Package ‘shrinkTVPVAR’

June 3, 2025

Type Package

Title Efficient Bayesian Inference for TVP-VAR-SV Models with Shrinkage

Version 1.0.1

Maintainer Peter Knaus <peter.knaus@wu.ac.at>

Description
Efficient Markov chain Monte Carlo (MCMC) algorithms for fully Bayesian estimation of time-varying parameter vector autoregressive models with stochastic volatility (TVP-VAR-SV) under shrinkage priors and dynamic shrinkage processes. Details on the TVP-VAR-SV model and the shrinkage priors can be found in Cadonna et al. (2020) <doi:10.3390/econometrics8020020>, details on the software can be found in Knaus et al. (2021) <doi:10.18637/jss.v100.i13>, while details on the dynamic shrinkage process can be found in Knaus and Frühwirth-Schnatter (2023) <doi:10.48550/arXiv.2312.10487>.

License GPL (>= 2)

Encoding UTF-8

Depends R (>= 3.3.0)

RoxygenNote 7.3.2

LinkingTo Rcpp, RcppProgress, RcppArmadillo, shrinkTVP (>= 3.1.0), stochvol

Imports Rcpp, shrinkTVP (>= 3.1.0), stochvol, coda, methods, grDevices, RColorBrewer, lattice, zoo, mvtnorm

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation yes

Author Peter Knaus [aut, cre] (ORCID: <https://orcid.org/0000-0001-6498-7084>)

Repository CRAN

Date/Publication 2025-06-03 13:40:07 UTC

Contents

`density_plotter` *Kernel density plots of posterior distribution for hyperparameters of time-varying coefficient matrix in a TVP-VAR model*

Description

`density_plotter` plots empirical kernel density estimates of the posterior distribution for hyperparameters of a time-varying parameter coefficient matrix in a TVP-VAR model. `beta_mean` and `theta_sr` will most likely be of interest here.

Usage

```
density_plotter(
  x,
  lag = 1,
  mgp = c(1.5, 0.5, 0),
  ylim,
  xlim,
  ylabs,
  mains,
  h_borders = c(0.075, 0.05),
  w_borders = c(0.05, 0.05),
  ...
)
```

Arguments

x	mcmc.var object
lag	single integer value, indicating the lag of the time-varying VAR to be plotted. The default value is 1.
mgp	vector of length 3, determining the margin line (in par) for the plot. The default value is c(1.5, 0.5, 0). See par for more information.
ylim	numeric vector of length 2, determining the y-axis limits of the plot. If missing, the limits are determined by the density function.
xlim	numeric vector of length 2, determining the x-axis limits of the plot. If missing, the limits are determined by the minimum and maximum values of the data.
ylabs	character vector of length m, determining the y-axis labels of the plot. If missing, the labels are taken from the column names of the data.
mains	character vector of length m, determining the main titles of the plot. If missing, the titles are taken from the column names of the data.
h_borders	numeric vector of length 2, determining the horizontal borders of the plot. The first value is the space between the plot and the left border, the second value is the space between the plot and the right border. Both are fractions of the total width of the plot. The default value is c(0.075, 0.05).
w_borders	numeric vector of length 2, determining the vertical borders of the plot. The first value is the space between the plot and the top border, the second value is the space between the plot and the bottom border. Both are fractions of the total height of the plot. The default value is c(0.05, 0.05).
...	further arguments to be passed to plot.

Value

Called for its side effects and returns invisibly.

Author(s)

Peter Knaus <peter.knaus@wu.ac.at>

See Also

Other plotting functions: TV_heatmap(), plot.mcmc.tvp.var(), plot.mcmc.var(), plot.shrinkTVPVAR(), plot.shrinkTVPVAR_fit(), plot.shrinkTVPVAR_forc(), state_plotter()

Examples

```
set.seed(123)
sim <- simTVPVAR(p = 2)
data <- sim$data

res <- shrinkTVPVAR(data, p = 2)
plot(res$theta_sr)

# Plot second lag
```

```
plot(res$theta_sr, lag = 2)
```

`fitted.shrinkTVPVAR` *Calculate fitted historical values for an estimated (D)TVP-VAR-SV model*

Description

Calculates the fitted values for an estimated (D)TVP-VAR-SV model.

Usage

```
## S3 method for class 'shrinkTVPVAR'
fitted(object, ...)
```

Arguments

`object`	A `shrinkTVPVAR` or `shrinkDTVPVAR` object
`...`	Currently ignored.

Value

An object of class `shrinkTVPVAR_fit`

Author(s)

Peter Knaus `<peter.knaus@wu.ac.at>`

See Also

Other prediction functions: `LPDS()`, `forecast_shrinkTVPVAR()`

Examples

```
set.seed(123)
sim <- simTVPVAR(p = 2)
data <- sim$data

res <- shrinkTVPVAR(data, p = 2)
fitted <- fitted(res)

# Visualize fitted values
plot(fitted)
```

forecast_shrinkTVPVAR *Draw from posterior predictive density of a fitted TVP-VAR-SV model*

Description

forecast_shrinkTVPVAR draws from the posterior predictive distribution of a fitted TVP-VAR-SV model resulting from a call to shrinkTVPVAR or shrinkDTVPVAR.

Usage

```
forecast_shrinkTVPVAR(mod, n.ahead = 1)
```

Arguments

mod	an object of class shrinkTVPVAR or shrinkDTVPVAR, containing the fitted model.
n.ahead	a single, positive integer indicating the forecasting horizon, i.e. how many time-points into the future the posterior predictive distribution should be sampled from. Can not be larger than the number of rows in newdata.

Value

The value returned is a list object of class shrinkTVPVAR_forc containing the samples from the posterior predictive density.

Author(s)

Peter Knaus <peter.knaus@wu.ac.at>

See Also

Other prediction functions: LPDS(), fitted.shrinkTVPVAR()

Examples

```
set.seed(123)
sim <- simTVPVAR(p = 2)
data <- sim$data

res <- shrinkTVPVAR(data, p = 2)
forc <- forecast_shrinkTVPVAR(res, n.ahead = 4)

# Visualize forecast
plot(forc)
```

gen_TVP_params *Generate TVP_params that can be used as input for a TVP-VAR-SV model*

Description

gen_TVP_params creates either a list or a list of lists of hyperparameters in the correct format to be used as input for a TVP-VAR-SV model estimated by shrinkTVPVAR.

Usage

```
gen_TVP_params(m = 2, for_each_eq = TRUE)
```

Arguments

m	The number of equations in the VAR model. Ignored if for_each_eq is set to FALSE. The default value is 2.
for_each_eq	Logical. If TRUE, a list of lists is returned, where each list contains the hyperparameters for one equation. If FALSE, a single list is returned.

Value

Either a list containing the hyperparameters for all equations or a list of lists containing the hyperparameters for each equation individually.

Author(s)

Peter Knaus <peter.knaus@wu.ac.at>

Examples

```
# For a 5 equation model
params <- gen_TVP_params(m = 5)

# For a model where all equations share the same hyperparameters
params <- gen_TVP_params(for_each_eq = FALSE)
```

LPDS *Calculate the log predictive density score (LPDS) for a fitted TVP-VAR-SV model*

Description

LPDS calcualtes the one-step ahead log predictive density score (LPDS) for a fitted TVP-VAR-SV model resulting from a call to shrinkTVPVAR or shrinkDTVPVAR. The LPDS is calculated by sampling from the posterior predictive distribution of the model and evaluating the log predictive density at the true value of the next time-point.

Usage

```
LPDS(mod, y_true)
```

Arguments

mod	an object of class shrinkTVPVAR or shrinkDTVPVAR, containing the fitted model.
y_true	a numeric vector of length m containing the true value of the next time-point, i.e. the value at time N + 1. The order of the vector has to match the order at time of fitting the model, i.e. the order of the columns in mod$data$Y.

Value

A single numeric value containing the log predictive density score (LPDS) for the fitted model evaluated at the true value of the next time-point.

Author(s)

Peter Knaus <peter.knaus@wu.ac.at>

See Also

Other prediction functions: fitted.shrinkTVPVAR(), forecast_shrinkTVPVAR()

Examples

```
set.seed(123)
sim <- simTVPVAR(p = 2)
data <- sim$data

train_dat <- data[1:(nrow(data) - 1), ]
test_dat <- data[nrow(data), ]

res <- shrinkTVPVAR(train_dat, p = 2)

LPDS(res, test_dat)

res_dyn <- shrinkDTVPVAR(train_dat, p = 2)
```

```
LPDS(res_dyn, test_dat)
```

---

plot.mcmc.tvp.var   *Plotting method for* mcmc.tvp.var *objects*

Description

Plotting method for mcmc.tvp.var objects

Usage

```
## S3 method for class 'mcmc.tvp.var'
plot(x, ...)
```

Arguments

x	mcmc.tvp.var object
...	further arguments to be passed to state_plotter function. For further details see state_plotter function.

Value

Called for its side effects and returns invisibly.

Author(s)

Peter Knaus <peter.knaus@wu.ac.at>

See Also

Other plotting functions: TV_heatmap(), density_plotter(), plot.mcmc.var(), plot.shrinkTVPVAR(), plot.shrinkTVPVAR_fit(), plot.shrinkTVPVAR_forc(), state_plotter()

Examples

```
set.seed(123)
sim <- simTVPVAR(p = 2)
data <- sim$data

res <- shrinkTVPVAR(data, p = 2)

plot(res$beta)
```

plot.mcmc.var *Plotting method for* mcmc.var *objects*

Description

Plotting method for mcmc.var objects

Usage

```
## S3 method for class 'mcmc.var'
plot(x, ...)
```

Arguments

x	mcmc.var object
...	further arguments to be passed to TV_heatmap function. For further details see TV_heatmap function.

Value

Called for its side effects and returns invisibly.

Author(s)

Peter Knaus <peter.knaus@wu.ac.at>

See Also

Other plotting functions: TV_heatmap(), density_plotter(), plot.mcmc.tvp.var(), plot.shrinkTVPVAR(), plot.shrinkTVPVAR_fit(), plot.shrinkTVPVAR_forc(), state_plotter()

Examples

```
set.seed(123)
sim <- simTVPVAR(p = 2)
data <- sim$data

res <- shrinkTVPVAR(data, p = 2)

plot(res$theta_sr)
```

plot.shrinkTVPVAR *Plotting method for* shrinkTVPVAR *objects*

Description

Plotting method for shrinkTVPVAR objects

Usage

```
## S3 method for class 'shrinkTVPVAR'
plot(x, ...)
```

Arguments

x	shrinkTVPVAR object
...	further arguments to be passed to state_plotter function. For further details see state_plotter function.

Value

Called for its side effects and returns invisibly.

Author(s)

Peter Knaus <peter.knaus@wu.ac.at>

See Also

Other plotting functions: TV_heatmap(), density_plotter(), plot.mcmc.tvp.var(), plot.mcmc.var(), plot.shrinkTVPVAR_fit(), plot.shrinkTVPVAR_forc(), state_plotter()

Examples

```
set.seed(123)
sim <- simTVPVAR(p = 2)
data <- sim$data

res <- shrinkTVPVAR(data, p = 2)

plot(res)
```

plot.shrinkTVPVAR_fit *Graphical summary of posterior distribution of fitted values for TVP-VAR model*

Description

plot.shrinkTVPVAR_fit generates plots visualizing the posterior distribution of fitted values of a model generated by a call to shrinkTVPVAR.

Usage

```
## S3 method for class 'shrinkTVPVAR_fit'
plot(x, nplot = 3, h_borders = c(0.05, 0.05), w_borders = c(0.02, 0.02), ...)
```

Arguments

x	a shrinkTVPVAR_fit object.
nplot	single integer value, determining the number of plots (i.e. number of equations to visualize at once) to be generated. The default value is 3.
h_borders	numeric vector of length 2, determining the horizontal borders of the plot. The first value is the space between the plot and the left border, the second value is the space between the plot and the right border. Both are fractions of the total width of the plot. The default value is c(0.05, 0.05).
w_borders	numeric vector of length 2, determining the vertical borders of the plot. The first value is the space between the plot and the top border, the second value is the space between the plot and the bottom border. Both are fractions of the total height of the plot. The default value is c(0.02, 0.02).
...	further arguments to be passed to plot.

Value

Called for its side effects and returns invisibly.

Author(s)

Peter Knaus <peter.knaus@wu.ac.at>

See Also

Other plotting functions: TV_heatmap(), density_plotter(), plot.mcmc.tvp.var(), plot.mcmc.var(), plot.shrinkTVPVAR(), plot.shrinkTVPVAR_forc(), state_plotter()

Examples

```
set.seed(123)
sim <- simTVPVAR(p = 2)
data <- sim$data

res <- shrinkTVPVAR(data, p = 2)
fit <- fitted(res)
plot(fit)
```

---

plot.shrinkTVPVAR_forc *Graphical summary of posterior predictive density for TVP-VAR-SV model*

Description

`plot.shrinkTVPVAR_forc` generates plots visualizing the posterior predictive density generated by `forecast_shrinkTVPVAR`.

Usage

```
## S3 method for class 'shrinkTVPVAR_forc'
plot(x, nplot = 3, h_borders = c(0.05, 0.05), w_borders = c(0.02, 0.02), ...)
```

Arguments

`x`	a `shrinkTVPVAR_forc` object.
`nplot`	single integer value, determining the number of plots (i.e. number of equations to visualize at once) to be generated. The default value is 3.
`h_borders`	numeric vector of length 2, determining the horizontal borders of the plot. The first value is the space between the plot and the left border, the second value is the space between the plot and the right border. Both are fractions of the total width of the plot. The default value is `c(0.05, 0.05)`.
`w_borders`	numeric vector of length 2, determining the vertical borders of the plot. The first value is the space between the plot and the top border, the second value is the space between the plot and the bottom border. Both are fractions of the total height of the plot. The default value is `c(0.02, 0.02)`.
`...`	further arguments to be passed to `plot`.

Value

Called for its side effects and returns invisibly.

Author(s)

Peter Knaus <peter.knaus@wu.ac.at>

See Also

Other plotting functions: `TV_heatmap()`, `density_plotter()`, `plot.mcmc.tvp.var()`, `plot.mcmc.var()`, `plot.shrinkTVPVAR()`, `plot.shrinkTVPVAR_fit()`, `state_plotter()`

Examples

```
set.seed(123)
sim <- simTVPVAR(p = 2)
data <- sim$data

res <- shrinkTVPVAR(data, p = 2)
forc <- forecast_shrinkTVPVAR(res, n.ahead = 2)

plot(forc)
```

---

print.shrinkDTVPVAR *Nicer printing of shrinkDTVPVAR objects*

Description

Nicer printing of shrinkDTVPVAR objects

Usage

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

Arguments

`x`	a `shrinkDTVPVAR` object.
`...`	Currently ignored.

Value

Called for its side effects and returns invisibly.

Author(s)

Peter Knaus <peter.knaus@wu.ac.at>

`print.shrinkTVPVAR` *Nicer printing of shrinkTVPVAR objects*

Description

Nicer printing of shrinkTVPVAR objects

Usage

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

Arguments

`x`	a `shrinkTVPVAR` object.
`...`	Currently ignored.

Value

Called for its side effects and returns invisibly.

Author(s)

Peter Knaus <`peter.knaus@wu.ac.at`>

`shrinkDTVPVAR` *Markov Chain Monte Carlo (MCMC) for TVP-VAR-SV models under dynamic shrinkage priors*

Description

`shrinkDTVPVAR` samples from the joint posterior distribution of the parameters of a TVP-VAR-SV model as described in Cadonna et al. (2020) and returns the MCMC draws. The prior on the VAR coefficients is a dynamic shrinkage prior, as described in Knaus and Frühwirth-Schnatter (2023). The model can be written as:

$$Y_t = c_t + \Phi_{1,t} Y_{t-1} + \Phi_{2,t} Y_{t-2} + \cdots + \Phi_{p,t} Y_{t-p} + \epsilon_t$$

where $\epsilon_t \sim \mathcal{N}_m(0, \Sigma_t)$.

Usage

```
shrinkDTVPVAR(
  y,
  p = 1,
  mod_type = "double",
  const = TRUE,
  niter = 5000,
  nburn = round(niter/2),
  nthin = 1,
  display_progress = TRUE,
  TVP_params_beta = list(),
  TVP_params_sigma = list()
)
```

Arguments

- `y` matrix or data frame containing the time series data. The rows correspond to the time points and the columns to the variables.
- `p` positive integer indicating the number of lags in the VAR model. The default value is 1.
- `mod_type` character string that reads either `"triple"`, `"double"` or `"ridge"`. Determines whether the triple gamma, double gamma or ridge prior are used for `theta_sr` and `beta_mean`. The default is "double".
- `const` logical value indicating whether a constant should be included in the model. The default value is `TRUE`.
- `niter` positive integer, indicating the number of MCMC iterations to perform, including the burn-in. Has to be larger than or equal to `nburn` + 2. The default value is 5000.
- `nburn` non-negative integer, indicating the number of iterations discarded as burn-in. Has to be smaller than or equal to `niter` - 2. The default value is `round(niter/2)`.
- `nthin` positive integer, indicating the degree of thinning to be performed. Every `nthin` draw is kept and returned. The default value is 1, implying that every draw is kept.
- `display_progress` logical value indicating whether the progress bar and other informative output should be displayed. The default value is `TRUE`.
- `TVP_params_beta` *optional* named list containing hyperparameter values for the TVP prior of the beta_mean matrix. Not all have to be supplied, with those missing being replaced by the default values. Any list elements that are misnamed will be ignored and a warning will be thrown. Can either be a list of depth 1, which results in all equations having the same hyperparameters, or a list of lists of length `m`, where each sub-list contains the hyperparameters for the respective equation. If sub-lists are provided, they can be unnamed, in which case the order of the elements is assumed to be the same as the order of the equations in the data.

Alternatively, they can be named eq1, eq2, ..., eqm. The meaning of the hyperparameters is the same as in shrinkDTVP, as this function is used to sample the TVP coefficients. A key difference lies in how the adaptive MH arguments for theta and beta_mean are provided. In this function, the adaptive MH arguments are provided as vectors of length 4, where the elements control the MH steps in following order: a_xi, a_tau, c_xi, c_tau. E.g. if adaptive = c(FALSE, TRUE, TRUE, FALSE), then the MH step for a_xi and c_tau are non-adaptive, while the MH step for a_tau and c_xi are adaptive. Most hyperparameters to be provided are the same as in shrinkTVPVAR, however, the following additional ones are required for the dynamic shrinkage prior:

- iid: logical. The default value is TRUE.
- a_psi: numeric vector of length m*p if const is FALSE and m*p + 1 if const is TRUE. The default value is a vector filled with 0.5.
- c_psi: numeric vector of length m*p if const is FALSE and m*p + 1 if const is TRUE. The default value is a vector filled with 0.5.
- a_rho: positive, real number. The default value is 2.
- b_rho:positive, real number between 0 and 1. The default value is 0.95.
- alpha_rho: numeric vector of length m*p if const is FALSE and m*p + 1 if const is TRUE. The default value is a vector filled with 0.5.
- beta_rho: numeric vector of length m*p if const is FALSE and m*p + 1 if const is TRUE. The default value is a vector filled with 3.
- tuning_par_rho: positive, real number. The default value is 1.
- adaptive_rho: logical. If TRUE, the MH step for rho is adaptive, otherwise it is not. The default value is TRUE.
- target_rate_rho: positive, real number. The default value is 0.44.
- batch_size_rho: positive integer. The default value is 50.
- max_adapt_rho: positive, real number. The default value is 0.01.

TVP_params_sigma

optional named list containing hyperparameter values for the TVP prior of the Sigma matrix. The structure is the same as for TVP_params_beta. The default values are the same as for TVP_params_beta.

Details

The elements of the VAR coefficients $\Phi_{i,t}$ are assumed to follow component-wise random walks.

For further details concerning the algorithms and the model please refer to the papers by Cadonna et al. (2020) and Knaus and Frühwirth-Schnatter (2023).

Value

A list of class "shrinkDTVPVAR" containing:

beta	an mcmc.tvp.var object with the VAR coefficient draws.
beta_mean	an mcmc.var object with the beta_mean draws.
theta_sr	an mcmc.var object with the theta_sr draws.
xi2	an mcmc.var object with the xi2 draws.

c_xi	an mcmc object with the c_xi draws.
kappa2	an mcmc.var object with the kappa2 draws.
kappa2_B	an mcmc object with the kappa2_B draws.
a_xi	an mcmc object with the a_xi draws.
tau2	an mcmc.var object with the tau2 draws.
c_tau	an mcmc object with the c_tau draws.
lambda2	an mcmc.var object with the lambda2 draws.
lambda2_B	an mcmc object with the lambda2_B draws.
a_tau	an mcmc object with the a_tau draws.
Sigma	an mcmc.tvp.var object with the covariance matrix draws.
psi	an mcmc.tvp.var object with the psi draws.
rho_p	an mcmc.var object with the rho draws.
pred_objs	a list with objects needed for prediction methods.
final_lambda	an mcmc.var with the values of lambda at time T of the dynamic shrinkage process. Used for predicting.
final_lambda_SIGMA	an array with the values of lambda of the variance-covariance matrix Sigma at time T of the dynamic shrinkage process. Used for predicting.
rho_p_SIGMA	an array with the rho_p values of the variance-covariance matrix Sigma. Used for predicting.
beta_consts	a list of mcmc.tvp objects with the intercept draws (if const is TRUE).
psi_consts	a list of mcmc.tvp objects with the psi draws (if const is TRUE).
data	a list with the original data used for estimation.

Author(s)

Peter Knaus <peter.knaus@wu.ac.at>

References

Cadonna, A., Frühwirth-Schnatter, S., & Knaus, P. (2020). Triple the Gamma—A Unifying Shrinkage Prior for Variance and Variable Selection in Sparse State Space and TVP Models. *Econometrics*, 8(2), 20.

Knaus, P., Bitto-Nemling, A., Cadonna, A., & Frühwirth-Schnatter, S. (2021). Shrinkage in the Time-Varying Parameter Model Framework Using the R Package shrinkTVP. *Journal of Statistical Software*, 100(13), 1–32.

Knaus, P., & Frühwirth-Schnatter, S. (2023). The Dynamic Triple Gamma Prior as a Shrinkage Process Prior for Time-Varying Parameter Models. *arXiv preprint* arXiv:2312.10487.

See Also

TV_heatmap, density_plotter, state_plotter

Examples

```
## Not run:
set.seed(123)
sim <- simTVPVAR(p = 2)
data <- sim$data

res <- shrinkDTVPVAR(data, p = 2)

# Visualize the results
plot(res)
plot(res$theta_sr)

# Change prior to triple gamma
res2 <- shrinkDTVPVAR(data, p = 2, mod_type = "triple")

# Modify the hyperparameter setup
hyperparam <- list(learn_a_xi = FALSE, learn_c_xi = FALSE,
                  learn_a_tau = FALSE, learn_c_tau = FALSE,
                  a_xi = 0.5, c_xi = 0.5, a_tau = 0.5, c_tau = 0.5)

res3 <- shrinkDTVPVAR(data, p = 2, mod_type = "triple",
                    TVP_params_beta = hyperparam,
                    TVP_params_sigma = hyperparam)

## End(Not run)
```

`shrinkTVPVAR` *Markov Chain Monte Carlo (MCMC) for TVP-VAR-SV models with shrinkage*

Description

`shrinkTVPVAR` samples from the joint posterior distribution of the parameters of a TVP-VAR-SV model with shrinkage as described in Cadonna et al. (2020) and returns the MCMC draws. The model can be written as:

$$Y_t = c_t + \Phi_{1,t} Y_{t-1} + \Phi_{2,t} Y_{t-2} + \cdots + \Phi_{p,t} Y_{t-p} + \epsilon_t$$

where $\epsilon_t \sim \mathcal{N}_m(0, \Sigma_t)$.

Usage

```
shrinkTVPVAR(
  y,
  p = 1,
  mod_type = "double",
  const = TRUE,
  niter = 5000,
```

```
  nburn = round(niter/2),
  nthin = 1,
  display_progress = TRUE,
  TVP_params_beta = list(),
  TVP_params_sigma = list()
)
```

Arguments

y
: matrix or data frame containing the time series data. The rows correspond to the time points and the columns to the variables.

p
: positive integer indicating the number of lags in the VAR model. The default value is 1.

mod_type
: character string that reads either "triple", "double" or "ridge". Determines whether the triple gamma, double gamma or ridge prior are used for theta_sr and beta_mean. The default is "double".

const
: logical value indicating whether a constant should be included in the model. The default value is TRUE.

niter
: positive integer, indicating the number of MCMC iterations to perform, including the burn-in. Has to be larger than or equal to nburn + 2. The default value is 10000.

nburn
: non-negative integer, indicating the number of iterations discarded as burn-in. Has to be smaller than or equal to niter - 2. The default value is round(niter / 2).

nthin
: positive integer, indicating the degree of thinning to be performed. Every nthin draw is kept and returned. The default value is 1, implying that every draw is kept.

display_progress
: logical value indicating whether the progress bar and other informative output should be displayed. The default value is TRUE.

TVP_params_beta
: *optional* named list containing hyperparameter values for the TVP prior of the beta_mean matrix. Not all have to be supplied, with those missing being replaced by the default values. Any list elements that are misnamed will be ignored and a warning will be thrown. Can either be a list of depth 1, which results in all equations having the same hyperparameters, or a list of lists of length m, where each sub-list contains the hyperparameters for the respective equation. If sub-lists are provided, they can be unnamed, in which case the order of the elements is assumed to be the same as the order of the equations in the data. Alternatively, they can be named eq1, eq2, ..., eqm. The meaning of the hyperparameters is the same as in shrinkTVP, as this function is used to sample the TVP coefficients. A key difference lies in how the adaptive MH arguments are provided. In this function, the adaptive MH arguments are provided as vectors of length 4, where the elements control the MH steps in following order: a_xi, a_tau, c_xi, c_tau. E.g. if adaptive = c(FALSE, TRUE, TRUE, FALSE), then the MH step for a_xi and c_tau are non-adaptive, while the MH step for a_tau and c_xi are adaptive. The following elements can be supplied:

- e1: positive, real number. The default value is 0.5.
- e2: positive, real number. The default value is 0.001.
- d1: positive, real number. The default value is 0.5.
- d2: positive, real number. The default value is 0.001.
- beta_a_xi: positive, real number. The default value is 10.
- beta_a_tau: positive, real number. The default value is 10.
- alpha_a_xi: positive, real number. The default value is 5.
- alpha_a_tau: positive, real number. The default value is 5.
- beta_c_xi: positive, real number. The default value is 2.
- beta_c_tau: positive, real number. The default value is 2.
- alpha_c_xi: positive, real number. The default value is 5.
- alpha_c_tau: positive, real number. The default value is 5.
- a_tuning_par_xi: positive, real number. The default value is 1.
- a_tuning_par_tau: positive, real number. The default value is 1.
- c_tuning_par_xi: positive, real number. The default value is 1.
- c_tuning_par_tau: positive, real number. The default value is 1.
- learn_a_xi: logical. The default value is TRUE.
- learn_a_tau: logical. The default value is TRUE.
- a_eq_c_xi: logical. The default value is FALSE.
- a_eq_c_tau: logical. The default value is FALSE.
- a_xi: positive, real number. The default value is 0.1.
- a_tau: positive, real number. The default value is 0.1.
- learn_c_xi: logical. The default value is TRUE.
- learn_c_tau: logical. The default value is TRUE.
- c_xi: positive, real number. The default value is 0.1.
- c_tau: positive, real number. The default value is 0.1.
- learn_kappa2_B: logical. The default value is TRUE.
- learn_lambda2_B: logical. The default value is TRUE.
- kappa2_B: positive, real number. The default value is 20.
- lambda2_B: positive, real number. The default value is 20.
- Bsigma_sv: positive, real number. The default value is 1.
- a0_sv: positive, real number. The default value is 5.
- b0_sv: positive, real number. The default value is 1.5.
- bmu: real number. The default value is 0.
- Bmu: positive, real number. The default value is 1.
- adaptive: logical vector of length 4. The default value is rep(TRUE, 4).
- target_rates: numeric vector of length 4. The default value is rep(0.44, 4).
- batch_sizes: numeric vector of length 4. The default value is rep(50, 4).
- max_adapts: numeric vector of length 4. The default value is rep(0.01, 4).

TVP_params_sigma
optional named list containing hyperparameter values for the TVP prior of the Sigma matrix. The structure is the same as for TVP_params_beta. The default values are the same as for TVP_params_beta.

Details

The elements of the VAR coefficients $\Phi_{i,t}$ are assumed to follow component-wise random walk.

For further details concerning the algorithms and the model please refer to the paper by Cadonna et al. (2020).

Value

The value returned is a shrinkTVPVAR object containing:

beta: an object of class "mcmc.tvp.var" containing the MCMC draws of the VAR coefficients.

beta_mean: an object of class "mcmc.var" containing the MCMC draws of beta_mean for the VAR coefficients.

theta_sr: an object of class "mcmc.var" containing the MCMC draws of theta_sr for the VAR coefficients.

xi2: an object of class "mcmc.var" containing the MCMC draws of xi2 for the VAR coefficients.

c_xi: an object of class "mcmc" containing the MCMC draws of c_xi for the VAR coefficients.

kappa2: an object of class "mcmc.var" containing the MCMC draws of kappa2 for the VAR coefficients.

kappa2_B: an object of class "mcmc" containing the MCMC draws of kappa2_B for the VAR coefficients.

a_xi: an object of class "mcmc" containing the MCMC draws of a_xi for the VAR coefficients.

tau2: an object of class "mcmc.var" containing the MCMC draws of tau2 for the VAR coefficients.

c_tau: an object of class "mcmc" containing the MCMC draws of c_tau for the VAR coefficients.

lambda2: an object of class "mcmc.var" containing the MCMC draws of lambda2 for the VAR coefficients.

lambda2_B: an object of class "mcmc" containing the MCMC draws of lambda2_B for the VAR coefficients.

a_tau: an object of class "mcmc" containing the MCMC draws of a_tau for the VAR coefficients.

Sigma: an object of class "mcmc.tvp.var" containing the MCMC draws of the covariance matrices.

pred_objs: a list containing objects required for prediction methods to work.

beta_consts: a list of mcmc.tvp objects containing the MCMC draws of the intercepts.

data a list containing the input data, as well as the synthetic "covariates" used for estimation.

Note that only the values pertaining to the VAR coefficients are returned. The values for the variance-covariance matrix are not returned.

To display the output, use `plot` on various elements of the list, as well as the `TV_heatmap`, `density_plotter` and `state_plotter` function. Many functions that can be applied to `coda::mcmc` objects (e.g. `coda::acfplot`) can be applied to all output elements that are `coda` compatible.

Author(s)

Peter Knaus <peter.knaus@wu.ac.at>

References

Cadonna, A., Frühwirth-Schnatter, S., & Knaus, P. (2020). "Triple the Gamma—A Unifying Shrinkage Prior for Variance and Variable Selection in Sparse State Space and TVP Models." *Econometrics*, 8(2), 20. <doi:10.3390/econometrics8020020>

Knaus, P., Bitto-Nemling, A., Cadonna, A., & Frühwirth-Schnatter, S. (2021) "Shrinkage in the Time-Varying Parameter Model Framework Using the R Package `shrinkTVP`." *Journal of Statistical Software* 100(13), 1–32. <doi:10.18637/jss.v100.i13>

See Also

`TV_heatmap` `density_plotter` `state_plotter`

Examples

```
set.seed(123)
sim <- simTVPVAR(p = 2)
data <- sim$data

res <- shrinkTVPVAR(data, p = 2)

# Visualize the results
plot(res)

plot(res$theta_sr)

# Change prior to triple gamma
res2 <- shrinkTVPVAR(data, p = 2, mod_type = "triple")

# Modify the hyperparameter setup
# Estimate under hierarchical horseshoe prior
hyperparam <- list(learn_a_xi = FALSE, learn_c_xi = FALSE,
                   learn_a_tau = FALSE, learn_c_tau = FALSE,
                   a_xi = 0.5, c_xi = 0.5, a_tau = 0.5, c_tau = 0.5)

res3 <- shrinkTVPVAR(data, p = 2, mod_type = "triple",
                     TVP_params_beta = hyperparam,
                     TVP_params_sigma = hyperparam)
```

```
# Can also specify different hyperparameters for each equation
# gen_TVP_params() is a helper function and returns a
# list of lists (if for_each_eq = TRUE) where each sub-list
# contains the hyperparameters for the respective equation
hyperparam2 <- gen_TVP_params(m = 3, for_each_eq = TRUE)

hyperparam2[[1]]$learn_a_xi <- FALSE
hyperparam2[[1]]$a_xi <- 0.5

res4 <- shrinkTVPVAR(data, p = 2, mod_type = "triple",
                     TVP_params_beta = hyperparam2)

# Now, a_xi is only fixed in first equation
plot(res4$a_xi)
```

---

simTVPVAR *Generate synthetic data from a TVP-VAR-SV model*

Description

simTVPVAR generates synthetic data from a TVP-VAR-SV model. The data is always generated as to be stationary. This is done via a trial and error approach, where the VAR coefficients are drawn from the data generating process until the VAR process is stationary. As such, very large models might take a long time to generate.

Usage

```
simTVPVAR(
  N = 200,
  p = 2,
  m = 3,
  prob_0_beta = 0.8,
  prob_0_theta = 0.8,
  simsig2_theta_sr = 0.2,
  simsig2_beta_mean = 0.2,
  intercept = TRUE,
  display_progress = TRUE
)
```

Arguments

N	integer > 2. Indicates the length of the time series to be generated. The default value is 200.
p	integer > 0. Indicates the number of lags in the VAR model. The default value is 2.

`m` integer > 1. Indicates the number of equations in the VAR model. The default value is 3.

`prob_0_beta` numeric. Indicates the probability of a zero element in the beta_mean matrix. Can be a single value or a vector of length p. The default value is 0.8.

`prob_0_theta` numeric. Indicates the probability of a zero element in the theta matrix. Can be a single value or a vector of length p. The default value is 0.8.

`simsig2_theta_sr` numeric. Indicates the standard deviation of the normal distribution from which the elements of the theta matrix are drawn. The default value is 0.2.

`simsig2_beta_mean` numeric. Indicates the standard deviation of the normal distribution from which the elements of the beta_mean matrix are drawn. The default value is 0.2.

`intercept` logical. Indicates whether an intercept should be included in the model. The default value is TRUE.

`display_progress` logical. Indicates whether a progress bar should be displayed. The default value is TRUE.

Value

The value returned is a list object containing:

- `data`: data frame that holds the simulated data.
- `true_vals`: list object containing:
  - `Phi`: array containing the true VAR coefficients.
  - `Sigma`: array containing the true covariance matrices.
  - `theta_sr`: array containing the true standard deviations of the theta matrix.
  - `beta_mean`: array containing the true means of the beta matrix.

Author(s)

Peter Knaus <peter.knaus@wu.ac.at>

Examples

```
# Generate a time series of length 300
res <- simTVPVAR(N = 300, m = 3, p = 3)

# Estimate a model
model <- shrinkTVPVAR(y = res$data, p = 3)
```

`state_plotter` *Graphical summary of posterior distribution for a time-varying coefficient matrix in a TVP-VAR model*

Description

`plot.mcmc.tvp` plots empirical posterior quantiles for a time-varying parameter coefficient matrix in a TVP-VAR model.

Usage

```
state_plotter(
  x,
  lag = 1,
  mgp = c(1.5, 0.5, 0),
  ylim,
  ylabs,
  mains,
  h_borders = c(0.075, 0.05),
  w_borders = c(0.05, 0.05),
  ...
)
```

Arguments

`x`	`mcmc.tvp.var` object
`lag`	single integer value, indicating the lag of the time-varying VAR to be plotted. The default value is 1.
`mgp`	vector of length 3, determining the margin line (in `par`) for the plot. The default value is `c(1.5, 0.5, 0)`. See `par` for more information.
`ylim`	numeric vector of length 2, determining the y-axis limits of the plot. If missing, the limits are determined by the lowest and largest quantiles of the data.
`ylabs`	character vector of length m, determining the y-axis labels of the plot. If missing, the labels are taken from the column names of the data.
`mains`	character vector of length m, determining the main titles of the plot. If missing, the titles are taken from the column names of the data.
`h_borders`	numeric vector of length 2, determining the horizontal borders of the plot. The first value is the space between the plot and the left border, the second value is the space between the plot and the right border. Both are fractions of the total width of the plot. The default value is `c(0.075, 0.05)`.
`w_borders`	numeric vector of length 2, determining the vertical borders of the plot. The first value is the space between the plot and the top border, the second value is the space between the plot and the bottom border. Both are fractions of the total height of the plot. The default value is `c(0.05, 0.05)`.
`...`	further arguments to be passed to `plot.mcmc.tvp` (see shrinkTVP package).

Value

Called for its side effects and returns invisibly.

Author(s)

Peter Knaus <peter.knaus@wu.ac.at>

See Also

Other plotting functions: TV_heatmap(), density_plotter(), plot.mcmc.tvp.var(), plot.mcmc.var(), plot.shrinkTVPVAR(), plot.shrinkTVPVAR_fit(), plot.shrinkTVPVAR_forc()

Examples

```
set.seed(123)
sim <- simTVPVAR(p = 2)
data <- sim$data

res <- shrinkTVPVAR(data, p = 2)
plot(res$beta)

# Plot second lag
plot(res$beta, lag = 2)
```

TV_heatmap — *Heatmap of hyperparameters of time-varying coefficient matrix in a TVP-VAR model*

Description

TV_heatmap plots a heatmap of posterior distribution for hyperparameters of a time-varying parameter coefficient matrix in a TVP-VAR model. This is achieved by plotting the median of the posterior of the absolute value for theta_sr and the median of the posterior for all others. The plot itself is generated by lattice::levelplot. beta_mean and theta_sr will most likely be of interest here.

Usage

```
TV_heatmap(x, cuts = 15, cols, max_val, flipcols, ...)
```

Arguments

x	mcmc.var object
cuts	single integer value, determining the number of cuts for the color palette. The default value is 15.

`cols`	character string, determining the color palette to be used. The default value is "Purples" for `theta_sr` and "RdBu" for all others. See `RColorBrewer::brewer.pal.info` for more information.
`max_val`	numeric value, determining the maximum value for the color palette. If missing, the maximum value is determined by the largest absolute value of the data.
`flipcols`	logical value, determining whether the color palette should be flipped. The default value is `FALSE` for `theta_sr` and `TRUE` for all others.
`...`	further arguments to be passed to `lattice::levelplot`.

Value

Called for its side effects and returns invisibly.

Author(s)

Peter Knaus <peter.knaus@wu.ac.at>

See Also

Other plotting functions: `density_plotter()`, `plot.mcmc.tvp.var()`, `plot.mcmc.var()`, `plot.shrinkTVPVAR()`, `plot.shrinkTVPVAR_fit()`, `plot.shrinkTVPVAR_forc()`, `state_plotter()`

Examples

```
set.seed(123)
sim <- simTVPVAR(p = 2)
data <- sim$data

res <- shrinkTVPVAR(data, p = 2)
TV_heatmap(res$theta_sr)
```

Index