Package 'MARX'

April 24, 2018

Title Simulation, Estimation, Model Selection and Forecasting for MARX Models

Version 0.2

Date 2018-04-24

Author Sean Telg [aut, cre, cph],
Alain Hecq [ctb],
Lenard Lieb [ctb]

Maintainer Sean Telg <j.telg@maastrichtuniversity.nl>

Description Simulate, estimate (by t-MLE), select and forecast mixed causal-noncausal autoregressive models with possibly exogenous regressors, using methods proposed in Lanne and Saikkonen (2011) <doi:10.2202/1941-1928.1080> and Hecq et al. (2016) <doi:10.15609/annaeconstat2009.123-124.0307>.

Imports matlab, fBasics, tseries, stabledist, metRology

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2018-04-24 18:44:32 UTC

R topics documented:

aic 2
arx.ls 3
bic 4
commodity 4
companion.form 5
compute.MA 6
forecast.marx 6
hq 7

inference 8
ll.max 9
marx 10
marx.t 11
mixed 12
pseudo 13
regressor.matrix 14
selection.lag 15
selection.lag.lead 16
sim.marx 16

Index **18**

`aic` *The Akaike information criterion (AIC) function*

Description

This function allows you to calculate the Akaike information criteria (AIC) for ARX models.

Usage

```
aic(y, x, p_max)
```

Arguments

`y`	Data vector of time series observations.
`x`	Matrix of data (every column represents one time series). Specify NULL or "not" if not wanted.
`p_max`	Maximum number of autoregressive terms to be included.

Value

`p`	Lag order chosen by AIC.
`values`	Vector containing values AIC for p = 0 up to p_max.

Author(s)

Sean Telg

Examples

```
data <- sim.marx(c('t',1,1), c('t',1,1),100,0.5,0.4,0.3)
aic(data$y, data$x,8)
```

arx.ls *The ARX estimation by OLS function*

Description

This function allows you to estimate ARX models by ordinary least squares (OLS).

Usage

```
arx.ls(y, x, p)
```

Arguments

y	Data vector of time series observations.
x	Matrix of data (every column represents one time series). Specify NULL or "not" if not wanted.
p	Number of autoregressive terms to be included.

Value

coefficients	Vector of estimated coefficients.
coef.auto	Vector of estimated autoregressive parameters.
coef.exo	Vector of estimated exogenous parameters.
mse	Mean squared error.
residuals	Residuals.
loglikelihood	Value of the loglikelihood.
fitted.values	Fitted values.
df	Degrees of freedom.
vcov	Variance-covariance matrix of residuals.

Author(s)

Sean Telg

Examples

```
data <- sim.marx(c('t',3,1),c('t',1,1),100,0.5,0.4,0.3)
arx.ls(data$y,data$x,2)
```

bic — *The Bayesian/Schwarz information criterion (BIC) function*

Description

This function allows you to calculate the Bayesian/Schwarz information criteria (BIC) for ARX models.

Usage

```
bic(y, x, p_max)
```

Arguments

y	Data vector of time series observations.
x	Matrix of data (every column represents one time series). Specify NULL or "not" if not wanted.
p_max	Maximum number of autoregressive terms to be included.

Value

p	Lag order chosen by BIC.
values	Vector containing values BIc for p = 0 up to p_max.

Author(s)

Sean Telg

Examples

```
data <- sim.marx(c('t',1,1), c('t',1,1),100,0.5,0.4,0.3)
bic(data$y, data$x,8)
```

commodity — *Data: Monthly growth rates of commodity prices, exchange rate and industrial production index.*

Description

Monthly growth rates of commodity prices, exchange rate and industrial production index from February 1980 until October 2010. Levels of these series can be downloaded from IMF and Federal Reserve Bank of St. Louis.

Usage

```
data("commodity")
```

Format

A data frame with 441 observations on the following 8 variables.

`X_date_` a vector with dates

`dlnbev` a numeric vector

`dlnind` a numeric vector

`dlnrawm` a numeric vector

`dlnmeta` a numeric vector

`dlnoil` a numeric vector

`dlnipi` a numeric vector

`dlnex` a numeric vector

Source

IMF Primary Commodity Prices (http://www.imf.org/external/np/res/commod/index.aspx) and Federal Reserve Bank of St. Louis (https://fred.stlouisfed.org).

Examples

```
data(dataset)
```

---

companion.form *Companion form function*

Description

This function allows you to compute a companion form matrix in order to check the stability of causal and noncausal part of the ARX model.

Usage

```
companion.form(pol)
```

Arguments

`pol`	Coefficient vector. If polynomial is 1 - ax - bx^2, coefficient vector is c(a, b).

Value

`C`	Companion matrix C.

Author(s)

Sean Telg

Examples

```
pol <- c(0.3,0.4)
C <- companion.form(pol)
```

`compute.MA` *Coefficients of the moving average representation function*

Description

This function allows you to invert a polynomial (either the causal or the noncausal one) and output the corresponding coefficients of the moving average representation.

Usage

```
compute.MA(pol, M)
```

Arguments

`pol`	Coefficient vector. If polynomial is 1 - ax - bx^2, coefficient vector is c(a, b).
`M`	Truncation value M (how many MA coefficients should be computed?).

Value

`psi`	Vector containing coefficients of the moving average representation.

Author(s)

Sean Telg

Examples

```
pol <- c(0.3,0.4)
psi <- companion.form(pol)
```

`forecast.marx` *Forecasting function for the MARX model*

Description

This function allows you to forecast with the mixed causal-noncausal model with possibly exogenous regressors.

Usage

```
forecast.marx(y, X, p_C, p_NC, X.for, h, M, N)
```

Arguments

y	Data vector y.
X	(optional) Matrix with data (column represent a series).
p_C	Number of lags (causal order).
p_NC	Number of leads (noncausal order).
X.for	(optional) Matrix with forecasted values for X (column represents series).
h	Forecast horizon h.
M	(optional) Truncation value M for MA representation. Default value: 50.
N	(optional) Number of simulations to forecast noncausal component. Default: 10,000.

Value

y.for	Vector containing forecasted values for y.

Author(s)

Sean Telg

Examples

```
## Forecasting MAR(0,1) model 4-periods ahead for lnbev (from dataset)
data <- MARX::dataset[,2]
y.for <- forecast.marx(y=data, p_C=0, p_NC=1, h=4, M=50, N=1000)
```

---

hq *The Hannan-Quinn (HQ) information criterion function*

Description

This function allows you to calculate the Hannan-Quinn (HQ) information criteria for ARX models.

Usage

```
hq(y, x, p_max)
```

Arguments

y	Data vector of time series observations.
x	Matrix of data (every column represents one time series). Specify NULL or "not" if not wanted.
p_max	Maximum number of autoregressive terms to be included.

Value

p	Lag order chosen by HQ.
values	Vector containing values HQ for p = 0 up to p_max.

Author(s)

Sean Telg

Examples

```
data <- sim.marx(c('t',1,1), c('t',1,1),100,0.5,0.4,0.3)
hq(data$y, data$x,8)
```

---

inference *Asymptotic inference for the MARX function*

Description

This function allows you to calculate standard errors and confidence intervals for parameters of the MARX model.

Usage

```
inference(y, x, B_C, B_NC, B_x, IC, sig, df, sig_level)
```

Arguments

y	Data vector of time series observations.
x	Matrix of data (every column represents one time series). Specify NULL or "not" if not wanted.
B_C	Estimated causal parameters of the MARX.
B_NC	Estimated noncausal parameters of the MARX.
B_x	Estimated parameters of the exogenous variables in the MARX.
IC	Estimated intercept.
sig	Estimated scale parameter of the assumed underlying Student-t distribution of the residuals.
df	Estimated degrees of freedom of the assumed underlying Student-t distribution of the residuals.
sig_level	Significance level for the construction of inference.

Value

CI.c	Confidence intervals for causal parameters.
CI.nc	Confidence intervals for noncausal parameters.
CI.exo	Confidence intervals for exogenous parameters.
CI.int	Confidence interval for intercept.
se.c	Standard errors of causal parameters.
se.nc	Standard errors of noncausal parameters.
se.exo	Standard errors of exogenous parameters.
se.int	Standard error of intercept.

Author(s)

Sean Telg

Examples

```
data <- sim.marx(c('t',1,1), c('t',1,1),100,0.5,0.4,0.3)
y <- data$y
x <- data$x
res <- marx.t(y,x,1,1)
inference(y,x,res$coef.c,res$coef.nc,res$coef.exo,res$coef.int,res$scale,res$df,0.05)
```

---

ll.max *The value of the t-log-likelihood for MARX function*

Description

This function allows you to determine the value of the t-log-likelihood for the MARX model.

Usage

```
ll.max(params, y, x, p_C, p_NC)
```

Arguments

params	List of parameters.
y	Data vector of time series observations.
x	Matrix of data (every column represents one time series). Specify NULL or "not" if not wanted.
p_C	Number of lags.
p_NC	Number of leads.

Value

`neg.loglikelihood`
: Minus the loglikelihood.

Author(s)

Sean Telg

Examples

```
data <- sim.marx(c('t',1,1), c('t',1,1),100,0.5,0.4,0.3)
y <- data$y
x <- data$x
p_C <- 1
p_NC <- 1
params <- c(0.5,0.4,0.3,0,1,1)
ll.max(params,y,x,p_C,p_NC)
```

---

marx *The MARX function*

Description

This interface-based function allows you to perform model selection for MARX models based on information criteria.

Usage

```
marx(y, x, p_max, sig_level, p_C, p_NC)
```

Arguments

`y`	Data vector of time series observations.
`x`	Matrix of data (every column represents one time series). Specify NULL or "not" if not wanted.
`p_max`	Maximum number of autoregressive parameters (leads + lags) to be included.
`sig_level`	Significance level for the construction of inference.
`p_C`	Number of lags (if not specified by the user a model selection procedure is used to determine the number of lags).
`p_NC`	Number of leads (if not specified by the user a model selection procedure is used to determine the number of leads).

Details

Mixed causal-noncausal autoregressions with exogenous regressors.

Value

The function returns the values of the information criteria for the pseudo-causal models. The user is asked to choose a value for "p". Extensive output for the MARX(r,s,q) model (with p = r + s) which maximizes the log-likelihood is reported.

Author(s)

Sean Telg

Examples

```
data <- sim.marx(c('t',1,1), c('t',1,1),100,0.5,0.4,0.3)
p_max <- 8
sig_level <- 0.05
marx(data$y, data$x, p_max, sig_level,1,1) ## p_C and p_NC chosen to be 1: MARX(1,1,1) output.
marx(data$y, NULL, p_max,sig_level,1,1) ## MAR(1,1), no exogenous variable specified.
```

---

marx.t *The estimation of the MARX model by t-MLE function*

Description

This function allows you to estimate the MARX model by t-MLE.

Usage

```
marx.t(y, x, p_C, p_NC, params0)
```

Arguments

y	Data vector of time series observations.
x	Matrix of data (every column represents one time series). Specify NULL or "not" if not wanted.
p_C	Number of lags.
p_NC	Number of leads.
params0	Starting values for the parameters to be estimated (both model and distributional parameters).

Value

coef.c	Estimated causal coefficients.
coef.nc	Estimated noncausal coefficients.
coef.exo	Estimated exogenous coefficients.
coef.int	Estimated intercept.
scale	Estimated scale parameter.

df	Estimated degrees of freedom.
residuals	Residuals.
se.dist	Standard errors of the distributional parameters.

Author(s)

Sean Telg

Examples

```
data <- sim.marx(c('t',3,1),c('t',3,1),100,0.5,0.4,0.3)
marx.t(data$y,data$x,1,1)
```

---

mixed *The MARX estimation function*

Description

This function allows you to estimate mixed causal-noncausal MARX models by t-MLE (compatible with most functions in lm() class).

Usage

```
mixed(y, x, p_C, p_NC)

## Default S3 method:
mixed(y, x, p_C, p_NC)

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

## S3 method for class 'mixed'
summary(object, ...)
```

Arguments

y	Data vector of time series observations.
x	Matrix of data (every column represents one time series). Specify NULL or "not" if not wanted.
p_C	Number of lags to be included.
p_NC	Number of leads to be included.
...	Other parameters.
object	An object of the class "mixed".

Value

An object of class "mixed" is a list containing the following components:

coefficients	Vector of estimated coefficients.
se	Standard errors of estimated coefficients.
df.residual	Degrees of freedom residuals.
residuals	Residuals.
fitted.values	Fitted values.
order	Vector containing (r,s,q), i.e. causal order r, noncausal order s, number of exogenous regressors q.

Examples

```
data <- sim.marx(c('t',1,1), c('t',1,1),100,0.5,0.4,0.3)
object <- mixed(data$y, data$x, 1, 1)
class(object) <- "mixed"
summary(object)
```

---

pseudo *The pseudo-causal model function*

Description

This function allows you to estimate pseudo-causal ARX models by OLS (compatible with most functions in lm() class).

Usage

```
pseudo(y, x, p)

## Default S3 method:
pseudo(y, x, p)

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

## S3 method for class 'pseudo'
summary(object, ...)
```

Arguments

y	Data vector of time series observations.
x	Matrix of data (every column represents one time series). Specify NULL or "not" if not wanted.
p	Number of lags to be included.
...	Other arguments
object	An object of the class "pseudo"

Value

An object of class "pseudo" is a list containing the following components:

coefficients	Vector of estimated coefficients.
coef.auto	Vector of estimated autoregressive parameters.
coef.exo	Vector of estimated exogenous parameters.
mse	Mean squared error.
residuals	Residuals.
loglikelihood	Value of the loglikelihood.
fitted.values	Fitted values.
df	Degrees of freedom.
vcov	Variance-covariance matrix of residuals.

Examples

```
data <- sim.marx(c('t',1,1), c('t',1,1),100,0.5,0.4,0.3)
object <- pseudo(data$y, data$x, 2)
class(object) <- "pseudo"
summary(object)
```

---

regressor.matrix *The regressor matrix function*

Description

This function allows you to create a regressor matrix.

Usage

```
regressor.matrix(y, x, p)
```

Arguments

y	Data vector of time series observations.
x	Matrix of data (every column represents one time series). Specify NULL or "not" if not wanted.
p	Number of autoregressive terms to be included.

Value

Z	Regressor matrix

Author(s)

Sean Telg

Examples

```
data <- sim.marx(c('t',3,1),c('t',1,1),100,0.5,0.4,0.3)
regressor.matrix(data$y, data$x, 2)
```

`selection.lag` *The model selection for pseudo-ARX function*

Description

This function allows you to calculate AIC, BIC, HQ for pseudo-ARX models.

Usage

```
selection.lag(y, x, p_max)
```

Arguments

`y`	Data vector of time series observations.
`x`	Matrix of data (every column represents one time series). Specify NULL or "not" if not wanted.
`p_max`	Maximum number of autoregressive terms to be included.

Value

`bic`	Vector containing values BIC for p=0 up to p_max.
`aic`	Vector containing values AIC for p=0 up to p_max.
`hq`	vector containing values HQ for p=0 up to p_max.

Author(s)

Sean Telg

Examples

```
data <- sim.marx(c('t',1,1), c('t',1,1),100,0.5,0.4,0.3)
selection.lag(data$y,data$x,8)
```

selection.lag.lead *The lag-lead model selection for MARX function*

Description

This function allows you to determine the MARX model (for p = r + s) that maximizes the t-log-likelihood.

Usage

```
selection.lag.lead(y, x, p_pseudo)
```

Arguments

y	Data vector of time series observations.
x	Matrix of data (every column represents one time series). Specify NULL or "not" if not wanted.
p_pseudo	Number of autoregressive terms to be included in the pseudo-causal model.

Value

p.C	The number of lags selected.
p.NC	The number of leads selected.
loglikelihood	The value of the loglikelihood for all models with p = r + s.

Author(s)

Sean Telg

Examples

```
data <- sim.marx(c('t',3,1), c('t',3,1),100,0.5,0.4,0.3)
selection.lag.lead(data$y,data$x,2)
```

sim.marx *The simulation of MARX processes*

Description

This function allows you to simulate MARX processes based on different underlying distribution.

Usage

```
sim.marx(dist.eps, dist.x, obs, c_par, nc_par, exo_par)
```

Arguments

dist.eps	vector containing the error distribution and its parameters (options: t, normal, stable).
dist.x	vector containing the distribution of x and its parameters (options: t, normal, stable). Specify NULL or "not" if not wanted.
obs	Number of observations for simulated process.
c_par	vector of causal parameters.
nc_par	vector of noncausal parameters.
exo_par	Parameter of the exogenous variable.

Value

y	Simulated data y.
x	Simulated data x (exogenous variable).

Author(s)

Sean Telg

Examples

```
dist.eps <- c('t',1,1) ## t-distributed errors with 1 degree of freedom and scale parameter 1
dist.x   <- c('normal',0,1) ## standard normally distributed x variable
obs <- 100
c_par <- c(0.2,0.4)
nc_par <- 0.8
exo_par <- 0.5
sim.marx(dist.eps,dist.x,obs,c_par,nc_par,exo_par) ## Simulates a MARX(2,1,1) process
```

Index

∗Topic **causal-noncausal**
inference, 8
marx.t, 11
mixed, 12
selection.lag.lead, 16
∗Topic **datasets**
commodity, 4
∗Topic **estimation**
arx.ls, 3
marx, 10
marx.t, 11
mixed, 12
pseudo, 13
regressor.matrix, 14
∗Topic **forecasting**
forecast.marx, 6
∗Topic **inference**
inference, 8
∗Topic **optimization**
ll.max, 9
∗Topic **pseudo-causal**
arx.ls, 3
pseudo, 13
selection.lag, 15
∗Topic **selection**
aic, 2
bic, 4
hq, 7
marx, 10
selection.lag, 15
selection.lag.lead, 16
∗Topic **simulation**
sim.marx, 16
∗Topic **stability,**
companion.form, 5
compute.MA, 6
∗Topic **stationarity**
companion.form, 5
compute.MA, 6

aic, 2
arx.ls, 3

bic, 4

commodity, 4
companion.form, 5
compute.MA, 6

dataset *(*commodity*)*, 4

forecast.marx, 6

hq, 7

inference, 8

ll.max, 9

marx, 10
marx.t, 11
mixed, 12

print.mixed *(*mixed*)*, 12
print.pseudo *(*pseudo*)*, 13
pseudo, 13

regressor.matrix, 14

selection.lag, 15
selection.lag.lead, 16
sim.marx, 16
summary.mixed *(*mixed*)*, 12
summary.pseudo *(*pseudo*)*, 13