Package ‘fitAutoReg’

August 25, 2023

Title Fit Autoregressive Models

Version 0.9.1

Description Fit autoregressive models using 'RcppArmadillo'.

URL https://github.com/ijapesigan/fitAutoReg,
https://ijapesigan.github.io/fitAutoReg/

BugReports https://github.com/ijapesigan/fitAutoReg/issues

License GPL (>= 3)

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

VignetteBuilder knitr

Depends R (>= 3.5.0)

LinkingTo Rcpp, RcppArmadillo

Imports Rcpp

Suggests knitr, rmarkdown, testthat

RoxygenNote 7.2.3

NeedsCompilation yes

Author Ivan Jacob Agaloos Pesigan [aut, cre, cph]
(<https://orcid.org/0000-0003-4818-8420>)

Maintainer Ivan Jacob Agaloos Pesigan <ijapesigan@gmail.com>

R topics documented:

`dat_demo` *Data from the Vector Autoregressive Model*

Description

Data from the Vector Autoregressive Model

Usage

```
dat_demo
```

Format

A matrix with 1000 rows (time points) and k = 3 columns (variables) generated from the p = 2 vector autoregressive model given by

$$Y_{1_t} = 1 + 0.4Y_{1_{t-1}} + 0.0Y_{2_{t-1}} + 0.0Y_{3_{t-1}} + 0.1Y_{1_{t-2}} + 0.0Y_{2_{t-2}} + 0.0Y_{3_{t-2}} + \varepsilon_{1_t},$$

$$Y_{2_t} = 1 + 0.0Y_{1_{t-1}} + 0.5Y_{2_{t-1}} + 0.0Y_{3_{t-1}} + 0.0Y_{1_{t-2}} + 0.2Y_{2_{t-2}} + 0.0Y_{3_{t-2}} + \varepsilon_{2_t},$$

and

$$Y_{3_t} = 1 + 0.0Y_{1_{t-1}} + 0.0Y_{2_{t-1}} + 0.6Y_{3_{t-1}} + 0.0Y_{1_{t-2}} + 0.0Y_{2_{t-2}} + 0.3Y_{3_{t-2}} + \varepsilon_{3_t}$$

which simplifies to

$$Y_{1_t} = 1 + 0.4Y_{1_{t-1}} + 0.1Y_{1_{t-2}} + \varepsilon_{1_t},$$

$$Y_{2_t} = 1 + 0.5Y_{2_{t-1}} + 0.2Y_{2_{t-2}} + \varepsilon_{2_t},$$

and

$$Y_{3_t} = 1 + 0.6Y_{3_{t-1}} + 0.3Y_{3_{t-2}} + \varepsilon_{3_t}.$$

The covariance matrix of process noise is an identity matrix.

`dat_demo_exo` *Data from the Vector Autoregressive Model with Exogenous Variables*

Description

Data from the Vector Autoregressive Model with Exogenous Variables

Usage

```
dat_demo_exo
```

Format

A matrix with 1000 rows (time points) and k = 3 (autoregressive variables) plus m = 3 columns (exogenous variables) generated from the p = 2 vector autoregressive model given by

$Y_{1_t} = 1 + 0.4 Y_{1_{t-1}} + 0.0 Y_{2_{t-1}} + 0.0 Y_{3_{t-1}} + 0.1 Y_{1_{t-2}} + 0.0 Y_{2_{t-2}} + 0.0 Y_{3_{t-2}} + 0.5 X_1 + 0.0 X_2 + 0.0 X_3 \varepsilon_{1_t},$

$Y_{2_t} = 1 + 0.0 Y_{1_{t-1}} + 0.5 Y_{2_{t-1}} + 0.0 Y_{3_{t-1}} + 0.0 Y_{1_{t-2}} + 0.2 Y_{2_{t-2}} + 0.0 Y_{3_{t-2}} + 0.0 X_1 + 0.5 X_2 + 0.0 X_3 \varepsilon_{2_t},$

and

$Y_{3_t} = 1 + 0.0 Y_{1_{t-1}} + 0.0 Y_{2_{t-1}} + 0.6 Y_{3_{t-1}} + 0.0 Y_{1_{t-2}} + 0.0 Y_{2_{t-2}} + 0.3 Y_{3_{t-2}} + 0.0 X_1 + 0.0 X_2 + 0.5 X_3 \varepsilon_{3_t}$

which simplifies to

$$Y_{1_t} = 1 + 0.4 Y_{1_{t-1}} + 0.1 Y_{1_{t-2}} + 0.5 X_1 \varepsilon_{1_t},$$
$$Y_{2_t} = 1 + 0.5 Y_{2_{t-1}} + 0.2 Y_{2_{t-2}} + 0.5 X_2 \varepsilon_{2_t},$$

and

$$Y_{3_t} = 1 + 0.6 Y_{3_{t-1}} + 0.3 Y_{3_{t-2}} + 0.5 X_3 \varepsilon_{3_t}.$$

The covariance matrix of process noise is an identity matrix.

`dat_demo_exo_yx` *Data from the Vector Autoregressive Model (Y) and Lagged Predictors and Exogenous Variables (X)*

Description

Data from the Vector Autoregressive Model (Y) and Lagged Predictors and Exogenous Variables (X)

Usage

```
dat_demo_exo_yx
```

Format

A list with elements Y and X where Y is equal to the k = 3 autoregressive variables of the `dat_demo_exo` data set minus p = 2 terminal rows and `X` is a matrix of ones for the first column, lagged values of `Y`, and m = 3 exogenous variables.

`dat_demo_yx` *Data from the Vector Autoregressive Model (Y) and Lagged Predictors (X)*

Description

Data from the Vector Autoregressive Model (Y) and Lagged Predictors (X)

Usage

```
dat_demo_yx
```

Format

A list with elements Y and X where Y is equal to the `dat_demo` data set minus p = 2 terminal rows and `X` is a matrix of ones for the first column and lagged values of `Y` for the rest of the columns.

`FitVARLasso` *Fit Vector Autoregressive (VAR) Model Parameters using Lasso Regularization*

Description

This function estimates the parameters of a VAR model using the Lasso regularization method with cyclical coordinate descent. The Lasso method is used to estimate the autoregressive and cross-regression coefficients with sparsity.

Usage

```
FitVARLasso(Ystd, Xstd, lambda, max_iter, tol)
```

Arguments

`Ystd`	Numeric matrix. Matrix of standardized dependent variables (Y).
`Xstd`	Numeric matrix. Matrix of standardized predictors (X).
`lambda`	Lasso hyperparameter. The regularization strength controlling the sparsity.
`max_iter`	Integer. The maximum number of iterations for the coordinate descent algorithm (e.g., `max_iter = 10000`).
`tol`	Numeric. Convergence tolerance. The algorithm stops when the change in coefficients between iterations is below this tolerance (e.g., `tol = 1e-5`).

Details

The FitVARLasso() function estimates the parameters of a Vector Autoregressive (VAR) model using the Lasso regularization method. Given the input matrices Ystd and Xstd, where Ystd is the matrix of standardized dependent variables, and Xstd is the matrix of standardized predictors, the function computes the autoregressive and cross-regression coefficients of the VAR model with sparsity induced by the Lasso regularization.

The steps involved in estimating the VAR model parameters using Lasso are as follows:

- **Initialization**: The function initializes the coefficient matrix beta with OLS estimates. The beta matrix will store the estimated autoregressive and cross-regression coefficients.
- **Coordinate Descent Loop**: The function performs the cyclical coordinate descent algorithm to estimate the coefficients iteratively. The loop iterates max_iter times, or until convergence is achieved. The outer loop iterates over the predictor variables (columns of Xstd), while the inner loop iterates over the outcome variables (columns of Ystd).
- **Coefficient Update**: For each predictor variable (column of Xstd), the function iteratively updates the corresponding column of beta using the coordinate descent algorithm with L1 norm regularization (Lasso). The update involves calculating the soft-thresholded value c, which encourages sparsity in the coefficients. The algorithm continues until the change in coefficients between iterations is below the specified tolerance tol or when the maximum number of iterations is reached.
- **Convergence Check**: The function checks for convergence by comparing the current beta matrix with the previous iteration's beta_old. If the maximum absolute difference between beta and beta_old is below the tolerance tol, the algorithm is considered converged, and the loop exits.

Value

Matrix of estimated autoregressive and cross-regression coefficients.

Author(s)

Ivan Jacob Agaloos Pesigan

See Also

The FitVAROLS() function for estimating VAR model parameters using OLS.

Other Fitting Autoregressive Model Functions: FitVARLassoSearch(), FitVAROLS(), LambdaSeq(), OrigScale(), PBootVARLasso(), PBootVAROLS(), SearchVARLasso(), StdMat()

Examples

```
Ystd <- StdMat(dat_demo_yx$Y)
Xstd <- StdMat(dat_demo_yx$X[, -1])
lambda <- 73.90722
FitVARLasso(Ystd = Ystd, Xstd = Xstd, lambda = lambda,
  max_iter = 10000, tol = 1e-5)
```

FitVARLassoSearch *Fit Vector Autoregressive (VAR) Model Parameters using Lasso Regularization with Lambda Search*

Description

Fit Vector Autoregressive (VAR) Model Parameters using Lasso Regularization with Lambda Search

Usage

```
FitVARLassoSearch(Ystd, Xstd, lambdas, crit, max_iter, tol)
```

Arguments

`Ystd`	Numeric matrix. Matrix of standardized dependent variables (Y).
`Xstd`	Numeric matrix. Matrix of standardized predictors (X).
`lambdas`	Numeric vector. Vector of lambda hyperparameters for Lasso regularization.
`crit`	Character string. Information criteria to use. Valid values include `"aic"`, `"bic"`, and `"ebic"`.
`max_iter`	Integer. The maximum number of iterations for the coordinate descent algorithm (e.g., `max_iter = 10000`).
`tol`	Numeric. Convergence tolerance. The algorithm stops when the change in coefficients between iterations is below this tolerance (e.g., `tol = 1e-5`).

Value

Matrix of estimated autoregressive and cross-regression coefficients.

Author(s)

Ivan Jacob Agaloos Pesigan

See Also

Other Fitting Autoregressive Model Functions: `FitVARLasso()`, `FitVAROLS()`, `LambdaSeq()`, `OrigScale()`, `PBootVARLasso()`, `PBootVAROLS()`, `SearchVARLasso()`, `StdMat()`

Examples

```
Ystd <- StdMat(dat_demo_yx$Y)
Xstd <- StdMat(dat_demo_yx$X[, -1])
lambdas <- LambdaSeq(Y = Ystd, X = Xstd, n_lambdas = 100)
FitVARLassoSearch(Ystd = Ystd, Xstd = Xstd, lambdas = lambdas,
  crit = "ebic", max_iter = 1000, tol = 1e-5)
```

FitVAROLS — *Fit Vector Autoregressive (VAR) Model Parameters using OLS*

Description

This function estimates the parameters of a VAR model using the Ordinary Least Squares (OLS) method. The OLS method is used to estimate the autoregressive and cross-regression coefficients.

Usage

```
FitVAROLS(Y, X)
```

Arguments

`Y`	Numeric matrix. Matrix of dependent variables (Y).
`X`	Numeric matrix. Matrix of predictors (X).

Details

The `FitVAROLS()` function estimates the parameters of a Vector Autoregressive (VAR) model using the Ordinary Least Squares (OLS) method. Given the input matrices `Y` and `X`, where `Y` is the matrix of dependent variables, and `X` is the matrix of predictors, the function computes the autoregressive and cross-regression coefficients of the VAR model. Note that if the first column of `X` is a vector of ones, the constant vector is also estimated.

The steps involved in estimating the VAR model parameters using OLS are as follows:

- Compute the QR decomposition of the lagged predictor matrix `X` using the `qr` function from the Armadillo library.
- Extract the `Q` and `R` matrices from the QR decomposition.
- Solve the linear system `R * coef = Q.t() * Y` to estimate the VAR model coefficients `coef`.
- The function returns a matrix containing the estimated autoregressive and cross-regression coefficients of the VAR model.

Value

Matrix of estimated autoregressive and cross-regression coefficients.

Author(s)

Ivan Jacob Agaloos Pesigan

See Also

The `qr_econ` function from the Armadillo library for QR decomposition.

Other Fitting Autoregressive Model Functions: `FitVARLassoSearch()`, `FitVARLasso()`, `LambdaSeq()`, `OrigScale()`, `PBootVARLasso()`, `PBootVAROLS()`, `SearchVARLasso()`, `StdMat()`

Examples

```
Y <- dat_demo_yx$Y
X <- dat_demo_yx$X
FitVAROLS(Y = Y, X = X)
```

LambdaSeq *Function to generate the sequence of lambdas*

Description

Function to generate the sequence of lambdas

Usage

```
LambdaSeq(Y, X, n_lambdas)
```

Arguments

Y	Numeric matrix. Matrix of dependent variables (Y).
X	Numeric matrix. Matrix of predictors (X).
n_lambdas	Integer. Number of lambdas to generate.

Value

Returns a vector of lambdas.

Author(s)

Ivan Jacob Agaloos Pesigan

See Also

Other Fitting Autoregressive Model Functions: `FitVARLassoSearch()`, `FitVARLasso()`, `FitVAROLS()`, `OrigScale()`, `PBootVARLasso()`, `PBootVAROLS()`, `SearchVARLasso()`, `StdMat()`

Examples

```
Ystd <- StdMat(dat_demo_yx$Y)
Xstd <- StdMat(dat_demo_yx$X[, -1])
LambdaSeq(Y = Ystd, X = Xstd, n_lambdas = 100)
```

`OrigScale` *Return Standardized Estimates to the Original Scale*

Description

Return Standardized Estimates to the Original Scale

Usage

```
OrigScale(coef_std, Y, X)
```

Arguments

`coef_std`	Numeric matrix. Standardized estimates of the autoregression and cross regression coefficients.
`Y`	Numeric matrix. Matrix of dependent variables (Y).
`X`	Numeric matrix. Matrix of predictors (X).

Author(s)

Ivan Jacob Agaloos Pesigan

See Also

Other Fitting Autoregressive Model Functions: `FitVARLassoSearch()`, `FitVARLasso()`, `FitVAROLS()`, `LambdaSeq()`, `PBootVARLasso()`, `PBootVAROLS()`, `SearchVARLasso()`, `StdMat()`

Examples

```
Y <- dat_demo_yx$Y
X <- dat_demo_yx$X[, -1]
Ystd <- StdMat(Y)
Xstd <- StdMat(X)
coef_std <- FitVAROLS(Y = Ystd, X = Xstd)
FitVAROLS(Y = Y, X = X)
OrigScale(coef_std = coef_std, Y = Y, X = X)
```

PBootCI *Parametric Bootstrap Confidence Intervals*

Description

Parametric Bootstrap Confidence Intervals

Usage

```
PBootCI(x, alpha = 0.05)
```

Arguments

x	Numeric matrix. Output of `PBootVAROLS()`.
alpha	Numeric. Significance level.

Value

A list with two elements, namely `ll` for the lower limit and `ul` for the upper limit.

Author(s)

Ivan Jacob Agaloos Pesigan

See Also

Other Simulation of Autoregressive Data Functions: `PBootSE()`, `SelectVARLasso()`, `SimVAR()`, `YX()`

Examples

```
set.seed(42)
system.time(pb <- PBootVAROLS(data = dat_demo, p = 2, B = 10, burn_in = 20))
pb$est
PBootCI(pb)
system.time(pb <- PBootVARLasso(
  data = dat_demo, p = 2, B = 10, burn_in = 20,
  n_lambdas = 100, crit = "ebic", max_iter = 1000, tol = 1e-5
))
pb$est
PBootCI(pb)
```

PBootSE *Parametric Bootstrap Standard Errors*

Description

Parametric Bootstrap Standard Errors

Usage

```
PBootSE(x)
```

Arguments

x	Numeric matrix. Output of `PBootVAROLS()`.

Value

A matrix of standard error.

Author(s)

Ivan Jacob Agaloos Pesigan

See Also

Other Simulation of Autoregressive Data Functions: `PBootCI()`, `SelectVARLasso()`, `SimVAR()`, `YX()`

Examples

```
set.seed(42)
system.time(pb <- PBootVAROLS(data = dat_demo, p = 2, B = 10, burn_in = 20))
pb$est
PBootSE(pb)
system.time(pb <- PBootVARLasso(
  data = dat_demo, p = 2, B = 10, burn_in = 20,
  n_lambdas = 100, crit = "ebic", max_iter = 1000, tol = 1e-5
))
pb$est
PBootSE(pb)
```

PBootVARLasso — *Parametric Bootstrap for the Vector Autoregressive Model Using Lasso Regularization*

Description

Parametric Bootstrap for the Vector Autoregressive Model Using Lasso Regularization

Usage

```
PBootVARLasso(data, p, B, burn_in, n_lambdas, crit, max_iter, tol)
```

Arguments

`data`	Numeric matrix. The time series data with dimensions `t` by `k`, where `t` is the number of observations and `k` is the number of variables.
`p`	Integer. The order of the VAR model (number of lags).
`B`	Integer. Number of bootstrap samples to generate.
`burn_in`	Integer. Number of burn-in observations to exclude before returning the results in the simulation step.
`n_lambdas`	Integer. Number of lambdas to generate.
`crit`	Character string. Information criteria to use. Valid values include `"aic"`, `"bic"`, and `"ebic"`.
`max_iter`	Integer. The maximum number of iterations for the coordinate descent algorithm (e.g., `max_iter = 10000`).
`tol`	Numeric. Convergence tolerance. The algorithm stops when the change in coefficients between iterations is below this tolerance (e.g., `tol = 1e-5`).

Value

List containing the estimates (`est`) and bootstrap estimates (`boot`).

Author(s)

Ivan Jacob Agaloos Pesigan

See Also

Other Fitting Autoregressive Model Functions: `FitVARLassoSearch()`, `FitVARLasso()`, `FitVAROLS()`, `LambdaSeq()`, `OrigScale()`, `PBootVAROLS()`, `SearchVARLasso()`, `StdMat()`

Examples

```
pb <- PBootVARLasso(data = dat_demo, p = 2, B = 10, burn_in = 20,
  n_lambdas = 100, crit = "ebic", max_iter = 1000, tol = 1e-5)
str(pb)
```

`PBootVAROLS`	*Parametric Bootstrap for the Vector Autoregressive Model Using Ordinary Least Squares*

Description

Parametric Bootstrap for the Vector Autoregressive Model Using Ordinary Least Squares

Usage

```
PBootVAROLS(data, p, B, burn_in)
```

Arguments

`data`	Numeric matrix. The time series data with dimensions `t` by `k`, where `t` is the number of observations and `k` is the number of variables.
`p`	Integer. The order of the VAR model (number of lags).
`B`	Integer. Number of bootstrap samples to generate.
`burn_in`	Integer. Number of burn-in observations to exclude before returning the results in the simulation step.

Value

List containing the estimates (`est`) and bootstrap estimates (`boot`).

Author(s)

Ivan Jacob Agaloos Pesigan

See Also

Other Fitting Autoregressive Model Functions: `FitVARLassoSearch()`, `FitVARLasso()`, `FitVAROLS()`, `LambdaSeq()`, `OrigScale()`, `PBootVARLasso()`, `SearchVARLasso()`, `StdMat()`

Examples

```
pb <- PBootVAROLS(data = dat_demo, p = 2, B = 10, burn_in = 20)
str(pb)
```

SearchVARLasso *Compute AIC, BIC, and EBIC for Lasso Regularization*

Description

This function computes the Akaike Information Criterion (AIC), Bayesian Information Criterion (BIC), and Extended Bayesian Information Criterion (EBIC) for a given matrix of predictors `X`, a matrix of outcomes `Y`, and a vector of lambda hyperparameters for Lasso regularization.

Usage

```
SearchVARLasso(Ystd, Xstd, lambdas, max_iter, tol)
```

Arguments

`Ystd`	Numeric matrix. Matrix of standardized dependent variables (Y).
`Xstd`	Numeric matrix. Matrix of standardized predictors (X).
`lambdas`	Numeric vector. Vector of lambda hyperparameters for Lasso regularization.
`max_iter`	Integer. The maximum number of iterations for the coordinate descent algorithm (e.g., `max_iter = 10000`).
`tol`	Numeric. Convergence tolerance. The algorithm stops when the change in coefficients between iterations is below this tolerance (e.g., `tol = 1e-5`).

Value

List containing two elements:

- Element 1: Matrix with columns for lambda, AIC, BIC, and EBIC values.
- Element 2: List of matrices containing the estimated autoregressive and cross-regression coefficients for each lambda.

Author(s)

Ivan Jacob Agaloos Pesigan

See Also

Other Fitting Autoregressive Model Functions: `FitVARLassoSearch()`, `FitVARLasso()`, `FitVAROLS()`, `LambdaSeq()`, `OrigScale()`, `PBootVARLasso()`, `PBootVAROLS()`, `StdMat()`

Examples

```
Ystd <- StdMat(dat_demo_yx$Y)
Xstd <- StdMat(dat_demo_yx$X[, -1])
lambdas <- 10^seq(-5, 5, length.out = 100)
search <- SearchVARLasso(Ystd = Ystd, Xstd = Xstd, lambdas = lambdas,
  max_iter = 10000, tol = 1e-5)
plot(x = 1:nrow(search$criteria), y = search$criteria[, 4],
  type = "b", xlab = "lambda", ylab = "EBIC")
```

`SelectVARLasso` *Select the Lasso Estimates from the Grid Search*

Description

Select the Lasso Estimates from the Grid Search

Usage

```
SelectVARLasso(search, crit = "ebic")
```

Arguments

`search`	Object. Output of the `SearchVARLasso()` function.
`crit`	Character string. Information criteria to use. Valid values include `"aic"`, `"bic"`, and `"ebic"`.

Value

Returns the Lasso estimates of autoregression and cross regression coefficients.

Author(s)

Ivan Jacob Agaloos Pesigan

See Also

Other Simulation of Autoregressive Data Functions: `PBootCI()`, `PBootSE()`, `SimVAR()`, `YX()`

Examples

```
Ystd <- StdMat(dat_demo_yx$Y)
Xstd <- StdMat(dat_demo_yx$X[, -1])
lambdas <- 10^seq(-5, 5, length.out = 100)
search <- SearchVARLasso(
  Ystd = Ystd, Xstd = Xstd, lambdas = lambdas,
  max_iter = 10000, tol = 1e-5
)
SelectVARLasso(search, crit = "ebic")
```

SimVAR *Simulate Data from a Vector Autoregressive (VAR) Model*

Description

This function generates synthetic time series data from a Vector Autoregressive (VAR) model.

Usage

```
SimVAR(time, burn_in, constant, coef, chol_cov)
```

Arguments

`time`	Integer. Number of time points to simulate.
`burn_in`	Integer. Number of burn-in observations to exclude before returning the results.
`constant`	Numeric vector. The constant term vector of length `k`, where `k` is the number of variables.
`coef`	Numeric matrix. Coefficient matrix with dimensions `k` by `(k * p)`. Each `k` by `k` block corresponds to the coefficient matrix for a particular lag.
`chol_cov`	Numeric matrix. The Cholesky decomposition of the covariance matrix of the multivariate normal noise. It should have dimensions `k` by `k`.

Details

The `SimVAR()` function generates synthetic time series data from a Vector Autoregressive (VAR) model. The VAR model is defined by the constant term `constant`, the coefficient matrix `coef`, and the Cholesky decomposition of the covariance matrix of the multivariate normal process noise `chol_cov`. The generated time series data follows a VAR(p) process, where `p` is the number of lags specified by the size of `coef`. The generated data includes a burn-in period, which is excluded before returning the results.

The steps involved in generating the VAR time series data are as follows:

- Extract the number of variables `k` and the number of lags `p` from the input.
- Create a matrix `data` of size `k` by (`time + burn_in`) to store the generated VAR time series data.
- Set the initial values of the matrix `data` using the constant term `constant`.
- For each time point starting from the `p`-th time point to `time + burn_in - 1`:
  - Generate a vector of random noise from a multivariate normal distribution with mean 0 and covariance matrix `chol_cov`.
  - Generate the VAR time series values for each variable `j` at time `t` using the formula:

$$Y_{tj} = constant_j + \sum_{l=1}^{p} \sum_{m=1}^{k} (coef_{jm} * Y_{im}) + \text{noise}_j$$

where Y_{tj} is the value of variable `j` at time `t`, $constant_j$ is the constant term for variable `j`, $coef_{jm}$ are the coefficients for variable `j` from lagged variables up to order `p`, Y_{tm} are the lagged values of variable `m` up to order `p` at time `t`, and $noise_j$ is the element `j` from the generated vector of random process noise.

- Transpose the matrix `data` and return only the required time period after the burn-in period, which is from column `burn_in` to column `time + burn_in - 1`.

Value

Numeric matrix containing the simulated time series data with dimensions `k` by `time`, where `k` is the number of variables and `time` is the number of observations.

Author(s)

Ivan Jacob Agaloos Pesigan

See Also

Other Simulation of Autoregressive Data Functions: `PBootCI()`, `PBootSE()`, `SelectVARLasso()`, `YX()`

Examples

```
set.seed(42)
time <- 50L
burn_in <- 10L
k <- 3
p <- 2
constant <- c(1, 1, 1)
coef <- matrix(
  data = c(
    0.4, 0.0, 0.0, 0.1, 0.0, 0.0,
    0.0, 0.5, 0.0, 0.0, 0.2, 0.0,
    0.0, 0.0, 0.6, 0.0, 0.0, 0.3
  ),
  nrow = k,
  byrow = TRUE
)
chol_cov <- chol(diag(3))
y <- SimVAR(
  time = time,
  burn_in = burn_in,
  constant = constant,
  coef = coef,
  chol_cov = chol_cov
)
head(y)
```

`StdMat` *Standardize Matrix*

Description

This function standardizes the given matrix by centering the columns and scaling them to have unit variance.

Usage

```
StdMat(X)
```

Arguments

`X`	Numeric matrix. The matrix to be standardized.

Value

Numeric matrix with standardized values.

Author(s)

Ivan Jacob Agaloos Pesigan

See Also

Other Fitting Autoregressive Model Functions: `FitVARLassoSearch()`, `FitVARLasso()`, `FitVAROLS()`, `LambdaSeq()`, `OrigScale()`, `PBootVARLasso()`, `PBootVAROLS()`, `SearchVARLasso()`

Examples

```
std <- StdMat(dat_demo)
colMeans(std)
var(std)
```

`YX` *Create Y and X Matrices*

Description

This function creates the dependent variable (Y) and predictor variable (X) matrices.

Usage

```
YX(data, p)
```

Arguments

`data`	Numeric matrix. The time series data with dimensions `t` by `k`, where `t` is the number of observations and `k` is the number of variables.
`p`	Integer. The order of the VAR model (number of lags).

Details

The `YX()` function creates the `Y` and `X` matrices required for fitting a Vector Autoregressive (VAR) model. Given the input `data` matrix with dimensions `t` by `k`, where `t` is the number of observations and `k` is the number of variables, and the order of the VAR model `p` (number of lags), the function constructs lagged predictor matrix `X` and the dependent variable matrix `Y`.

The steps involved in creating the `Y` and `X` matrices are as follows:

- Determine the number of observations `t` and the number of variables `k` from the input data matrix.
- Create matrices `X` and `Y` to store lagged variables and the dependent variable, respectively.
- Populate the matrices `X` and `Y` with the appropriate lagged data. The predictors matrix `X` contains a column of ones and the lagged values of the dependent variables, while the dependent variable matrix `Y` contains the original values of the dependent variables.
- The function returns a list containing the `Y` and `X` matrices, which can be used for further analysis and estimation of the VAR model parameters.

Value

List containing the dependent variable (Y) and predictor variable (X) matrices. Note that the resulting matrices will have `t - p` rows.

Author(s)

Ivan Jacob Agaloos Pesigan

See Also

The `SimVAR()` function for simulating time series data from a VAR model.

Other Simulation of Autoregressive Data Functions: `PBootCI()`, `PBootSE()`, `SelectVARLasso()`, `SimVAR()`

Examples

```
set.seed(42)
time <- 50L
burn_in <- 10L
k <- 3
p <- 2
constant <- c(1, 1, 1)
coef <- matrix(
  data = c(
    0.4, 0.0, 0.0, 0.1, 0.0, 0.0,
    0.0, 0.5, 0.0, 0.0, 0.2, 0.0,
    0.0, 0.0, 0.6, 0.0, 0.0, 0.3
  ),
  nrow = k,
  byrow = TRUE
)
chol_cov <- chol(diag(3))
y <- SimVAR(
  time = time,
  burn_in = burn_in,
  constant = constant,
  coef = coef,
  chol_cov = chol_cov
)
yx <- YX(data = y, p = 2)
str(yx)
```

Index