specify_structural_restrictions. If specified, the IRFs to the structural shocks identified by the restrictions given in this argument will be calculated. If not specified, the orthogonal IRFs based on the Cholesky decomposition will be calculated. Note that the orthogonal IRFs depend on the ordering of the variables and do not necessarily correspond to shocks with a meaningful interpretation. |
| shocks | an matrix with r rows, where r is the number of shocks (see 'Details'). Each column specifies a shock. Default: diag(r), will calculate the responses to all structural (or factor) shocks of one standard deviation. |
| hairy | set to TRUE in order to plot each path seperately. To show valid quantiles, an Bayes optimal order of the posterior samples will be calculated which can take a long time even for moderately many samples. Default: FALSE. |
| ... | Following expert arguments can be specified: |

**solver:** "randomized" or "lp". If some columns have more than one sign restriction, "lp" might find a solution, even if "randomized" is unable to. However "lp" can produce artifically narrow confidence bands which do not properly reflect the uncertainty in the identification scheme. Default: "randomized"

**randomized_max_rotations_per_sample:** if using the "randomized" solver, how many rotations are drawn for each sample of the reduced form parameters in x. Default: 2.

## Details

If a factor model was used, then the number of shocks is equal to the number of factors.

If the Cholesky model was used, then the number of shocks is equal to the number of variables.

## Value

Returns a `bayesianVARs_irf` object.

## Author(s)

Stefan Haan <sthaan@edu.aau.at>

## References

Arias, J. and Rubio-Ramírez, J. and Waggoner, D. (2014). Inference Based on SVARs Identified with Sign and Zero Restrictions: Theory and Applications. *FRB Atlanta Working Paper Series*, doi:10.2139/ssrn.2580264.

## See Also

`specify_structural_restrictions`, `extractB0`

## Examples

```
train_data <- 100 * usmacro_growth[,c("GDPC1", "GDPCTPI", "GS1", "M2REAL", "CPIAUCSL")]
prior_sigma <- specify_prior_sigma(train_data, type="cholesky", cholesky_heteroscedastic=FALSE)
mod <- bvar(train_data, lags=5L, prior_sigma=prior_sigma, quiet=TRUE)

structural_restrictions <- specify_structural_restrictions(
 mod,
 restrictions_B0=rbind(
   c(1 ,NA,0 ,NA,NA),
   c(0 ,1 ,0 ,NA,NA),
   c(0 ,NA,1 ,NA,NA),
   c(0 ,0 ,NA,1 ,NA),
   c(0 ,0 ,0 ,0 ,1 )
 )
)
irf_structural <- irf(
 mod, ahead=8,
 structural_restrictions=structural_restrictions
)
plot(irf_structural)
```

## my_gig *Draw from generalized inverse Gaussian*

### Description

Vectorized version of rgig

### Usage

```
my_gig(n, lambda, chi, psi)
```

### Arguments

| | |
|---|---|
| n | A single integer indicating the number of draws to generate. |
| lambda | vector of shape parameters. |
| chi | vector of shape/scale parameters. Must be nonnegative for positive lambdas and positive else. |
| psi | vector of shape/scale parameters. Must be nonnegative for negative lambdas and positive else. |

### Value

Matrix of dimension c(n,m), where m is the maximum length of lambda, psi and chi.

### Examples

```
gigsamples <- my_gig(2, c(1,1), c(1,1), c(1,1))
```

## pairs_predict *Pairwise visualization of out-of-sample posterior predictive densities.*

### Description

Pairwise visualization of out-of-sample posterior predictive densities.

### Usage

```
## S3 method for class 'bayesianVARs_predict'
pairs(x, vars, ahead, ...)
```

### Arguments

| | |
|---|---|
| x | An object of class bayesianVARs_predict obtained via predict.bayesianVARs_bvar(). |
| vars | Integer vector (or coercible to such) indicating which variables to plot. |
| ahead | Integer vector (or coercible to such) indicating which step ahead to plot. max(ahead) must be smaller equal to dim(x$predictions)[1]. |
| ... | Currently ignored! |

## Value

Returns x invisibly.

## Note

Note that that bayesianVARs_predict can also be used withing plot.bayesianVARs_bvar().

## See Also

Other plotting plot.bayesianVARs_bvar(), plot.bayesianVARs_fitted(), plot.bayesianVARs_predict() posterior_heatmap().

## Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]

# Estimate a model
mod <- bvar(data, sv_keep = "all", quiet = TRUE)

# Simulate from posterior predictive
predictions <- predict(mod, ahead = 1:3)

# Visualize
pairs(predictions, vars = 1:3, ahead = 1:3)
```

---

# plot.bayesianVARs_bvar

*Plot method for bayesianVARs_bvar*

---

## Description

Visualization of in-sample fit. Can also be used to display prediction intervals of future values.

## Usage

```
## S3 method for class 'bayesianVARs_bvar'
plot(
  x,
  predictions = NULL,
  quantiles = c(0.05, 0.5, 0.95),
  dates = NULL,
  n_col = 1,
  ...
)
```

### Arguments

| | |
|---|---|
| x | An object of class bayesianVARs_bvar obtained via bvar(). |
| predictions | Optional array of out of sample predictions, e.g. obtained via predict.bayesianVARs_bvar(). |
| quantiles | numeric vector indicating which quantiles to plot. |
| dates | optional vector of dates for labelling the x-axis. The default values is NULL; in this case, the axis will be labeled with numbers. |
| n_col | integer indicating the number of columns to use for plotting. |
| ... | Currently ignored! |

### Value

Returns x invisibly.

### See Also

Other plotting plot.bayesianVARs_fitted(), plot.bayesianVARs_predict(), pairs.bayesianVARs_predict(), posterior_heatmap().

### Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]

# Estimate a model
mod <- bvar(data, sv_keep = "all", quiet = TRUE)

# Simulate from posterior predictive
predictions <- predict(mod, ahead = 1:3)

# Visualize
plot(mod, predictions = predictions)
```

---

## plot.bayesianVARs_fitted

*Visualization of in-sample fit of an estimated VAR.*

---

### Description

Visualization of in-sample fit of an estimated VAR.

## Usage

```
## S3 method for class 'bayesianVARs_fitted'
plot(
  x,
  dates = NULL,
  vars = "all",
  quantiles = c(0.05, 0.5, 0.95),
  n_col = 1L,
  ...
)
```

## Arguments

| | |
|---|---|
| x | A bayesianVARs_fitted object. |
| dates | optional vector of dates for labelling the x-axis. The default values is NULL; in this case, the axis will be labeled with numbers. |
| vars | character vector containing the names of the variables to be visualized. The default is "all" indicating that the fit of all variables is visualized. |
| quantiles | numeric vector indicating which quantiles to plot. |
| n_col | integer indicating the number of columns to use for plotting. |
| ... | Currently ignored. |

## Value

returns x invisibly

## See Also

- fitted method for class 'bayesianVARs_bvar': fitted.bayesianVARs_bvar().
- Other plotting plot.bayesianVARs_bvar(), plot.bayesianVARs_fitted(), plot.bayesianVARs_predict(), pairs.bayesianVARs_predict(), posterior_heatmap().

## Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]

# Estimate a model
mod <- bvar(data, sv_keep = "all", quiet = TRUE)

# Simulate predicted historical values including the error term.
pred <- fitted(mod, error_term = TRUE)

# Visualize
plot(pred)
```

`plot.bayesianVARs_irf` *Impulse Responses Plot*

## Description

Visualization of the impulse responses. Responses are plotted on a grid, where rows correspond to variables and columns correspond to shocks.

## Usage

```
## S3 method for class 'bayesianVARs_irf'
plot(
  x,
  vars = "all",
  quantiles = c(0.05, 0.25, 0.5, 0.75, 0.95),
  default_hair_color = "#FF000003",
  true_irf = NULL,
  ...
)
```

## Arguments

| | |
|---|---|
| `x` | An object of type `bayesianVARs_irf` obtained via `irf`. |
| `vars` | character vector containing the names of the variables to be visualized. The default is `"all"` indicating that all variables are visualized. |
| `quantiles` | numeric vector indicating which quantiles to plot. If `hairy=TRUE` was specified when calling `irf`, a proportion of `max(quantiles)` IRFs will be plotted. Specify `0` to plot a point-estimate only. If `hairy=FALSE` was specified (the default), point-wise quantiles will be plotted. Note that the curve of point-wise medians is not necessarily in the set of IRFs (see Inoue 2022). |
| `default_hair_color` | the color of the IRF samples, if `hairy=TRUE` was specified. |
| `true_irf` | If the true IRFs are known (because the data was simulated) they can be plotted alongside the estimates, such that the quality of the estimates may be judged. `true_irf` should be a numeric array with dimensions variables, shocks and time, in that order. |
| `...` | Currently ignored! |

## Author(s)

Stefan Haan <sthaan@edu.aau.at>

## References

Inoue, A. and Kilian, L. (2022). Joint Bayesian inference about impulse responses in VAR models. *Journal of Econometrics*, doi:10.1016/j.jeconom.2021.05.010.

## See Also

irf

---

plot.bayesianVARs_predict *Fan chart*

---

## Description

Visualization of (out-of-sample) predictive distribution.

## Usage

```
## S3 method for class 'bayesianVARs_predict'
plot(
  x,
  dates = NULL,
  vars = "all",
  ahead = NULL,
  quantiles = c(0.05, 0.25, 0.5, 0.75, 0.95),
  n_col = 1L,
  first_obs = 1L,
  ...
)
```

## Arguments

| | |
|---|---|
| x | An object of type bayesianVARs_predict obtained via predict.bayesianVARs_bvar(). |
| dates | optional vector of dates for labeling the x-axis. The default values is NULL; in this case, the axis will be labeled with numbers. |
| vars | character vector containing the names of the variables to be visualized. The default is "all" indicating that all variables are visualized. |
| ahead | Integer vector (or coercible to such) indicating which step ahead to plot. max(ahead) must be smaller equal to dim(x$predictions)[1]. |
| quantiles | numeric vector indicating which quantiles to plot. |
| n_col | integer indicating the number of columns to use for plotting. |
| first_obs | integer indicating the first observation to be used for plotting. |
| ... | Currently ignored! |

## Value

Returns x invisibly!

## See Also

Other plotting plot.bayesianVARs_bvar(), plot.bayesianVARs_fitted(), pairs.bayesianVARs_predict() posterior_heatmap().

## Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]

# Estimate a model
mod <- bvar(data, sv_keep = "all", quiet = TRUE)

# Simulate from posterior predictive
predictions <- predict(mod, ahead = 1:3)

# Visualize
plot(predictions, vars = 1:3, ahead = 1:3)
```

---

# plot.bayesianVARs_residuals

*Visualization of the residuals of an estimated VAR.*

---

## Description

Visualization of the residuals of an estimated VAR.

## Usage

```
## S3 method for class 'bayesianVARs_residuals'
plot(
  x,
  dates = NULL,
  vars = "all",
  quantiles = c(0.05, 0.5, 0.95),
  n_col = 1L,
  ...
)
```

## Arguments

| | |
|---|---|
| x | A bayesianVARs_residuals object. |
| dates | optional vector of dates for labelling the x-axis. The default values is NULL; in this case, the axis will be labeled with numbers. |
| vars | character vector containing the names of the variables to be visualized. The default is "all" indicating that the fit of all variables is visualized. |
| quantiles | numeric vector indicating which quantiles to plot. |
| n_col | integer indicating the number of columns to use for plotting. |
| ... | Currently ignored. |

## Value

returns x invisibly

## See Also

- residuals method for class 'bayesianVARs_bvar': `residuals.bayesianVARs_bvar()`.
- Other plotting `plot.bayesianVARs_bvar()`, `plot.bayesianVARs_fitted()`, `plot.bayesianVARs_predict()`, `pairs.bayesianVARs_predict()`, `posterior_heatmap()`.

## Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]

# Estimate a model
mod <- bvar(data, sv_keep = "all", quiet = TRUE)

mod.resids <- residuals(mod)

# Visualize
plot(mod.resids)
```

---

# posterior_heatmap *Posterior heatmaps for matrix valued parameters*

---

## Description

`posterior_heatmap()` generates a heatmap for draws of matrix values parameters visualizing point wise summaries, such as mean, median, variance, standard deviation, interquartile range etc. etc.

## Usage

```
posterior_heatmap(
  x,
  FUN,
  ...,
  transpose = FALSE,
  colorbar = TRUE,
  colorbar_width = 0.1,
  gap_width = 0.25,
  xlabels = NULL,
  ylabels = NULL,
  add_numbers = FALSE,
  zlim = NULL,
  colspace = NULL,
  border_color = NA,
```

```
  zero_color = NA,
  main = "",
  detect_lags = TRUE,
  cex.axis = 0.75,
  cex.colbar = 1,
  cex.numbers = 1,
  asp = NULL
)
```

## Arguments

| | |
|---|---|
| x | An array of dimension $a \times b \times draws$, where $a \times b$ is the dimension of the parameter to visualize and draws is the number of posterior draws. |
| FUN | The summary function to be applied to margins c(1,2) of x. E.g. "median", "mean", "IQR", "sd" or "var". apply(x, 1:2, FUN, ...) must return a matrix! |
| ... | optional arguments to FUN. |
| transpose | logical indicating whether to transpose the matrix or not, i.e. whether to plot an $a \times b$ or an $b \times a$ matrix. Default is FALSE. |
| colorbar | logical indicating whether to display a colorbar or not. Default is TRUE. |
| colorbar_width | numeric. A value between 0 and 1 indicating the proportion of the width of the plot for the colorbar. |
| gap_width | numeric. A value between 0 and 1 indicating the width of the gap between the heatmap and the colorbar. The width is computed as gap_width*colorbar_width. |
| xlabels | ylabels=NULL, the default, indicates that colnames(x) will be displayed. ylabels="" indicates that no ylabels will be displayed. |
| ylabels | xlabels=NULL, the default, indicates that rownames(x) will be displayed. xlabels="" indicates that no ylabels are displayed. |
| add_numbers | logical. add_numbers=TRUE, the default indicates that the actual values of summary will be displayed. |
| zlim | numeric vector of length two indicating the minimum and maximum values for which colors should be plotted. By default this range is determined by the maximum of the absolute values of the selected summary. |
| colspace | Optional argument indicating the color palette to be used. If not specified, colorspace::diverge_hcl() will be used, or, if FUN returns only positive values colorspace::sequential_hcl(). See below for a more detailed description of the default usage. |
| border_color | The color of the rectangles borders. If not specified no borders will be displayed. |
| zero_color | The color of exact zero elements. By default this is not specified and then will depend on colspace. |
| main | main title for the plot. |
| detect_lags | logical. If class(x) is "bayesianVARs_coef", then detect_lags=TRUE will separate the sub matrices corresponding to the lags with black lines. |

| | |
|---|---|
| cex.axis | The magnification to be used for y-axis annotation relative to the current setting of cex. |
| cex.colbar | The magnification to be used for colorbar annotation relative to the current setting of cex. |
| cex.numbers | The magnification to be used for the actual values (if add_numbers=TRUE) relative to the current setting of cex. |
| asp | aspect ratio. |

## Details

**colspace:**

If not specified either `colorspace::diverge_hcl(1001, alpha = alpha, palette = "Blue-Red")` or `colorspace::sequential_hcl(1001, alpha = alpha, rev = TRUE, palette = "Reds 2")` will be used.

## Value

Returns x invisibly.

## See Also

Other plotting `plot.bayesianVARs_bvar()`, `plot.bayesianVARs_fitted()`, `plot.bayesianVARs_predict()`, `pairs.bayesianVARs_predict()`.

## Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]

# Estimate a model
mod <- bvar(100*data, sv_keep = "all", quiet = TRUE)

# Extract posterior draws of VAR coefficients
phi_post <- coef(mod)

# Visualize posterior median of VAR coefficients
posterior_heatmap(phi_post, median, detect_lags = TRUE, border_color = rgb(0, 0, 0, alpha = 0.2))

# Extract posterior draws of variance-covariance matrices (for each point in time)
sigma_post <- vcov(mod)
# Visualize posterior interquartile-range of variance-covariance matrix of the first observation
posterior_heatmap(sigma_post[1,,,], IQR)
```

predict.bayesianVARs_bvar *Predict method for Bayesian VARs*

## Description

Simulates from (out-of-sample) predictive density for Bayesian VARs estimated via `bvar()` and computes log predictive likelhoods if ex-post observed data is supplied.

## Usage

```
## S3 method for class 'bayesianVARs_bvar'
predict(
  object,
  ahead = 1L,
  each = 1L,
  stable = TRUE,
  simulate_predictive = TRUE,
  LPL = FALSE,
  Y_obs = NA,
  LPL_VoI = NA,
  ...
)
```

## Arguments

| | |
|---|---|
| `object` | A `bayesianVARs_bvar` object, obtained from `bvar()`. |
| `ahead` | Integer vector (or coercible to such), indicating the number of steps ahead at which to predict. |
| `each` | Single integer (or coercible to such) indicating how often should be drawn from the posterior predictive distribution for each draw that has been stored during MCMC sampling. |
| `stable` | logical indicating whether to consider only those draws from the posterior that fulfill the 'stable' criterion. Default is `TRUE`. |
| `simulate_predictive` | logical, indicating whether the posterior predictive distribution should be simulated. |
| `LPL` | logical indicating whether `ahead`-step-ahead log predictive likelihoods should be computed. If `LPL=TRUE`, `Y_obs` has to be specified. |
| `Y_obs` | Data matrix of observed values for computation of log predictive likelihood. Each of `ncol(object$Yraw)` columns is assumed to contain a single time-series of length `length(ahead)`. |
| `LPL_VoI` | either integer vector or character vector of column-names indicating for which subgroup of time-series in `object$Yraw` a joint log predictive likelihood shall be computed. |
| `...` | Currently ignored! |

## Value

Object of class `bayesianVARs_predict`, a list that may contain the following elements:

- `predictions` array of dimensions `c(length(ahead), ncol(object$Yraw), each * dim(object$PHI)[3])` containing the simulations from the predictive density (if `simulate_predictive=TRUE`).
- `LPL` vector of length `length(ahead)` containing the log-predictive-likelihoods (taking into account the joint distribution of all variables) (if `LPL=TRUE`).
- `LPL_univariate` matrix of dimension `c(length(ahead), ncol(object$Yraw)` containing the marginalized univariate log-predictive-likelihoods of each series (if `LPL=TRUE`).
- `LPL_VoI` vector of length `length(ahead)` containing the log-predictive-likelihoods for a subset of variables (if `LPL=TRUE` and `LPL_VoI != NA`).
- `Yraw` matrix containing the data used for the estimation of the VAR.
- `LPL_draws` matrix containing the simulations of the log-predictive-likelihood (if `LPL=TRUE`).
- `PL_univariate_draws` array containing the simulations of the univariate predictive-likelihoods (if `LPL=TRUE`).
- `LPL_sub_draws` matrix containing the simulations of the log-predictive-likelihood for a subset of variables (if `LPL=TRUE` and `LPL_VoI != NA`).

## See Also

`stable_bvar()`, `plot.bayesianVARs_predict()`, `pairs.bayesianVARs_predict()`.

## Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]

# Split data in train and test
train <- data[1:(nrow(data)-4),]
test <- data[-c(1:(nrow(data)-4)),]

# Estimate model using train data only
mod <- bvar(train, quiet = TRUE)

# Simulate from 1-step to 4-steps ahead posterior predictive and compute
# log-predictive-likelihoods
predictions <- predict(mod, ahead = 1:4, LPL = TRUE, Y_obs = test)

# Summary
summary(predictions)

# Visualize via fan-charts
plot(predictions)


# In order to evaluate the joint predictive density of a subset of the
# variables (variables of interest), consider specifying 'LPL_VoI':
predictions <- predict(mod, ahead = 1:4, LPL = TRUE, Y_obs = test, LPL_VoI = c("GDPC1","FEDFUNDS"))
predictions$LPL_VoI
```

## print.bayesianVARs_bvar

*Pretty printing of a bvar object*

### Description

Pretty printing of a bvar object

### Usage

```
## S3 method for class 'bayesianVARs_bvar'
print(x, ...)
```

### Arguments

| | |
|---|---|
| x | Object of class bayesianVARs_bvar, usually resulting from a call of bvar(). |
| ... | Ignored. |

### Value

Returns x invisibly.

### Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]

# Estimate a model
mod <- bvar(data, sv_keep = "all", quiet = TRUE)

# Print model
mod
```

## print.bayesianVARs_predict

*Print method for bayesianVARs_predict objects*

### Description

Print method for bayesianVARs_predict objects.

### Usage

```
## S3 method for class 'bayesianVARs_predict'
print(x, ...)
```

### Arguments

| | |
|---|---|
| x | A bayesianVARs_predict object obtained via predict.bayesianVARs_bvar(). |
| ... | Currently ignored! |

### Value

Returns x invisibly.

### Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]

# Split data in train and test
train <- data[1:(nrow(data)-4),]
test <- data[-c(1:(nrow(data)-4)),]

# Estimate model using train data only
mod <- bvar(train, quiet = TRUE)

# Simulate from 1-step ahead posterior predictive
predictions <- predict(mod, ahead = 1L)
print(predictions)
```

---

## print.summary.bayesianVARs_bvar

*Print method for summary.bayesianVARs_bvar objects*

---

### Description

Print method for summary.bayesianVARs_bvar objects.

### Usage

```
## S3 method for class 'summary.bayesianVARs_bvar'
print(x, ...)
```

### Arguments

| | |
|---|---|
| x | A summary.bayesianVARs_bvar object obtained via summary.bayesianVARs_bvar(). |
| ... | Currently ignored! |

### Value

Returns x invisibly!

## Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]

# Estimate model
mod <- bvar(data, quiet = TRUE)

# Print summary
summary(mod)
```

---

`print.summary.bayesianVARs_predict` *Print method for summary.bayesianVARs_predict objects*

---

## Description

Print method for `summary.bayesianVARs_predict` objects.

## Usage

```
## S3 method for class 'summary.bayesianVARs_predict'
print(x, ...)
```

## Arguments

| | |
|---|---|
| `x` | A `summary.bayesianVARs_predict` object obtained via `summary.bayesianVARs_predict()`. |
| `...` | Currently ignored. |

## Value

Returns `x` invisibly.

## Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]

# Split data in train and test
train <- data[1:(nrow(data)-4),]
test <- data[-c(1:(nrow(data)-4)),]

# Estimate model using train data only
mod <- bvar(train, quiet = TRUE)

# Simulate from 1-step ahead posterior predictive
predictions <- predict(mod, ahead = 1L)
sum <- summary(predictions)
print(sum)
```

`residuals.bayesianVARs_bvar` *Extract Model Residuals*

## Description

Extract model residuals, defined as the difference between the observed time-series and the in-sample predictions of the VAR model. Because in-sample prediction is subject to uncertainty of the VAR parameter estimates, this uncertainty carries over to the model residuals.

## Usage

```
## S3 method for class 'bayesianVARs_bvar'
residuals(object, ...)
```

## Arguments

| | |
|---|---|
| `object` | A `bayesianVARs_bvar` object estimated via `bvar()`. |
| `...` | Passed to `fitted.bayesianVARs_bvar()`. |

## Value

An object of class `bayesianVARs_residuals`.

## See Also

`fitted.bayesianVARs_bvar`

## Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]

# Estimate a model
mod <- bvar(data, sv_keep = "all", quiet = TRUE)

mod.resids <- residuals(mod)
plot(mod.resids)
```

specify_prior_phi *Specify prior on PHI*

## Description

Configures prior on PHI, the matrix of reduced-form VAR coefficients.

## Usage

```
specify_prior_phi(
  data = NULL,
  M = ncol(data),
  lags = 1L,
  prior = "HS",
  priormean = 0,
  PHI_tol = 1e-18,
  DL_a = "1/K",
  DL_tol = 0,
  R2D2_a = 0.1,
  R2D2_b = 0.5,
  R2D2_tol = 0,
  NG_a = 0.1,
  NG_b = 1,
  NG_c = 1,
  NG_tol = 0,
  SSVS_c0 = 0.01,
  SSVS_c1 = 100,
  SSVS_semiautomatic = TRUE,
  SSVS_p = 0.5,
  HMP_lambda1 = c(0.01, 0.01),
  HMP_lambda2 = c(0.01, 0.01),
  normal_sds = 10,
  global_grouping = "global",
  ...
)
```

## Arguments

| | |
|---|---|
| data | Optional. Data matrix (can be a time series object). Each of $M$ columns is assumed to contain a single time-series of length $T$. |
| M | positive integer indicating the number of time-series of the VAR. |
| lags | positive integer indicating the order of the VAR, i.e. the number of lags of the dependent variables included as predictors. |
| prior | character, one of "HS", "R2D2", "NG", "DL", "SSVS", "HMP" or "normal". |

priormean — real numbers indicating the prior means of the VAR coefficients. One single number means that the prior mean of all own-lag coefficients w.r.t. the first lag equals priormean and 0 else. A vector of length M means that the prior mean of the own-lag coefficients w.r.t. the first lag equals priormean and 0 else. If priormean is a matrix of dimension c(lags*M,M), then each of the $M$ columns is assumed to contain lags*M prior means for the VAR coefficients of the respective VAR equations.

PHI_tol — Minimum number that the absolute value of a VAR coefficient draw can take. Prevents numerical issues that can appear when strong shrinkage is enforced if chosen to be greater than zero.

DL_a — (Single) positive real number. The value is interpreted as the concentration parameter for the local scales. Smaller values enforce heavier shrinkage. If the argument global_grouping specifies e.g. k groups, then DL_a can be a numeric vector of length k and the elements indicate the shrinkage in each group. A matrix of dimension c(s,2) specifies a discrete hyperprior, where the first column contains s support points and the second column contains the associated prior probabilities. DL_a has only to be specified if prior="DL".

DL_tol — Minimum number that a parameter draw of one of the shrinking parameters of the Dirichlet Laplace prior can take. Prevents numerical issues that can appear when strong shrinkage is enforced if chosen to be greater than zero. DL_tol has only to be specified if prior="DL".

R2D2_a — (Single) positive real number. The value is interpreted as the concentration parameter for the local scales. Smaller values enforce heavier shrinkage. If the argument global_grouping specifies e.g. k groups, then R2D2_a can be a numeric vector of length k and the elements indicate the shrinkage in each group. A matrix of dimension c(s,2) specifies a discrete hyperprior, where the first column contains s support points and the second column contains the associated prior probabilities. R2D2_a has only to be specified if prior="R2D2".

R2D2_b — (Single) positive real number. The value indicates the shape parameter of the inverse gamma prior on the (semi-)global scales. If the argument global_grouping specifies e.g. k groups, then NG_b can be a numeric vector of length k and the elements determine the shape parameter in each group. R2D2_b has only to be specified if prior="R2D2".

R2D2_tol — Minimum number that a parameter draw of one of the shrinking parameters of the R2D2 prior can take. Prevents numerical issues that can appear when strong shrinkage is enforced if chosen to be greater than zero. R2D2_tol has only to be specified if prior="R2D2".

NG_a — (Single) positive real number. The value is interpreted as the concentration parameter for the local scales. Smaller values enforce heavier shrinkage. If the argument global_grouping specifies e.g. k groups, then NG_a can be a numeric vector of length k and the elements indicate the shrinkage in each group. A matrix of dimension c(s,2) specifies a discrete hyperprior, where the first column contains s support points and the second column contains the associated prior probabilities. NG_a has only to be specified if prior="NG".

NG_b — (Single) positive real number. The value indicates the shape parameter of the inverse gamma prior on the (semi-)global scales. If the argument global_grouping

specifies e.g. k groups, then NG_b can be a numeric vector of length k and the elements determine the shape parameter in each group. NG_b has only to be specified if prior="NG".

NG_c (Single) positive real number. The value indicates the scale parameter of the inverse gamma prior on the (semi-)global scales. If the argument global_grouping specifies e.g. k groups, then NG_c can be a numeric vector of length k and the elements determine the scale parameter in each group. Expert option would be to set the scale parameter proportional to NG_a. E.g. in the case where a discrete hyperprior for NG_a is chosen, a desired proportion of let's say 0.2 is achieved by setting NG_c="0.2*a" (character input!). NG_c has only to be specified if prior="NG".

NG_tol Minimum number that a parameter draw of one of the shrinking parameters of the normal-gamma prior can take. Prevents numerical issues that can appear when strong shrinkage is enforced if chosen to be greater than zero. NG_tol has only to be specified if prior="NG".

SSVS_c0 single positive number indicating the (unscaled) standard deviation of the spike component. SSVS_c0 has only to be specified if prior="SSVS". It should be that $SSVS_{c0} \ll SSVS_{c1}$! SSVS_c0 has only to be specified if prior="SSVS".

SSVS_c1 single positive number indicating the (unscaled) standard deviation of the slab component. SSVS_c0 has only to be specified if prior="SSVS". It should be that $SSVS_{c0} \ll SSVS_{c1}$!

SSVS_semiautomatic

logical. If SSVS_semiautomatic=TRUE both SSVS_c0 and SSVS_c1 will be scaled by the variances of the posterior of PHI under a FLAT conjugate (dependent Normal-Wishart prior). SSVS_semiautomatic has only to be specified if prior="SSVS".

SSVS_p Either a single positive number in the range (0,1) indicating the (fixed) prior inclusion probability of each coefficient. Or numeric vector of length 2 with positive entries indicating the shape parameters of the Beta distribution. In that case a Beta hyperprior is placed on the prior inclusion probability. SSVS_p has only to be specified if prior="SSVS".

HMP_lambda1 numeric vector of length 2. Both entries must be positive. The first indicates the shape and the second the rate of the Gamma hyperprior on own-lag coefficients. HMP_lambda1 has only to be specified if prior="HMP".

HMP_lambda2 numeric vector of length 2. Both entries must be positive. The first indicates the shape and the second the rate of the Gamma hyperprior on cross-lag coefficients. HMP_lambda2 has only to be specified if prior="HMP".

normal_sds numeric vector of length $n$, where $n = lagsM^2$ is the number of all VAR coefficients (excluding the intercept), indicating the prior variances. A single number will be recycled accordingly! Must be positive. normal_sds has only to be specified if prior="normal".

global_grouping

One of "global", "equation-wise", "covariate-wise", "olcl-lagwise" "fol" indicating the sub-groups of the semi-global(-local) modifications to HS, R2D2, NG, DL and SSVS prior. Works also with user-specified indicator matrix of dimension c(lags*M,M). Only relevant if prior="HS", prior="DL", prior="R2D2", prior="NG" or prior="SSVS".

... Do not use!

## Details

For details concerning prior-elicitation for VARs please see Gruber & Kastner (2025).

Currently one can choose between six hierarchical shrinkage priors and a normal prior: `prior="HS"` stands for the Horseshoe-prior, `prior="R2D2` for the $R^2$-induced-Dirichlet-decompostion-prior, `prior="NG"` for the normal-gamma-prior, `prior="DL"` for the Dirichlet-Laplace-prior, `prior="SSVS"` for the stochastic-search-variable-selection-prior, `prior="HMP"` for the semi-hierarchical Minnesota prior and `prior=normal` for the normal-prior.

Semi-global shrinkage, i.e. group-specific shrinkage for pre-specified subgroups of the coefficients, can be achieved through the argument `global_grouping`.

## Value

A `baysianVARs_prior_phi`-object.

## References

Gruber, L. and Kastner, G. (2025). Forecasting macroeconomic data with Bayesian VARs: Sparse or dense? It depends! *International Journal of Forecasting*. doi:10.1016/j.ijforecast.2025.02.001.

## Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]

# Horseshoe prior for a VAR(2)
phi_hs <- specify_prior_phi(data = data, lags = 2L ,prior = "HS")

# Semi-global-local Horseshoe prior for a VAR(2) with semi-global shrinkage parameters for
# cross-lag and own-lag coefficients in each lag
phi_hs_sg <- specify_prior_phi(data = data, lags = 2L, prior = "HS",
global_grouping = "olcl-lagwise")

# Semi-global-local Horseshoe prior for a VAR(2) with equation-wise shrinkage
# construct indicator matrix for equation-wise shrinkage
semi_global_mat <- matrix(1:ncol(data), 2*ncol(data), ncol(data),
byrow = TRUE)
phi_hs_ew <- specify_prior_phi(data = data, lags = 2L, prior = "HS",
global_grouping = semi_global_mat)
# (for equation-wise shrinkage one can also use 'global_grouping = "equation-wise"')


# Estimate model with your prior configuration of choice
mod <- bvar(data, lags = 2L, prior_phi = phi_hs_sg, quiet = TRUE)
```

specify_prior_sigma *Specify prior on Sigma*

## Description

Configures prior on the variance-covariance of the VAR.

## Usage

```
specify_prior_sigma(
  data = NULL,
  M = ncol(data),
  type = c("factor", "cholesky"),
  factor_factors = 1L,
  factor_restrict = c("none", "upper"),
  factor_priorfacloadtype = c("rowwiseng", "colwiseng", "normal"),
  factor_priorfacload = 0.1,
  factor_facloadtol = 1e-14,
  factor_priorng = c(1, 1),
  factor_priormu = c(0, 10),
  factor_priorphiidi = c(10, 3),
  factor_priorphifac = c(10, 3),
  factor_priorsigmaidi = 1,
  factor_priorsigmafac = 1,
  factor_priorh0idi = "stationary",
  factor_priorh0fac = "stationary",
  factor_heteroskedastic = TRUE,
  factor_priorhomoskedastic = NA,
  factor_interweaving = 4,
  cholesky_U_prior = c("HS", "DL", "R2D2", "NG", "SSVS", "normal", "HMP"),
  cholesky_U_tol = 1e-18,
  cholesky_heteroscedastic = TRUE,
  cholesky_priormu = c(0, 100),
  cholesky_priorphi = c(20, 1.5),
  cholesky_priorsigma2 = c(0.5, 0.5),
  cholesky_priorh0 = "stationary",
  cholesky_priorhomoscedastic = as.numeric(NA),
  cholesky_DL_a = "1/n",
  cholesky_DL_tol = 0,
  cholesky_R2D2_a = 0.4,
  cholesky_R2D2_b = 0.5,
  cholesky_R2D2_tol = 0,
  cholesky_NG_a = 0.5,
  cholesky_NG_b = 0.5,
  cholesky_NG_c = 0.5,
  cholesky_NG_tol = 0,
  cholesky_SSVS_c0 = 0.001,
```

```
  cholesky_SSVS_c1 = 1,
  cholesky_SSVS_p = 0.5,
  cholesky_HMP_lambda3 = c(0.01, 0.01),
  cholesky_normal_sds = 10,
  expert_sv_offset = 0,
  quiet = FALSE,
  ...
)
```

## Arguments

`data` Optional. Data matrix (can be a time series object). Each of $M$ columns is assumed to contain a single time-series of length $T$.

`M` positive integer indicating the number of time-series of the VAR.

`type` character, one of `"factor"` (the default) or `"cholesky"`, indicating which decomposition to be applied to the covariance-matrix.

`factor_factors` Number of latent factors to be estimated. Only required if `type="factor"`.

`factor_restrict`

Either "upper" or "none", indicating whether the factor loadings matrix should be restricted to have zeros above the diagonal ("upper") or whether all elements should be estimated from the data ("none"). Setting `restrict` to "upper" often stabilizes MCMC estimation and can be important for identifying the factor loadings matrix, however, it generally is a strong prior assumption. Setting `restrict` to "none" is usually the preferred option if identification of the factor loadings matrix is of less concern but covariance estimation or prediction is the goal. Only required if `type="factor"`.

`factor_priorfacloadtype`

Can be `"normal"`, `"rowwiseng"`, `"colwiseng"`. Only required if `type="factor"`.

`"normal"`**:** Normal prior. The value of `priorfacload` is interpreted as the standard deviations of the Gaussian prior distributions for the factor loadings.

`"rowwiseng"`**:** Row-wise Normal-Gamma prior. The value of `priorfacload` is interpreted as the shrinkage parameter `a`.

`"colwiseng"`**:** Column-wise Normal-Gamma prior. The value of `priorfacload` is interpreted as the shrinkage parameter `a`.

For details please see Kastner (2019).

`factor_priorfacload`

Either a matrix of dimensions `M` times `factor_factors` with positive elements or a single number (which will be recycled accordingly). Only required if `type="factor"`. The meaning of `factor_priorfacload` depends on the setting of `factor_priorfacloadtype` and is explained there.

`factor_facloadtol`

Minimum number that the absolute value of a factor loadings draw can take. Prevents numerical issues that can appear when strong shrinkage is enforced if chosen to be greater than zero. Only required if `type="factor"`.

factor_priorng Two-element vector with positive entries indicating the Normal-Gamma prior's hyperhyperparameters c and d (cf. Kastner (2019)). Only required if type="factor".

factor_priormu Vector of length 2 denoting prior mean and standard deviation for unconditional levels of the idiosyncratic log variance processes. Only required if type="factor".

factor_priorphiidi
: Vector of length 2, indicating the shape parameters for the Beta prior distributions of the transformed parameters (phi+1)/2, where phi denotes the persistence of the idiosyncratic log variances. Only required if type="factor".

factor_priorphifac
: Vector of length 2, indicating the shape parameters for the Beta prior distributions of the transformed parameters (phi+1)/2, where phi denotes the persistence of the factor log variances. Only required if type="factor".

factor_priorsigmaidi
: Vector of length M containing the prior volatilities of log variances. If factor_priorsigmaidi has exactly one element, it will be recycled for all idiosyncratic log variances. Only required if type="factor".

factor_priorsigmafac
: Vector of length factor_factors containing the prior volatilities of log variances. If factor_priorsigmafac has exactly one element, it will be recycled for all factor log variances. Only required if type="factor".

factor_priorh0idi
: Vector of length 1 or M, containing information about the Gaussian prior for the initial idiosyncratic log variances. Only required if type="factor". If an element of factor_priorh0idi is a nonnegative number, the conditional prior of the corresponding initial log variance h0 is assumed to be Gaussian with mean 0 and standard deviation factor_priorh0idi times $sigma$. If an element of factor_priorh0idi is the string 'stationary', the prior of the corresponding initial log volatility is taken to be from the stationary distribution, i.e. h0 is assumed to be Gaussian with mean 0 and variance $sigma^2/(1 - phi^2)$.

factor_priorh0fac
: Vector of length 1 or factor_factors, containing information about the Gaussian prior for the initial factor log variances. Only required if type="factor". If an element of factor_priorh0fac is a nonnegative number, the conditional prior of the corresponding initial log variance h0 is assumed to be Gaussian with mean 0 and standard deviation factor_priorh0fac times $sigma$. If an element of factor_priorh0fac is the string 'stationary', the prior of the corresponding initial log volatility is taken to be from the stationary distribution, i.e. h0 is assumed to be Gaussian with mean 0 and variance $sigma^2/(1 - phi^2)$.

factor_heteroskedastic
: Vector of length 1, 2, or M + factor_factors, containing logical values indicating whether time-varying (factor_heteroskedastic = TRUE) or constant (factor_heteroskedastic = FALSE) variance should be estimated. If factor_heteroskedastic is of length 2 it will be recycled accordingly, whereby the first element is used for all idiosyncratic variances and the second element is used for all factor variances. Only required if type="factor".

factor_priorhomoskedastic
: Only used if at least one element of factor_heteroskedastic is set to FALSE.

In that case, factor_priorhomoskedastic must be a matrix with positive entries and dimension c(M, 2). Values in column 1 will be interpreted as the shape and values in column 2 will be interpreted as the rate parameter of the corresponding inverse gamma prior distribution of the idiosyncratic variances. Only required if type="factor".

factor_interweaving

The following values for interweaving the factor loadings are accepted (Only required if type="factor"):

**0:** No interweaving.

**1:** Shallow interweaving through the diagonal entries.

**2:** Deep interweaving through the diagonal entries.

**3:** Shallow interweaving through the largest absolute entries in each column.

**4:** Deep interweaving through the largest absolute entries in each column.

For details please see Kastner et al. (2017). A value of 4 is the highly recommended default.

cholesky_U_prior

character, one of "HS", "R2D2", "NG", "DL", "SSVS", "HMP" or "normal". Only required if type="cholesky".

cholesky_U_tol Minimum number that the absolute value of an free off-diagonal element of an $U$-draw can take. Prevents numerical issues that can appear when strong shrinkage is enforced if chosen to be greater than zero. Only required if type="cholesky".

cholesky_heteroscedastic

single logical indicating whether time-varying (cholesky_heteroscedastic = TRUE) or constant (cholesky_heteroscedastic = FALSE) variance should be estimated. Only required if type="cholesky".

cholesky_priormu

Vector of length 2 denoting prior mean and standard deviation for unconditional levels of the log variance processes. Only required if type="cholesky".

cholesky_priorphi

Vector of length 2, indicating the shape parameters for the Beta prior distributions of the transformed parameters (phi+1)/2, where phi denotes the persistence of the log variances. Only required if type="cholesky".

cholesky_priorsigma2

Vector of length 2, indicating the shape and the rate for the Gamma prior distributions on the variance of the log variance processes. (Currently only one global setting for all $M$ processes is supported). Only required if type="cholesky".

cholesky_priorh0

Vector of length 1 or M, containing information about the Gaussian prior for the initial idiosyncratic log variances. Only required if type="cholesky". If an element of cholesky_priorh0 is a nonnegative number, the conditional prior of the corresponding initial log variance h0 is assumed to be Gaussian with mean 0 and standard deviation cholesky_priorh0 times $sigma$. If an element of cholesky_priorh0 is the string 'stationary', the prior of the corresponding initial log volatility is taken to be from the stationary distribution, i.e. h0 is assumed to be Gaussian with mean 0 and variance $sigma^2/(1 - phi^2)$.

cholesky_priorhomoscedastic
: Only used if cholesky_heteroscedastic=FALSE. In that case, cholesky_priorhomoscedastic must be a matrix with positive entries and dimension c(M, 2). Values in column 1 will be interpreted as the shape and values in column 2 will be interpreted as the scale parameter of the corresponding inverse gamma prior distribution of the variances. Only required if type="cholesky".

cholesky_DL_a
: (Single) positive real number. The value is interpreted as the concentration parameter for the local scales. Smaller values enforce heavier shrinkage. A matrix of dimension c(s,2) specifies a discrete hyperprior, where the first column contains s support points and the second column contains the associated prior probabilities. cholesky_DL_a has only to be specified if cholesky_U_prior="DL".

cholesky_DL_tol
: Minimum number that a parameter draw of one of the shrinking parameters of the Dirichlet Laplace prior can take. Prevents numerical issues that can appear when strong shrinkage is enforced if chosen to be greater than zero. DL_tol has only to be specified if cholesky_U_prior="DL".

cholesky_R2D2_a
: (Single) positive real number. The value is interpreted as the concentration parameter for the local scales. Smaller values enforce heavier shrinkage. A matrix of dimension c(s,2) specifies a discrete hyperprior, where the first column contains s support points and the second column contains the associated prior probabilities. cholesky_R2D2_a has only to be specified if cholesky_U_prior="R2D2".

cholesky_R2D2_b
: single positive number, where greater values indicate heavier regularization. cholesky_R2D2_b has only to be specified if cholesky_U_prior="R2D2".

cholesky_R2D2_tol
: Minimum number that a parameter draw of one of the shrinking parameters of the R2D2 prior can take. Prevents numerical issues that can appear when strong shrinkage is enforced if chosen to be greater than zero. cholesky_R2D2_tol has only to be specified if cholesky_U_prior="R2D2".

cholesky_NG_a
: (Single) positive real number. The value is interpreted as the concentration parameter for the local scales. Smaller values enforce heavier shrinkage. A matrix of dimension c(s,2) specifies a discrete hyperprior, where the first column contains s support points and the second column contains the associated prior probabilities. cholesky_NG_a has only to be specified if cholesky_U_prior="NG".

cholesky_NG_b
: (Single) positive real number. The value indicates the shape parameter of the inverse gamma prior on the global scales. cholesky_NG_b has only to be specified if cholesky_U_prior="NG".

cholesky_NG_c
: (Single) positive real number. The value indicates the scale parameter of the inverse gamma prior on the global scales. Expert option would be to set the scale parameter proportional to NG_a. E.g. in the case where a discrete hyperprior for NG_a is chosen, a desired proportion of let's say 0.2 is achieved by setting NG_c="0.2a" (character input!). cholesky_NG_c has only to be specified if cholesky_U_prior="NG".

cholesky_NG_tol
: Minimum number that a parameter draw of one of the shrinking parameters of the normal-gamma prior can take. Prevents numerical issues that can appear

when strong shrinkage is enforced if chosen to be greater than zero. `cholesky_NG_tol` has only to be specified if `cholesky_U_prior="NG"`.

`cholesky_SSVS_c0`
: single positive number indicating the (unscaled) standard deviation of the spike component. `cholesky_SSVS_c0` has only to be specified if `choleksy_U_prior="SSVS"`. It should be that $SSVS_{c0} \ll SSVS_{c1}$!

`cholesky_SSVS_c1`
: single positive number indicating the (unscaled) standard deviation of the slab component. `cholesky_SSVS_c1` has only to be specified if `choleksy_U_prior="SSVS"`. It should be that $SSVS_{c0} \ll SSVS_{c1}$!

`cholesky_SSVS_p`
: Either a single positive number in the range `(0,1)` indicating the (fixed) prior inclusion probability of each coefficient. Or numeric vector of length 2 with positive entries indicating the shape parameters of the Beta distribution. In that case a Beta hyperprior is placed on the prior inclusion probability. `cholesky_SSVS_p` has only to be specified if `choleksy_U_prior="SSVS"`.

`cholesky_HMP_lambda3`
: numeric vector of length 2. Both entries must be positive. The first indicates the shape and the second the rate of the Gamma hyperprior on the contemporaneous coefficients. `cholesky_HMP_lambda3` has only to be specified if `choleksy_U_prior="HMP"`.

`cholesky_normal_sds`
: numeric vector of length $\frac{M^2-M}{2}$, indicating the prior variances for the free off-diagonal elements in $U$. A single number will be recycled accordingly! Must be positive. `cholesky_normal_sds` has only to be specified if `choleksy_U_prior="normal"`.

`expert_sv_offset`
: ... Do not use!

`quiet`
: logical indicating whether informative output should be omitted.

`...`
: Do not use!

## Details

`bvar` offers two different specifications for the errors: The user can choose between a factor stochastic volatility structure or a cholesky stochastic volatility structure. In both cases the disturbances $\boldsymbol{\epsilon}_t$ are assumed to follow a $M$-dimensional multivariate normal distribution with zero mean and variance-covariance matrix $\boldsymbol{\Sigma}_t$. In case of the cholesky specification $\boldsymbol{\Sigma}_t = \boldsymbol{U'}^{-1}\boldsymbol{D}_t\boldsymbol{U}^{-1}$, where $\boldsymbol{U}^{-1}$ is upper unitriangular (with ones on the diagonal). The diagonal matrix $\boldsymbol{D}_t$ depends upon latent log-variances, i.e. $\boldsymbol{D}_t = diag(exp(h_{1t}), \ldots, exp(h_{Mt}))$. The log-variances follow a priori independent autoregressive processes $h_{it} \sim N(\mu_i + \phi_i(h_{i,t-1} - \mu_i), \sigma_i^2)$ for $i = 1, \ldots, M$. In case of the factor structure, $\boldsymbol{\Sigma}_t = \boldsymbol{\Lambda}\boldsymbol{V}_t\boldsymbol{\Lambda}' + \boldsymbol{G}_t$. The diagonal matrices $\boldsymbol{V}_t$ and $\boldsymbol{G}_t$ depend upon latent log-variances, i.e. $\boldsymbol{G}_t = diag(exp(h_{1t}), \ldots, exp(h_{Mt}))$ and $\boldsymbol{V}_t = diag(exp(h_{M+1,t}), \ldots, exp(h_{M+r,t}))$. The log-variances follow a priori independent autoregressive processes $h_{it} \sim N(\mu_i + \phi_i(h_{i,t-1} - \mu_i), \sigma_i^2)$ for $i = 1, \ldots, M$ and $h_{M+j,t} \sim N(\phi_i h_{M+j,t-1}, \sigma_{M+j}^2)$ for $j = 1, \ldots, r$.

## Value

Object of class `bayesianVARs_prior_sigma`.

## References

Kastner, G. (2019). Sparse Bayesian Time-Varying Covariance Estimation in Many Dimensions *Journal of Econometrics*, **210**(1), 98–115, doi:10.1016/j.jeconom.2018.11.007

Kastner, G., Frühwirth-Schnatter, S., and Lopes, H.F. (2017). Efficient Bayesian Inference for Multivariate Factor Stochastic Volatility Models. *Journal of Computational and Graphical Statistics*, **26**(4), 905–917, doi:10.1080/10618600.2017.1322091.

## See Also

`specify_prior_phi()`.

## Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]

# examples with stochastic volatility (heteroscedasticity) -----------------
# factor-decomposition with 2 factors and colwise normal-gamma prior on the loadings
sigma_factor_cng_sv <- specify_prior_sigma(data = data, type = "factor",
factor_factors = 2L, factor_priorfacloadtype = "colwiseng", factor_heteroskedastic = TRUE)

# cholesky-decomposition with Dirichlet-Laplace prior on U
sigma_cholesky_dl_sv <- specify_prior_sigma(data = data, type = "cholesky",
cholesky_U_prior = "DL", cholesky_DL_a = 0.5, cholesky_heteroscedastic = TRUE)

# examples without stochastic volatility (homoscedasticity) ----------------
# factor-decomposition with 2 factors and colwise normal-gamma prior on the loadings
sigma_factor_cng <- specify_prior_sigma(data = data, type = "factor",
factor_factors = 2L, factor_priorfacloadtype = "colwiseng",
factor_heteroskedastic = FALSE, factor_priorhomoskedastic = matrix(c(0.5,0.5),
ncol(data), 2))

# cholesky-decomposition with Horseshoe prior on U
sigma_cholesky_dl <- specify_prior_sigma(data = data, type = "cholesky",
cholesky_U_prior = "HS", cholesky_heteroscedastic = FALSE)


# Estimate model with your prior configuration of choice
mod <- bvar(data, prior_sigma = sigma_factor_cng_sv, quiet = TRUE)
```

---

`specify_structural_restrictions` *Set identifying restrictions for the structural VAR parameters.*

---

## Description

Set identifying restrictions for the structural VAR parameters.

## Usage

```
specify_structural_restrictions(
  x,
  restrictions_facload = NULL,
  restrictions_B0_inv_t = NULL,
  restrictions_B0 = NULL,
  restrictions_structural_coeff = NULL,
  restrictions_long_run_ir = NULL
)
```

## Arguments

| | |
|---|---|
| x | An object of type bayesianVARs_bvar. |
| restrictions_facload | an M times r matrix of restrictions on the factor loadings. This is equivalent to the instantaneous effects of the factor shocks. Can only be used, if the factor decomposition for sigma was specified. |
| restrictions_B0_inv_t | an M times M matrix of restrictions on the instantaneous effects of the structural shocks. Columns correspond to shocks, rows to variables. Can only be used, if the cholesky decomposition for sigma was specified. |
| restrictions_B0 | an M times M matrix of restrictions on the structural matrix. Can only be used, if the cholesky decomposition for sigma was specified. |
| restrictions_structural_coeff | a matrix of restrictions on the structural VAR coefficients. Its size should match the dimensions of x$PHI. Can only be used, if the cholesky decomposition for sigma was specified. |
| restrictions_long_run_ir | a matrix of restrictions on the long run impulse responses. The long run impulse responses are the sum of the impulse responses summed over all time horizons. Restrictions on the long run impulse responses can be specified for both the factor and the cholesky decomposition of sigma. In the case of a factor decomposition its size is expected to be M times r. In the case of a cholesky decomposition the size must be M times M. |

## Details

All restrictions_* entries have following meaning

**NA:** an unrestricted entry.

**0:** The entry at this position is restricted to be zero (i.e. an exclusion restriction).

**A positive number:** The sign of this entry should be positive.

**A negative number:** The sign of this entry should be negative.

The structural VAR(p) model is of the following form:

$$\boldsymbol{y}_t' \boldsymbol{B}_0 = \boldsymbol{x}_t' \boldsymbol{\Phi} \boldsymbol{B}_0 + \boldsymbol{\omega}_t'$$

## Author(s)

Stefan Haan <sthaan@edu.aau.at>

## See Also

irf, extractB0, specify_prior_sigma

## Examples

```
train_data <- 100 * usmacro_growth[,c("GDPC1", "GDPCTPI", "GS1", "M2REAL", "CPIAUCSL")]
prior_sigma <- specify_prior_sigma(train_data, type="cholesky", cholesky_heteroscedastic=FALSE)
mod <- bvar(train_data, lags=5L, prior_sigma=prior_sigma, quiet=TRUE)

structural_restrictions <- specify_structural_restrictions(
 mod,
 restrictions_B0=rbind(
   c(1 ,NA,0 ,NA,NA),
   c(0 ,1 ,0 ,NA,NA),
   c(0 ,NA,1 ,NA,NA),
   c(0 ,0 ,NA,1 ,NA),
   c(0 ,0 ,0 ,0 ,1 )
 )
)
irf_structural <- irf(
 mod, ahead=8,
 structural_restrictions=structural_restrictions
)
plot(irf_structural)
```

---

# stable_bvar *Stable posterior draws*

---

## Description

stable_bvar() detects and discards all posterior draws of an bayesianVARs_bvar object that do not fulfill the stability condition: A VAR(p) model is considered as stable only if the eigenvalues of the companion form matrix lie inside the unit circle.

## Usage

```
stable_bvar(object, quiet = FALSE)
```

## Arguments

| | |
|---|---|
| object | A bayesianVARs_bvar object obtained via bvar(). |
| quiet | logical indicating whether informative output should be omitted. |

## Value

An object of type bayesianVARs_bvar.

## Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]

# Estimate a model
mod <- bvar(data, sv_keep = "all", quiet = TRUE)

# Discard "unstable" draws
stable_mod <- stable_bvar(mod)
```

---

# summary.bayesianVARs_bvar

*Summary method for bayesianVARs_bvar objects*

---

## Description

Summary method for bayesianVARs_bvar objects.

## Usage

```
## S3 method for class 'bayesianVARs_bvar'
summary(object, quantiles = c(0.025, 0.25, 0.5, 0.75, 0.975), digits = 3L, ...)
```

## Arguments

| | |
|---|---|
| object | A bayesianVARs_bvar object obtained via bvar(). |
| quantiles | numeric vector which quantiles to compute. |
| digits | Single integer indicating the number of decimal places to be used for rounding the summary statistics. Negative values are not allowed. |
| ... | Currently ignored! |

## Value

An object of type summary.bayesianVARs_bvar.

## Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]

# Estimate model
mod <- bvar(data, quiet = TRUE)
```

```
# Summary
sum <- summary(mod)
```

---

`summary.bayesianVARs_draws` *Summary statistics for bayesianVARs posterior draws.*

---

## Description

Summary statistics for bayesianVARs posterior draws.

## Usage

```
## S3 method for class 'bayesianVARs_draws'
summary(object, quantiles = c(0.25, 0.5, 0.75), ...)
```

## Arguments

| | |
|---|---|
| `object` | An object of class `bayesianVARs_draws` usually obtained through extractors like `coef.bayesianVARs_bvar()` and `vcov.bayesianVARs_bvar()`. |
| `quantiles` | A vector of quantiles to evaluate. |
| `...` | Currently ignored. |

## Value

A list object of class `bayesianVARs_draws_summary` holding

- `mean`: Vector or matrix containing the posterior mean.
- `sd`: Vector or matrix containing the posterior standard deviation .
- `quantiles`: Array containing the posterior quantiles.

## See Also

Available extractors: `coef.bayesianVARs_bvar()`, `vcov.bayesianVARs_bvar()`.

## Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]
# Estimate a model
mod <- bvar(data, sv_keep = "all", quiet = TRUE)

# Extract posterior draws of VAR coefficients
bvar_coefs <- coef(mod)

# Compute summary statistics
summary_stats <- summary(bvar_coefs)
```

```
# Compute summary statistics of VAR coefficients without using coef()
summary_stats <- summary(mod$PHI)

# Test which list elements of 'mod' are of class 'bayesianVARs_draws'.
names(mod)[sapply(names(mod), function(x) inherits(mod[[x]], "bayesianVARs_draws"))]
```

---

`summary.bayesianVARs_predict` *Summary method for bayesianVARs_predict objects*

---

## Description

Summary method for `bayesianVARs_predict` objects.

## Usage

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

## Arguments

| | |
|---|---|
| `object` | A `bayesianVARs_predict` object obtained via `predict.bayesianVARs_bvar()`. |
| `...` | Currently ignored! |

## Value

A `summary.bayesianVARs_predict` object.

## Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]

# Split data in train and test
train <- data[1:(nrow(data)-4),]
test <- data[-c(1:(nrow(data)-4)),]

# Estimate model using train data only
mod <- bvar(train, quiet = TRUE)

# Simulate from 1-step ahead posterior predictive
predictions <- predict(mod, ahead = 1L)
summary(predictions)
```

## usmacro_growth *Data from the US-economy*

### Description

21 selected quarterly time-series from 1953:Q1 to 2021:Q2. From FRED-QD data base (McCracken and Ng, 2021). Release date 2021-07. Data is transformed to be interpreted as growth-rates (first log-differences with the exception of interest rates, which are already growth rates).

### Usage

```
usmacro_growth
```

### Format

A matrix with 247 rows and 21 columns.

### Source

Raw (untransformed) data available at https://www.stlouisfed.org/research/economists/mccracken/fred-databases, https://www.stlouisfed.org/-/media/project/frbstl/stlouisfed/research/fred-md/historical-vintages-of-fred-qd-2018-05-to-2024-12.zip.

### References

McCracken, M. W. and Ng, S. (2021). FRED-QD: A Quarterly Database for Macroeconomic Research, *Review, Federal Reserve Bank of St. Louis*, **103**(1), 1–44, doi:10.20955/r.103.144.

## vcov.bayesianVARs_bvar *Extract posterior draws of the (time-varying) variance-covariance matrix for a VAR model*

### Description

Returns the posterior draws of the possibly time-varying variance-covariance matrix of a VAR estimated via `bvar()`. Returns the full paths if `sv_keep="all"` when calling `bvar()`. Otherwise, the draws of the variance-covariance matrix for the last observation are returned, only.

### Usage

```
## S3 method for class 'bayesianVARs_bvar'
vcov(object, t = seq_len(nrow(object$logvar)), ...)
```

## Arguments

| | |
|---|---|
| object | An object of class bayesianVARs_bvar obtained via bvar(). |
| t | Vector indicating which points in time should be extracted, defaults to all. |
| ... | Currently ignored. |

## Value

An array of class bayesianVARs_draws of dimension $T \times M \times M \times draws$, where $T$ is the number of observations, $M$ the number of time-series and $draws$ the number of stored posterior draws.

## See Also

summary.bayesianVARs_draws, coef.bayesianVARs_bvar().

## Examples

```
# Access a subset of the usmacro_growth dataset
data <- usmacro_growth[,c("GDPC1", "CPIAUCSL", "FEDFUNDS")]
# Estimate a model
mod <- bvar(data, sv_keep = "all", quiet = TRUE)

# Extract posterior draws of the variance-covariance matrix
bvar_vcov <- vcov(mod)
```

# Index


∗ **datasets**
    usmacro_growth, 45
[.bayesianVARs_coef, 3
[.bayesianVARs_draws, 3

bvar, 4
bvar(), *8*, *10*, *15*, *23*, *25*, *28*, *41, 42*, *45, 46*

coef, 8
coef.bayesianVARs_bvar(), *7*, *43*, *46*
colorspace::diverge_hcl(), *21*
colorspace::sequential_hcl(), *21*

extractB0, 9, *12*, *41*

fitted.bayesianVARs_bvar, 10, *28*
fitted.bayesianVARs_bvar(), *7*, *16*, *28*

irf, 11, *17, 18*, *41*

my_gig, 13

pairs.bayesianVARs_predict
        *(*pairs_predict*)*, 13
pairs.bayesianVARs_predict(), *15, 16*, *19, 20*, *22*, *24*
pairs_predict, 13
plot.bayesianVARs_bvar, 14
plot.bayesianVARs_bvar(), *7*, *14*, *16*, *19, 20*, *22*
plot.bayesianVARs_fitted, 15
plot.bayesianVARs_fitted(), *14–16*, *19, 20*, *22*
plot.bayesianVARs_irf, 17
plot.bayesianVARs_predict, 18
plot.bayesianVARs_predict(), *14–16*, *20*, *22*, *24*
plot.bayesianVARs_residuals, 19
posterior_heatmap, 20
posterior_heatmap(), *14–16*, *19, 20*
predict.bayesianVARs_bvar, 23
predict.bayesianVARs_bvar(), *7*, *13*, *15*, *18*, *26*, *44*
print.bayesianVARs_bvar, 25
print.bayesianVARs_predict, 25
print.summary.bayesianVARs_bvar, 26
print.summary.bayesianVARs_predict, 27

residuals.bayesianVARs_bvar, 28
residuals.bayesianVARs_bvar(), *20*
rgig, *13*

specify_prior_phi, *5*, 29
specify_prior_phi(), *7*, *39*
specify_prior_sigma, *5*, 33, *41*
specify_prior_sigma(), *7*
specify_structural_restrictions, *9*, *11, 12*, 39
stable_bvar, 41
stable_bvar(), *7*, *24*
stochvol, *7*
summary.bayesianVARs_bvar, 42
summary.bayesianVARs_bvar(), *7*, *26*
summary.bayesianVARs_draws, 43, *46*
summary.bayesianVARs_draws(), *8*
summary.bayesianVARs_predict, 44
summary.bayesianVARs_predict(), *27*

update_fast_sv, *7*
usmacro_growth, 45

vcov.bayesianVARs_bvar, 45
vcov.bayesianVARs_bvar(), *7, 8*, *43*