Package 'tseriesTARMA'

September 18, 2023

Title Analysis of Nonlinear Time Series Through TARMA Models

Version 0.3-4

Depends R (>= 3.5.0)

Suggests knitr, rmarkdown

Roxygen list(markdown = TRUE)

LazyData yes

LazyLoad yes

Encoding UTF-8

NeedsCompilation yes

Imports methods,
stats,
Rsolnp,
lbfgsb3c,
Matrix,
Rdpack,
mathjaxr,
rugarch,
zoo

RdMacros Rdpack, mathjaxr

Description Routines for nonlinear time series analysis based on Threshold Autoregressive Moving Average models.
It provides functions and methods for: TARMA model fitting and forecasting, tests for threshold effects, unit-root tests based on TARMA models.

Maintainer Simone Giannerini <simone.giannerini@unibo.it>

License GPL (>= 2)

RoxygenNote 7.2.3

R topics documented:

`ACValues` *Andrews Tabulated Critical Values*

Description

The data is taken from Table 1 of (Andrews 2003), which provides asymptotic critical values for sup Wald, LM, and LR tests for parameter instability. Critical values are given for degrees of freedom $p = 1, \ldots, 20$. They can be used as asymptotic critical values for the following tests:

- `TAR.test`
- `TARMA.test`
- `TARMAGARCH.test`

Provided `pb = 1- pa`.

Usage

```
ACValues
```

Format

`ACValues`**:**
A matrix with 13 rows and 62 columns. The first two columns contain the parameters, whereas the remaining 60 columns contain 3 critical values (at level 10%, 5%, and 1%) for each value of the parameter p:

`pi` π_0 in the paper, corresponds to `pa`.
`lambda` λ in the paper, not relevant here.
`1-90` $p = 1$, critical level 10%.
`1-95` $p = 1$, critical level 5%.
`1-99` $p = 1$, critical level 1%. ...
`20-90` $p = 20$, critical level 10%.
`20-95` $p = 20$, critical level 5%.
`20-99` $p = 20$, critical level 1%.

Note that p are the degrees of freedom and correspond to the total number of tested parameter in the above tests.

Source

Andrews DWK (2003). "Tests for Parameter Instability and Structural Change with Unknown Change Point: A Corrigendum." *Econometrica*, **71**(1), 395-397. doi:10.1111/14680262.00405.

plot.tsfit *Plot from fitted/forecasted time series models.*

Description

Plots a time series model fit, possibly together with its forecast and forecast bands.

Usage

```
## S3 method for class 'tsfit'
plot(
  x,
  fit,
  plot.fit = TRUE,
  fore = NULL,
  lcols = list(series = "lightblue", fit = 4, pred = "red4", band = "red"),
  ptype = list(series = 20, fit = 1, pred = 16, band = 20),
  ltype = list(series = 1, fit = 1, pred = 1, band = 1),
  lwdt = 2,
  ...
)
```

Arguments

x	A time series
fit	A time series model fitted upon x, e.g. an object obtained with TARMA.fit
plot.fit	Logical. If TRUE adds the fitted values from the model.
fore	Forecast derived from fit, e.g. an object obtained with predict.TARMA if not null it adds the prediction together with its confidence bands.
lcols	List of colours for the plots.
ptype	List of point types (pch) for the plots.
ltype	List of line types (lty) for the plots.
lwdt	A common line width for the plots.
...	Additional graphical parameters.

Value

No return value, called for side effects

Author(s)

Simone Giannerini, <simone.giannerini@unibo.it>

Greta Goracci, <greta.goracci@unibz.it>

See Also

TARMA.fit and TARMA.fit2 for TARMA modelling. predict.TARMA for prediction and forecasting.

Examples

```
## a TARMA(1,1,1,1) model
set.seed(13)
x   <- TARMA.sim(n=200, phi1=c(0.5,-0.5), phi2=c(0.0,0.5), theta1=-0.5, theta2=0.7, d=1, thd=0.2)
fit1 <- TARMA.fit(x,tar1.lags = 1, tar2.lags = 1, tma1.lags = 1, tma2.lags = 1, d=1,
         estimate.thd = FALSE, threshold=0.2)
xp1  <- predict(fit1,x,n.ahead=5)

# plots both the fitted and the forecast
plot.tsfit(x,fit=fit1,fore=xp1);grid();

# plots only the forecast
plot.tsfit(x,fit=fit1,plot.fit=FALSE,fore=xp1);grid();

# plots only the fitted
plot.tsfit(x,fit=fit1);grid();
```

`predict.TARMA` *Forecast from fitted TARMA models.*

Description

Forecasting with TARMA models

Usage

```
## S3 method for class 'TARMA'
predict(
  object,
  x,
  n.ahead = 0,
  n.sim = 1000,
  quant = c(0.05, 0.95),
  pred.matrix = FALSE,
  ...
)
```

Arguments

`object`	A `TARMA` fit upon `x`.
`x`	The fitted time series.
`n.ahead`	The number of steps ahead for which prediction is required.
`n.sim`	The number of Monte Carlo replications used to simulate the prediction density.
`quant`	Vector of quantiles (in the interval `[0, 1]`) to be computed upon the prediction density.
`pred.matrix`	Logical. if `TRUE` prints also the whole simulated prediction density for each prediction horizon from 1 to `n.ahead`.
`...`	Additional arguments.

Details

If n.ahead = 0 it gives the fitted values from the model. If the fit is from TARMA.fit2 and includes covariates, these are ignored.

Value

A list with components pred.matrix, pred, and pred.interval. The latter two are ts objects that contain the prediction and the quantiles of the prediction density, respectively. If pred.matrix = TRUE then the prediction density from which the quantiles are computed is also returned.

Author(s)

Simone Giannerini, <simone.giannerini@unibo.it>

Greta Goracci, <greta.goracci@unibo.it>

References

- Giannerini S, Goracci G (2021). "Estimating and Forecasting with TARMA models." University of Bologna.

See Also

TARMA.fit and TARMA.fit2 for TARMA modelling. plot.tsfit for plotting TARMA fits and forecasts.

Examples

```
## a TARMA(1,1,1,1) model
set.seed(13)
x1  <- TARMA.sim(n=200, phi1=c(0.5,-0.5), phi2=c(0.0,0.5), theta1=-0.5, theta2=0.7, d=1, thd=0.2)
fit1 <- TARMA.fit(x1, method='L-BFGS-B',tar1.lags = 1, tar2.lags = 1, tma1.lags = 1,
        tma2.lags = 1, d=1, threshold=0.2)
xp1  <- predict(fit1,x1,n.ahead=2)
xp1
```

---

print.TARMA *Methods for TARMA fits*

Description

Methods for TARMA fits

Usage

```
## S3 method for class 'TARMA'
print(x, digits = max(3L, getOption("digits") - 3L), se = TRUE, ...)

## S3 method for class 'TARMA'
coef(object, ...)

## S3 method for class 'TARMA'
vcov(object, ...)
```

```
## S3 method for class 'TARMA'
residuals(object, ...)
```

Arguments

x	A TARMA fit.
digits	Number of decimal digits for the output.
se	Logical. if TRUE (the default) prints also the standard errors.
...	Further parameters.
object	A TARMA fit.

Value

No return value, called for side effects

See Also

TARMA.fit and TARMA.fit2 for TARMA modelling, plot.tsfit for plotting TARMA fits and forecasts.

---

print.TARMAtest *Methods for TARMA tests*

Description

Methods for TARMA tests

Usage

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

Arguments

x	A TARMAtest object.
...	Further parameters passed to print.htest.

Details

Print method for TARMAtest objects. Prints the results using the method for class htest and adds critical values extracted from ACValues for the test for threshold nonlinearity and supLMQur for the unit root test against the TARMA alternative. Note that the bootstrap version of the tests also print the bootstrap p-value.

Value

No return value, called for side effects

Author(s)

Simone Giannerini, <simone.giannerini@unibo.it>

Greta Goracci, <greta.goracci@unibz.it>

See Also

print.htest

---

supLMQur *Tabulated Critical Values for the Unit Root IMA vs TARMA test*

Description

The data provides asymptotic null critical values for the unit root supLM test described in (Chan et al. 2024). They are used with the following tests:

- TARMAur.test
- TARMAur.test.B

Provided pb = 1- pa.

Usage

```
supLMQur
```

Format

supLMQur**:**

A 4-dimensional array that contains 4 critical values (at level 10%, 5%, 1%, 0.1%) for each combination of

pa Lower bound for the threshold range. From 0.01 to 0.4

th MA(1) parameter.

n Sample size.

Source

Chan K, Giannerini S, Goracci G, Tong H (2024). "Testing for threshold regulation in presence of measurement error." *Statistica Sinica*, **34**(3). https://doi.org/10.5705/ss.202022.0125.

TAR.test *AR versus TARMA supLM robust test for nonlinearity*

Description

Implements a heteroskedasticity robust supremum Lagrange Multiplier test for a AR specification versus a TAR specification. Includes the classic (non robust) AR versus TAR test.

Usage

```
TAR.test(x, pa = 0.25, pb = 0.75, ar.ord, d = 1)
```

Arguments

x	A univariate time series.
pa	Real number in [0,1]. Sets the lower limit for the threshold search to the 100*pa-th sample percentile. The default is 0.25
pb	Real number in [0,1]. Sets the upper limit for the threshold search to the 100*pb-th sample percentile. The default is 0.75
ar.ord	Order of the AR part.
d	Delay parameter. Defaults to 1.

Details

Implements a heteroskedasticity robust asymptotic supremum Lagrange Multiplier test to test an AR specification versus a TAR specification. This is an asymptotic test and the value of the test statistic has to be compared with the critical values tabulated in (Goracci et al. 2021) or (Andrews 2003). Both the non-robust supLM and the robust supLMh statistics are returned.

Value

An object of class TARMAtest with components:

statistic A named vector with the values of the classic supLM and robust supLMh statistics.

parameter A named vector: threshold is the value that maximises the Lagrange Multiplier values.

test.v Matrix of values of the LM statistic for each threshold given in thd.range.

thd.range Range of values of the threshold.

fit The null model: AR fit over x.

sigma2 Estimated innovation variance from the AR fit.

data.name A character string giving the name of the data.

prop Proportion of values of the series that fall in the lower regime.

p.value The p-value of the test. It is NULL for the asymptotic test.

method A character string indicating the type of test performed.

d The delay parameter.

pa Lower threshold quantile.

dfree Effective degrees of freedom. It is the number of tested parameters.

Author(s)

Simone Giannerini, <simone.giannerini@unibo.it>

Greta Goracci, <greta.goracci@unibz.it>

References

- Goracci G, Giannerini S, Chan K, Tong H (2023). "Testing for threshold effects in the TARMA framework." *Statistica Sinica*, **33**(3), 1879-1901. https://doi.org/10.5705/ss.202021.0120.
- Andrews DWK (2003). "Tests for Parameter Instability and Structural Change with Unknown Change Point: A Corrigendum." *Econometrica*, **71**(1), 395-397. doi:10.1111/1468-0262.00405.

See Also

TAR.test.B for the bootstrap version of the test. TARMA.test for the ARMA vs TARMA asymptotic version of the test, which includes also the AR vs TAR test, with different defaults. TARMAGARCH.test for the robust version of the ARMA vs TARMA test with respect to GARCH innovations. TARMA.sim to simulate from a TARMA process.

Examples

```
set.seed(123)
## a TAR(1,1) ---------------
x1  <- TARMA.sim(n=100, phi1=c(0.5,-0.5), phi2=c(0.0,0.8), theta1=0, theta2=0, d=1, thd=0.2)
TAR.test(x1, ar.ord=1, d=1)

## a AR(1)    ----------------
x2   <- arima.sim(n=100, model=list(order=c(1,0,0), ar=0.5))
TAR.test(x2, ar.ord=1, d=1)
```

TAR.test.B *AR versus TAR bootstrap supLM test for nonlinearity*

Description

Implements various bootstrap supremum Lagrange Multiplier tests for a AR specification versus a TAR specification.

Usage

```
TAR.test.B(
  x,
  B = 1000,
  pa = 0.25,
  pb = 0.75,
  ar.ord,
  d = 1,
  btype = c("iid", "wb.h", "wb.r", "wb.n"),
  ...
)
```

Arguments

x	A univariate time series.
B	Integer. Number of bootstrap resamples. Defaults to 1000.
pa	Real number in [0,1]. Sets the lower limit for the threshold search to the 100*pa-th sample percentile. The default is 0.25
pb	Real number in [0,1]. Sets the upper limit for the threshold search to the 100*pb-th sample percentile. The default is 0.75
ar.ord	Order of the AR part.
d	Delay parameter. Defaults to 1.
btype	Bootstrap type, can be one of 'iid','wb.h','wb.r','wb.n', see Details.
...	Additional arguments to be passed to arima.

Details

Implements the bootstrap version of TAR.test the supremum Lagrange Multiplier test to test an AR specification versus a TARMA specification. The option btype specifies the type of bootstrap as follows:

iid Residual iid bootstrap. See (Giannerini et al. 2022), (Giannerini et al. 2023).

wb.h Stochastic permutation of (Hansen 1996).

wb.r Residual wild bootstrap with Rademacher auxiliary distribution. See (Giannerini et al. 2022), (Giannerini et al. 2023).

wb.n Residual wild bootstrap with Normal auxiliary distribution. See (Giannerini et al. 2022), (Giannerini et al. 2023).

Value

A list of class htest with components:

statistic The value of the supLM statistic.

parameter A named vector: threshold is the value that maximises the Lagrange Multiplier values.

test.v Vector of values of the LM statistic for each threshold given in thd.range.

thd.range Range of values of the threshold.

fit The null model: AR fit over x.

sigma2 Estimated innovation variance from the AR fit.

data.name A character string giving the name of the data.

prop Proportion of values of the series that fall in the lower regime.

p.value The bootstrap p-value of the test.

method A character string indicating the type of test performed.

Tb The bootstrap null distribution.

Author(s)

Simone Giannerini, <simone.giannerini@unibo.it>

Greta Goracci, <greta.goracci@unibz.it>

References

- Giannerini S, Goracci G, Rahbek A (2022). "The validity of bootstrap testing in the threshold framework." doi:10.48550/ARXIV.2201.00028, https://arxiv.org/abs/2201.00028.
- Giannerini S, Goracci G, Rahbek A (2023). "The validity of bootstrap testing in the threshold framework." *Journal of Econometrics*, in press. https://doi.org/10.1016/j.jeconom.2023.01.004.
- Goracci G, Giannerini S, Chan K, Tong H (2023). "Testing for threshold effects in the TARMA framework." *Statistica Sinica*, **33**(3), 1879-1901. https://doi.org/10.5705/ss.202021.0120.
- Giannerini S, Goracci G (2021). "Estimating and Forecasting with TARMA models." University of Bologna.
- Hansen BE (1996). "Inference When a Nuisance Parameter Is Not Identified Under the Null Hypothesis." *Econometrica*, **64**(2), 413–430. ISSN 00129682, 14680262, https://doi.org/10.2307/2171789.

See Also

`TAR.test` for the heteroskedastic robust asymptotic test. `TARMAGARCH.test` for the robust version of the test with respect to GARCH innovations. `TARMA.sim` to simulate from a TARMA process.

Examples

```
## a TAR(1,1) where the threshold effect is on the AR parameters
set.seed(123)
x1 <- TARMA.sim(n=100, phi1=c(0.5,-0.5), phi2=c(0.0,0.8), theta1=0, theta2=0, d=1, thd=0.2)
TAR.test.B(x1, ar.ord=1, d=1)
TAR.test.B(x1, ar.ord=1, d=1, btype='wb.r')
TAR.test.B(x1, ar.ord=1, d=1, btype='wb.h')

## a AR(1)
x2 <- arima.sim(n=100, model=list(order = c(1,0,0),ar=0.5))
TAR.test.B(x2, ar.ord=1, d=1)
TAR.test.B(x2, ar.ord=1, d=1, btype='wb.r')
TAR.test.B(x2, ar.ord=1, d=1, btype='wb.h')
```

`TARMA.fit` *TARMA Modelling of Time Series*

Description

Implements a Least Squares fit of full subset two-regime `TARMA(p1,p2,q1,q2)` model to a univariate time series

Usage

```
TARMA.fit(
  x,
  tar1.lags = c(1),
  tar2.lags = c(1),
  tma1.lags = c(1),
```

```
  tma2.lags = c(1),
  estimate.thd = TRUE,
  threshold,
  d = 1,
  pa = 0.25,
  pb = 0.75,
  method = c("L-BFGS-B", "solnp", "lbfgsb3c", "robust"),
  alpha = 0,
  qu = c(0.05, 0.95),
  optim.control = list(trace = 0),
  ...
)
```

Arguments

x	A univariate time series.
tar1.lags	Vector of AR lags for the lower regime. It can be a subset of 1 ... p1 = max(tar1.lags).
tar2.lags	Vector of AR lags for the upper regime. It can be a subset of 1 ... p2 = max(tar2.lags).
tma1.lags	Vector of MA lags for the lower regime. It can be a subset of 1 ... q1 = max(tma1.lags).
tma2.lags	Vector of MA lags for the upper regime. It can be a subset of 1 ... q2 = max(tma2.lags).
estimate.thd	Logical. If TRUE estimates the threshold over the threshold range specified by pa and pb.
threshold	Threshold parameter. Used only if estimate.thd = FALSE.
d	Delay parameter. Defaults to 1.
pa	Real number in [0,1]. Sets the lower limit for the threshold search to the 100*pa-th sample percentile. The default is 0.25
pb	Real number in [0,1]. Sets the upper limit for the threshold search to the 100*pb-th sample percentile. The default is 0.75
method	Optimization/fitting method, can be one of "L-BFGS-B", "solnp", "lbfgsb3c", "robust".
alpha	Real positive number. Tuning parameter for robust estimation. Only used if method is "robust".
qu	Quantiles for initial trimmed estimation. Tuning parameter for robust estimation. Only used if method is "robust".
optim.control	List of control parameters for the optimization method.
...	Additional arguments for the optimization.

Details

Implements the Least Squares fit of the following two-regime TARMA(p1,p2,q1,q2) process:

$$X_t = \begin{cases} \phi_{1,0} + \sum_{i \in I_1} \phi_{1,i} X_{t-i} + \sum_{j \in M_1} \theta_{1,j} \varepsilon_{t-j} + \varepsilon_t & \text{if} X_{t-d} \leq \text{thd} \\ \\ \phi_{2,0} + \sum_{i \in I_2} \phi_{2,i} X_{t-i} + \sum_{j \in M_2} \theta_{2,j} \varepsilon_{t-j} + \varepsilon_t & \text{if} X_{t-d} > \text{thd} \end{cases}$$

where $\phi_{1,i}$ and $\phi_{2,i}$ are the TAR parameters for the lower and upper regime, respectively, and I1 = tar1.lags and I2 = tar2.lags are the corresponding vectors of TAR lags. $\theta_{1,j}$ and $\theta_{2,j}$ are the

TMA parameters and $j \in M_1, M_2$, where `M1 = tma1.lags` and `M2 = tma2.lags`, are the vectors of TMA lags.
The most demanding routines have been reimplemented in Fortran and dynamically loaded.

Value

A list of class `TARMA` with components:

- `fit` - List with the following components
  - `coef` - Vector of estimated parameters which can be extracted by the coef method.
  - `sigma2` - Estimated innovation variance.
  - `var.coef` - The estimated variance matrix of the coefficients coef, which can be extracted by the vcov method
  - `residuals` - Vector of residuals from the fit.
  - `nobs` - Effective sample size used for fitting the model.
- `se` - Standard errors for the parameters. Note that they are computed conditionally upon the threshold so that they are generally smaller than the true ones.
- `thd` - Estimated threshold.
- `aic` - Value of the AIC for the minimised least squares criterion over the threshold range.
- `bic` - Value of the BIC for the minimised least squares criterion over the threshold range.
- `rss` - Minimised value of the target function. Coincides with the residual sum of squares for ordinary least squares estimation.
- `rss.v` - Vector of values of the rss over the threshold range.
- `thd.range` - Vector of values of the threshold range.
- `d` - Delay parameter.
- `phi1` - Estimated AR parameters for the lower regime.
- `phi2` - Estimated AR parameters for the upper regime.
- `theta1` - Estimated MA parameters for the lower regime.
- `theta2` - Estimated MA parameters for the upper regime.
- `tlag1` - TAR lags for the lower regime
- `tlag2` - TAR lags for the upper regime
- `mlag1` - TMA lags for the lower regime
- `mlag2` - TMA lags for the upper regime
- `method` - Estimation method.
- `alpha` - Tuning parameter for robust estimation.
- `qu` - Tuning parameter for robust estimation.
- `call` - The matched call.
- `convergence` - Convergence code from the optimization routine.

Fitting methods

method has the following options:

L-BFGS-B Calls the corresponding method of optim. Linear ergodicity constraints are imposed.

solnp Calls the function solnp. It is a nonlinear optimization using augmented Lagrange method with linear and nonlinear inequality bounds. This allows to impose all the ergodicity constraints so that in theory it always return an ergodic solution. In practice the solution should be checked since this is a local solver and there is no guarantee that the minimum has been reached.

lbfgsb3c Calls the function lbfgsb3c in package lbfgsb3c. Improved version of the L-BFGS-B in optim.

robust Robust M-estimator of Ferrari and La Vecchia (Ferrari and La-Vecchia 2011). Based on the L-BFGS-B in optim and an additional iterative reweighted least squares step to estimate the robust weights. Uses the tuning parameters alpha and qu. Robust standard errors are derived from the sandwich estimator of the variance/covariance matrix of the estimates

Where possible, the ergodicity bounds are imposed to the optimization routines but there is no guarantee that the solution will be ergodic so that it is advisable to check the fitted parameters.

Author(s)

Simone Giannerini, <simone.giannerini@unibo.it>

Greta Goracci, <greta.goracci@unibz.it>

References

- Giannerini S, Goracci G (2021). "Estimating and Forecasting with TARMA models." University of Bologna.
- Chan K, Goracci G (2019). "On the Ergodicity of First-Order Threshold Autoregressive Moving-Average Processes." *J. Time Series Anal.*, **40**(2), 256-264.
- Goracci G, Ferrari D, Giannerini S, Ravazzolo F (2023). "Robust estimation for Threshold Autoregressive Moving-Average models." Free University of Bolzano, University of Bologna. doi:10.48550/ARXIV.2211.08205.
- Ferrari D, La-Vecchia D (2011). "On robust estimation via pseudo-additive information." *Biometrika*, **99**(1), 238-244. ISSN 0006-3444, doi:10.1093/biomet/asr061.

See Also

TARMA.fit2 for Maximum Likelihood estimation of TARMA models with common MA part. print.TARMA for print methods for TARMA fits. predict.TARMA for prediction and forecasting. plot.tsfit for plotting TARMA fits and forecasts.

Examples

```
## a TARMA(1,1,1,1) model
set.seed(13)
x   <- TARMA.sim(n=200, phi1=c(0.5,-0.5), phi2=c(0.0,0.5), theta1=-0.5, theta2=0.7, d=1, thd=0.2)
fit1 <- TARMA.fit(x,tar1.lags=1, tar2.lags=1, tma1.lags=1, tma2.lags=1, d=1)

## ------------------------------------------------------------------------
## In the following examples the threshold is fixed to speed up computations
```

```
## ------------------------------------------------------------------------

## ------------------------------------------------------------------------
## Least Squares fit
## ------------------------------------------------------------------------

set.seed(26)
n    <- 200
y   <- TARMA.sim(n=n, phi1=c(0.6,0.6), phi2=c(-1.0,0.4), theta1=-0.7, theta2=0.5, d=1, thd=0.2)

fit1 <- TARMA.fit(y,tar1.lags=1, tar2.lags=1, tma1.lags=1, tma2.lags=1, d=1,
       estimate.thd=FALSE, threshold=0.2)
fit1

## ------------------------------------------------------------------------
## Contaminating the data with one additive outlier
## ------------------------------------------------------------------------
x     <- y           # contaminated series
x[54] <- x[54] + 10

## ------------------------------------------------------------------------
## Compare the non-robust LS fit with the robust fit
## ------------------------------------------------------------------------

fitls  <- TARMA.fit(x,tar1.lags=1, tar2.lags=1, tma1.lags=1, tma2.lags=1, d=1,
            estimate.thd=FALSE, threshold=0.2)
fitrob <- TARMA.fit(x,tar1.lags=1, tar2.lags=1, tma1.lags=1, tma2.lags=1, d=1,
           method='robust',alpha=0.7,qu=c(0.1,0.95),estimate.thd = FALSE, threshold=0.2)

par.true <- c(0.6,0.6,-1,0.4,-0.7,0.5)
pnames   <- c("int.1", "ar1.1", "int.2", "ar2.1", "ma1.1", "ma2.1")
names(par.true) <- pnames

par.ls  <- round(fitls$fit$coef,2)  # Least Squares
par.rob <- round(fitrob$fit$coef,2) # robust

rbind(par.true,par.ls,par.rob)
```

`TARMA.fit2` *TARMA Modelling of Time Series*

Description

Maximum Likelihood fit of a two-regime `TARMA(p1,p2,q,q)` model with common MA parameters, possible common AR parameters and possible covariates.

Usage

```
TARMA.fit2(
  x,
  ar.lags = NULL,
  tar1.lags = c(1),
  tar2.lags = c(1),
  ma.ord = 1,
```

```
  sma.ord = 0L,
  period = NA,
  estimate.thd = TRUE,
  threshold,
  d = 1,
  pa = 0.25,
  pb = 0.75,
  thd.var = NULL,
  include.int = TRUE,
  x.reg = NULL,
  optim.control = list(),
  ...
)
```

Arguments

x	A univariate time series.
ar.lags	Vector of common AR lags. Defaults to NULL. It can be a subset of lags.
tar1.lags	Vector of AR lags for the lower regime. It can be a subset of 1 ... p1 = max(tar1.lags).
tar2.lags	Vector of AR lags for the upper regime. It can be a subset of 1 ... p2 = max(tar2.lags).
ma.ord	Order of the MA part (also called q below).
sma.ord	Order of the seasonal MA part (also called Q below).
period	Period of the seasonal MA part (also called s below).
estimate.thd	Logical. If TRUE estimates the threshold over the threshold range specified by pa and pb.
threshold	Threshold parameter. Used only if estimate.thd = FALSE.
d	Delay parameter. Defaults to 1.
pa	Real number in [0,1]. Sets the lower limit for the threshold search to the 100*pa-th sample percentile. The default is 0.25
pb	Real number in [0,1]. Sets the upper limit for the threshold search to the 100*pb-th sample percentile. The default is 0.75
thd.var	Optional exogenous threshold variable. If NULL it is set to lag(x,-d). If not NULL it has to be a ts object.
include.int	Logical. If TRUE includes the intercept terms in both regimes, a common intercept is included otherwise.
x.reg	Covariates to be included in the model. These are passed to arima. If they are not ts objects they must have the same length as x.
optim.control	List of control parameters for the optimization method.
...	Additional arguments passed to arima.

Details

Fits the following two-regime TARMA process with optional components: linear AR part, seasonal MA and covariates.

$$X_t = \phi_0 + \sum_{h\in I} \phi_h X_{t-h} + \sum_{l=1}^{Q} \Theta_l \epsilon_{t-ls} + \sum_{j=1}^{q} \theta_j \epsilon_{t-j} + \sum_{k=1}^{K} \delta_k Z_k + \epsilon_t + \begin{cases} \phi_{1,0} + \sum_{i\in I_1} \phi_{1,i} X_{t-i} & \text{if} X_{t-d} \leq \text{thd} \\ \phi_{2,0} + \sum_{i\in I_2} \phi_{2,i} X_{t-i} & \text{if} X_{t-d} > \text{thd} \end{cases}$$

where ϕ_h are the common AR parameters and h ranges in `I = ar.lags`. θ_j are the common MA parameters and $j = 1, \ldots, q$ (`q = ma.ord`), Θ_l are the common seasonal MA parameters and $l = 1, \ldots, Q$ (`Q = sma.ord`) δ_k are the parameters for the covariates. Finally, $\phi_{1,i}$ and $\phi_{2,i}$ are the TAR parameters for the lower and upper regime, respectively and `I1 = tar1.lags` `I2 = tar2.lags` are the vector of TAR lags.

Value

A list of class `TARMA` with components:

- `fit` - The output of the fit. It is a `arima` object.
- `aic` - Value of the AIC for the minimised least squares criterion over the threshold range.
- `bic` - Value of the BIC for the minimised least squares criterion over the threshold range.
- `aic.v` - Vector of values of the AIC over the threshold range.
- `thd.range` - Vector of values of the threshold range.
- `d` - Delay parameter.
- `thd` - Estimated threshold.
- `phi1` - Estimated AR parameters for the lower regime.
- `phi2` - Estimated AR parameters for the upper regime.
- `theta1` - Estimated MA parameters for the lower regime.
- `theta2` - Estimated MA parameters for the upper regime.
- `delta` - Estimated parameters for the covariates `x.reg`.
- `tlag1` - TAR lags for the lower regime
- `tlag2` - TAR lags for the upper regime
- `mlag1` - TMA lags for the lower regime
- `mlag2` - TMA lags for the upper regime
- `arlag` - Same as the input slot `ar.lags`
- `include.int` - Same as the input slot `include.int`
- `se` - Standard errors for the parameters. Note that they are computed conditionally upon the threshold so that they are generally smaller than the true ones.
- `rss` - Minimised residual sum of squares.
- `method` - Estimation method.
- `call` - The matched call.

Author(s)

Simone Giannerini, <simone.giannerini@unibo.it>

Greta Goracci, <greta.goracci@unibz.it>

References

- Giannerini S, Goracci G (2021). "Estimating and Forecasting with TARMA models." University of Bologna.
- Chan K, Goracci G (2019). "On the Ergodicity of First-Order Threshold Autoregressive Moving-Average Processes." *J. Time Series Anal.*, **40**(2), 256-264.

See Also

TARMA.fit for Least Square estimation of full subset TARMA models. print.TARMA for print methods for TARMA fits. predict.TARMA for prediction and forecasting.

Examples

```
## a TARMA(1,1,1,1)
set.seed(127)
x   <- TARMA.sim(n=100, phi1=c(0.5,-0.5), phi2=c(0,0.8), theta1=0.5, theta2=0.5, d=1, thd=0.2)
fit1 <- TARMA.fit2(x, tar1.lags=1, tar2.lags=1, ma.ord=1, d=1)


## Showcase of the fit with covariates ---
## simulates from a TARMA(3,3,1,1) model with common MA parameter
## and common AR(1) and AR(2) parameters. Only the lag 3 parameter varies across regimes
set.seed(212)
n <- 300
x <- TARMA.sim(n=n, phi1=c(0.5,0.3,0.2,0.4), phi2=c(0.5,0.3,0.2,-0.2), theta1=0.4, theta2=0.4,
     d=1, thd=0.2, s1=1, s2=1)

## FIT 1: estimates lags 1,2,3 as threshold lags ---
fit1 <- TARMA.fit2(x, ma.ord=1, tar1.lags=c(1,2,3), tar2.lags=c(1,2,3), d=1)

## FIT 2: estimates lags 1 and 2 as fixed AR and lag 3 as the threshold lag
fit2 <- TARMA.fit2(x, ma.ord=1, tar1.lags=c(3),  tar2.lags=c(3), ar.lags=c(1,2), d=1)

## FIT 3: creates lag 1 and 2 and fits them as covariates ---
z1   <- lag(x,-1)
z2   <- lag(x,-2)
fit3 <- TARMA.fit2(x, ma.ord=1,  tar1.lags=c(3), tar2.lags=c(3), x.reg=ts.intersect(z1,z2), d=1)

## FIT 4: estimates lag 1 as a covariate, lag 2 as fixed AR and lag 3 as the threshold lag
fit4 <- TARMA.fit2(x, ma.ord = 1,  tar1.lags=c(3), tar2.lags=c(3), x.reg=z1, ar.lags=2, d=1)
```

TARMA.sim *Simulation of a two-regime* TARMA(p1,p2,q1,q2) *process*

Description

Simulates from the following two-regime TARMA(p1,p2,q1,q2) process:

$$X_t = \begin{cases} \phi_{1,0} + \sum_{i=1}^{p_1} \phi_{1,i} X_{t-i} + \sum_{j=1}^{q_1} \theta_{1,j} \varepsilon_{t-j} + \varepsilon_t, & \text{if} \quad X_{t-d} \leq \text{thd} \\ \phi_{2,0} + \sum_{i=1}^{p_2} \phi_{2,i} X_{t-i} + \sum_{j=1}^{q_2} \theta_{2,j} \varepsilon_{t-j} + \varepsilon_t, & \text{if} \quad X_{t-d} > \text{thd} \end{cases}$$

Usage

```
TARMA.sim(
  n,
  phi1,
```

```
  phi2,
  theta1,
  theta2,
  d = 1,
  thd = 0,
  s1 = 1,
  s2 = 1,
  rand.gen = rnorm,
  innov = rand.gen(n, ...),
  n.start = 500,
  xstart,
  start.innov = rand.gen(n.start, ...),
  ...
)
```

Arguments

n	Length of the series.
phi1	Vector of p1+1 Autoregressive parameters of the lower regime. The first element is the intercept.
phi2	Vector of p2+1 Autoregressive parameters of the upper regime. The first element is the intercept.
theta1	Vector of q1 Moving Average parameters of the lower regime.
theta2	Vector of q2 Moving Average parameters of the upper regime.
d	Delay parameter. Defaults to 1.
thd	Threshold parameter. Defaults to 0.
s1	Innovation variance for the lower regime. Defaults to 1.
s2	Innovation variance for the upper regime. Defaults to 1.
rand.gen	Optional: a function to generate the innovations. Defaults to rnorm.
innov	Optional: a time series of innovations. If not provided, rand.gen is used.
n.start	Length of the burn-in period. Defaults to 500.
xstart	Initial condition as a named list: $ar: AR part length k = max(p1,p2,d), X[k], X[k-1], ... ,X[1]; $ma: MA part length q = ma.ord, e[q], ... , e[1].
start.innov	Optional: a time series of innovations for the burn-in period.
...	Additional arguments for rand.gen.

Details

Note that the parameters are not checked for ergodicity.

Value

A time series object of class ts generated from the above model.

Author(s)

Simone Giannerini, <simone.giannerini@unibo.it>

Greta Goracci, <greta.goracci@unibz.it>

References

- Giannerini S, Goracci G (2021). "Estimating and Forecasting with TARMA models." University of Bologna.

Examples

```
## a TARMA(1,1,1,1) model
set.seed(123)
x <- TARMA.sim(n=100, phi1=c(0.5,-0.5), phi2=c(0.0,0.8), theta1=-0.5, theta2=0.5, d=1, thd=0.2)

## a TARMA(1,2,1,1) model
x <- TARMA.sim(n=100,phi1=c(0.5,-0.5,0),phi2=c(0,0.5,0.3),theta1=-0.5,theta2=0.5,d=1,thd=0.2)
```

TARMA.test *ARMA versus TARMA supLM test for nonlinearity*

Description

Implements a supremum Lagrange Multiplier test for a ARMA specification versus a TARMA specification. Includes the AR versus TAR test.

Usage

```
TARMA.test(
  x,
  pa = 0.25,
  pb = 0.75,
  ar.ord,
  ma.ord,
  ma.fixed = TRUE,
  d,
  thd.range,
  method = "CSS-ML",
  ...
)
```

Arguments

x	A univariate time series.
pa	Real number in [0,1]. Sets the lower limit for the threshold search to the 100*pa-th sample percentile. The default is 0.25
pb	Real number in [0,1]. Sets the upper limit for the threshold search to the 100*pb-th sample percentile. The default is 0.75
ar.ord	Order of the AR part.
ma.ord	Order of the MA part.
ma.fixed	Logical. Only applies to testing ARMA vs TARMA. If TRUE computes the test where only the AR parameters are tested, see (Goracci et al. 2021) for details.
d	Delay parameter. Defaults to 1.

thd.range	Vector of optional user defined threshold range. If missing then pa and pb are used.
method	Fitting method to be passed to arima.
...	Additional arguments to be passed to arima.

Details

Implements an asymptotic supremum Lagrange Multiplier test to test an ARMA specification versus a TARMA specification. If ma.fixed=TRUE (the default), the AR parameters are tested whereas the MA parameters are fixed. If ma.fixed=FALSE both the AR and the MA parameters are tested. This is an asymptotic test and the value of the test statistic has to be compared with the critical values tabulated in (Goracci et al. 2021) and (Andrews 2003). If ma.ord=0 then the AR versus TAR test is used. Note that when method='CSS', this is equivalent to TAR.test, which uses least squares.

Value

An object of class TARMAtest with components:

statistic The value of the supLM statistic.

parameter A named vector: threshold is the value that maximises the Lagrange Multiplier values.

test.v Vector of values of the LM statistic for each threshold given in thd.range.

thd.range Range of values of the threshold.

fit.ARMA The null model: ARMA fit over x.

sigma2 Estimated innovation variance from the ARMA fit.

data.name A character string giving the name of the data.

prop Proportion of values of the series that fall in the lower regime.

p.value The p-value of the test. It is NULL for the asymptotic test.

method A character string indicating the type of test performed.

d The delay parameter.

pa Lower threshold quantile.

dfree Effective degrees of freedom. It is the number of tested parameters.

Author(s)

Simone Giannerini, <simone.giannerini@unibo.it>

Greta Goracci, <greta.goracci@unibz.it>

References

- Goracci G, Giannerini S, Chan K, Tong H (2023). "Testing for threshold effects in the TARMA framework." *Statistica Sinica*, **33**(3), 1879-1901. https://doi.org/10.5705/ss.202021.0120.
- Andrews DWK (2003). "Tests for Parameter Instability and Structural Change with Unknown Change Point: A Corrigendum." *Econometrica*, **71**(1), 395-397. doi:10.1111/1468-0262.00405.

See Also

TAR.test.B for the bootstrap version of the test. TARMAGARCH.test for the robust version of the test with respect to GARCH innovations. TARMA.sim to simulate from a TARMA process.

Examples

```
## a TARMA(1,1,1,1) where the threshold effect is on the AR parameters
set.seed(123)
x1   <- TARMA.sim(n=100, phi1=c(0.5,-0.5), phi2=c(0.0,0.8), theta1=0.5, theta2=0.5, d=1, thd=0.2)
TARMA.test(x1, ar.ord=1, ma.ord=1, d=1)
TARMA.test(x1, ar.ord=1, ma.ord=1, d=1, ma.fixed=FALSE) # full TARMA test

## a TARMA(1,1,1,1) where the threshold effect is on the MA parameters
set.seed(212)
x2   <- TARMA.sim(n=100, phi1=c(0.5,0.2), phi2=c(0.5,0.2), theta1=0.6, theta2=-0.6, d=1, thd=0.2)
TARMA.test(x2, ar.ord=1, ma.ord=1, d=1)
TARMA.test(x2, ar.ord=1, ma.ord=1, d=1, ma.fixed=FALSE) # full TARMA test

## a ARMA(1,1)
x3   <- arima.sim(n=100, model=list(order = c(1,0,1),ar=0.5, ma=0.5))
TARMA.test(x3, ar.ord=1, ma.ord=1, d=1)

## a TAR(1,1)
x4  <- TARMA.sim(n=100, phi1=c(0.5,-0.5), phi2=c(0.0,0.8), theta1=0, theta2=0, d=1, thd=0.2)
TARMA.test(x4, ar.ord=1, ma.ord=0, d=1)

## a AR(1)
x5   <- arima.sim(n=100, model=list(order = c(1,0,0),ar=0.5))
TARMA.test(x5, ar.ord=1, ma.ord=0, d=1)
```

TARMAGARCH.test *ARMA GARCH versus TARMA GARCH supLM test for nonlinearity*

Description

Implements a supremum Lagrange Multiplier test for a ARMA-GARCH specification versus a TARMA-GARCH specification. Both the AR and MA parameters are tested

Usage

```
TARMAGARCH.test(
  x,
  pa = 0.25,
  pb = 0.75,
  ar.ord = 1,
  ma.ord = 1,
  arch.ord = 1,
  garch.ord = 1,
  d = 1,
  thd.range,
  method = "CSS",
```

```
  ...
)
```

Arguments

x	A univariate time series.
pa	Real number in [0,1]. Sets the lower limit for the threshold search to the 100*pa-th sample percentile. The default is 0.25
pb	Real number in [0,1]. Sets the upper limit for the threshold search to the 100*pb-th sample percentile. The default is 0.75
ar.ord	Order of the AR part.
ma.ord	Order of the MA part.
arch.ord	Order of the ARCH part.
garch.ord	Order of the GARCH part.
d	Delay parameter. Defaults to 1.
thd.range	Vector of optional user defined threshold range. If missing then pa and pb are used.
method	Fitting method to be passed to arima.
...	Additional arguments to be passed to arima.

Details

Implements an asymptotic supremum Lagrange Multiplier test to test an ARMA-GARCH specification versus a TARMA-GARCH specification. In other words, the test is robust with respect to heteroskedasticity. Both the AR parameters and the MA parameters are tested. This is an asymptotic test and the value of the test statistic has to be compared with the critical values tabulated in (Angelini et al. 2022) or (Andrews 2003).

Value

A list of class htest with components:

statistic The value of the supLM statistic.

parameter A named vector: threshold is the value that maximises the Lagrange Multiplier values.

test.v Vector of values of the LM statistic for each threshold given in thd.range.

thd.range Range of values of the threshold.

fit.ARMA The null model: ARMA fit over x.

fit.GARCH The null model: GARCH fit over the residuals of the ARMA fit.

sigma2 Estimated innovation variance from the ARMA fit.

data.name A character string giving the name of the data.

prop Proportion of values of the series that fall in the lower regime.

p.value The p-value of the test. It is NULL for the asymptotic test.

method A character string indicating the type of test performed.

d The delay parameter.

pa Lower threshold quantile.

dfree Effective degrees of freedom. It is the number of tested parameters.

Author(s)

Simone Giannerini, <simone.giannerini@unibo.it>

Greta Goracci, <greta.goracci@unibz.it>

References

- Angelini F, Castellani M, Goracci G, Giannerini S (2022). "Testing for Threshold Effects in Presence of Volatility and Measurement Error: The Case of Italian Strikes." University of Bologna, University of Bolzano.
- Goracci G, Giannerini S, Chan K, Tong H (2021). "Testing for threshold effects in the TARMA framework." University of Bologna, Free University of Bolzano, University of Iowa, London School of Economics. https://arxiv.org/abs/2103.13977.
- Andrews DWK (2003). "Tests for Parameter Instability and Structural Change with Unknown Change Point: A Corrigendum." *Econometrica*, **71**(1), 395-397. doi:10.1111/1468-0262.00405.

See Also

TARMA.test and TAR.test.B for the asymptotic and bootstrap test without the GARCH component. TARMA.sim to simulate from a TARMA process. TARMA.fit and TARMA.fit2 for TARMA modelling.

Examples

```
## Function to simulate from a ARMA-GARCH process

arma11.garch11 <- function(n, ph, th, a, b, a0=1, rand.gen = rnorm, innov = rand.gen(n, ...),
n.start = 500, start.innov = rand.gen(n.start, ...),...){

  #  Simulates a ARMA(1,1)-GARCH(1,1) process
  #  with parameeters ph, th, a, b, a0.
  #         x[t] <- ph*x[t-1] + th*eps[t-1] + eps[t]
  #       eps[t] <- e[t]*sqrt(v[t])
  #         v[t] <- a0 + a*eps[t-1]^2 + b*v[t-1];
  # ph   : AR
  # th   : MA
  # a    : ARCH
  # b    : GARCH

  # checks
  if(abs(a+b)>=1)   stop("model is not stationary")
  if(b/(1-a)>=1) stop("model has infinite fourth moments")

  if (!missing(start.innov) && length(start.innov) < n.start)
    stop(gettextf("'start.innov' is too short: need %d points", n.start), domain = NA)
  e <- c(start.innov[1L:n.start], innov[1L:n])
  ntot <- length(e)
  x <- v <- eps <- double(ntot)
  v[1]   <- a0/(1.0-a-b);
  eps[1] <- e[1]*sqrt(v[1])
  x[1]   <- eps[1];
  for(i in 2:ntot){
    v[i]   <- a0 + a*eps[i-1]^2 + b*v[i-1];
    eps[i] <- e[i]*sqrt(v[i])
```

```
    x[i]   <- ph*x[i-1] + th*eps[i-1] + eps[i]
  }
  if (n.start > 0)  x <- x[-(1L:n.start)]
  return(ts(x));
}

## ************************************************************************
## Comparison between the robust and the non-robust test in presence of GARCH errors
## Simulates from the ARMA(1,1)-GARCH(1,1)

set.seed(12)
x1 <- arma11.garch11(n=100, ph=0.9, th=0.5, a=0.85, b=0.1, a0=1,n.start=500)
TARMAGARCH.test(x1, ar.ord=1, ma.ord=1, arch.ord=1, garch.ord=1, d=1)
TARMA.test(x1, ar.ord=1, ma.ord=1, d=1, ma.fixed=FALSE)

## a TARMA(1,1,1,1) where the threshold effect is on the AR parameters
set.seed(123)
x2 <- TARMA.sim(n=100, phi1=c(0.5,-0.5), phi2=c(0.0,0.8), theta1=0.5, theta2=0.5, d=1, thd=0.2)
TARMAGARCH.test(x2, ar.ord=1, ma.ord=1, d=1)
```

`TARMAur.test` *Unit root supLM test for an integrated MA versus a stationary TARMA process*

Description

Implements a supremum Lagrange Multiplier unit root test for the null hypiythesis of a integrated MA process versus a stationary TARMA process.

Usage

```
TARMAur.test(x, pa = 0.25, pb = 0.75, thd.range, method = "ML", ...)
```

Arguments

`x`	A univariate vector or time series.
`pa`	Real number in `[0,1]`. Sets the lower limit for the threshold search to the `100*pa`-th sample percentile. The default is `0.25`
`pb`	Real number in `[0,1]`. Sets the upper limit for the threshold search to the `100*pb`-th sample percentile. The default is `0.75`
`thd.range`	Vector of optional user defined threshold range. If missing then `pa` and `pb` are used.
`method`	Fitting method to be passed to `arima`.
`...`	Additional arguments to be passed to `arima`.

Details

Implements an asymptotic supremum Lagrange Multiplier test to test an integrate MA(1) specification versus a stationary TARMA(1,1) specification. This is an asymptotic test and the value of the test statistic has to be compared with the critical values tabulated in (Chan et al. 2024) and available in `supLMQur`. The relevant critical values are automatically shown upon printing the test, see `print.TARMAtest`.

Value

An object of class `TARMAtest` with components:

`statistic` The value of the supLM statistic.

`parameter` A named vector: `threshold` is the value that maximises the Lagrange Multiplier values.

`test.v` Vector of values of the LM statistic for each threshold given in `thd.range`.

`thd.range` Range of values of the threshold.

`fit.ARMA` The null model: IMA(1) fit over `x`.

`sigma2` Estimated innovation variance from the IMA fit.

`data.name` A character string giving the name of the data.

`p.value` The p-value of the test. It is `NULL` for the asymptotic test.

`method` A character string indicating the type of test performed.

`d` The delay parameter.

`pa` Lower threshold quantile.

Author(s)

Simone Giannerini, <simone.giannerini@unibo.it>

Greta Goracci, <greta.goracci@unibz.it>

References

- Chan K, Giannerini S, Goracci G, Tong H (2024). "Testing for threshold regulation in presence of measurement error." *Statistica Sinica*, **34**(3). https://doi.org/10.5705/ss.202022.0125.

See Also

`TARMAur.test.B` for the bootstrap version of the test. `print.TARMAtest` for the print method.

Examples

```
## a TARMA(1,1,1,1)
set.seed(123)
x1   <- TARMA.sim(n=100, phi1=c(0.5,-0.5), phi2=c(0.0,0.8), theta1=0.5, theta2=0.5, d=1, thd=0.2)
TARMAur.test(x1)


## a IMA(1,1)
x2   <- arima.sim(n=100, model=list(order = c(0,1,1),ma=0.6))
TARMAur.test(x2)
```

`TARMAur.test.B` *Unit root supLM test for an integrated MA versus a stationary TARMA process*

Description

Implements a supremum Lagrange Multiplier unit root test for the null hypothesis of a integrated MA process versus a stationary TARMA process.

Usage

```
TARMAur.test.B(
  x,
  B = 1000,
  pa = 0.25,
  pb = 0.75,
  thd.range,
  method = "ML",
  btype = c("wb.r", "wb.n", "iid"),
  ...
)
```

Arguments

`x`	A univariate vector or time series.
`B`	Integer. Number of bootstrap resamples. Defaults to 1000.
`pa`	Real number in `[0,1]`. Sets the lower limit for the threshold search to the `100*pa`-th sample percentile. The default is `0.25`
`pb`	Real number in `[0,1]`. Sets the upper limit for the threshold search to the `100*pb`-th sample percentile. The default is `0.75`
`thd.range`	Vector of optional user defined threshold range. If missing then `pa` and `pb` are used.
`method`	Fitting method to be passed to `arima`.
`btype`	Bootstrap type, can be one of `'iid'`, `'wb.h'`, `'wb.r'`, `'wb.n'`, see Details.
`...`	Additional arguments to be passed to `arima`.

Details

Implements the bootstrap version of `TARMAur.test` the supremum Lagrange Multiplier test to test an integrate MA(1) specification versus a stationary TARMA(1,1) specification. The option `btype` specifies the type of bootstrap as follows:

`wb.r` Residual wild bootstrap with Rademacher auxiliary distribution. See (Giannerini et al. 2022).

`wb.n` Residual wild bootstrap with Normal auxiliary distribution. See (Giannerini et al. 2022).

`iid` Residual iid bootstrap. See (Goracci et al. 2021).

Value

An object of class `TARMAtest` with components:

`statistic` The value of the supLM statistic.

`parameter` A named vector: `threshold` is the value that maximises the Lagrange Multiplier values.

`test.v` Vector of values of the LM statistic for each threshold given in `thd.range`.

`thd.range` Range of values of the threshold.

`fit.ARMA` The null model: IMA(1) fit over `x`.

`sigma2` Estimated innovation variance from the IMA fit.

`data.name` A character string giving the name of the data.

`p.value` The bootstrap p-value of the test.

`method` A character string indicating the type of test performed.

`d` The delay parameter.

`pa` Lower threshold quantile.

`Tb` The bootstrap null distribution.

Author(s)

Simone Giannerini, <simone.giannerini@unibo.it>

Greta Goracci, <greta.goracci@unibz.it>

References

- Chan K, Giannerini S, Goracci G, Tong H (2024). "Testing for threshold regulation in presence of measurement error." *Statistica Sinica*, **34**(3). https://doi.org/10.5705/ss.202022.0125.

See Also

`TARMAur.test` for the asymptotic version of the test. `print.TARMAtest` for the print method.

Examples

```
## a TARMA(1,1,1,1)
set.seed(123)
x1   <- TARMA.sim(n=100, phi1=c(0.5,-0.5), phi2=c(0.0,0.8), theta1=0.5, theta2=0.5, d=1, thd=0.2)
TARMAur.test.B(x1, B=100) # B=100 for speedup


## a IMA(1,1)
x2   <- arima.sim(n=100, model=list(order = c(0,1,1),ma=0.6))
TARMAur.test.B(x2, B=100) # B=100 for speedup
```

Index