Package ‘BVAR’

September 21, 2019

Type Package

Title Hierarchical Bayesian Vector Autoregression

Version 0.2.1

Date 2019-09-05

Author Nikolas Kuschnig [aut, cre], Lukas Vashold [aut], Michael McCracken [dtc] (author of the FRED-QD dataset)

Maintainer Nikolas Kuschnig <nikolas.kuschnig@wu.ac.at>


URL https://github.com/nk027/bvar

BugReports https://github.com/nk027/bvar/issues

Depends R (>= 3.3.0)

Imports mvtnorm, stats, graphics, utils

Suggests coda, vars

License GPL-3 | file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2019-09-21 11:40:04 UTC
R topics documented:

BVAR-package .................................................. 2
as.mcmc.bvar .................................................. 3
bvar .............................................................. 4
bv_dummy .......................................................... 6
bv_fcast ........................................................... 7
bv_irf ............................................................. 8
bv_metropolis .................................................... 10
bv_minnesota ..................................................... 11
bv_plot_trace .................................................... 13
bv_priors .......................................................... 14
coef.bvar .......................................................... 15
companion.bvar .................................................. 16
density.bvar ....................................................... 17
fitted.bvar ........................................................ 19
fred_qd ............................................................. 20
irf.bvar ............................................................. 21
logLik.bvar ........................................................ 23
plot.bvar ........................................................... 23
plot.bvar_fcast .................................................... 25
plot.bvar_irf ....................................................... 26
predict.bvar ....................................................... 27
summary.bvar ..................................................... 29

Index 31

<table>
<thead>
<tr>
<th>BVAR-package</th>
<th>BVAR</th>
</tr>
</thead>
</table>

Description


References

as.mcmc.bvar

Method for coda Markov chain Monte Carlo objects

Description

Method to convert chains of hyperparameters and marginal likelihoods obtained from \texttt{bvar} or coefficient values to objects compatible for further processing with \texttt{coda}, i.e., objects of class \texttt{mcmc} or \texttt{mcmc.list}. Multiple chains, that is comparable \texttt{bvar} objects may be converted using the \texttt{chains} argument.

Usage

\begin{verbatim}
as.mcmc.bvar(x, vars = NULL, vars_response = NULL, vars_impulse = NULL, chains = list(), ...)
\end{verbatim}

Arguments

\begin{description}
\item \texttt{x} \hspace{1cm} A \texttt{bvar} object, obtained from \texttt{bvar}.
\item \texttt{vars} \hspace{1cm} Optional character vector used to subset the converted hyperparameters. The elements need to match the names of hyperparameters (plus \texttt{"ml"}). Defaults to \texttt{NULL}, i.e. all variables.
\item \texttt{vars_response}, \texttt{vars_impulse} \hspace{1cm} Optional integer vector with the positions of coefficient values to convert. \texttt{vars_response} corresponds to a specific dependent variable, \texttt{vars_impulse} to an independent one. Note that the constant is found at position one.
\item \texttt{chains} \hspace{1cm} List with additional \texttt{bvar} objects. If provided, contents are converted to an object of class \texttt{mcmc.list}.
\item \texttt{...} \hspace{1cm} Other parameters for \texttt{as.mcmc} and \texttt{as.mcmc.list}.
\end{description}

See Also

\texttt{bvar}; \texttt{mcmc}

Examples

\begin{verbatim}
library("coda")

data <- matrix(rnorm(200), ncol = 2)
x <- bvar(data, lags = 2)
y <- bvar(data, lags = 2)

# Convert hyperparameter lambda and the marginal likelihood
as.mcmc(x, vars = c("ml", "lambda"))

# Add second chain for further processing
as.mcmc(x, vars = c("ml", "lambda"), chains = list(y = y))
\end{verbatim}
Hierarchical Bayesian Vector Autoregression

Description

Hierarchical Bayesian estimation of Vector Autoregression (VAR) models in the fashion of Giannone et al. (2015). Options for the priors and mh arguments are provided via the functions bv_priors and bv_metropolis. Several methods facilitate analysis, including summary.bvar, plot.bvar, predict.bvar and irf.bvar.

Usage

bvar(data, lags, n_draw = 10000, n_burn = 5000, n_thin = 1,
     priors = bv_priors(), mh = bv_mh(), fcast = NULL, irf = NULL,
     verbose = TRUE, ...)

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

Arguments

data Numeric matrix or dataframe. Note that observations need to be ordered from earliest to latest one.
lags Integer scalar. Number of lags to apply to the data.
n_draw Integer scalar. Number of total iterations for the model to cycle through.
n_burn Integer scalar. Number of iterations to discard.
n_thin Integer scalar. Provides the option of reducing the number of stored iterations to every n_thin'th one. The number of saved iterations thus equals \((n_{\text{draw}} - n_{\text{burn}})/n_{\text{thin}}\).
priors bv_priors object containing priors and their settings. See bv_priors.
mh bv_metropolis object with settings regarding the acceptance rate of the Metropolis-Hastings step. See bv_mh.
fcast bv_fcast object of forecast options set with bv_fcast. May be set to NULL to skip forecasting. Forecasts may also be calculated ex-post using predict.bvar.
irf bv_irf object with options regarding impulse responses and forecast error variance decompositions. Set via bv_irf or skipped when set to NULL. May also be computed ex-post using irf.bvar.
verbose Logical scalar. Whether to print intermediate results and progress.
... Not used.
x A bvar object.
Value

Returns a bvar object with the following elements:

- **beta** - Numeric array with saved draws from the posterior distribution of the VAR coefficients. See `coef.bvar`.
- **sigma** - Numeric array with saved draws from the posterior distribution of the model’s VCOV-matrix. See `vcov.bvar`.
- **hyper** - Numeric matrix with saved draws from the posterior distributions of the hierarchical priors’ hyperparameters.
- **ml** - Numeric vector with the values of the posterior marginal likelihood corresponding to each draw of hyperparameters and associated VAR coefficients.
- **optim** - List with outputs from `optim`, which is used to find suitable starting values.
- **prior** - bv_priors object. See `bv_priors`.
- **call** - Call to the function. See `match.call`.
- **meta** - List with meta information such as number of variables, accepted draws, number of iterations, et cetera.
- **variables** - Character vector with the column names of `data`.
- **fcast** - bvar_fcast object with posterior forecast draws, quantiles as well as the forecast’s setup from the `fcast` argument.
- **irf** - bvar_irf object with posterior impulse response and their quantiles, forecast error variance decomposition draws, as well as the setup from the `irf` argument.

Author(s)

Nikolas Kuschnig, Lukas Vashold

References


See Also

`bv_priors`; `bv_mh`; `bv_fcast`; `bv_irf`; `predict.bvar`; `irf.bvar`; `plot.bvar`;

Examples

# Access a subset of the fred_qd dataset and transform it to be stationary
data("fred_qd")
data <- fred_qd[, c("CPIAUCSL", "UNRATE", "FEDFUNDS")]
data[5:nrow(data), 1] <- diff(log(data[, 1]), lag = 4) * 100
data <- data[5:nrow(data), ]

# Compute VAR using 2 lags and a ridiculously low number of draws
x <- bvar(data = data, lags = 1,
          n_draw = 500, n_burn = 400, n_thin = 2, verbose = FALSE)
# Check out some of the outputs generated
plot(x)
predict(x)
plot(predict(x))
irf(x)
plot(irf(x))

---

**bv_dummy**

**Dummy prior settings**

**Description**

Allows the creation of dummy observation priors for `bv_priors`. See the Details section for information on common dummy priors.

**Usage**

```
bv_dummy(mode = 1, sd = 1, min = 0.0001, max = 5, fun)
bv_soc(mode = 1, sd = 1, min = 0.0001, max = 50)
bv_sur(mode = 1, sd = 1, min = 0.0001, max = 50)
```

**Arguments**

- **mode**: Numeric scalar (/vector). Mode (or the like) of the parameter.
- **sd**: Numeric scalar with the standard deviation.
- **min**: Numeric scalar (/vector). Minimum allowed value.
- **max**: Numeric scalar (/vector). Maximum allowed value.
- **fun**: Function taking `Y`, `lags` and the prior’s parameter `par` to generate and return a named list with elements `X` and `Y` (numeric matrices).

**Details**

Dummy priors are often used to "reduce the importance of the deterministic component implied by VARs estimated conditioning on the initial observations" (Giannone et al. 2015, p. 440). One such prior is the sum-of-coefficients (SOC) prior, which imposes the notion that a no-change forecast is optimal at the beginning of a time series. Its key parameter $\mu$ controls the tightness - i.e. for low values the model is pulled towards a form with as many unit roots as variables and no cointegration. Another such prior is the single-unit-root (SUR) prior, that allows for cointegration relationships in the data. It pushes variables either towards their unconditional mean or towards the presence of at least one unit root. These priors are implemented via Theil mixed estimation, i.e. by adding dummy-observations on top of the data matrix. They are readily available via the shorthand functions `bv_soc` and `bv_sur`. 
Value

Returns a named list of class bv_dummy for bv_priors.

References


See Also

bv_priors; bv_minnesota

Examples

# Create a sum-of-coefficients prior
add_soc <- function(Y, lags, par) {
  soc <- if(lags == 1) {diag(Y[1, ,]) / par} else {
    diag(colMeans(Y[1:lags, ,])) / par
  }
  Y_soc <- soc
  X_soc <- cbind(rep(0, ncol(Y)), matrix(rep(soc, lags), nrow = ncol(Y))

  return(list("Y" = Y_soc, "X" = X_soc))
}
soc <- bv_dummy(mode = 1, sd = 1, min = 0.0001, max = 50, fun = add_soc)

# Create a single-unit-root prior
add_sur <- function(Y, lags, par) {
  sur <- if(lags == 1) {Y[1, ,] / par} else {
    colMeans(Y[1:lags, ,]) / par
  }
  Y_sur <- sur
  X_sur <- c(1 / par, rep(sur, lags))

  return(list("Y" = Y_sur, "X" = X_sur))
}
sur <- bv_dummy(mode = 1, sd = 1, min = 0.0001, max = 50, fun = add_sur)

# Adding them to the prior list with `bv_priors()`
priors_dum <- bv_priors(hyper = "auto", soc = soc, sur = sur)

---

**Forecast settings**

Provide forecast settings to `bvar` or `predict.bvar`. Allows adjusting the horizon of the forecast.
Usage

```r
bv_fcast(horizon = 12, conditional = FALSE)
## S3 method for class 'bv_fcast'
print(x, ...)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>horizon</td>
<td>Integer scalar. Horizon for which forecasts should be computed.</td>
</tr>
<tr>
<td>conditional</td>
<td>Logical scalar. Not yet implemented.</td>
</tr>
<tr>
<td>x</td>
<td>Object of class bv_fcast.</td>
</tr>
<tr>
<td>...</td>
<td>Not used.</td>
</tr>
</tbody>
</table>

Details

As of now only unconditional forecasts are implemented and conditional must be set to FALSE. Conditional forecasts will require further options, such as variable paths.

Value

Returns a named list of class `bv_fcast` with options for `bvar` or `predict.bvar`.

See Also

`bvar`; `predict.bvar`; `plot.bvar_fcast`

Examples

```r
# Set the forecast-horizon to 20 time periods instead of 12
bv_fcast(horizon = 20)
```

---

### bv_irf

#### Impulse response settings

**Description**

Provide settings for the computation of impulse responses to `bvar`, `irf.bvar` or `fevd.bvar`. Allows setting the horizon for which impulse responses should be computed, whether or not forecast error variance decompositions (FEVDs) should be included and if and what kind of identification should be used.

**Usage**

```r
bv_irf(horizon = 12, fevd = FALSE, identification = TRUE,
       sign_restr = NULL, sign_lim = 1000)
```
Arguments

horizon  Integer scalar. The horizon for which impulse responses (and FEVDs) should be computed.

fevd  Logical scalar. Whether or not forecast error variance decompositions should be calculated.

identification  Logical scalar. Whether or not the shocks used for calculating impulses should be identified. Defaults to TRUE, i.e. identification via Cholesky decomposition unless sign_restr is provided.

sign_restr  Numeric matrix. Sign restrictions for identification. Elements should be set to 1 (−1) to restrict for positive (negative) impacts. If no presumption about the impact can be made the corresponding elements can be set to 0. The default value is NULL, meaning identification would be performed via Cholesky decomposition.

sign_lim  Integer scalar. Maximum number of rotational matrices to draw and check for fitting sign restrictions.

Details

Identification can be performed via Cholesky decomposition and sign restrictions. The algorithm for generating suitable sign restrictions follows Rubio-Ramirez et al. (2010). Note the possibility of finding no suitable sign restrictions.

Value

Returns a named list of class bv_irf with options for bvar, irf.bvar or fevd.bvar.

References


See Also

bvar; irf.bvar; plot.bvar_irf

Examples

# Set impulse responses to a horizon of 20 time periods and enable FEVD
# (Identification is performed via Cholesky decomposition)
bv_irf(horizon = 20, fevd = TRUE)

# Identify impulse responses using sign restrictions
data("fred_qd")
data <- fred_qd[, c("CPIAUCSL", "UNRATE", "FEDFUNDS")]
data[5:nrow(data), 1] <- diff(log(data[, 1]), lag = 4) * 100
data <- data[5:nrow(data), ]

# Signs should be based on economic theory
signs <- matrix(c(1, 1, -1, -1, -1, -1, -1, 1, 1), nrow = 3)
irf_signs <- bv_irf(sign_restr = signs)

bvar(data, lags = 5, irf = irf_signs)

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

**Arguments**

- `scale_hess` Numeric scalar or vector. Scaling parameter, determining the range of hyper-parameter draws. **Should be calibrated so a reasonable rate of acceptance is reached.** If provided as vector the length must equal the number of hyperparameters (one per variable for psi).
- `adjust_acc` Logical scalar. Whether or not to further scale the variability of parameter draws during the burn-in phase.
- `acc_lower`, `acc_upper` Numeric scalar. Lower (upper) bound of the target acceptance rate. Required if `adjust_acc` is set to TRUE.
- `acc_change` Numeric scalar. Percent change applied to the Hessian matrix. Required if `adjust_acc` is set to TRUE.
- `x` An object of class `bv_metropolis`.
- `...` Not used

**Value**

Returns a named list of class `bv_metropolis` with options for `bvar`. 
Examples

# Only adjust the scale parameter
bv_mh(scale_hess = 10)

# Turn on automatic scaling of the acceptance rate to [20%, 40%]
bv_mh(adjust_acc = TRUE, acc_lower = 0.2, acc_upper = 0.4)

# Increase the rate of automatic scaling
bv_mh(adjust_acc = TRUE, acc_lower = 0.2, acc_upper = 0.4, acc_change = 0.1)

Description

Provide settings for the Minnesota prior to `bv_priors`. See the Details section for further information.

Usage

```r
bv_minnesota(lambda = bv_lambda(mode = 0.2, sd = 0.4, min = 0.0001, max = 5), alpha = bv_alpha(2, 0.25, 1, 3), psi = bv_psi(0.004, 0.004, "auto"), var = 10000000, b = "auto")
```

```
BV_minnesota(lambda = bv_lambda(mode = 0.2, sd = 0.4, min = 0.0001, max = 5),
alpha = bv_alpha(2, 0.25, 1, 3), psi = bv_psi(0.004, 0.004, "auto"),
var = 10000000, b = "auto")
```

**Arguments**

- `lambda` List constructed via `bv_lambda`. Arguments are `mode`, `sd`, `min` and `max`. May also be provided as a numeric vector of length 4. See the Details section for further information.

- `alpha` List constructed via `bv_alpha`. Arguments are `mode`, `min` and `max`. High values for `mode` may affect invertibility of the augmented data matrix. May also be provided as a numeric vector of length 4. See the Details section for further information.

- `psi` List with elements `scale`, `shape` of the prior as well as `mode` and optionally `min` and `max`. The length of these needs to match the number of variables (i.e. columns) in the data. By default `mode` is set automatically to the square root of
the innovations variance after fitting an $AR(p)$ model to the data. By default $min / max$ are set to $mode$ divided / multiplied by 100. See the Details section for further information.

- **var**: Numeric scalar with the prior variance on the model’s constant.
- **b**: Numeric matrix with the prior mean.
- **mode**: Numeric scalar (/vector). Mode (or the like) of the parameter.
- **sd**: Numeric scalar with the standard deviation.
- **min**: Numeric scalar (/vector). Minimum allowed value.
- **max**: Numeric scalar (/vector). Maximum allowed value.
- **scale, shape**: Numeric scalar. Scale and shape parameters of a Gamma distribution.

### Details

Essentially this prior imposes the hypothesis, that the individual variables all follow random walk processes. This parsimonious specification typically performs well in forecasts of macroeconomic time series and is often used as a benchmark for evaluating accuracy (Kilian and Lütkepohl, 2017). The key parameter is $\lambda$ ($\text{lambda}$), which controls the tightness of the prior. The parameter $\alpha$ ($\text{alpha}$) governs variance decay with increasing lag order, while $\psi$ ($\text{psi}$) controls the prior’s standard deviation on lags of variables other than the dependent. The Minnesota prior is often refined with additional priors, trying to minimise the importance of conditioning on initial observations. See `bv_dummy` for more information on such priors.

### Value

Returns a list of class `bv_minnesota` with options for `bvar`.

### References


### See Also

`bv_priors`; `bv_dummy`

### Examples

```r
# Adjust alpha and the Minnesota prior variance.
bv_mn(
  alpha = bv_alpha(mode = 0.5, sd = 1, min = 1e-12, max = 10),
  var = 1e6
)

# Optionally use a vector as shorthand
bv_mn(alpha = c(0.5, 1, 1e-12, 10), var = 1e6)

# Only adjust lambda's standard deviation
bv_mn(lambda = bv_lambda(sd = 2))
```
**bv_plot_trace**

**Hyperparameter trace & density plot**

**Description**

**Deprecated. Use plot.bvar instead.** Diagnostic plots of the trace / density of a single hyperparameter. A parameter may be plotted across multiple iterations of `bvar` via the ellipsis parameter. Given that the settings for `bvar` are identical this can be used to assess convergence.

**Usage**

```r
bv_plot_trace(x, name, ...)

bv_plot_density(x, name, ...)
```

**Arguments**

- `x`: A `bvar` object, obtained from `bvar`.
- `name`: String with the name of the hyperparameter to plot. Possible values are names of hyperparameters and "ml" for the posterior marginal likelihood.
- `...`: Further `bvar` objects to include in the plot. The desired hyperparameter must be available and priors of the objects should match.

**Examples**

```r
data <- matrix(rnorm(200), ncol = 2)
x <- bvar(data, lags = 2)

# Plot the trace of lambda
bv_plot_trace(x, "lambda")

# Adjust par() and plot the density of the posterior marginal likelihood
op <- par(mar = c(2, 2, 2, 0.5))
bv_plot_density(x, "ml")
par(op)

# Assess parameter convergence of several chains via their trace
y <- bvar(data, lags = 2)
z <- bvar(data, lags = 2)
bv_plot_trace(x, "lambda", y, z)
```
**Description**

Function to provide priors and their parameters to `bvar`. Used for adjusting the parameters treated as hyperparameters, the Minnesota prior and adding various dummy priors through the ellipsis parameter. Note that treating $\psi$ ($psi$) as a hyperparameter in a model with many variables may lead to very low acceptance rates and thus hinder convergence.

**Usage**

```r
bv_priors(hyper = "auto", mn = bv_mn(), ...)
```

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

## S3 method for class 'bv_minnesota'
print(x, indent = FALSE, ...)

## S3 method for class 'bv_dummy'
print(x, indent = FALSE, ...)

## S3 method for class 'bv_psi'
print(x, indent = FALSE, ...)
```

**Arguments**

- `hyper`: Character vector. Used to specify the parameters to be treated as hyperparameters. May also be set to "auto" or "full" for an automatic / full subset. Other allowed values are the Minnesota prior's parameters "lambda", "alpha" and "psi" as well as the names of additional dummy priors included via `...`.
- `mn`: List of class "bv_minnesota". Options for the Minnesota prior, set via `bv_mn`.
- `...`: Optional lists of class "bv_dummy" with options for dummy priors. **Must be assigned a name in the function call.** Created with `bv_dummy`.
- `x`: Object of class `bv_priors`, `bv_dummy` or `bv_psi`.
- `indent`: Logical scalar. Used internally to format printed outputs.

**Value**

Returns a named list of class `bv_priors` with options for `bvar`.

**See Also**

`bv_mn`; `bv_dummy`
Examples

# Extending hyperparameters to the full Minnesota prior
bv_priors(c("lambda", "alpha", "psi"))
# Alternatively
bv_priors("full")

# Adding a dummy prior via 'bv_dummy()'

# First create a single-unit-root prior
add_sur <- function(Y, lags, par) {
  sur <- if(lags == 1) (Y[1,] / par) else {
    colMeans(Y[1:lags,]) / par
  }
  Y_sur <- sur
  X_sur <- c(1 / par, rep(sur, lags))
  return(list("Y" = Y_sur, "X" = X_sur))
}
sur <- bv_dummy(mode = 1, sd = 1, min = 0.0001, max = 50, fun = add_sur)

# Then add the prior to 'bv_priors()'
priors_dum <- bv_priors(hyper = "auto", sur = sur)

coef.bvar

Coefficient and VCOV methods for Bayesian VARs

Description

Retrieves coefficient / variance-covariance values for Bayesian VARs generated with bvar. Note that coefficients are available for every stored draw and credible intervals may be set via the conf_bands argument.

Usage

## S3 method for class 'bvar'
coef(object, conf_bands = 0.5, companion = FALSE, ...)

## S3 method for class 'bvar'
vcov(object, conf_bands = 0.5, ...)

## S3 method for class 'bvar_coefs'
print(x, digits = 3L, complete = FALSE, ...)

## S3 method for class 'bvar_vcovs'
print(x, digits = 3L, complete = FALSE, ...)
Arguments

- **object**: A bvar object, obtained from `bvar`.
- **conf_bands**: Numeric vector of desired confidence bands to apply. E.g. for bands at 5%, 10%, 90% and 95% set this to `c(0.05, 0.1)`. Note that the median, i.e. 0.5 is always included.
- **companion**: Logical scalar. Whether to retrieve the companion matrix of coefficients. See `companion.bvar`.
- **...**: Not used.
- **x**: Object of class `bvar_coefs` or `bvar_vcovs`.
- **digits**: Integer scalar. Fed to `round` and applied to numeric outputs (i.e. the quantiles).
- **complete**: Logical scalar. Whether to print only medians or all available confidence bands.

Value

Returns a numeric array of class `bvar_coefs` / `bvar_vcovs` with desired values at the specified confidence bands.

See Also

`bvar`; `companion.bvar`

Examples

```r
  data <- matrix(rnorm(200), ncol = 2)
  x <- bvar(data, lags = 2)

  # Get coefficient values at the 10%, 50% and 90% quantiles
  coef(x, conf_bands = 0.10)

  # Only get the median of the variance-covariance matrix
  vcov(x, conf_bands = 0.5)
```

Description

Calculates the companion matrix for Bayesian VARs generated via `bvar`.

Usage

```r
  ## S3 method for class 'bvar'
  companion(object, conf_bands = 0.5, complete = FALSE, ...)
```

```r
  companion(object, ...)
```
### Arguments

- **object**: A bvar object, obtained from `bvar`.
- **conf_bands**: Numeric vector of desired confidence bands to apply. E.g. for bands at 5%, 10%, 90% and 95% set this to c(0.05, 0.1). Note that the median, i.e. 0.5 is always included.
- **complete**: Logical value. Whether to retrieve the companion matrix for all saved draws of the VAR coefficients. Overrides `conf_bands` if set `TRUE`.
- **...**: Not used.

### Value

Returns a numeric array/matrix of class `bvar_comp` with the VAR’s coefficients in companion form, at the specified confidence bands.

### See Also

- `bvar`: coef

### Examples

```r
data <- matrix(rnorm(200), ncol = 2)
x <- bvar(data, lags = 2)

# Get companion matrices for confidence bands at to 10%, 50% and 90%
companion(x, conf_bands = 0.10, complete = FALSE)

# Get companion matrices for all draws of the VAR coefficients
companion(x, complete = TRUE)
```

---

**Density methods for Bayesian VARs**

### Description

Calculates densities of hyperparameters or coefficient values of Bayesian VARs generated via `bvar`. Wraps standard `density` functionality into a list.

### Usage

```r
## S3 method for class 'bvar'
density(x, vars = NULL, vars_response = NULL, vars_impulse = NULL, ...)

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

plot(x, mar = c(2, 2, 2, 0.5),
     mfrow = c(length(x), 1), ...)

independent_index(var, n_vars, lag)

### Arguments

- **x**: A `bvar` object, obtained from `bvar`.
- **vars**: Optional character vector used to specify hyperparameters to retrieve the density of. The elements need to match the names of hyperparameters (plus "ml"). Defaults to NULL, i.e. all hyperparameters.
- **vars_response, vars_impulse**: Optional integer vector with the positions of coefficient values to retrieve densities of. `vars_response` corresponds to a specific dependent variable, `vars_impulse` to an independent one. Note that the constant is found at position one.
- **...**: Fed to `density` or `par`.
- **mar**: Numeric vector. Margins for `par`.
- **mfrow**: Numeric vector. Rows for `par`.
- **var, n_vars, lag**: Integer scalars.

### Value

Returns a list with outputs of `density`.

### See Also

- `bvar`

### Examples

```r
data <- matrix(rnorm(200), ncol = 2)
x <- bvar(data, lags = 2)

# Get densities of standard hyperparameters
density(x)

# Plot them
plot(density(x))

# Only get the density of the marginal likelihood
density(x, vars = "ml")

# Check out the constant's density on both dependents
plot(density(x, vars_impulse = 1))
```
# Get the density of the 1st lag of variable 2's coefficients with
# respect to variable 1
idx <- independent_index(var = 2, n_vars = 2, lag = 1)
density(x, vars_response = 1, vars_impulse = idx)

---

**fitted.bvar**

**Fitted and residual methods for Bayesian VARs**

### Description

Calculates fitted values / residuals for Bayesian VARs generated with `bvar`.

### Usage

```r
## S3 method for class 'bvar'
fitted(object, conf_bands = 0.5, ...)

## S3 method for class 'bvar'
residuals(object, conf_bands = 0.5, ...)

## S3 method for class 'bvar_resid'
plot(x, vars = NULL, mar = c(2, 2, 2, 0.5), ...)

## S3 method for class 'bvar_fitted'
print(x, digits = 2L, ...)

## S3 method for class 'bvar_resid'
print(x, digits = 2L, ...)
```

### Arguments

- **object**
  - A `bvar` object, obtained from `bvar`.
- **conf_bands**
  - Numeric vector of desired confidence bands to apply. E.g. for bands at 5%, 10%, 90% and 95% set this to `c(0.05, 0.1)`. Note that the median, i.e. 0.5 is always included.
- **...**
  - Graphical parameters for `par`.
- **x**
  - Object of class `bvar_fitted` / `bvar_resid`.
- **vars**
  - Optional numeric vector. Used to subset the plot to certain variables by position. Defaults to `NULL`, i.e. all variables.
- **mar**
  - Numeric vector. Margins for `par`.
- **digits**
  - Integer scalar. Fed to `round` and applied to numeric outputs (i.e. the quantiles).

### Value

Returns a numeric array of class `bvar_fitted` / `bvar_resid` with desired values at the specified confidence bands.
See Also

bvar

Examples

data <- matrix(rnorm(200), ncol = 2)
x <- bvar(data, lags = 2)

# Get fitted values and adjust confidence bands to 10%, 50% and 90%
fitted(x, conf_bands = 0.10)

# Get residuals
residuals(x)

Description

FRED-QD is a quarterly frequency companion to FRED-MD, a large macroeconomic database. It is designed to emulate the dataset used in "Disentangling the Channels of the 2007-2009 Recession" by Stock and Watson (2012), but also contains several additional series. The currently included dataset is from April 2019, contains observations from 1959Q1 until 2018Q4, and has been subset to series that either are in public domain, or we were given permission to use.

Usage

fred_qd

Format

A data.frame with 240 observations of 234 (248) variables.

Details

For further details see McCracken and Ng (2016) or the dataset’s appendix at https://research.stlouisfed.org/econ/mccracken/fred-databases/. The FRED-QD database is made available under a modified ODC-BY 1.0 license, which can be found in the provided LICENSE file. Thanks to Michael McCracken, Adrienne Brennecke and the Federal Reserve Bank of St. Louis for making this database available and their responsiveness and help regarding licensing issues.

Source

https://research.stlouisfed.org/econ/mccracken/fred-databases/
References


iRF.bvar

**Impulse response and forecast error methods for Bayesian VARs**

**Description**

Retrieves / calculates impulse response functions (IRFs) and/or forecast error variance decompositions (FEVDs) for Bayesian VARs generated via `bvar`. If the object is already present and no settings are supplied it is simply retrieved, otherwise it will be calculated ex-post. Note that FEVDs require the presence / calculation of IRFs. To store IRFs you may want to assign the output of `irf.bvar` to `x$irf`. May also be used to update confidence bands, i.e. credible intervals.

**Usage**

```r
## S3 method for class 'bvar'
irf(x, ..., conf_bands, n_thin = 1L)

## S3 method for class 'bvar'
fevd(x, ..., conf_bands = 0.5, n_thin = 1L)

irf(x, ...)

fevd(x, ...)

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

## S3 method for class 'bvar_fevd'
print(x, digits = 4L, complete = FALSE, ...)

## S3 method for class 'bvar_irf'
summary(object, vars_impulse = NULL, vars_response = NULL, ...)
```

**Arguments**

- `x, object` A `bvar` object, obtained from `bvar`. Summary and print methods take in a `bvar_irf / bvar_fevd` object.
- `...` A `bv_irf` object or arguments to be fed into `bv_irf`. Contains settings for the IRFs / FEVDs.
conf_bands  Numeric vector of desired confidence bands to apply. E.g. for bands at 5%, 10%, 90% and 95% set this to c(0.05, 0.1). Note that the median, i.e. 0.5 is always included.

n_thin    Integer scalar. Every n_thin'th draw in x is used for calculations, others are dropped.

digits   Integer scalar. Fed to round and applied to numeric outputs (i.e. the quantiles).

complete Logical scalar. Whether to print only medians or all available confidence bands of a a bvar_fevd object.

vars_impulse, vars_response
Optional numeric or character vector. Used to subset the summary’s impulses / responses to certain variables by position or name (must be available). Defaults to NULL, i.e. all variables.

Value

Returns a list of class bvar_irf including IRFs and optionally FEVDs at desired confidence bands. Also see bvar. Note that the fevd method only returns a numeric array of FEVDs at desired confidence bands. The summary method returns a numeric array of impulse responses at the specified confidence bands.

See Also

bvar; plot.bvar_irf; bv_irf

Examples

data <- matrix(rnorm(400), ncol = 4)
x <- bvar(data, lags = 2)

# Add IRFs
x$irf <- irf(x)

# Access IRFs and update confidence bands
irf(x, conf_bands = 0.01)

# Compute and store IRFs with a longer horizon
x$irf <- irf(x, horizon = 24L)

# Lower draws, use ‘bv_irf()’ to set options and add confidence bands
irf(x, bv_irf(24L), n_thin = 10L, conf_bands = c(0.05, 0.16))

# Get a summary of the last saved IRFs
summary(x)

# Limit the summary to responses of variable #2
summary(x, vars_response = 2L)
logLik.bvar

Log-Likelihood method for Bayesian VARs

Description

Calculates the log-likelihood of Bayesian VARs generated with \texttt{bvar}.

Usage

## S3 method for class 'bvar'
logLik(object, ...)

Arguments

\textbf{object} \hspace{1cm} A \texttt{bvar} object, obtained from \texttt{bvar}.

\textbf{...} \hspace{1cm} Other graphical parameters for \texttt{par}.

Value

Returns an object of class \texttt{logLik}.

See Also

\texttt{bvar}

Examples

```r
data <- matrix(rnorm(200), ncol = 2)
x <- bvar(data, lags = 2)
# Get log-likelihood
logLik(x)
```

plot.bvar

Plotting method for Bayesian VARs

Description

Method to plot trace and densities of hyperparameters and marginal likelihood or coefficient values obtained from \texttt{bvar}. Plots may be subset to certain types using \texttt{type} and to hyperparameters using \texttt{vars}. Multiple chains, that is comparable \texttt{bvar} objects, may be plotted together using the \texttt{chains} argument. The \texttt{type} argument may be used to access \texttt{plot.bvar_irf} and \texttt{plot.bvar_fcast}. 
Usage

```r
## S3 method for class 'bvar'
plot(x, type = c("full", "trace", "density", "irf", "fcast"), vars = NULL, vars_response = NULL, vars_impulse = NULL,
chains = list(), mar = c(2, 2, 2, 0.5), ...)
```

```r
bv_plot(x, mar = c(2, 2, 2, 0.5), ...)
```

Arguments

- **x**: A `bvar` object, obtained from `bvar`.
- **type**: A string with the type of plot desired. The standard method “full” includes both density and trace plots.
- **vars**: Optional character vector used to subset the plot. The elements need to match the names of hyperparameters (plus "ml"). Defaults to NULL, i.e. all hyperparameters.
- **vars_response**, **vars_impulse**: Optional integer vector with the positions of coefficient values used to subset the plot. `vars_response` corresponds to a specific dependent variable, `vars_impulse` to an independent one. Note that the constant is found at position one.
- **chains**: List with additional `bvar` objects. Contents are then added to trace and density plots.
- **mar**: Numeric vector. Margins for `par`.
- **...**: Other graphical parameters for `par`.

Value

Returns `x` invisibly.

See Also

- `bvar`
- `plot.bvar_fcast`
- `plot.bvar_irf`

Examples

```r
data <- matrix(rnorm(200), ncol = 2)
x <- bvar(data, lags = 2, irf = bv_irf(), fcast = bv_fcast())
y <- bvar(data, lags = 2)

# Plot full traces and densities
plot(x)

# Compare with second chain
plot(x, chains = y)

# Only plot the marginal likelihood’s density
plot(x, "dens", "ml")
```
# Use plot as an alternative to plot(irf(x)) and plot(predict(x))
plot(x, "irf")
plot(x, "fcast", vars = 2)

plot.bvar_fcast  Plotting method for Bayesian VAR forecasts

Description

Plotting method for forecasts obtained from bvar or predict.bvar. Forecasts of all or a subset of the available variables can be plotted.

Usage

## S3 method for class 'bvar_fcast'
plot(x, conf_bands, vars = NULL, variables = NULL,
     orientation = c("vertical", "horizontal"), mar = c(2, 2, 2, 0.5),
     ...)  
bv_plot_fcast(x, conf_bands = 0.16, variables = NULL, vars = NULL,
    orientation = c("vertical", "horizontal"), mar = c(2, 2, 2, 0.5),
    ...)

Arguments

x  A bvar / bvar_fcast object, obtained from bvar / predict.bvar.
vars  Optional numeric or character vector. Used to subset the plot to certain variables by position or name (must be available). Defaults to NULL, i.e. all variables.
variables  Optional character vector. Names of all variables in the object. Used to subset and title. Taken from x$variables if available.
orientation  String indicating the orientation of the plots. Defaults to "v" (i.e. vertical); may be set to "h" (i.e. horizontal).
mar  Numeric vector. Margins for par.
...  Other graphical parameters for par.

Value

Returns x invisibly.

See Also

bvar; predict.bvar
Examples

data <- matrix(rnorm(400), ncol = 4)
x <- bvar(data, lags = 2, fcast = bv_fcast())

# Plot forecasts for all available variables
plot(predict(x))
# Alternatively
plot(x$fcast)

# Subset to variables in positions 1, 2 and 4 via position and name
plot(x$fcast, vars = c(1, 2, 4))
plot(x$fcast, variables = c("gdp", "flux", "cpi", "capacitor"),
     vars = c("gdp", "flux", "capacitor"))

# Use the method to plot and adjust orientation
plot(x$fcast, orientation = "h")

# Adjust confidence bands via predict
plot(predict(x, conf_bands = c(0.01, 0.05)))

plot.bvar_irf

Plotting method for Bayesian VAR impulse responses

Description

Plotting method for impulse responses obtained from bvar or irf.bvar. Impulse responses of all or a subset of the available variables can be plotted.

Usage

## S3 method for class 'bvar_irf'
plot(x, conf_bands, vars_response = NULL,
     vars_impulse = NULL, variables = NULL, mar = c(2, 2, 2, 0.5), ...)

bv_plot_irf(x, conf_bands = 0.16, variables = NULL,
            vars_impulse = NULL, vars_response = NULL, mar = c(2, 2, 2, 0.5),
            ...)

Arguments

x A bvar/bvar_irf object, obtained from bvar/irf.bvar.
vars_impulse, vars_response
Optional numeric or character vector. Used to subset the plot's impulses / responses to certain variables by position or name (must be available). Defaults to NULL, i.e. all variables.

variables
Optional character vector. Names of all variables in the object. Used to subset and title. Taken from x$variables if available.

mar
Numeric vector. Margins for par.

... Other graphical parameters for par.

Value
Returns x invisibly.

See Also
bvar; irf.bvar

Examples

data <- matrix(rnorm(400), ncol = 4)
x <- bvar(data, lags = 2, irf = bv_irf())

# Plot impulse responses for all available variables
plot(irf(x))
# Alternatively
plot(x$irf)

# Subset to impulse variables in positions 2 and 4 via position and name
plot(x$irf, vars_impulse = c(2, 4))
plot(x$irf,
variables = c("solved", "for", "many", "decades"),
vars_impulse = c("for", "decades")
)

# Adjust confidence bands via irf
plot(irf(x, conf_bands = c(0.01, 0.05)))

predict.bvar

Predict method for Bayesian VARs

Description
Retrieves / calculates forecasts for Bayesian VARs generated via bvar. If a forecast is already present and no settings are supplied it is simply retrieved, otherwise it will be calculated ex-post. To store a prediction you may want to assign the output of predict.bvar to object$fcast. May also be used to update confidence bands.
Usage

## S3 method for class 'bvar'
predict(object, ..., conf_bands, n_thin = 1L, newdata)

## S3 method for class 'bvar_fcast'
print(x, vars = NULL, ...)

## S3 method for class 'bvar_fcast'
summary(object, vars = NULL, digits = 2L, ...)

## S3 method for class 'bvar_fcast_summary'
print(x, digits = 2L, ...)

Arguments

object, x  A bvar object, obtained from bvar. Summary and print methods take in a
 bv_fcast object.
...
A bv_fcast object or parameters to be fed into bv_fcast. Contains settings for
the forecast.
conf_bands  Numeric vector of desired confidence bands to apply. E.g. for bands at 5%,
10%, 90% and 95% set this to c(0.05,0.1). Note that the median, i.e. 0.5 is
always included.
n_thin  Integer scalar. Every n_thin’th draw in object is used for forecasting, others are
dropped.
newdata  Optional numeric matrix or dataframe. Used to base the prediction on. Fitted
values are used by default.
vars  Optional numeric or character vector. Used to subset the summary to certain
variables by position or name (must be available). Defaults to NULL, i.e. all
variables.
digits  Integer scalar. Fed to round and applied to numeric outputs (i.e. the quantiles).

Value

Returns a list of class bvar_fcast including forecasts at desired confidence bands. See bvar. The
summary method returns a numeric array of forecast paths at the specified confidence bands.

See Also

bvar; plot.bvar_fcast; bv_fcast

Examples

data <- matrix(rnorm(200), ncol = 2)
x <- bvar(data, lags = 2)

# Access forecast and update confidence bands
predict(x, conf_bands = 0.01)

# Adjust, compute and store a longer forecast
x$fcast <- predict(x, horizon = 24L)

# Lower draws, use `bv_fcast()` to set options and add confidence bands
predict(x, bv_fcast(24L), n_thin = 10L, conf_bands = c(0.05, 0.16))

# Use new data to calculate a prediction
predict(x, newdata = matrix(rnorm(200), ncol = 2))

# Get a summary of the last saved forecast
summary(x)

# Limit the summary to variable #2
summary(x, vars = 2L)

---

**summary.bvar**

*Summary method for Bayesian VARs*

**Description**

Retrieves several outputs of interest, including the median coefficient matrix, the median variance-covariance matrix, and the Log-Likelihood. Separate summary methods exist for impulse responses and forecasts.

**Usage**

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

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

**Arguments**

- `object` A `bvar` object, obtained from `bvar`.
- `...` Not used.
- `x` A `bvar_summary` object.

**Value**

Returns a list of class `bvar_summary` with elements that can be accessed individually:

- `bvar` - `object`, the `bvar` object provided.
- `coef` - Coefficient values from `coef.bvar`.
- `vcov` - VCOV values from `vcov.bvar`.
- `logLik` - The Log-Likelihood from `logLik.bvar`.
See Also

bvar; coef.bvar; logLik.bvar

Examples

data <- matrix(rnorm(200), ncol = 2)
x <- bvar(data, lags = 2)
summary(x)
Index

*Topic **BVAR**
  as.mcmc.bvar, 3
  bv_fcast, 7
  bv_irf, 8
  bv_metropolis, 10
  bv_priors, 14
  bvar, 4
  irf.bvar, 21
  plot.bvar, 23
  plot.bvar_fcast, 25
  plot.bvar_irf, 26
  predict.bvar, 27

*Topic **FRED**
  fred_qd, 20

*Topic **MH**
  bv_metropolis, 10

*Topic **VAR**
  as.mcmc.bvar, 3
  bv_fcast, 7
  bv_irf, 8
  bv_metropolis, 10
  bv_priors, 14
  bvar, 4
  irf.bvar, 21
  plot.bvar, 23
  plot.bvar_fcast, 25
  plot.bvar_irf, 26
  predict.bvar, 27

*Topic **acceptance**
  bv_metropolis, 10

*Topic **coda**
  as.mcmc.bvar, 3

*Topic **convergence**
  as.mcmc.bvar, 3
  plot.bvar, 23

*Topic **dataset**
  fred_qd, 20

*Topic **density**
  plot.bvar, 23

*Topic **dummy**
  bv_priors, 14

*Topic **fevd**
  bv_irf, 8
  irf.bvar, 21
  plot.bvar_irf, 26

*Topic **forecast-error-variance-decomposition**
  irf.bvar, 21

*Topic **forecasts**
  bv_fcast, 7
  plot.bvar_fcast, 25
  predict.bvar, 27

*Topic **hierarchical**
  bvar, 4

*Topic **impulse-responses**
  irf.bvar, 21

*Topic **impulse**
  bv_irf, 8

*Topic **irf**
  bv_irf, 8
  irf.bvar, 21
  plot.bvar_irf, 26

*Topic **macroeconomics**
  bvar, 4
  fred_qd, 20

*Topic **mcmc**
  as.mcmc.bvar, 3
  bv_metropolis, 10

*Topic **metropolis-hastings**
  bv_metropolis, 10

*Topic **minnesota**
  bv_priors, 14

*Topic **plot**
  plot.bvar, 23
  plot.bvar_fcast, 25
  plot.bvar_irf, 26

*Topic **prediction**
  bv_fcast, 7
INDEX

plot.bvar_fcast, 25
predict.bvar, 27

*Topic priors
bv_priors, 14
bvar, 4

*Topic quantiles
predict.bvar, 27

*Topic responses
bv_irf, 8

*Topic settings
bv_fcast, 7
bv_irf, 8

*Topic trace
plot.bvar, 23

*Topic vector-autoregression
bvar, 4

as.mcmc, 3
as.mcmc.bvar, 3
as.mcmc.list, 3

bv_alpha, 11
bv_alpha (bv_minnesota), 11
bv_dummy, 6, 12, 14
bv_fcast, 4, 5, 7, 28
bv_irf, 4, 5, 8, 21, 22
bv_lambda, 11
bv_lambda (bv_minnesota), 11
bv_metropolis, 4, 10
bv_mh, 4, 5
bv_mh (bv_metropolis), 10
bv_minnesota, 7, 11
bv_mn, 14
bv_mn (bv_minnesota), 11
bv_plot (plot.bvar), 23
bv_plot_density (bv_plot_trace), 13
bv_plot_fcast (plot.bvar_fcast), 25
bv_plot_irf (plot.bvar_irf), 26
bv_plot_trace, 13
bv_priors, 4–7, 11, 12, 14
bv_psi (bv_minnesota), 11
bv_soc, 6
bv_soc (bv_dummy), 6
bv_sur, 6
bv_sur (bv_dummy), 6
bvar, 3, 4, 7–10, 12–30
BVAR-package, 2

coeff, 17

coeff.bvar, 5, 15, 29, 30
companion (companion.bvar), 16
companion.bvar, 16, 16
density, 17, 18
density.bvar, 17
fevd (irf.bvar), 21
fevd.bvar, 8, 9
fitted.bvar, 21
fred_qd, 20
independent_index (density.bvar), 17
irf (irf.bvar), 21
irf.bvar, 4, 5, 8, 9, 21, 26, 27
logLik.bvar, 23, 29, 30
match.call, 5
mcmc, 3
mcmc.list, 3

optim, 5

par, 18, 19, 23–25, 27
plot.bvar, 4, 5, 13, 23
plot.bvar_density (density.bvar), 17
plot.bvar_fcast, 8, 23, 24, 25, 28
plot.bvar_irf, 9, 22–24, 26
plot.bvar_resid (fitted.bvar), 19
predict.bvar, 4, 5, 7, 8, 25, 27
print.bv_dummy (bv_priors), 14
print.bv_fcast (bv_fcast), 7
print.bv_metropolis (bv_metropolis), 10
print.bv_minnesota (bv_priors), 14
print.bv_priors (bv_priors), 14
print.bv_psi (bv_priors), 14
print.bvar (bvar), 4
print.bvar_coefs (coeff.bvar), 15
print.bvar_density (density.bvar), 17
print.bvar_fcast (predict.bvar), 27
print.bvar_fcast_summary (predict.bvar), 27
print.bvar_fevd (irf.bvar), 21
print.bvar_fitted (fitted.bvar), 19
print.bvar_irf (irf.bvar), 21
print.bvar_resid (fitted.bvar), 19
print.bvar_summary (summary.bvar), 29
print.bvar_vcovs (coeff.bvar), 15
INDEX

residuals.bvar (fitted.bvar), 19
round, 16, 19, 22, 28

summary.bvar, 4, 29
summary.bvar_fcast (predict.bvar), 27
summary.bvar_irf (irf.bvar), 21

vcov.bvar, 5, 29
vcov.bvar (coef.bvar), 15