Package ‘SteadyStateBVAR’

July 28, 2026

Title Bayesian Vector Autoregressions with Steady-State Priors

Version 0.1.1

Description Provides estimation of Bayesian vector autoregression (BVAR) models with steady-state priors via 'Stan', along with functions for unconditional and conditional forecasting, as well as impulse response analysis. For details on the steady-state BVAR model see Villani (2009) <doi:10.1002/jae.1065>.

License GPL (>= 3)

URL https://github.com/markjwbecker/SteadyStateBVAR, https://markjwbecker.github.io/SteadyStateBVAR/

Depends R (>= 3.6.0)

Imports graphics, grDevices, methods, MTS (>= 1.2.1), Rcpp (>= 0.12.0), RcppParallel (>= 5.0.1), rstan (>= 2.32.7), rstantools (>= 2.6.0), stats, utils

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

LazyData true

RoxygenNote 8.0.0

Biarch true

LinkingTo BH (>= 1.66.0), Rcpp (>= 0.12.0), RcppEigen (>= 0.3.3.3.0), RcppParallel (>= 5.0.1), rstan (>= 2.18.1), StanHeaders (>= 2.18.0)

SystemRequirements GNU make

BugReports https://github.com/markjwbecker/SteadyStateBVAR/issues

NeedsCompilation yes

Author Mark Becker [aut, cre, cph]

Maintainer Mark Becker <mark.jw.becker@gmail.com>

Repository CRAN

Date/Publication 2026-07-28 09:30:16 UTC

Contents

`bvar` *Create a steady-state BVAR model object*

Description

Initialises a steady-state `bvar` object. This is the starting point for all models in `SteadyStateBVAR`. After creation, pass the object sequentially to `setup`, `priors`, and `fit` to build and estimate the model.

Usage

```
bvar(data)
```

Arguments

`data`	A numeric matrix or time series of data where each column is a variable and each row is a time period.

Details

The model takes the form

$$y_t = \Psi d_t + \Pi_1(y_{t-1} - \Psi d_{t-1}) + \cdots + \Pi_p(y_{t-p} - \Psi d_{t-p}) + u_t$$

where y_t is an k-dimensional vector of endogenous variables at time t, and d_t is a q-dimensional vector of deterministic (exogenous) variables at time t. Here Π_ℓ for $\ell = 1, \ldots, p$ is a $(k \times k)$ matrix of autoregressive parameters, and Ψ is a $(k \times q)$ matrix of steady-state parameters. Now

$$\mathrm{E}(y_t) = \mu_t = \Psi d_t$$

is the unconditional mean, or the **steady state** of the process. Note that the current version of this package only allows for d_t to contain either a constant, a constant and a dummy variable, or a constant and a time trend. One can stack the (transposed) Π_i matrices in the $(kp \times k)$ matrix β

$$\beta = \begin{bmatrix} \Pi_1' \\ \vdots \\ \Pi_p' \end{bmatrix}$$

Then the model can be rewritten as a nonlinear regression (Karlsson, 2013)

$$y_t' = d_t'\Psi' + [w_t' - q_t'(I_p \otimes \Psi')]\,\beta + u_t'$$

where where $w_t' = (y_{t-1}', \ldots, y_{t-p}')$ is a kp-dimensional vector of lagged endogenous variables and $q_t' = (d_{t-1}', \ldots, d_{t-p}')$ is a qp-dimensional vector of lagged deterministic (exogenous) variables, I_p is the $(p \times p)$ identity matrix and $\otimes$ denotes the Kronecker product. This is how the likelihood is written in the Stan code. The goal is to estimate β, Ψ, and Σ_u.

For the innovations to the model, in the case of the homoscedastic steady-state BVAR, they are $u_t \overset{\text{iid}}{\sim} \mathrm{N_k}(0, \Sigma_u)$. However, for models with stochastic volatility, there is instead a time-varying covariance matrix $u_t \sim \mathrm{N_k}(0, \Sigma_{u,t})$. The innovations then take the form

$$u_t = A^{-1}\Lambda_t^{0.5}\epsilon_t$$
$$\epsilon_t \overset{\text{iid}}{\sim} \mathrm{N}(0, \mathrm{I}_k)$$

where A is a lower triangular matrix with ones on the diagonal that describes the contemporaneous interaction of the endogenous variables, and

$$\Lambda_t = \mathrm{diag}(\lambda_{1,t}, \ldots, \lambda_{k,t})$$

contains the time-varying volatilities. For the `AR1` stochastic volatility specification, the log volatilities follow AR(1) processes

$$\ln \lambda_{i,t} = \gamma_{0,i} + \gamma_{1,i} \ln \lambda_{i,t-1} + \nu_{i,t},\ i = 1, \ldots, k$$

where the log volatility AR(1) processes are restricted to the stationary region, i.e. $|\gamma_{1,i}| < 1\ \forall i$. For the `RW` stochastic volatility specification, the log volatilities follow Random Walk processes

$$\gamma_{0,i} = 0,\ \gamma_{1,i} = 1\ \forall i$$

The innovations to the log volatilities follow in the `AR1` case

$$\nu_t = (\nu_{1,t}, \ldots, \nu_{k,t})' \overset{\text{iid}}{\sim} \mathrm{N}(0, \Phi)$$

where Φ *is not diagonal* and as such the innovations to the log volatilities are allowed to be correlated across variables. For the `RW` case, Φ *is diagonal* with variances ϕ_i for $i = 1, \ldots, k$.

Note that under both stochastic volatility specifications, the time-varying covariance matrix is

$$\Sigma_{u,t} = A^{-1}\Lambda_t(A^{-1})'$$

For details on the homoscedastic steady-state BVAR model, see Villani (2009). For details on the Random Walk stochastic volatility steady-state BVAR model, see Clark (2011). See Carriero, Clark, and Marcellino (2024) for the above-mentioned AR(1) stochastic volatility specification applied to a conventional BVAR.

Value

An object of class `bvar`.

References

Carriero, A., Clark, T. E., and Marcellino, M. (2024). Capturing macro-economic tail risks with Bayesian vector autoregressions. *Journal of Money, Credit and Banking*, 56(5), pp. 1099–1127.

Clark, T. E. (2011). Real-time density forecasts from Bayesian vector autoregressions with stochastic volatility. *Journal of Business & Economic Statistics*, 29(3), pp. 327–341.

Karlsson, S. (2013). Forecasting with Bayesian vector autoregression. In: Elliott, G. and Timmermann, A. (eds), *Handbook of Economic Forecasting*. Elsevier B.V., Vol. 2, Part B, pp. 791–897.

Villani, M. (2009). Steady-state priors for vector autoregressions. *Journal of Applied Econometrics*, 24(4), pp. 630–650.

Examples

```
yt <- matrix(rnorm(50), 25, 2)

bvar_obj <- bvar(data = yt)
```

`conditional_forecast` *Conditional forecasts from a fitted steady-state BVAR model*

Description

Computes and plots conditional forecasts from a fitted steady-state `bvar` object. Conditions are imposed on specific variables at specific horizons using Algorithm 3.3.1 from Dieppe, Legrand, and van Roye (2016). Both conditional and unconditional forecasts are plotted for comparison. Please note that for the moment, conditional forecasting is only enabled for the homoscedastic steady-state BVAR, i.e. when `SV=FALSE` in `priors()`.

Usage

```
conditional_forecast(
  bvar_obj,
  conditions,
  pi = 0.95,
  fcst_type = c("mean", "median"),
  growth_rate_idx = NULL,
  plot_idx = NULL
)
```

Arguments

- `bvar_obj` — A steady-state `bvar` object that has been passed through `fit`.
- `conditions` — A data frame with three columns: `var` (integer index of the conditioned variable), `horizon` (integer forecast horizon at which the condition is imposed), and `value` (numeric, the imposed condition). Note that `var` and `horizon` must be integers, as they are used for indexing.
- `pi` — Numeric. The prediction interval width. Default `0.95`, i.e. 95% prediction interval.
- `fcst_type` — Character. Whether to use `"mean"` or `"median"` as the point forecast. Default `"mean"`.
- `growth_rate_idx` — Integer vector. Indices of variables of which to convert forecasts to annual growth rates $\ln x_t - \ln x_{t-f}$, where f is the frequency of the data (4 for quarterly, 12 for monthly). Only suitable for variables specified as $\ln x_t - \ln x_{t-1}$, i.e. `diff(log(x))` or `100*diff(log(x))`. Computed by summing up to f log first differences. Default is `NULL`.
- `plot_idx` — Integer vector. Indices of variables to plot. If `NULL` (default), all variables are plotted. Forecasts are always computed and returned for all variables, regardless of `plot_idx`.

Details

See Section 5.3 of Dieppe, Legrand, and van Roye (2016) for more details. Please note the limitations of this method, see the detailed discussion in Section 5.4 of Dieppe, Legrand, and van Roye (2016).

Value

Invisibly returns a list with three matrices: `forecast`, `lower`, and `upper`, each of dimension `H x k`.

References

Dieppe, A., Legrand, R., and van Roye, B. (2016). The BEAR toolbox. Working Paper Series, No. 1934. European Central Bank.

Examples

```
#homoscedastic with Jeffreys prior
yt <- matrix(rnorm(50), 25, 2)

bvar_obj <- bvar(data = yt)

bvar_obj <- setup(bvar_obj, p=1, deterministic = "constant")

bvar_obj <- priors(bvar_obj,
                   lambda_1 = 0.2,
                   lambda_2 = 0.5,
                   lambda_3 = 1,
                   first_own_lag_prior_mean = rep(1,2),
```

```
                 theta_Psi = rep(0, 2),
                 Omega_Psi = diag(0.1, 2, 2),
                 Jeffreys = TRUE,
                 SV = FALSE,
                 SV_type = NULL,
                 SV_priors = NULL)

bvar_obj <- fit(bvar_obj,
                H = 8,
                d_pred = matrix(rep(1,8)),
                iter = 200,
                warmup = 50,
                chains = 1,
                cores = 1)

conditions <- data.frame(var = rep(2,8),
                         horizon = rep(1:8),
                         value   = rep(1,8))

(cond_fcst <- conditional_forecast(bvar_obj,
                                   conditions,
                                   pi=0.68,
                                   fcst_type = "mean"))
```

---

fit *Estimate the steady-state BVAR model using Stan*

Description

Estimates the steady-state BVAR model using the No-U-Turn sampler (a variant of Hamiltonian Monte Carlo) via Stan. Also generates draws from the joint predictive distribution. Uses the data, setup, and priors stored in the steady-state bvar object.

Usage

```
fit(x, H = 1, d_pred = NULL, ...)
```

Arguments

x	A steady-state bvar object that has been passed through setup and priors.
H	Positive Integer. Forecast horizon. Default is 1.
d_pred	Matrix of size $H \times q$. Future values of the deterministic variables d_t. Default is NULL, must be provided by the user.
...	Additional arguments passed directly to the rstan function sampling (e.g. iter, warmup, chains, cores, control, seed, init, thin, algorithm, pars, include, refresh, verbose, save_warmup, sample_file, diagnostic_file). If pars/include is used to exclude model parameters required by fit() for posterior summaries, an error will be raised.

Details

The function selects the appropriate precompiled Stan model based on settings from priors:

- `steady_state_bvar_homoscedastic_jeffreys_prior`
- `steady_state_bvar_homoscedastic_inverse_wishart_prior`
- `steady_state_bvar_RW_stochastic_volatility`
- `steady_state_bvar_AR1_stochastic_volatility`

The function estimates the following parameters (see bvar for details):

- `beta`: $kp \times k$ VAR coefficient matrix
- `Psi`: $k \times q$ steady-state parameter matrix
- `Sigma_u`: innovation covariance matrix ($k \times k$ for homoscedastic, $T \times k \times k$ for stochastic volatility)
- If Random Walk stochastic volatility:
  - `A`: $k \times k$ lower triangular matrix with ones on the diagonal that describes the contemporaneous interaction of the endogenous variables
  - `phi`: k-dimensional vector of log volatility innovation variances
- If AR1 stochastic volatility:
  - `A`: $k \times k$ lower triangular matrix with ones on the diagonal that describes the contemporaneous interaction of the endogenous variables
  - `gamma_0`: k-dimensional vector of log volatility intercepts
  - `gamma_1`: k-dimensional vector of log volatility slopes
  - `Phi`: $k \times k$ log volatility innovation covariance matrix

Value

A fitted steady-state `bvar` object with:

- `x$fit$stan`: An object of class `stanfit`
- `x$fit$posterior_means`: List of posterior mean estimates
- `x$fit$posterior_medians`: List of posterior median estimates

Examples

```
#homoscedastic with Jeffreys prior, d_t = constant

yt <- matrix(rnorm(20), 10, 2)

bvar_obj <- bvar(data = yt)

bvar_obj <- setup(bvar_obj, p = 1, deterministic = "constant")

bvar_obj <- priors(bvar_obj,
                   lambda_1 = 0.2,
                   lambda_2 = 0.5,
                   lambda_3 = 1,
```

```
                   first_own_lag_prior_mean = rep(1,2),
                   theta_Psi = rep(0, 2),
                   Omega_Psi = diag(0.1, 2, 2),
                   Jeffreys = TRUE)

H <- 8
d_pred <- matrix(rep(1,8))
colnames(d_pred) <- c("constant")
rownames(d_pred) <- paste("Horizon", 1:H)
print(d_pred)

bvar_obj <- fit(bvar_obj,
                H = H,
                d_pred = d_pred,
                iter = 200,
                warmup = 50,
                chains = 1,
                cores = 1)


#homoscedastic with inverse-Wishart prior, d_t = constant and dummy

yt <- matrix(rnorm(20), 10, 2)

bvar_obj <- bvar(data = yt)

dummy_variable <- c(rep(1,5), rep(0,5))

bvar_obj <- setup(bvar_obj, p = 1,
                  deterministic = "constant_and_dummy",
                  dummy = dummy_variable)

k <- bvar_obj$setup$k
q <- bvar_obj$setup$q

bvar_obj <- priors(bvar_obj,
                   lambda_1 = 0.2,
                   lambda_2 = 0.5,
                   lambda_3 = 1,
                   first_own_lag_prior_mean = rep(1,2),
                   theta_Psi = rep(0, k*q),
                   Omega_Psi = diag(0.1, k*q, k*q),
                   Jeffreys = FALSE) #inverse-Wishart

H <- 8
d_pred <- cbind(rep(1, H), rep(0, H))
colnames(d_pred) <- c("constant", "dummy")
rownames(d_pred) <- paste("Horizon", 1:H)
print(d_pred)

bvar_obj <- fit(bvar_obj,
                H = H,
                d_pred = d_pred,
```

```
                iter = 200,
                warmup = 50,
                chains = 1,
                cores = 1)


#RW stochastic volatility

yt <- matrix(rnorm(20), 10, 2)

bvar_obj <- bvar(data = yt)

bvar_obj <- setup(bvar_obj, p=1, deterministic = "constant")

k <- bvar_obj$setup$k
n_free_params_A <- bvar_obj$setup$n_free_params_A

SV_priors_RW <- list(
theta_A             =  rep(0, n_free_params_A),
Omega_A             =  diag(1000, n_free_params_A),
mu_log_lambda_1     =  rep(0, k),
sigma2_log_lambda_1  =  rep(1000, k),
alpha_phi           =  rep(5, k),
beta_phi            = (rep(5, k) - 1) * rep(0.1, k)
)

bvar_obj <- priors(bvar_obj,
                   lambda_1 = 0.2,
                   lambda_2 = 0.5,
                   lambda_3 = 1,
                   first_own_lag_prior_mean = rep(1,2),
                   theta_Psi = rep(0, 2),
                   Omega_Psi = diag(0.1, 2, 2),
                   SV = TRUE,
                   SV_type = "RW",
                   SV_priors = SV_priors_RW)

bvar_obj <- fit(bvar_obj,
                H = 8,
                d_pred = matrix(rep(1,8)),
                iter = 200,
                warmup = 50,
                chains = 1,
                cores = 1,
                control = list(max_treedepth = 12, adapt_delta = 0.85)
                )


#AR1 stochastic volatility

yt <- matrix(rnorm(20), 10, 2)

bvar_obj <- bvar(data = yt)
```

```
bvar_obj <- setup(bvar_obj, p=1, deterministic = "constant")

k <- bvar_obj$setup$k
n_free_params_A <- bvar_obj$setup$n_free_params_A

SV_priors_AR1 <- list(
theta_A              =  rep(0, n_free_params_A),
Omega_A              =  diag(1000, n_free_params_A),
theta_gamma_0        =  rep(0.1, k),
Omega_gamma_0        =  diag(1000, k),
theta_gamma_1        =  rep(0.9, k),
Omega_gamma_1        =  diag(10, k),
theta_log_lambda_1   =  rep(0.1, k)/(1-rep(0.9, k)),
Omega_log_lambda_1   =  diag(1000, k),
V_Phi                = (10 - k - 1) * diag(k),
m_Phi                =  10
)

bvar_obj <- priors(bvar_obj,
                   lambda_1 = 0.2,
                   lambda_2 = 0.5,
                   lambda_3 = 1,
                   first_own_lag_prior_mean = rep(1,2),
                   theta_Psi = rep(0, 2),
                   Omega_Psi = diag(0.1, 2, 2),
                   SV = TRUE,
                   SV_type = "AR1",
                   SV_priors = SV_priors_AR1)

bvar_obj <- fit(bvar_obj,
                H = 8,
                d_pred = matrix(rep(1,8)),
                iter = 200,
                warmup = 50,
                chains = 1,
                cores = 1,
                control = list(max_treedepth = 12, adapt_delta = 0.85)
                )
```

`forecast` *Forecast from a fitted steady-state BVAR model*

Description

Computes and plots forecasts from a fitted `bvar` object. Draws from the joint predictive distribution are used to construct point forecasts and prediction intervals. Optionally converts monthly or quarterly growth rate forecasts to annual growth rates for selected variables.

Usage

```
forecast(
  x,
  pi = 0.95,
  fcst_type = c("mean", "median"),
  growth_rate_idx = NULL,
  plot_idx = NULL,
  show_all = FALSE
)
```

Arguments

- `x` — A steady-state `bvar` object that has been passed through `fit`.
- `pi` — Numeric. The prediction interval width. Default `0.95`, i.e. 95% prediction interval.
- `fcst_type` — Character. Whether to use `"mean"` or `"median"` as the point forecast. Default `"mean"`.
- `growth_rate_idx` — Integer vector. Indices of variables of which to convert forecasts to annual growth rates $\ln x_t - \ln x_{t-f}$, where f is the frequency of the data (4 for quarterly, 12 for monthly). Only suitable for variables specified as $\ln x_t - \ln x_{t-1}$, i.e. `diff(log(x))` or `100*diff(log(x))`. Computed by summing up to f log first differences. Default is `NULL`.
- `plot_idx` — Integer vector. Indices of variables to plot. If `NULL` (default), all variables are plotted. Forecasts are always computed and returned for all variables, regardless of `plot_idx`.
- `show_all` — Logical. If `FALSE` (default), the last eight years of history are shown alongside the forecast. If `TRUE`, the full history is shown.

Value

Invisibly returns a list with three matrices: `forecast`, `lower`, and `upper`, each of dimension `H x k` where `H` is the forecast horizon and `k` is the number of variables.

Examples

```
#homoscedastic with Jeffreys prior
yt <- matrix(rnorm(50), 25, 2)

bvar_obj <- bvar(data = yt)

bvar_obj <- setup(bvar_obj, p=1, deterministic = "constant")

bvar_obj <- priors(bvar_obj,
                   lambda_1 = 0.2,
                   lambda_2 = 0.5,
                   lambda_3 = 1,
                   first_own_lag_prior_mean = rep(1,2),
```

```
                   theta_Psi = rep(0, 2),
                   Omega_Psi = diag(0.1, 2, 2),
                   Jeffreys = TRUE,
                   SV = FALSE,
                   SV_type = NULL,
                   SV_priors = NULL)

bvar_obj <- fit(bvar_obj,
                H = 8,
                d_pred = matrix(rep(1,8)),
                iter = 200,
                warmup = 50,
                chains = 1,
                cores = 1)

(fcst <- forecast(bvar_obj, pi = 0.90, show_all = TRUE))
```

`IRF` *Impulse Response Functions for a fitted steady-state BVAR model*

Description

Computes and plots impulse response functions (IRFs) from a fitted steady-state `bvar` object. Supports both orthogonalized (OIRF) and generalized (GIRF) impulse responses, with optional conversion to annual growth rates.

Usage

```
IRF(
  x,
  H = 16,
  response = NULL,
  shock = NULL,
  type = c("median", "mean"),
  method = c("OIRF", "GIRF"),
  ci = 0.95,
  t = NULL,
  growth_rate_idx = NULL
)
```

Arguments

`x`	A steady-state `bvar` object that has been passed through `fit`.
`H`	Integer. The forecast horizon for the IRF. Default `16`.
`response`	Integer. Index of the response variable to plot. If `NULL` (default), all responses are plotted.

shock
: Integer. Index of the shock variable to plot. If `NULL` (default), all shocks are plotted.

type
: Character. Whether to use `"median"` or `"mean"` as the point estimate. Default `"median"`.

method
: Character. The IRF method: `"OIRF"` for orthogonalized or `"GIRF"` for generalized impulse responses. Default `"OIRF"`.

ci
: Numeric. The credible interval width. Default `0.95`, i.e. 95%.

t
: Integer. Time index for the covariance matrix when using stochastic volatility models. If `NULL` (default), the last time `t` is used.

growth_rate_idx
: Integer vector. Indices of variables for which the impulse response is converted from a quarterly or monthly log first difference to an annual growth rate response, i.e. $\ln x_t - \ln x_{t-f}$, where f is the frequency of the data (4 for quarterly, 12 for monthly). Only suitable for variables specified as $\ln x_t - \ln x_{t-1}$, i.e. `diff(log(x))` or `100*diff(log(x))`. Computed by summing up to f periods of the impulse response, treating the response in periods prior to the shock as zero. Default is `NULL`.

Value

Invisibly returns a list with three arrays: the point estimate IRF, `lower`, and `upper` credible bounds, each of dimension `k x k x (H+1)`.

Examples

```
#homoscedastic with Jeffreys prior
yt <- matrix(rnorm(50), 25, 2)

bvar_obj <- bvar(data = yt)

bvar_obj <- setup(bvar_obj, p=1, deterministic = "constant")

bvar_obj <- priors(bvar_obj,
                   lambda_1 = 0.2,
                   lambda_2 = 0.5,
                   lambda_3 = 1,
                   first_own_lag_prior_mean = rep(1,2),
                   theta_Psi = rep(0, 2),
                   Omega_Psi = diag(0.1, 2, 2),
                   Jeffreys = TRUE,
                   SV = FALSE,
                   SV_type = NULL,
                   SV_priors = NULL)

bvar_obj <- fit(bvar_obj,
                H = 8,
                d_pred = matrix(rep(1,8)),
                iter = 200,
                warmup = 50,
                chains = 1,
```

```
                 cores = 1)

(IRF(bvar_obj))
```

KoopKorobilis2010 *Koop and Korobilis (2010) dataset*

Description

Quarterly US macroeconomic data (1953Q1-2006Q3) from Koop and Korobilis (2010)

Usage

```
KoopKorobilis2010
```

Format

`KoopKorobilis2010`**:**

A multivariate time series object with 215 rows and 3 columns:

delta pi inflation rate (annual percentage change in a chain-weighted GDP price index)

u unemployment rate (seasonally adjusted civilian unemployment rate, all civilian workers aged 16 years or older)

r interest rate (yield on the three-month Treasury bill rate)

Source

Koop, G. and Korobilis, D. (2010). Bayesian multivariate time series methods for empirical macroeconomics. *Foundations and Trends in Econometrics*, 3(4), pp. 267-358.

---

ppi *Prior Probability Interval for a Normal Distribution*

Description

Calculates the mean and variance of a normal prior probability interval. Given a lower and upper bound of a prior probability interval this function recovers the implied normal prior parameters. Useful for specifying informative priors on the steady-state parameters (elements of Ψ).

Usage

```
ppi(l, u, interval = 0.95, annualized_growthrate = FALSE, freq = 4)
```

Arguments

`l`	Numeric. The lower bound of the prior probability interval.
`u`	Numeric. The upper bound of the prior probability interval.
`interval`	Numeric. The prior probability mass within the interval. Default `0.95`, i.e. 95%.
`annualized_growthrate`	Logical. If `TRUE`, treats `l` and `u` as bounds on the annualized steady-state growth rate and calculates the implied mean and variance on the corresponding quarterly/monthly scale. Useful if you are working with a variable `diff(log(x))` (which may be scaled by 100). Default `FALSE`.
`freq`	Integer. The data frequency (e.g. 4 for quarterly). Only used when `annualized_growthrate = TRUE`. Default 4.

Details

Consider a CPI variable `CPI <- data$CPI`, observed at quarterly frequency. In the model, quarter-on-quarter inflation is used `x <- 100*diff(log(CPI))`. Suppose the prior belief is that annualized steady-state inflation lies between 1.7 and 2.3 with 95% probability (mean 2). On the quarterly scale used by `x`, this corresponds to a 95% interval of 0.425 to 0.575 (mean 0.5). Since it is typically more natural to elicit a prior on the annualized scale, the `annualized_growthrate` argument performs this conversion. See `vignette("Homoscedastic-steady-state-BVAR")` for usage in practice.

Value

A list with two elements: `mean` and `var`, giving the mean and variance of the implied normal distribution.

Examples

```
ppi(l = 1.7, u = 2.3, interval = 0.95)
ppi(l = 1.7, u = 2.3, interval = 0.95, annualized_growthrate = TRUE, freq = 4)
```

`priors` *Specify priors for the steady-state BVAR model*

Description

This function prepares the priors. The Minnesota prior is used for the autoregressive parameters, and is determined by the overall tightness, cross-equation tightness, and the lag decay rate. For the steady-state parameters, a normal prior is used. For the covariance matrix of the innovations, the user can choose between Jeffreys prior or an uninformative inverse-Wishart prior. Optionally enables stochastic volatility where the covariance matrix varies over time.

Usage

```
priors(
  x,
  lambda_1 = 0.2,
  lambda_2 = 0.5,
  lambda_3 = 1,
  first_own_lag_prior_mean = NULL,
  theta_Psi = NULL,
  Omega_Psi = NULL,
  Jeffreys = TRUE,
  SV = FALSE,
  SV_type = NULL,
  SV_priors = NULL
)
```

Arguments

`x`	A steady-state `bvar` object that has been passed through `setup`.
`lambda_1`	Numeric. Overall tightness of the Minnesota prior. Default `0.2`.
`lambda_2`	Numeric. Cross-equation tightness of the Minnesota prior. Default `0.5`.
`lambda_3`	Numeric. Lag decay rate of the Minnesota prior. Default `1`.
`first_own_lag_prior_mean`	Numeric vector of length `k`. Prior means for the first own lags of the variables. If `NULL` (default), a zero vector is used.
`theta_Psi`	Numeric vector. Prior mean vector for $\text{vec}(\Psi)$, i.e. the steady-state parameters. If `NULL` (default), the OLS estimates are used.
`Omega_Psi`	Numeric matrix. Prior covariance matrix for $\text{vec}(\Psi)$, i.e. the steady-state parameters. If `NULL` (default), a diagonal matrix with variances `1000` is used.
`Jeffreys`	Logical. If `TRUE` (default), uses Jeffreys prior for the innovation covariance matrix. If `FALSE`, uses an uninformative inverse-Wishart prior. Only considered if `SV=FALSE`.
`SV`	Logical. If `TRUE`, enables stochastic volatility specification. Default `FALSE`.
`SV_type`	Character. Type of stochastic volatility model. Must be either `"RW"` or `"AR1"`. Required if `SV = TRUE`.
`SV_priors`	List. User-supplied stochastic volatility priors. Required when `SV = TRUE`. The list must contain the following named elements depending on `SV_type`: • For `"RW"`: `theta_A`, `Omega_A`, `mu_log_lambda_1`, `sigma2_log_lambda_1`, `alpha_phi`, `beta_phi`. • For `"AR1"`: `theta_A`, `Omega_A`, `theta_gamma_0`, `Omega_gamma_0`, `theta_gamma_1`, `Omega_gamma_1`, `theta_log_lambda_1`, `Omega_log_lambda_1`, `V_Phi`, `m_Phi`.

Details

The goal is to estimate β, Ψ, and Σ_u, so priors are needed. Following Villani (2009), prior independence between β, Ψ and Σ_u is assumed. For β, i.e. the autoregressive parameter matrix, the Minnesota prior is used

$$\text{vec}(\beta) \sim \text{N}_{kpk}(\theta_\beta, \Omega_\beta)$$

The prior means (the elements of θ_β) are set to

$$\text{E}\left(\Pi_\ell^{(i,j)}\right) = \begin{cases} \kappa & \text{if } \ell = 1 \text{ and } i = j \\ 0 & \text{otherwise} \end{cases}$$

$$\kappa = \begin{cases} \kappa^{level} & \text{if variable } i \text{ is in level} \\ \kappa^{\Delta} & \text{if variable } i \text{ is in difference} \end{cases}$$

Here, the autoregressive coefficient $\Pi_\ell^{(i,j)}$ is element (i, j) of Π_ℓ for $\ell = 1, \ldots, p$. As such, the Minnesota prior sets all prior means for the elements in β to 0, except for the elements that relate to the first own lags of the variables, which are set to κ. If variable i is in level (e.g. nominal interest rate), then $\kappa = \kappa^{level}$, and typical choices for κ^{level} are 1 or 0.9. Evaluating the equations at their prior means, equation i becomes a random walk if $\kappa^{level} = 1$ and a persistent stationary AR(1) process if $\kappa^{level} = 0.9$. Since the steady state only exists if the process is stationary, 0.9 is recommended for the steady-state BVAR. If variable i is in difference (e.g. output growth), then $\kappa = \kappa^{\Delta}$, and the most common choice for κ^{Δ} is 0, i.e. equation i becomes (when evaluating it at its prior means) a random walk expressed in first differences. If a differenced variable still shows some degree of persistence (can be examined with an ACF plot), a suitable value for κ^{Δ} can be (for example) 0.6 instead of 0. Moving on to the prior variances, Ω_β is a diagonal matrix containing the prior variances for the elements in β. They are specified as

$$\text{Var}\left(\Pi_\ell^{(i,j)}\right) = \begin{cases} \left(\frac{\lambda_1}{\ell^{\lambda_3}}\right)^2 & \text{if } i = j \\ \left(\frac{\lambda_1 \lambda_2 \sigma_i}{\ell^{\lambda_3} \sigma_j}\right)^2 & \text{if } i \neq j \end{cases}$$

Here λ_1, λ_2, and λ_3 are scalar hyperparameters known as the overall tightness, the cross-equation tightness and the lag decay rate. Furthermore, σ_i^2 is the (i, i):th element of Σ_u, which is unknown and therefore replaced with an estimate. In this package, it is replaced by the least squares residual variance from a univariate autoregression for variable i with p lags (including the constant and dummy/trend variable if applicable). Moving on to Ψ, the steady-state parameter matrix, the prior is

$$\text{vec}(\Psi) \sim \text{N}_{kq}(\theta_\Psi, \Omega_\Psi)$$

This is the core of the steady-state BVAR model. In θ_Ψ, one specifies the prior beliefs about the location of the steady state, and in Ω_Ψ, which is assumed to be a diagonal matrix, one specifies the degree of certainty in those prior beliefs. The prior for Σ_u is either the usual non-informative Jeffreys prior

$$p(\Sigma_u) \propto |\Sigma_u|^{-(k+1)/2}$$

or a proper uninformative inverse-Wishart prior

$$\Sigma_u \sim \text{IW}(V, m)$$

where V is the scale matrix and m is the number of degrees of freedom. An uninformative prior is specified by setting $V = (m - k - 1)\hat{\Sigma}_u$ where $\hat{\Sigma}_u$ is the least squares estimate from the VAR(p) (including the constant and dummy/trend variable if applicable), and $m = k + 2$. For the stochastic volatility specifications, the innovation covariance matrix is now time-varying $\Sigma_{u,t}$. Therefore, stochastic volatility priors are needed, see bvar for more details. For the Random Walk (`"RW"`) stochastic volatility specification, the following priors are used

$$\begin{aligned} a &\sim \mathrm{N}(\theta_A, \Omega_A) \\ \ln \lambda_{i,1} &\sim \mathrm{N}(\mu_{\ln \lambda_{i,1}}, \sigma^2_{\ln \lambda_{i,1}}) \\ \phi_i &\sim \mathrm{IG}(\alpha_{\phi_i}, \beta_{\phi_i}) \end{aligned}$$

Here a is a $k(k-1)/2$-dimensional vector that collects the free parameters in A in row-major order, and $\ln \lambda_{i,1}$ are the time $t = 1$ values (initial conditions) of $\ln \lambda_{i,t}$ for $i = 1, \ldots, k$. Furthermore, ϕ_i for $i = 1, \ldots, k$ are the log volatility innovation variances. For the AR(1) (`"AR1"`) stochastic volatility specification, the following priors are used

$$\begin{aligned} a &\sim \mathrm{N}(\theta_A, \Omega_A) \\ \gamma_0 &\sim \mathrm{N}(\theta_{\gamma_0}, \Omega_{\gamma_0}) \\ \gamma_1 &\sim \mathrm{N}(\theta_{\gamma_1}, \Omega_{\gamma_1}) \\ \ln \lambda_1 &\sim \mathrm{N}(\theta_{\ln \lambda_1}, \Omega_{\ln \lambda_1}) \\ \Phi &\sim \mathrm{IW}(V_\Phi, m_\Phi) \end{aligned}$$

Here a is again the $k(k-1)/2$-dimensional vector that collects the free parameters in A in row-major order, and $\ln \lambda_1$ is a k-dimensional vector containing the time $t = 1$ values (initial conditions) of $\ln \lambda_t$. Furthermore, γ_0 is a k-dimensional vector of log volatility intercepts, γ_1 is a k-dimensional vector of log volatility slopes, and Φ is the $k \times k$ log volatility innovation covariance matrix.

For details on the homoscedastic steady-state BVAR model, see Villani (2009). For details on the Random Walk stochastic volatility steady-state BVAR model, see Clark (2011). See Carriero, Clark, and Marcellino (2024) for the AR(1) stochastic volatility specification applied to a conventional BVAR.

Value

The steady-state `bvar` object with an appended `priors` list containing:

`theta_beta`	Prior mean vector for vec(β) constructed with the Minnesota prior
`Omega_beta`	Prior covariance matrix for vec(β) constructed with the Minnesota prior
`theta_Psi`	Prior mean vector for vec(Ψ), i.e. the steady-state parameters
`Omega_Psi`	Prior covariance matrix for vec(Ψ), i.e. the steady-state parameters
`Jeffreys`	Indicator for Jeffreys prior usage
`Sigma_AR`	Residual variance estimates from univariate AR fits, which are used by the Minnesota prior
`m`	Inverse-Wishart prior degrees of freedom (if `Jeffreys = FALSE`)
`V`	Inverse-Wishart prior scale matrix (if `Jeffreys = FALSE`)
`SV`	Logical indicator for stochastic volatility specification

SV_type	Stochastic volatility specification type
SV_priors	User-supplied SV prior list (if SV = TRUE)

References

Carriero, A., Clark, T. E., and Marcellino, M. (2024). Capturing macro-economic tail risks with Bayesian vector autoregressions. *Journal of Money, Credit and Banking*, 56(5), pp. 1099–1127.

Clark, T. E. (2011). Real-time density forecasts from Bayesian vector autoregressions with stochastic volatility. *Journal of Business & Economic Statistics*, 29(3), pp. 327–341.

Villani, M. (2009). Steady-state priors for vector autoregressions. *Journal of Applied Econometrics*, 24(4), pp. 630–650.

Examples

```
#homoscedastic with Jeffreys prior
yt <- matrix(rnorm(50), 25, 2)

bvar_obj <- bvar(data = yt)

bvar_obj <- setup(bvar_obj, p=1)

bvar_obj <- priors(bvar_obj,
                   lambda_1 = 0.2,
                   lambda_2 = 0.5,
                   lambda_3 = 1,
                   first_own_lag_prior_mean = rep(1,2),
                   theta_Psi = rep(0, 2),
                   Omega_Psi = diag(0.1, 2, 2),
                   Jeffreys = TRUE,
                   SV = FALSE,
                   SV_type = NULL,
                   SV_priors = NULL)

#RW stochastic volatility
yt <- matrix(rnorm(50), 25, 2)

bvar_obj <- bvar(data = yt)

bvar_obj <- setup(bvar_obj, p=1)

k <- bvar_obj$setup$k
n_free_params_A <- bvar_obj$setup$n_free_params_A

SV_priors_RW <- list(
theta_A             =  rep(0, n_free_params_A),
Omega_A             =  diag(1000, n_free_params_A),
mu_log_lambda_1     =  rep(0, k),
sigma2_log_lambda_1 =  rep(1000, k),
alpha_phi           =  rep(5, k),
beta_phi            = (rep(5, k) - 1) * rep(0.1, k)
)
```

```
bvar_obj <- priors(bvar_obj,
                   lambda_1 = 0.2,
                   lambda_2 = 0.5,
                   lambda_3 = 1,
                   first_own_lag_prior_mean = rep(1,2),
                   theta_Psi = rep(0, 2),
                   Omega_Psi = diag(0.1, 2, 2),
                   SV = TRUE,
                   SV_type = "RW",
                   SV_priors = SV_priors_RW)

#AR1 stochastic volatility
yt <- matrix(rnorm(50), 25, 2)

bvar_obj <- bvar(data = yt)

bvar_obj <- setup(bvar_obj, p=1)

k <- bvar_obj$setup$k
n_free_params_A <- bvar_obj$setup$n_free_params_A

SV_priors_AR1 <- list(
theta_A              =  rep(0, n_free_params_A),
Omega_A              =  diag(1000, n_free_params_A),
theta_gamma_0        =  rep(0.1, k),
Omega_gamma_0        =  diag(1000, k),
theta_gamma_1        =  rep(0.9, k),
Omega_gamma_1        =  diag(10, k),
theta_log_lambda_1   =  rep(0.1, k)/(1-rep(0.9, k)),
Omega_log_lambda_1   =  diag(1000, k),
V_Phi                = (10 - k - 1) * diag(k),
m_Phi                =  10
)

bvar_obj <- priors(bvar_obj,
                   lambda_1 = 0.2,
                   lambda_2 = 0.5,
                   lambda_3 = 1,
                   first_own_lag_prior_mean = rep(1,2),
                   theta_Psi = rep(0, 2),
                   Omega_Psi = diag(0.1, 2, 2),
                   SV = TRUE,
                   SV_type = "AR1",
                   SV_priors = SV_priors_AR1)
```

---

restrict_beta *Restrict VAR coefficients to zero*

Description

Applies zero restrictions to the VAR coefficient matrix β by setting the corresponding prior variances in `Omega_beta` to a value near zero (prior means are zero by default in the Minnesota prior).

Usage

```
restrict_beta(x, restriction_matrix)
```

Arguments

`x` A steady-state `bvar` object that has been passed through `priors`.

`restriction_matrix` A numeric matrix of dimension `k*p x k` where entries of `0` indicate coefficients to be restricted to zero and entries of `1` indicate unrestricted coefficients.

Details

The steady-state BVAR model takes the form

$$y_t = \Psi d_t + \Pi_1(y_{t-1} - \Psi d_{t-1}) + \cdots + \Pi_p(y_{t-p} - \Psi d_{t-p}) + u_t$$

where y_t is an k-dimensional vector of endogenous variables at time t, and d_t is a q-dimensional vector of deterministic (exogenous) variables at time t. Here Π_ℓ for $\ell = 1, \ldots, p$ is a $(k \times k)$ matrix of autoregressive parameters, and Ψ is a $(k \times q)$ matrix of steady-state parameters. One can stack the (transposed) Π_i matrices in the $(kp \times k)$ matrix β

$$\beta = \begin{bmatrix} \Pi_1' \\ \vdots \\ \Pi_p' \end{bmatrix}$$

This function puts zero restrictions on the elements of β.

Value

The steady-state `bvar` object with the restriction matrix stored in `setup` and the prior covariance matrix `Omega_beta` updated accordingly.

Examples

```
yt <- matrix(rnorm(50), 25, 2)

bvar_obj <- bvar(data = yt)

bvar_obj <- setup(bvar_obj, p=2)

bvar_obj <- priors(bvar_obj,
                   theta_Psi = rep(0, 2),
                   Omega_Psi = diag(0.1, 2, 2))
```

```
p <- bvar_obj$setup$p
k <- bvar_obj$setup$k

restriction_matrix <- matrix(1,k*p,k, dimnames=list(NULL,c("y1","y2")))

#restrict beta so y2 does not granger-cause y1

restriction_matrix[2, 1] <- 0
restriction_matrix[4, 1] <- 0

print(restriction_matrix)

bvar_obj <- restrict_beta(bvar_obj, restriction_matrix)
```

---

setup *Set up the steady-state BVAR model*

Description

Prepares the components needed for prior specification and estimation. Also computes OLS estimates.

Usage

```
setup(
  x,
  p = 1,
  deterministic = c("constant", "constant_and_dummy", "constant_and_trend"),
  dummy = NULL
)
```

Arguments

x	A steady-state bvar object created by `bvar`.
p	Integer. The lag order of the VAR. Default `1`.
deterministic	Character. The deterministic component to include. One of `"constant"` (default), `"constant_and_dummy"`, or `"constant_and_trend"`.
dummy	Numeric vector of a dummy variable. Only used when `deterministic = "constant_and_dummy"`. Default `NULL`.

Value

The steady-state `bvar` object with a `setup` list containing the required components for prior specification and estimation, and also the OLS estimates.

Examples

```
yt <- matrix(rnorm(50), 25, 2)

bvar_obj <- bvar(data = yt)

bvar_obj <- setup(bvar_obj)
```

---

stochastic_volatility_plot

Plot stochastic volatility estimates and forecasts

Description

Produces time series plots of posterior stochastic volatility estimates over the estimation sample together with predictive paths over the forecast horizon for a fitted steady-state `bvar` object with stochastic volatility.

Usage

```
stochastic_volatility_plot(
  x,
  ci = 0.95,
  vol = "log_lambda",
  stat = "mean",
  plot_idx = NULL,
  xlim = NULL,
  ylim = NULL
)
```

Arguments

`x`	A steady-state `bvar` object that has been passed through `fit` with stochastic volatility enabled.
`ci`	Numeric. Interval level. Default is `0.95`.
`vol`	Character. Volatility representation: `"log_lambda"` (default) or `"sd"`.
`stat`	Character. Posterior summary statistic to use as point estimate: `"mean"` (default) or `"median"`.
`plot_idx`	Integer vector. Indices of variables to plot. If `NULL` (default), all variables are plotted.
`xlim`	Numeric vector of length 2. Optional x-axis limits.
`ylim`	Numeric vector of length 2. Optional y-axis limits.

Details

The function supports two volatility representations:

- `"log_lambda"`: log variances
- `"sd"`: implied standard deviations

For each selected variable, the function plots the posterior point estimate path and credible bands over the estimation sample, and the predictive point estimate path and prediction intervals over the forecast horizon.

Value

An invisible list with:

- `estimated`: List of posterior summaries for the estimation sample:
  - `point`: $T \times k$ matrix of posterior point estimates (mean or median depending on `stat`)
  - `lower`: $T \times k$ matrix of lower credible bounds
  - `upper`: $T \times k$ matrix of upper credible bounds
- `predicted`: List of posterior summaries for the forecast horizon:
  - `point`: $(T + H) \times k$ matrix of predictive point estimates (mean or median depending on `stat`)
  - `lower`: $(T + H) \times k$ matrix of lower bound of prediction interval
  - `upper`: $(T + H) \times k$ matrix of upper bound of prediction interval

Examples

```
yt <- matrix(rnorm(50), 25, 2)

bvar_obj <- bvar(data = yt)

bvar_obj <- setup(bvar_obj, p = 1, deterministic = "constant")

k <- bvar_obj$setup$k
n_free_params_A <- bvar_obj$setup$n_free_params_A

SV_priors <- list(
theta_A             =  rep(0, n_free_params_A),
Omega_A             =  diag(1000, n_free_params_A),
mu_log_lambda_1     =  rep(0, k),
sigma2_log_lambda_1 =  rep(1000, k),
alpha_phi           =  rep(5, k),
beta_phi            = (rep(5, k) - 1) * rep(0.1, k)
)

bvar_obj <- priors(bvar_obj,
                   theta_Psi = rep(0, 2),
                   Omega_Psi = diag(0.1, 2, 2),
                   SV = TRUE,
                   SV_type = "RW",
                   SV_priors = SV_priors)
```

```
bvar_obj <- fit(bvar_obj,
                H = 8,
                d_pred = matrix(rep(1, 8)),
                iter = 200,
                warmup = 50,
                chains = 1,
                cores = 1,
                verbose = FALSE)

stochastic_volatility_plot(bvar_obj, ci = 0.95)
```

`summary.bvar` *Summarise a fitted steady-state BVAR model*

Description

Computes and prints posterior summaries from a fitted steady-state bvar object. The printed output depends on whether the model is homoscedastic or includes stochastic volatility (RW or AR1 specification).

Usage

```
## S3 method for class 'bvar'
summary(object, pars = NULL, stat = "mean", t = NULL, ...)
```

Arguments

object	A steady-state bvar object that has been passed through fit.
pars	Character vector of parameter names to include. If NULL (default), all available parameters are displayed.
stat	Character. Posterior summary statistic to display: "mean" (default) or "median".
t	Integer. Time index for the innovation covariance matrix if stochastic volatility was estimated. If NULL, the latest available t is used.
...	Additional arguments (currently unused).

Details

The function summarises the following estimated parameters:

- beta: $kp \times k$ VAR coefficient matrix
- Psi: $k \times q$ steady-state parameter matrix
- Sigma_u: innovation covariance matrix ($k \times k$ for homoscedastic, $T \times k \times k$ for stochastic volatility)
- If Random Walk stochastic volatility:

  - `A`: $k \times k$ lower triangular matrix with ones on the diagonal that describes the contemporaneous interaction of the endogenous variables
  - `phi`: k-dimensional vector of log volatility innovation variances
- If AR1 stochastic volatility:
  - `A`: $k \times k$ lower triangular matrix with ones on the diagonal that describes the contemporaneous interaction of the endogenous variables
  - `gamma_0`: k-dimensional vector of log volatility intercepts
  - `gamma_1`: k-dimensional vector of log volatility slopes
  - `Phi`: $k \times k$ log volatility innovation covariance matrix

Value

Returns the input object invisibly.

Examples

```
yt <- matrix(rnorm(20), 10, 2)
bvar_obj <- bvar(data = yt)
bvar_obj <- setup(bvar_obj, p = 1, deterministic = "constant")
bvar_obj <- priors(bvar_obj,
                   theta_Psi = rep(0, 2),
                   Omega_Psi = diag(0.1, 2, 2))

bvar_obj <- fit(bvar_obj,
                H = 1,
                d_pred = matrix(1),
                iter = 100,
                warmup = 50,
                chains = 1,
                cores = 1)

summary(bvar_obj)
```

`Villani2009` *Villani (2009) dataset*

Description

Swedish macroeconomic data (1980Q1–2005Q4) from Section 4.1 in Villani (2009)

Usage

```
Villani2009
```

Format

`Villani2009`:

A multivariate time series object with 104 rows and 7 columns:

delta y_f foreign GDP growth

pi_f foreign CPI inflation

i_f foreign 3-month interest rate

delta y domestic GDP growth

pi domestic CPI inflation

i domestic 3-month interest rate

q level of the real exchange rate defined as $q = s + p_f - p$, where p_f and p are the foreign and domestic CPI levels in logs and s is the log of the trade-weighted nominal exchange rate

Source

Villani, M. (2009). Steady-state priors for vector autoregressions. *Journal of Applied Econometrics*, 24(4), pp. 630-650.

Index