Package ‘bigtime’

August 9, 2021

Type Package

Title Sparse Estimation of Large Time Series Models

Version 0.2.1

Maintainer Ines Wilms <i.wilms@maastrichtuniversity.nl>


Depends R (>= 3.5.0), methods

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Imports Rcpp (>= 1.0.7), stats, utils, grDevices, graphics, corrplot, dplyr, ggplot2, tidyr, magrittr

SystemRequirements C++11

LinkingTo Rcpp, RcppArmadillo, RcppEigen

URL https://github.com/ineswilms/bigtime

NeedsCompilation yes

Author Ines Wilms [cre, aut], David S. Matteson [aut], Jacob Bien [aut], Sumanta Basu [aut], Will Nicholson [aut], Enrico Wegner [aut]

Repository CRAN

Date/Publication 2021-08-09 14:10:02 UTC
R topics documented:

- bigtime
- create_rand_coef_mat
- diagnostics_plot
- diagnostics_plot.bigtime.VAR
- diagnostics_plot.bigtime.VARMA
- diagnostics_plot.bigtime.VARX
- directforecast
- fitted.bigtime.VAR
- fitted.bigtime.VARMA
- fitted.bigtime.VARX
- get_ic_vals
- get_ic_vals.bigtime.VAR
- get_ic_vals.bigtime.VARX
- ic_selection
- is.stable
- lagmatrix
- plot.bigtime.recursiveforecast
- plot.bigtime.simVAR
- plot_cv
- recursiveforecast
- residuals.bigtime.VAR
- residuals.bigtime.VARMA
- residuals.bigtime.VARX
- simVAR
- sparseVAR
- sparseVARMA
- sparseVARX
- summary.bigtime.simVAR
- X.varx
- Y.var
- Y.varma
- Y.varx

Index

---

bigtime  

`bigtime`: A package for obtaining sparse estimates of large time series models.

**Description**

The `bigtime` package provides sparse estimators for three large time series models: Vector AutoRegressive Models, Vector AutoRegressive Models with Exogenous variables, and Vector AutoRegressive Moving Average Models. The univariate cases are also supported.
Details

To use the facilities of this package, start with a T by k time series matrix \( Y \) (for the VAR and VARMA), and an exogenous time series matrix \( X \) (for the VARX). Run `sparseVAR`, `sparseVARX` or `sparseVARMA` to get the estimated model. The function `lagmatrix` returns the lag matrix of estimated coefficients of the estimated model. The function `directforecast` gives h-step ahead forecasts based on the estimated model. The function `recursiveforecast` can be used to recursively forecast a VAR model. The function `is.stable` returns whether an estimated VAR model is stable. The function `diagnostics_plot` returns a plot of the fitted vs. observed values as well as of the residuals. The functions `fitted` and `residuals` return the fitted, respectively the residuals of the estimated model. The function `simVAR` can be used to simulate a VAR model with various sparsity patterns.

Author(s)

Ines Wilms <i.wilms@maastrichtuniversity.nl>, Jacob Bien, David S. Matteson, Sumanta Basu, Will Nicholson, Enrico Wegner

References


Examples

```r
# Fit a sparse VAR model
data(var.example)
VARfit <- sparseVAR(Y=scale(Y.var), selection = "cv") # using time series cross-validation
Lhat <- lagmatrix(fit=VARfit) # get estimated lagmatrix
VARforecast <- directforecast(fit=VARfit, h=1) # get one-step ahead forecasts
```

---

**create_rand_coef_mat**  
*Creates a random coefficient matrix*

Description

Creates a random coefficient matrix

Usage

```r
create_rand_coef_mat(
  k,
  p,
  max_abs_eigval = 0.8,
  sparsity_pattern = c("none", "lasso", "hvar"),
```

```r
```
sparsity_options = NULL,
decay = 0.5,
...
}

Arguments

k Number of time series
p Number of lags
max_abs_eigval if < 1, then the VAR will be stable
sparsity_pattern

The sparsity pattern that should be simulated. Options are: "none" for a dense VAR, "lasso" for a VAR with random zeroes, and "hvar" for an elementwise hierarchical sparsity pattern

sparsity_options

Named list of additional options for when sparsity pattern is lasso or hvar. For lasso the option num_zero determines the number of zeros. For hvar, the options zero_min (zero_max) give the minimum (maximum) of zeroes for each variable in each equation, and the option zeroes_in_self (boolean) determines if any of the coefficients of a variable on itself should be zero.

decay How fast should coefficients shrink when the lag increases.
...

Not currently used

Value

Returns a coefficient matrix in companion form of dimension kpxkp.

---

**diagnostics_plot**  
*Create a Diagnostic Plot*

**Description**

Create a Diagnostic Plot

**Usage**

diagnostics_plot(mod, variable = 1, dates = NULL)

**Arguments**

mod  
VAR model estimated using sparseVAR, sparseVARMA, or sparseVARX

variable  
Variable to show. Either numeric (which column) or character (variable name)

dates  
Optional Date vector.
**diagnostics_plot** function for **VAR** models

**Description**

Not supposed to be called directly. Rather call **diagnostics_plot**

**Usage**

```r
## S3 method for class 'bigtime.VAR'
diagnostics_plot(mod, variable = 1, dates = NULL)
```

**Arguments**

- **mod**: VAR model estimated using **sparseVAR**
- **variable**: Variable to show. Either numeric (which column) or character (variable name)
- **dates**: Optional Date vector.
diagnostics_plot.bigtime.VARMA

*diagnostics_plot function for VARMA models*

Description

Not supposed to be called directly. Rather call `diagnostics_plot`

Usage

```r
## S3 method for class 'bigtime.VARMA'
diagnostics_plot(mod, variable = 1, dates = NULL)
```

Arguments

- `mod` VAR model estimated using `sparseVARMA`
- `variable` Variable to show. Either numeric (which column) or character (variable name)
- `dates` Optional Date vector.

---

diagnostics_plot.bigtime.VARX

*diagnostics_plot function for VARX models*

Description

Not supposed to be called directly. Rather call `diagnostics_plot`

Usage

```r
## S3 method for class 'bigtime.VARX'
diagnostics_plot(mod, variable = 1, dates = NULL)
```

Arguments

- `mod` VARX model estimated using `sparseVARX`
- `variable` Variable to show. Either numeric (which column) or character (variable name)
- `dates` Optional Date vector.
directforecast  

Function to obtain h-step ahead direct forecast based on estimated VAR, VARX or VARMA model

Description

Function to obtain h-step ahead direct forecast based on estimated VAR, VARX or VARMA model

Usage

directforecast(fit, h = 1)

Arguments

- **fit**: Fitted sparse VAR, VARX or VARMA model.
- **h**: Desired forecast horizon. Default is h=1.

Value

Vector of length k containing the h-step ahead forecasts for the k time series.

Examples

data(var.example)
VARfit <- sparseVAR(Y=scale(Y), selection = "cv") # sparse VAR
VARforecast <- directforecast(fit=VARfit, h=1)

fitted.bigtime.VAR  

Gives the fitted values of a model estimated using sparseVAR

Description

Gives the fitted values of a model estimated using sparseVAR

Usage

```r
# S3 method for class 'bigtime.VAR'
fitted(object, ...)
```

Arguments

- **object**: Model estimated using sparseVAR
- **...**: Not currently used
Value

Returns a matrix of fitted values

Examples

dat <- simVAR(periods=200, k=2, p=5, decay = 0.001, seed = 6150533)
mod <- sparseVAR(Y=scale(dat$Y))
f <- fitted(mod)

fitted.bigtime.VARMA

Gives the fitted values of a model estimated using \texttt{sparseVARMA}

Description

Gives the fitted values of a model estimated using \texttt{sparseVARMA}

Usage

\texttt{## S3 method for class \'bigtime.VARMA\'
fit\_ted(object, ...)}

Arguments

\texttt{object} \hspace{1cm} \text{Model estimated using \texttt{sparseVARMA}}

\texttt{...} \hspace{1cm} \text{Not currently used}

Value

Returns a matrix of fitted values \texttt{data(varma.example)} varma <- sparseVARMA(Y = scale(Y.varma), VARMAselection="cv") f <- fitted(varma)

fitted.bigtime.VARX

Gives the fitted values of a model estimated using \texttt{sparseVARX}

Description

Gives the fitted values of a model estimated using \texttt{sparseVARX}

Usage

\texttt{## S3 method for class \'bigtime.VARX\'
fit\_ted(object, ...)}
get_ic_vals

Arguments

  object  Model estimated using sparseVARX
  ...  Not currently used

Value

Returns a matrix of fitted values data(varx.example) varx <- sparseVARX(Y=scale(Y.varx), X=scale(X.varx), selection="cv") fit <- fitted(varx)

get_ic_vals  Calculates the Information Criteria for a VAR, VARX, VARMA model

Description

The number of non-zero coefficients are taken as the degrees of freedom. Use with care for VARMA.

Usage

get_ic_vals(mod, verbose = TRUE)

Arguments

  mod  Model estimated using sparseVAR, sparseVARX, or sparseVARMA
  verbose  Should information about the optimal selection be printed?

Examples

dat <- simVAR(periods=200, k=2, p=5, decay = 0.01)
mod <- sparseVAR(Y=scale(dat$Y))
ics <- get_ic_vals(mod)

get_ic_vals.bigtime.VAR  Calculates the Information Criteria for a model estimated using sparseVAR

Description

The number of non-zero coefficients are taken as the degrees of freedom.

Usage

## S3 method for class 'bigtime.VAR'
get_ic_vals(mod, verbose = TRUE)
get_ic_vals.bigtime.VARX

Arguments

mod  Model estimated using sparseVAR
verbose  Should information about the optimal selection be printed?

Value

Returns a list containing

ics  Values of the ICs for all lambdas
mins  Which IC lead to the minimum (the row number)
selected_lambdas  Which lambdas were selected

Examples

dat <- simVAR(periods = 200, k=2, p=5, decay = 0.01)
mod <- sparseVAR(Y=scale(dat$Y))
ics <- get_ic_vals(mod)

Description

The number of non-zero coefficients in both the \( \Phi \) and \( B \) matrix are taken as the degrees of freedom.

Usage

## S3 method for class 'bigtime.VAR'
get_ic_vals(mod, verbose = TRUE)

Arguments

mod  Model estimated using sparseVAR
verbose  Should information about the optimal selection be printed?

Value

Returns a list containing

ics  Values of the ICs for all lambdas
mins  Which IC lead to the minimum (the row number)
selected_lambdas  Which lambdas were selected
selected_lamPhi  Which lambda \( \Phi \) were selected
selected_lamB  Which lambda \( B \) were selected
ic_selection

Selects the optimal penalty parameter using information criteria

Description

Selects the optimal penalty parameter using information criteria

Usage

ic_selection(mod, ic = c("bic", "aic", "hq"), verbose = FALSE)

Arguments

mod Model estimated Model estimated using sparseVAR, sparseVARX, or sparseVARMA
ic Which information criteria should be used. Must be one of "bic", "aic" or "hq"
verbose If true, some useful information will be printed during the process

Value

Returns a model that uses the optimal penalty

is.stable

Checks whether a VAR is stable

Description

Using a model estimated by sparseVAR, this function checks whether the resulting VAR is stable. This is the case, whenever the maximum absolute eigenvalue of the companion matrix corresponding to the VAR is less than one. This is sometimes also referred to as that the root lies outside the unit circle.

Usage

is.stable(mod, verbose = FALSE)

Arguments

mod Model estimated using sparseVAR. Can only be a model with one coefficient vector. Hence, the model must be estimated using a selection method. See sparseVAR for more details.
verbose If TRUE, then the actual maximum absolute eigenvalue of the companion matrix will be printed to the console. Default is FALSE

Value

Returns TRUE if the VAR is stable and FALSE otherwise
lagmatrix

**Description**

Creates Lagmatrix of Estimated Coefficients

**Usage**

```r
lagmatrix(fit, returnplot = F)
```

**Arguments**

- `fit` Fitted VAR, VARX or VARMA model.
- `returnplot` TRUE or FALSE: return plot of lag matrix or not.

**Value**

A list with estimated lag matrix of the VAR model, or lag matrices of the VARX or VARMA model. The rows contain the responses, the columns contain the predictors.

**Examples**

```r
data(var.example)
mod <- sparseVAR(Y=scale(Y.var), selection="cv")
Lhat <- lagmatrix(fit=mod)
```

---

plot.bigtime.recursiveforecast

**Description**

Plots Recursive Forecasts

Plots the recursive forecast obtained using `recursiveforecast` When forecasts were made for multiple lambdas and `lmbda` is not a single number, then a ribbon will be plotted that reaches from the minimum estimate of all lambdas to the maximum.

**Usage**

```r
## S3 method for class 'bigtime.recursiveforecast'
plot(x, series = NULL, lmbda = NULL, last_n = floor(nrow(fcst$Y) * 0.1), ...)
```

---
plot.bigtime.simVAR

Arguments

- `x`  Recursive Forecast obtained using \texttt{recursiveforecast}
- `series`  Series name. If original data has no names, then use Y1 for the first series, Y2 for the second, and so on.
- `lmbda`  Lambdas to be used for plotting. If forecast was done using only one lambda, then this will be ignored.
- `last_n`  Last \( n \) observations of the original data to include in the plot
- `...`  Not currently used

Details

If \texttt{lmbda} is of length one or forecasts were made using only one lambda, then only a line will be plotted.

Default names for series are Y1, Y2, ... if the original data does not have any column names.

Value

Returns a ggplot2 plot

---

\texttt{plot.bigtime.simVAR}  \textit{Plots a simulated VAR}

Description

Plots a simulated VAR

Usage

```r
## S3 method for class 'bigtime.simVAR'
plot(x, ...)
```

Arguments

- `x`  Simulated data of class \texttt{bigtime.simVAR} obtained from the \texttt{simVAR} function
- `...`  Not currently used

Value

Returns a ggplot2 plot
plot_cv

Plot the Cross Validation Error Curve for a Sparse VAR or VARX

Description
Plot the Cross Validation Error Curve for a Sparse VAR or VARX

Usage
plot_cv(fit, ...)

Arguments
fit Fitted VAR, VARMA or VARX model. returned by \texttt{sparseVAR}, \texttt{sparseVARMA} or \texttt{sparseVARX}.
... Not currently used

recursiveforecast
Recursively Forecasts a VAR

Description
Recursively forecasts a VAR estimated using \texttt{sparseVAR}. lambda can either be NULL, in which case all lambdas that were used for model estimation are used for forecasting, or a single value, in which case only the model using this lambda will be used for forecasting.

Usage
recursiveforecast(mod, h = 1, lambda = NULL)

Arguments
mod VAR model estimated using \texttt{sparseVAR}
\textit{h} Desired forecast horizon. Default is \textit{h}=1.
lambda Either NULL in which case a forecast will be made for all lambdas for which the model was estimated, or a single value in which case a forecast will only be made for the model using this lambda. Choice is redundant if the model was estimated using a selection procedure.

Value
Returns an object of S3 class \texttt{bigtime.recursiveforecast} containing
\textit{fcst} Matrix or 3D array of forecasts
\textit{h} Selected forecast horizon
\textit{lambda} List of lambdas for which the forecasts were made
\textit{Y} Data used for recursive forecasting
Examples

```r
sim_data <- simVAR(periods=200, k=5, p=5, seed = 12345)
summary(sim_data)
mod <- sparseVAR(Y=scale(sim_data$Y), selection = "bic")
is.stable(mod)
fcst_recursive <- recursiveforecast(mod, h = 4)
plot(fcst_recursive, series = "Y1")
fcst_direct <- directforecast(mod)
fcst_direct
fcst_recursive$fcst
```

---

residuals.bigtime.VAR  
*Gives the residuals for VAR models estimated using `sparseVAR`*

Description

Gives the residuals for VAR models estimated using `sparseVAR`

Usage

```r
## S3 method for class 'bigtime.VAR'
residuals(object, ...)
```

Arguments

- `object`  
  Model estimated using `sparseVAR`
- `...`  
  Not currently used

Value

Returns a matrix of residuals.

Examples

```r
dat <- simVAR(periods=200, k=2, p=5, decay = 0.001, seed = 6150533)
mod <- sparseVAR(Y=scale(dat$Y))
res <- resid(mod)
```
residuals.bigtime.VARMA

Gives the residuals for VARMA models estimated using `sparseVARMA`

Description

Gives the residuals for VARMA models estimated using `sparseVARMA`

Usage

```r
## S3 method for class 'bigtime.VARMA'
residuals(object, ...)
```

Arguments

- `object` Model estimated using `sparseVARMA`
- `...` Not currently used

Value

Returns a matrix of residuals.

Examples

```r
## Not run:
data(varma.example)
varma <- sparseVARMA(Y = scale(Y.varma), VARMAselection="cv")
res <- residuals(varma)
## End(Not run)
```

residuals.bigtime.VARX

Gives the residuals for VARX models estimated using `sparseVARX`

Description

Gives the residuals for VARX models estimated using `sparseVARX`

Usage

```r
## S3 method for class 'bigtime.VARX'
residuals(object, ...)
```

Arguments

- `object` Model estimated using `sparseVARX`
- `...` Not currently used

Value

Returns a matrix of residuals.
simVAR

Arguments

object  Model estimated using sparseVARX
...  Not currently used

Value

Returns a matrix of residuals.

Examples

## Not run:
data(varx.example)
varx <- sparseVARX(Y=scale(Y.varx), X=scale(X.varx), selection="cv")
res <- residuals(varx)

## End(Not run)
Arguments

- **periods**: Scalar indicating the desired time series length
- **k**: Number of time series
- **p**: Maximum lag number. In case of `sparsity_patter="none"` this will be the actual number of lags for all variables
- **coef_mat**: Coefficient matrix in companion form. If not provided, one will be simulated
- **const**: Constant term of VAR. Default is zero. Must be either a scalar, in which case it will be broadcasted to a k-vector, or a k-vector
- **e_dist**: Either a function taking argument n indicating the number of variables in the system, or a matrix of dimensions k x (periods+burnin)
- **init_y**: Initial values. Defaults to zero. Expects either a scalar or a vector of length (k*p)
- **max_abs_eigval**: Maximum allowed eigenvalue of companion matrix. Only applicable if coefficient matrix is being simulated
- **burnin**: Number of time points to be used for burnin
- **sparsity_pattern**: The sparsity pattern that should be simulated. Options are: "none" for a dense VAR, "lasso" (or "L1") for a VAR with random zeroes, and "hvar" (or "HLag") for an elementwise hierarchical sparsity pattern
- **sparsity_options**: Named list of additional options for when sparsity pattern is lasso (L1) or hvar (HLag). For lasso (L1) the option `num_zero` determines the number of zeros. For hvar (HLag), the options `zero_min` (`zero_max`) give the minimum (maximum) of zeroes for each variable in each equation, and the option `zeroes_in_self` (boolean) determines if any of the coefficients of a variable on itself should be zero.
- **decay**: How much smaller should parameters for later lags be. The smaller, the larger will early parameters be w.r.t. later ones.
- **seed**: Seed to be used for the simulation
- **...**: Additional arguments passed to `e_dist`

Value

Returns an object of S3 class `bigtime.simVAR` containing the following

- **Y**: Simulated Data
- **periods**: Time series length
- **k**: Number of endogenous variables
- **p**: Maximum lag length; effective lag length might be shorter due to sparsity patterns
- **coef_mat**: Companion form of the coefficient matrix. Will be of dimensions (kp)x(kp). First k rows correspond to the actual coefficient matrix.
- **is_coef_mat_simulated**: TRUE if the coef_mat was simulated, FALSE if it was user provided
sparseVAR

const Constant term
e_dist Errors used in the construction of the data
init_y Initial conditions
max_abs_eigval Maximum eigenvalue to which the companion matrix was constraint
burnin Burnin period used
sparsity_pattern Sparsity pattern used
sparsity_options Extra options for the sparsity patterns used
seed Seed used for the simulation

Examples

```r
periods <- 200 # time series length
k <- 5 # number of variables
p <- 10 # maximum lag
sparsity_pattern <- "HLag" # HLag sparsity structure
sparsity_options <- list(zero_min = 0, # variables can be included with all lags
zero_max = 10, # but some could also include no lags
zeroes_in_self = TRUE)

sim <- simVAR(periods=periods, k=k, p=p, sparsity_pattern=sparsity_pattern,
sparsity_options=sparsity_options, seed = 12345)

summary(sim)
```

---

**sparseVAR**

*Sparse Estimation of the Vector AutoRegressive (VAR) Model*

**Description**

Sparse Estimation of the Vector AutoRegressive (VAR) Model

**Usage**

```r
sparseVAR(
  Y,
  p = NULL,
  VARpen = "HLag",
  VARlseq = NULL,
  VARgran = NULL,
  selection = c("none", "cv", "bic", "aic", "hq"),
  cvcut = 0.9,
  h = 1,
  eps = 0.001,
  check_std = TRUE
)
```
Arguments

Y A $T$ by $k$ matrix of time series. If $k=1$, a univariate autoregressive model is estimated.

p User-specified maximum autoregressive lag order of the VAR. Typical usage is to have the program compute its own maximum lag order based on the time series length.

VARpen "HLag" (hierarchical sparse penalty) or "L1" (standard lasso penalty) penalization.

VARlseq User-specified grid of values for regularization parameter corresponding to sparse penalty. Typical usage is to have the program compute its own grid. Supplying a grid of values overrides this. WARNING: use with care.

VARgran User-specified vector of granularity specifications for the penalty parameter grid: First element specifies how deep the grid should be constructed. Second element specifies how many values the grid should contain.

selection One of "none" (default), "cv" (Time Series Cross-Validation), "bic", "aic", "hq". Used to select the optimal penalization.

cvcut Proportion of observations used for model estimation in the time series cross-validation procedure. The remainder is used for forecast evaluation. Redundant if selection is not "cv".

h Desired forecast horizon in time-series cross-validation procedure.

eps a small positive numeric value giving the tolerance for convergence in the proximal gradient algorithm.

check_std Check whether data is standardised. Default is TRUE and is not recommended to be changed.

Value

A list with the following components

Y $T$ by $k$ matrix of time series.

k Number of time series.

p Maximum autoregressive lag order of the VAR.

Phihat Matrix of estimated autoregressive coefficients of the VAR.

phihat vector of VAR intercepts.

series_names names of time series

lambdas sparsity parameter grid

MSFEcv MSFE cross-validation scores for each value of the sparsity parameter in the considered grid

MSFEcv_all MSFE cross-validation full output

lambda_opt Optimal value of the sparsity parameter as selected by the time-series cross-validation procedure

lambda_SEopt Optimal value of the sparsity parameter as selected by the time-series cross-validation procedure and after applying the one-standard-error rule. This is the value used.

h Forecast horizon h
References


See Also

lagmatrix and directforecast

Examples

data(var.example)
VARfit <- sparseVAR(Y = scale(Y.var)) # sparse VAR
ARfit <- sparseVAR(Y=scale(Y.var[,2])) # sparse AR

Description

Sparse Estimation of the Vector AutoRegressive Moving Average (VARMA) Model

Usage

sparseVARMA(
  Y,
  U = NULL,
  VARp = NULL,
  VARpen = "HLag",
  VARMAlPhiseq = NULL,
  VARMAThetaseq = NULL,
  VARMAPhigran = NULL,
  VARMAThetagran = NULL,
  VARMAalpha = 0,
  VARMApen = "HLag",
  VARMAPhiseq = NULL,
  VARMAPhigran = NULL,
  VARMAThetaseq = NULL,
  VARMAThetagran = NULL,
  VARMAalpha = 0,
  VARMApen = "HLag",
  VARMAPhiseq = NULL,
  VARMAPhigran = NULL,
  VARMAThetaseq = NULL,
  VARMAThetagran = NULL,
  h = 1,
  cvcut = 0.9,
  eps = 10^-3,
  check_std = TRUE
)
Arguments

Y
A T by k matrix of time series. If k=1, a univariate autoregressive moving average model is estimated.

U
A T by k matrix of (approximated) error terms. Typical usage is to have the program estimate a high-order VAR model (Phase I) to get approximated error terms U.

VARp
User-specified maximum autoregressive lag order of the Phase I VAR. Typical usage is to have the program compute its own maximum lag order based on the time series length.

VARpen
"HLag" (hierarchical sparse penalty) or "L1" (standard lasso penalty) penalization in Phase I VAR.

VAR1seq
User-specified grid of values for regularization parameter in the Phase I VAR. Typical usage is to have the program compute its own grid. Supplying a grid of values overrides this. WARNING: use with care.

VARgran
User-specified vector of granularity specifications for the penalty parameter grid of the Phase I VAR: First element specifies how deep the grid should be constructed. Second element specifies how many values the grid should contain.

VARselection
Selection procedure for the first stage. Default is time series Cross-Validation. Alternatives are BIC, AIC, HQ

VARMap
User-specified maximum autoregressive lag order of the VARMA. Typical usage is to have the program compute its own maximum lag order based on the time series length.

VARMAq
User-specified maximum moving average lag order of the VARMA. Typical usage is to have the program compute its own maximum lag order based on the time series length.

VARMApen
"HLag" (hierarchical sparse penalty) or "L1" (standard lasso penalty) penalization in the VARMA.

VARMA1Phiseq
User-specified grid of values for regularization parameter corresponding to the autoregressive coefficients in the VARMA. Typical usage is to have the program compute its own grid. Supplying a grid of values overrides this. WARNING: use with care.

VARMAPhigran
User-specified vector of granularity specifications for the penalty parameter grid corresponding to the autoregressive coefficients in the VARMA: First element specifies how deep the grid should be constructed. Second element specifies how many values the grid should contain.

VARMA1Thetaseq
User-specified grid of values for regularization parameter corresponding to the moving average coefficients in the VARMA. Typical usage is to have the program compute its own grid. Supplying a grid of values overrides this. WARNING: use with care.

VARMAThetagran
User-specified vector of granularity specifications for the penalty parameter grid corresponding to the moving average coefficients in the VARMA: First element specifies how deep the grid should be constructed. Second element specifies how many values the grid should contain.
VARMAalpha  a small positive regularization parameter value corresponding to squared Frobenius penalty in VARMA. The default is zero.
VARMAselection  selection procedure in the second stage. Default is "none"; Alternatives are cv, bic, aic, hq
h     Desired forecast horizon in time-series cross-validation procedure.
cvcut     Proportion of observations used for model estimation in the time series cross-validation procedure. The remainder is used for forecast evaluation.
eps     a small positive numeric value giving the tolerance for convergence in the proximal gradient algorithms.
check_std  Check whether data is standardised. Default is TRUE and is not recommended to be changed

Value

A list with the following components

Y     $T \times k$ matrix of time series.
U     Matrix of (approximated) error terms.
k     Number of time series.
VARp     Maximum autoregressive lag order of the PhaseI VAR.
VARPhihat  Matrix of estimated autoregressive coefficients of the Phase I VAR.
VARphi0hat  Vector of Phase I VAR intercepts.
VARMAp     Maximum autoregressive lag order of the VARMA.
VARMAq     Maximum moving average lag order of the VARMA.
Phihat     Matrix of estimated autoregressive coefficients of the VARMA.
Thetahat     Matrix of estimated moving average coefficients of the VARMA.
phi0hat     Vector of VARMA intercepts.
series_names  names of time series
PhaseI_lambas  Phase I sparsity parameter grid
PhaseI_MSFEcv  MSFE cross-validation scores for each value of the sparsity parameter in the considered grid
PhaseI_lambda_opt  Phase I Optimal value of the sparsity parameter as selected by the time-series cross-validation procedure
PhaseI_lambda_SEopt  Phase I Optimal value of the sparsity parameter as selected by the time-series cross-validation procedure and after applying the one-standard-error rule
PhaseII_lambdaPhi  Phase II sparsity parameter grid corresponding to Phi parameters
PhaseII_lambdaTheta  Phase II sparsity parameter grid corresponding to Theta parameters
PhaseII_lambdaPhil_opt
Phase II Optimal value of the sparsity parameter (corresponding to Phi parameters) as selected by the time-series cross-validation procedure

PhaseII_lambdaPhiSEopt
Phase II Optimal value of the sparsity parameter (corresponding to Theta parameters) as selected by the time-series cross-validation procedure and after applying the one-standard-error rule

PhaseII_lambdaTheta_opt
Phase II Optimal value of the sparsity parameter (corresponding to Phi parameters) as selected by the time-series cross-validation procedure

PhaseII_lambdaThetasEopt
Phase II Optimal value of the sparsity parameter (corresponding to Theta parameters) as selected by the time-series cross-validation procedure and after applying the one-standard-error rule

PhaseII_MSFEcv
Phase II MSFE cross-validation scores for each value in the two-dimensional sparsity grid

h
Forecast horizon h

References

See Also
lagmatrix and directforecast

Examples

data(varma.example)
VARMAfit <- sparseVARMA(Y = scale(Y.varma)) # sparse VARMA
y <- matrix(Y.varma[,1], ncol=1)
ARMAdiff <- sparseVARMA(y=scale(y)) # sparse ARMA

sparseVARX

Sparse Estimation of the Vector AutoRegressive with Exogenous Variables X (VARX) Model

Description
Sparse Estimation of the Vector AutoRegressive with Exogenous Variables X (VARX) Model
sparseVARX

Usage

sparseVARX(
  Y, 
  X, 
  p = NULL, 
  s = NULL, 
  VARXpen = "HLag", 
  VARX1Phiseq = NULL, 
  VARXPhigran = NULL, 
  VARX1Bseq = NULL, 
  VARXBgran = NULL, 
  VARXalpha = 0, 
  h = 1, 
  cvcut = 0.9, 
  eps = 10^-3, 
  selection = c("none", "cv", "bic", "aic", "hq"), 
  check_std = TRUE
)

Arguments

Y  A $T$ by $k$ matrix of time series. If $k=1$, a univariate autoregressive model is estimated.

X  A $T$ by $m$ matrix of time series.

p  User-specified maximum endogenous autoregressive lag order. Typical usage is to have the program compute its own maximum lag order based on the time series length.

s  User-specified maximum exogenous autoregressive lag order. Typical usage is to have the program compute its own maximum lag order based on the time series length.

VARXpen  "HLag" (hierarchical sparse penalty) or "L1" (standard lasso penalty) penalization in VARX.

VARX1Phiseq  User-specified grid of values for regularization parameter corresponding to the endogenous autoregressive coefficients in the VARX. Typical usage is to have the program compute its own grid. Supplying a grid of values overrides this. WARNING: use with care.

VARXPhigran  User-specified vector of granularity specifications for the penalty parameter grid corresponding to the endogenous autoregressive coefficients in the VARX: First element specifies how deep the grid should be constructed. Second element specifies how many values the grid should contain.

VARX1Bseq  User-specified grid of values for regularization parameter corresponding to the exogenous autoregressive coefficients in the VARX. Typical usage is to have the program compute its own grid. Supplying a grid of values overrides this. WARNING: use with care.

VARXBgran  User-specified vector of granularity specifications for the penalty parameter grid corresponding to the exogenous autoregressive coefficients in the VARX: First
element specifies how deep the grid should be constructed. Second element specifies how many values the grid should contain.

VARXalpha a small positive regularization parameter value corresponding to squared Frobenius penalty. The default is zero.

h Desired forecast horizon in time-series cross-validation procedure.

cvcut Proportion of observations used for model estimation in the time series cross-validation procedure. The remainder is used for forecast evaluation.

eps a small positive numeric value giving the tolerance for convergence in the proximal gradient algorithm.

selection Model selection method to be used. Default is none, which will return all values for all penalisations.

check_std Check whether data is standardised. Default is TRUE and is not recommended to be changed.

Value

A list with the following components:

Y \(T\) by \(k\) matrix of endogenous time series.

X \(T\) by \(m\) matrix of exogenous time series.

k Number of endogenous time series.

m Number of exogenous time series.

p Maximum endogenous autoregressive lag order of the VARX.

s Maximum exogenous autoregressive lag order of the VARX.

\(\Phi\) matrix of estimated endogenous autoregressive coefficients.

\(B\) matrix of estimated exogenous autoregressive coefficients.

\(\phi\) vector of VARX intercepts.

exogenous_series_names names of the exogenous time series

endogenous_series_names names of the endogenous time series

\(\lambda\) sparsity parameter grid corresponding to endogenous autoregressive parameters

\(\lambda\) sparsity parameter grid corresponding to exogenous autoregressive parameters

\(\lambda\) Optimal value of the sparsity parameter (corresponding to the endogenous autoregressive parameters) as selected by the time-series cross-validation procedure.

\(\lambda\) Optimal value of the sparsity parameter (corresponding to the endogenous autoregressive parameters) as selected by the time-series cross-validation procedure and after applying the one-standard-error rule.

\(\lambda\) Optimal value of the sparsity parameter (corresponding to the exogenous autoregressive parameters) as selected by the time-series cross-validation procedure.
lambdaB_SEopt  Optimal value of the sparsity parameter (corresponding to the exogenous autoregressive parameters) as selected by the time-series cross-validation procedure and after applying the one-standard-error rule

MSFEcv  MSFE cross-validation scores for each value in the two-dimensional sparsity grid

h  Forecast horizon h

References


See Also

lagmatrix and directforecast

Examples

data(varx.example)
VARXfit <- sparseVARX(Y=scale(Y.varx), X=scale(X.varx)) # sparse VARX
y <- matrix(Y.varx[,1], ncol=1)
ARXfit <- sparseVARX(Y=y, X=X.varx) # sparse ARX

summary.bigtime.simVAR

Gives a small summary of a VAR simulation

Description

Gives a small summary of a VAR simulation

Usage

## S3 method for class 'bigtime.simVAR'
summary(object, plot = TRUE, ...)

Arguments

object  Simulated data of class bigtime.simVAR obtained from the simVAR function
plot  Should the VAR be plotted. Default is TRUE
...  Not currently used

Value

If ‘plot=TRUE’, then a ggplot2 plot will be returned
X.varx  \hspace{1cm} VARX Time Series Example (varx.example)

**Description**

The data consists of a 200x3 matrix of endogenous variables, Y.varx, and a 200x3 matrix of exogenous variables, X.varx.

**Usage**

X.varx

**Format**

Two matrices, X.varx and Y.varx, both of dimension 200x3

Y.var  \hspace{1cm} VAR Time Series Example (var.example)

**Description**

The data consists of a 200x5 data matrix, Y.var, and was simulated from a sparse VAR model with H1ag sparsity pattern.

**Usage**

Y.var

**Format**

A matrix of dimension 200x5

Y.varma  \hspace{1cm} VARMA Time Series Example (varma.example)

**Description**

The data consists of a 200x3 data matrix, Y.varma, and was simulated from a sparse VARMA model.

**Usage**

Y.varma

**Format**

A matrix of dimension 200x3
Y.varx

VARX Time Series Example (varx.example)

Description

The data consists of a 200x3 matrix of endogenous variables, Y.varx, and a 200x3 matrix of exogenous variables, X.varx.

Usage

Y.varx

Format

Two matrices, X.varx and Y.varx, both of dimension 200x3
# Index

* **datasets**
  - X.varx, 28
  - Y.var, 28
  - Y.varma, 28
  - Y.varx, 29

  bigtime, 2

  create_rand_coef_mat, 3

  diagnostics_plot, 3, 4, 5, 6
  diagnostics_plot.bigtime.VAR, 5
  diagnostics_plot.bigtime.VARMA, 6
  diagnostics_plot.bigtime.VARX, 6
  directforecast, 3, 7, 21, 24, 27

  fitted, 3
  fitted.bigtime.VAR, 7
  fitted.bigtime.VARMA, 8
  fitted.bigtime.VARX, 8

  get_ic_vals, 9
  get_ic_vals.bigtime.VAR, 9
  get_ic_vals.bigtime.VARX, 10

  ic_selection, 11
  is.stable, 3, 11

  lagmatrix, 3, 12, 21, 24, 27

  plot.bigtime.recursiveforecast, 12
  plot.bigtime.simVAR, 13
  plot_cv, 14

  recursiveforecast, 3, 12, 13, 14
  residuals, 3
  residuals.bigtime.VAR, 15
  residuals.bigtime.VARMA, 16
  residuals.bigtime.VARX, 16

  simVAR, 3, 13, 17, 27

  sparseVAR, 3–5, 7, 9–11, 14, 15, 19
  sparseVARMA, 3, 4, 6, 8, 9, 11, 14, 16, 21
  sparseVARX, 3, 4, 6, 8–11, 14, 16, 17, 24
  summary.bigtime.simVAR, 27

  X.var, 28
  Y.var, 28
  Y.varma, 28
  Y.varx, 29