Package 'sparsevar'

February 4, 2026

Version 1.0.0

Date 2026-01-27

Title Sparse VAR (Vector Autoregression) / VECM (Vector Error Correction Model) Estimation

Author Simone Vazzoler [aut, cre]

Maintainer Simone Vazzoler <svazzole@gmail.com>

Imports Matrix, ncvreg, parallel, doParallel, glmnet, ggplot2, reshape2, grid, mvtnorm, corpcor, checkmate, rlang,

Suggests knitr, rmarkdown, testthat,

Depends R (>= 4.5.0)

Description A wrapper for sparse VAR (Vector Autoregression) and VECM (Vector Error Correction Model) time series models estimation using penalties like ENET (Elastic Net), SCAD (Smoothly Clipped Absolute Deviation) and MCP (Minimax Concave Penalty). Based on the work of Basu and Michailidis (2015) <doi:10.1214/15-AOS1315>.

License GPL-2

URL https://github.com/svazzole/sparsevar

BugReports https://github.com/svazzole/sparsevar/issues

VignetteBuilder knitr

RoxygenNote 7.3.3

Encoding UTF-8

NeedsCompilation no

Repository CRAN

Date/Publication 2026-02-04 17:50:02 UTC

Contents

accuracy *Accuracy metric*

Description

Compute the accuracy of a fit

Usage

```
accuracy(reference_m, a)
```

Arguments

reference_m	the matrix to use as reference
a	the matrix obtained from a fit

Value

A numeric value between 0 and 1 representing the accuracy of the fit, computed as the proportion of correctly identified zero/non-zero entries.

bootstrapped_var *Bootstrap VAR*

Description

Build the bootstrapped series from the original var

Usage

```
bootstrapped_var(v)
```

Arguments

v	the VAR object as from fitVAR or simulateVAR

Value

A matrix containing the bootstrapped time series with the same dimensions as the original series.

check_impulse_zero *Check Impulse Zero*

Description

A function to find which entries of the impulse response function are zero.

Usage

```
check_impulse_zero(irf)
```

Arguments

irf	irf output from impulseResponse function

Value

a matrix containing the indices of the impulse response function that are 0.

check_is_var *Check is var*

Description

Check if the input is a var object

Usage

```
check_is_var(v)
```

Arguments

v	the object to test

Value

A logical value: TRUE if the input is a var or varx object, FALSE otherwise.

companion_var *Companion VAR*

Description

Build the VAR(1) representation of a VAR(p) process

Usage

```
companion_var(v)
```

Arguments

v	the VAR object as from `fitVAR` or `simulateVAR`

Value

A sparse matrix (of class `dgCMatrix`) representing the companion form of the VAR(p) process.

compute_forecasts *Computes forecasts for VARs*

Description

This function computes forecasts for a given VAR.

Usage

```
compute_forecasts(v, num_steps)
```

Arguments

v	a VAR object as from fitVAR.
num_steps	the number of forecasts to produce.

Value

A matrix of dimension (number of variables) x (num_steps) containing the forecasted values for each variable at each forecast horizon.

`create_sparse_matrix` *Create Sparse Matrix*

Description

Creates a sparse square matrix with a given sparsity and distribution.

Usage

```
create_sparse_matrix(
  n,
  sparsity,
  method = "normal",
  stationary = FALSE,
  p = 1,
  ...
)
```

Arguments

`n`	the dimension of the square matrix
`sparsity`	the density of non zero elements
`method`	the method used to generate the entries of the matrix. Possible values are `"normal"` (default) or `"bimodal"`.
`stationary`	should the spectral radius of the matrix be smaller than 1? Possible values are `TRUE` or `FALSE`. Default is `FALSE`.
`p`	normalization constant (used for VAR of order greater than 1, default = 1)
`...`	other options for the matrix (you can specify the mean `mu_mat` and the standard deviation `sd_mat`).

Value

An nxn sparse matrix.

Examples

```
M <- create_sparse_matrix(
  n = 30, sparsity = 0.05, method = "normal",
  stationary = TRUE
)
```

decompose_pi *Decompose Pi VECM matrix*

Description

A function to estimate a (possibly big) multivariate VECM time series using penalized least squares methods, such as ENET, SCAD or MC+.

Usage

```
decompose_pi(vecm, rk, ...)
```

Arguments

vecm	the VECM object
rk	rank
...	options for the function (TODO: specify)

Value

alpha

beta

error_bands_irf *Error bands for IRF*

Description

A function to estimate the confidence intervals for irf and oirf.

Usage

```
error_bands_irf(v, irf, alpha, m, resampling, ...)
```

Arguments

v	a var object as from fitVAR or simulateVAR
irf	irf output from impulseResponse function
alpha	level of confidence (default alpha = 0.01)
m	number of bootstrapped series (default m = 100)
resampling	type of resampling: "bootstrap" or "jackknife"
...	some options for the estimation: verbose = TRUE or FALSE, mode = "fast" or "slow", threshold = TRUE or FALSE.

Value

a matrix containing the indices of the impulse response function that are 0.

`fit_var` *Multivariate VAR estimation*

Description

A function to estimate a (possibly high-dimensional) multivariate VAR time series using penalized least squares methods, such as ENET, SCAD or MC+.

Usage

```
fit_var(data, p = 1, penalty = "ENET", method = "cv", ...)
```

Arguments

`data`	the data from the time series: variables in columns and observations in rows
`p`	order of the VAR model
`penalty`	the penalty function to use. Possible values are `"ENET"`, `"SCAD"` or `"MCP"`
`method`	possible values are `"cv"` or `"timeSlice"`
`...`	the options for the estimation. Global options are: `threshold`: if `TRUE` all the entries smaller than the oracle threshold are set to zero; `scale`: scale the data (default = FALSE)? `nfolds`: the number of folds used for cross validation (default = 10); `parallel`: if `TRUE` use multicore backend (default = FALSE); `ncores`: if `parallel` is `TRUE`, specify the number of cores to use for parallel evaluation. Options for ENET estimation: `alpha`: the value of alpha to use in elastic net (0 is Ridge regression, 1 is LASSO (default)); `type.measure`: the measure to use for error evaluation (`"mse"` or `"mae"`); `nlambda`: the number of lambdas to use in the cross validation (default = 100); `leaveOut`: in the time slice validation leave out the last `leaveOutLast` observations (default = 15); `horizon`: the horizon to use for estimating mse/mae (default = 1).

Value

`A` the list (of length `p`) of the estimated matrices of the process

`fit` the results of the penalized LS estimation

`mse` the mean square error of the cross validation

`time` elapsed time for the estimation

`residuals` the time series of the residuals

fit_varx *Multivariate VARX estimation*

Description

A function to estimate a (possibly high-dimensional) multivariate VARX time series using penalized least squares methods, such as ENET, SCAD or MC+.

Usage

```
fit_varx(data, p = 1, xt_matrix, m = 1,
penalty = "ENET", method = "cv", ...)
```

Arguments

data	the data from the time series: variables in columns and observations in rows
p	order of the VAR model
xt_matrix	the exogenous variables
m	order of the exogenous variables
penalty	the penalty function to use. Possible values are "ENET", "SCAD" or "MCP"
method	possible values are "cv" or "timeSlice"
...	the options for the estimation. Global options are: threshold: if TRUE all the entries smaller than the oracle threshold are set to zero; scale: scale the data (default = FALSE)? nfolds: the number of folds used for cross validation (default = 10); parallel: if TRUE use multicore backend (default = FALSE); ncores: if parallel is TRUE, specify the number of cores to use for parallel evaluation. Options for ENET estimation: alpha: the value of alpha to use in elastic net (0 is Ridge regression, 1 is LASSO (default)); type.measure: the measure to use for error evaluation ("mse" or "mae"); nlambda: the number of lambdas to use in the cross validation (default = 100); leaveOut: in the time slice validation leave out the last leaveOutLast observations (default = 15); horizon: the horizon to use for estimating mse/mae (default = 1).

Value

A the list (of length p) of the estimated matrices of the process

fit the results of the penalized LS estimation

mse the mean square error of the cross validation

time elapsed time for the estimation

residuals the time series of the residuals

fit_vecm *Multivariate VECM estimation*

Description

A function to estimate a (possibly big) multivariate VECM time series using penalized least squares methods, such as ENET, SCAD or MC+.

Usage

```
fit_vecm(data, p, penalty, method, log_scale, ...)
```

Arguments

data	the data from the time series: variables in columns and observations in rows
p	order of the VECM model
penalty	the penalty function to use. Possible values are "ENET", "SCAD" or "MCP"
method	"cv" or "timeSlice"
log_scale	should the function consider the log of the inputs? By default this is set to TRUE
...	options for the function (TODO: specify)

Value

Pi the matrix Pi for the VECM model

G the list (of length p-1) of the estimated matrices of the process

fit the results of the penalized LS estimation

mse the mean square error of the cross validation

time elapsed time for the estimation

frob_norm *Frobenius norm of a matrix*

Description

Compute the Frobenius norm of m

Usage

```
frob_norm(m)
```

Arguments

m	the matrix (real or complex valued)

Value

A numeric value representing the Frobenius norm of the matrix.

---

impulse_response *Impulse Response Function*

Description

A function to estimate the Impulse Response Function of a given VAR.

Usage

```
impulse_response(v, len = 20)
```

Arguments

v	the data in the for of a VAR
len	length of the impulse response function

Value

irf a 3d array containing the impulse response function.

---

inform_crit *Computes information criteria for VARs*

Description

This function computes information criteria (AIC, Schwartz and Hannan-Quinn) for VARs.

Usage

```
inform_crit(v)
```

Arguments

v	a list of VAR objects as from fitVAR.

Value

A data frame with columns AIC, BIC, and HannanQuinn containing the information criteria values for each VAR model in the input list.

l1norm *L1 matrix norm*

Description

Compute the L1 matrix norm of m

Usage

```
l1norm(m)
```

Arguments

m	the matrix (real or complex valued)

Value

A numeric value representing the L1 matrix norm (maximum absolute column sum).

l2norm *L2 matrix norm*

Description

Compute the L2 matrix norm of M

Usage

```
l2norm(m)
```

Arguments

m	the matrix (real or complex valued)

Value

A numeric value representing the L2 matrix norm.

l_infty_norm *L-infinity matrix norm*

Description

Compute the L-infinity matrix norm of m

Usage

```
l_infty_norm(m)
```

Arguments

m	the matrix (real or complex valued)

Value

A numeric value representing the L-infinity matrix norm (maximum absolute row sum).

max_norm *Max-norm of a matrix*

Description

Compute the max-norm of m

Usage

```
max_norm(m)
```

Arguments

m	the matrix (real or complex valued)

Value

A numeric value representing the maximum absolute entry of the matrix.

multiplot *Multiplots with ggplot*

Description

Multiple plot function. ggplot objects can be passed in ..., or to plotlist (as a list of ggplot objects)

Usage

```
multiplot(..., plotlist = NULL, cols = 1, layout = NULL)
```

Arguments

...	a sequence of ggplots to be plotted in the grid.
plotlist	a list containing ggplots as elements.
cols	number of columns in layout
layout	a matrix specifying the layout. If present, 'cols' is ignored. If the layout is something like matrix(c(1,2,3,3), nrow=2, byrow=TRUE), then plot 1 will go in the upper left, 2 will go in the upper right, and 3 will go all the way across the bottom. Taken from R Cookbook

Value

A ggplot containing the plots passed as arguments

plot_irf *IRF plot*

Description

Plot a IRF object

Usage

```
plot_irf(irf, eb, i, j, type, bands)
```

Arguments

irf	the irf object to plot
eb	the errorbands to plot
i	the first index
j	the second index
type	type = "irf" or type = "oirf"
bands	"quantiles" or "sd"

Value

An image plot relative to the impulse response function.

`plot_irf_grid` *IRF grid plot*

Description

Plot a IRF grid object

Usage

```
plot_irf_grid(irf, eb, indexes, type, bands)
```

Arguments

`irf`	the irf object computed using impulseResponse
`eb`	the error bands estimated using errorBands
`indexes`	a vector containing the indices that you want to plot
`type`	plot the irf (`type = "irf"` by default) or the orthogonal irf (`type = "oirf"`)
`bands`	which type of bands to plot ("quantiles" (default) or "sd")

Value

An image plot relative to the impulse response function.

`plot_matrix` *Matrix plot*

Description

Plot a sparse matrix

Usage

```
plot_matrix(m, colors)
```

Arguments

`m`	the matrix to plot
`colors`	dark or light

Value

An image plot with a particular color palette (black zero entries, red for the negative ones and green for the positive)

plot_var *Plot VARs*

Description

Plot all the matrices of a VAR model

Usage

```
plot_var(..., colors)
```

Arguments

...	a sequence of VAR objects (one or more than one, as from simulateVAR or fitVAR)
colors	the gradient used to plot the matrix. It can be "light" (low = red – mid = white – high = blue) or "dark" (low = red – mid = black – high = green)

Value

An image plot with a specific color palette

plot_vecm *Plot VECMs*

Description

Plot all the matrices of a VECM model

Usage

```
plot_vecm(v)
```

Arguments

v	a VECM object (as from fit_vecm)

Value

An image plot with a specific color palette (black zero entries, red for the negative ones and green for the positive)

simulate_var *VAR simulation*

Description

This function generates a simulated multivariate VAR time series.

Usage

```
simulate_var(n, p, nobs, rho, sparsity, mu, method, covariance, ...)
```

Arguments

n	dimension of the time series (default n = 100).
p	number of lags of the VAR model (default p = 1).
nobs	number of observations to be generated (default nobs = 250).
rho	base value for the covariance matrix (default rho = 0.5).
sparsity	density (in percentage) of the number of nonzero elements of the VAR matrices (default sparsity = 0.05).
mu	a vector containing the mean of the simulated process (default mu = 0).
method	which method to use to generate the VAR matrix. Possible values are "normal" or "bimodal" ((default method = "normal")).
covariance	type of covariance matrix to use in the simulation. Possible values: "Toeplitz", "block1", "block2", "Wishart" or simply "diagonal" (default covariance = "Toeplitz").
...	the options for the simulation. These are: muMat: the mean of the entries of the VAR matrices; sdMat: the sd of the entries of the matrices;

Value

A a list of NxN matrices ordered by lag

data a list with two elements: series the multivariate time series and noises the time series of errors

S the variance/covariance matrix of the process

simulate_varx *VARX simulation*

Description

This function generates a simulated multivariate VAR time series.

Usage

```
simulate_varx(n, k, p, m, nobs, rho,
                    sparsity_a1, sparsity_a2, sparsity_a3,
                    mu, method, covariance, ...)
```

Arguments

n	dimension of the time series.
k	TODO
p	number of lags of the VAR model.
m	TODO
nobs	number of observations to be generated.
rho	base value for the covariance matrix.
sparsity_a1	density (in percentage) of the number of nonzero elements of the A1 block.
sparsity_a2	density (in percentage) of the number of nonzero elements of the A2 block.
sparsity_a3	density (in percentage) of the number of nonzero elements of the A3 block.
mu	a vector containing the mean of the simulated process.
method	which method to use to generate the VAR matrix. Possible values are "normal" or "bimodal".
covariance	type of covariance matrix to use in the simulation. Possible values: "toeplitz", "block1", "block2" or simply "diagonal".
...	the options for the simulation. These are: muMat: the mean of the entries of the VAR matrices; sdMat: the sd of the entries of the matrices;

Value

A a list of NxN matrices ordered by lag

data a list with two elements: series the multivariate time series and noises the time series of errors

S the variance/covariance matrix of the process

sparsevar *sparsevar: A package to estimate multivariate time series models (such as VAR and VECM), under the sparsity hypothesis.*

Description

It performs the estimation of the matrices of the models using penalized least squares methods such as LASSO, SCAD and MCP.

sparsevar functions

`fit_var`, `fit_vecm`, `simulate_var`, `create_sparse_matrix`, `plotMatrix`, `plotVAR`, `plotVECM` `l2norm`, `l1norm`, `lInftyNorm`, `maxNorm`, `frobNorm`, `spectralRadius`, `spectralNorm`, `impulseResponse`

Author(s)

Maintainer: Simone Vazzoler <svazzole@gmail.com>

See Also

Useful links:

- https://github.com/svazzole/sparsevar
- Report bugs at https://github.com/svazzole/sparsevar/issues

spectral_norm *Spectral norm*

Description

Compute the spectral norm of m

Usage

```
spectral_norm(m)
```

Arguments

m	the matrix (real or complex valued)

Value

A numeric value representing the spectral norm (largest singular value).

spectral_radius *Spectral radius*

Description

Compute the spectral radius of m

Usage

```
spectral_radius(m)
```

Arguments

m	the matrix (real or complex valued)

Value

A numeric value representing the spectral radius (largest absolute eigenvalue).

test_granger *Test for Ganger Causality*

Description

This function should retain only the coefficients of the matrices of the VAR that are statistically significative (from the bootstrap)

Usage

```
test_granger(v, eb)
```

Arguments

v	the VAR object as from fitVAR or simulateVAR
eb	the error bands as obtained from errorBands

Value

A list of matrices containing only the statistically significant VAR coefficients (non-significant coefficients are set to zero).

transform_data *Transform data*

Description

Transform the input data

Usage

```
transform_data(data, p, opt)
```

Arguments

data	the data
p	the order of the VAR
opt	a list containing the options

Value

A list containing:

- X: the design matrix for the VAR estimation
- y: the response vector
- series: the (possibly centered/scaled) data matrix
- mu: the column means used for centering

true_negative_rate *True Negative Rate*

Description

Computes the True Negative Rate (Specificity) between a reference matrix and an estimated matrix. TNR = TN / (TN + FP), where negatives are zero entries in the reference matrix.

Usage

```
true_negative_rate(reference_m, a)
```

Arguments

reference_m	the reference (ground truth) matrix
a	the estimated matrix to compare against the reference

Value

The true negative rate (between 0 and 1), or NA if there are no actual negatives in the reference matrix

true_positive_rate *True Positive Rate*

Description

Computes the True Positive Rate (Sensitivity/Recall) between a reference matrix and an estimated matrix. TPR = TP / (TP + FN), where positives are non-zero entries in the reference matrix.

Usage

```
true_positive_rate(reference_m, a)
```

Arguments

reference_m	the reference (ground truth) matrix
a	the estimated matrix to compare against the reference

Value

The true positive rate (between 0 and 1), or NA if there are no actual positives in the reference matrix

var_enet *VAR ENET*

Description

Estimate VAR using ENET penalty

Usage

```
var_enet(data, p, lambdas, opt)
```

Arguments

data	the data
p	the order of the VAR
lambdas	a vector containing the lambdas to be used in the fit
opt	a list containing the options

Value

A glmnet object containing the fitted model.

var_mcp *VAR MCP*

Description

Estimate VAR using MCP penalty

Usage

```
var_mcp(data, p, lambdas, opt)
```

Arguments

data	the data
p	the order of the VAR
lambdas	a vector containing the lambdas to be used in the fit
opt	a list containing the options

Value

An ncvreg object containing the fitted model.

var_scad *VAR SCAD*

Description

Estimate VAR using SCAD penalty

Usage

```
var_scad(data, p, lambdas, opt, penalty)
```

Arguments

data	the data
p	the order of the VAR
lambdas	a vector containing the lambdas to be used in the fit
opt	a list containing the options
penalty	a string "SCAD" or something else

Value

An ncvreg object containing the fitted model.

Index