Package ‘TSA’

September 13, 2020

Type Package

Title Time Series Analysis

Version 1.3

Date 2020-9-11

Author Kung-Sik Chan, Brian Ripley

Maintainer Kung-Sik Chan <kungsik.chan@gmail.com>

Depends R (>= 2.5.1)

Imports leaps, locfit, mgcv, tseries

Description Contains R functions and datasets detailed in the book ``Time Series Analysis with Applications in R (second edition)'' by Jonathan Cryer and Kung-Sik Chan.

URL https://stat.uiowa.edu/~kchan/TSA.htm

RoxygenNote 7.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2020-09-13 15:00:11 UTC

R topics documented:

TSA-package 3
acf 4
airmiles 5
airpass 6
ar1.2.s 6
ar1.s 7
ar2.s 8
arima 8
arima.boot 10
arimax 11
arma11.s 13

ARMAspec . . . 13
armasubsets . . . 14
beersales . . . 15
bluebird . . . 15
bluebirdlite . . . 16
boardings . . . 17
BoxCox.ar . . . 17
co2 . . . 18
color . . . 19
CREF . . . 19
cref.bond . . . 20
days . . . 20
deere1 . . . 21
deere2 . . . 21
deere3 . . . 22
detectAO . . . 22
detectIO . . . 24
eacf . . . 25
eeg . . . 26
electricity . . . 27
euph . . . 27
explode.s . . . 28
fitted.Arima . . . 28
flow . . . 29
garch.sim . . . 29
gBox . . . 30
gold . . . 31
google . . . 32
hare . . . 33
harmonic . . . 33
hours . . . 34
ima22.s . . . 35
JJ . . . 35
Keenan.test . . . 36
kurtosis . . . 37
lagplot . . . 38
larain . . . 39
LB.test . . . 39
ma1.1.s . . . 40
ma1.2.s . . . 41
ma2.s . . . 41
McLeod.Li.test . . . 42
milk . . . 43
oil.price . . . 43
oilfilters . . . 44
periodogram . . . 44
plot.Arima . . . 45
plot.armasubsets . . . 47

plot1.acf . . . 48
predict.TAR . . . 48
prescrip . . . 49
prewhiten . . . 50
prey.eq . . . 51
qar.sim . . . 51
retail . . . 52
robot . . . 53
rstandard.Arima . . . 54
runs . . . 54
rwalk . . . 55
season . . . 56
skewness . . . 57
SP . . . 58
spec . . . 58
spots . . . 60
spots1 . . . 61
star . . . 61
summary.armasubsets . . . 62
tar . . . 62
tar.sim . . . 64
tar.skeleton . . . 65
tbone . . . 67
tempdub . . . 67
tlrt . . . 68
Tsay.test . . . 69
tsdiag.Arimax . . . 70
tsdiag.TAR . . . 71
tuba . . . 72
units . . . 73
usd.hkd . . . 73
veilleux . . . 74
wages . . . 75
winnebago . . . 75
zlag . . . 76

Index **77**

`TSA-package` *Time Series Analysis*

Description

Contains R functions and datasets detailed in the book "Time Series Analysis with Applications in R (second edition)" by J.D. Cryer and K.S. Chan

Details

Package:	TSA
Type:	Package
Version:	1.3
Date:	2020-9-9
License:	GPL version 2 or newer

Author(s)

Kung-Sik Chan, Brian Ripley

`acf` *Auto- and Cross- Covariance and -Correlation Function Estimation*

Description

This function calls the acf function in the stats package and processes to drop lag-0 of the acf. It only works for univariate time series, so x below should be 1-dimensional.

Usage

```
acf(x, lag.max = NULL, type = c("correlation", "covariance", "partial")[1],
plot = TRUE, na.action = na.fail, demean = TRUE, drop.lag.0 = TRUE, ...)
```

Arguments

`x`	a univariate or multivariate (not ccf) numeric time series object or a numeric vector or matrix, or an "acf" object.
`lag.max`	maximum number of lags at which to calculate the acf. Default is 10*log10(N/m) where N is the number of observations and m the number of series.
`type`	character string giving the type of acf to be computed. Allowed values are "correlation" (the default), "covariance" or "partial".
`plot`	logical. If TRUE (the default) the acf is plotted.
`na.action`	function to be called to handle missing values. na.pass can be used.
`demean`	logical. Should the covariances be about the sample means?
`drop.lag.0`	logical. Should lag 0 be dropped
`...`	further arguments to be passed to plot.acf.

Value

An object of class "acf", which is a list with the following elements:

`lag`	A three dimensional array containing the lags at which the acf is estimated.
`acf`	An array with the same dimensions as lag containing the estimated acf.
`type`	The type of correlation (same as the type argument).
`n.used`	The number of observations in the time series.
`series`	The name of the series x.
`snames`	The series names for a multivariate time series.

Author(s)

Original authors of stats:::acf are: Paul Gilbert, Martyn Plummer, B.D. Ripley. This wrapper is written by Kung-Sik Chan

References

~put references to the literature/web site here ~

See Also

`plot.acf`, `ARMAacf` for the exact autocorrelations of a given ARMA process.

Examples

```
data(rwalk)
model1=lm(rwalk~time(rwalk))
summary(model1)
acf(rstudent(model1),main='')
```

`airmiles` *Monthly Airline Passenger-Miles in the US*

Description

Monthly U.S. airline passenger-miles: 01/1996 - 05/2005.

Usage

```
data(airmiles)
```

Format

The format is: 'ts' int [1:113, 1] 30983174 32147663 38342975 35969113 36474391 38772238 40395657 41738499 33580773 36389842 ... - attr(*, "dimnames")=List of 2 ..$: NULL ..$: chr "airmiles" - attr(*, "tsp")= num [1:3] 1996 2005 12

Source

www.bts.gov/xml/air_traffic/src/index.xml#MonthlySystem

Examples

```
data(airmiles)
## maybe str(airmiles) ; plot(airmiles) ...
```

---

airpass *Monthly total international airline passengers*

Description

Monthly total international airline passengers from 01/1960- 12/1971.

Usage

```
data(airpass)
```

Format

The format is: Time-Series [1:144] from 1960 to 1972: 112 118 132 129 121 135 148 148 136 119 ...

Source

Box, G. E. P., Jenkins, G. M. and Reinsel, G. C. (1994) Time Series Analysis, Forecasting and Control. Second Edition. New York: Prentice-Hall.

Examples

```
data(airpass)
## maybe str(airpass) ; plot(airpass) ...
```

---

ar1.2.s *A simulated AR(1) series*

Description

A simulated AR(1) series with the AR coefficient equal to 0.4.

Usage

```
data(ar1.2.s)
```

Format

The format is: Time-Series [1:60] from 1 to 60: -0.0678 1.4994 0.4888 0.3987 -0.5162 ...

Details

The model is Y(t)=0.4*Y(t-1)+e(t) where the e's are iid standard normal.

Examples

```
data(ar1.2.s)
## maybe str(ar1.2.s) ; plot(ar1.2.s) ...
```

`ar1.s` *A simulated AR(1) series*

Description

A simulated AR(1) series with the AR coefficient equal to 0.9.

Usage

```
data(ar1.s)
```

Format

The format is: Time-Series [1:60] from 1 to 60: -1.889 -1.691 -1.962 -0.566 -0.627 ...

Details

The model is Y(t)=0.9*Y(t-1)+e(t) where the e's are iid standard normal.

Examples

```
data(ar1.s)
## maybe str(ar1.s) ; plot(ar1.s) ...
```

ar2.s *Asimulated AR(2) series / time series*

Description

Asimulated AR(2) series with AR coefficients being equal to 1.5 and -0.75

Usage

```
data(ar2.s)
```

Format

The format is: Time-Series [1:120] from 1 to 120: -2.064 -1.937 0.406 2.039 2.953 ...

Details

The model is Y(t)=1.5*Y(t-1)-0.75*Y(t-2)+e(t) where the e's are iid standard normal random variables.

Examples

```
data(ar2.s)
## maybe str(ar2.s) ; plot(ar2.s) ...
```

arima *Fitting an ARIMA model with Exogeneous Variables*

Description

This function is identical to the arimax function which builds on and extends the capability of the arima function in R stats by allowing the incorporation of transfer functions, and innovative and additive outliers. For backward compatitibility, the function is also named arima. Note in the computation of AIC, the number of parameters excludes the noise variance. This function is heavily based on the arima function of the stats core of R.

Usage

```
arima(x, order = c(0, 0, 0), seasonal = list(order = c(0, 0, 0), period = NA),
 xreg = NULL, include.mean = TRUE, transform.pars = TRUE, fixed = NULL,
init = NULL, method = c("CSS-ML", "ML", "CSS"), n.cond, optim.control = list(),
 kappa = 1e+06, io = NULL, xtransf, transfer = NULL)
```

Arguments

- `x` — time series response
- `order` — regular ARIMA order
- `seasonal` — seasonal ARIMA order
- `xreg` — a dataframe containing covariates
- `include.mean` — if true, an intercept term is incorporated in the model; applicable only to stationary models.
- `transform.pars` — if true, the AR parameters are transformed to ensure stationarity
- `fixed` — a vector indicating which coefficients are fixed or free
- `init` — initial values
- `method` — estimation method
- `n.cond` — number of initial values to be conditioned on in a conditional analysis
- `optim.control` — control parameters for the optimization procedure
- `kappa` — prior variance; used in dealing with initial values

  All of the above parameters have the same usage as those in the arima function. Please check the help manual of the arima function. Below are new options.
- `io` — a list of time points at which the model may have an innovative outlier. The time point of the outlier can be given either as absolute time point or as c(a,b), i.e. at the b-th 'month' of the a-th 'year' where each year has frequency(x) months, assuming x is a time series.
- `xtransf` — xtranf is a matrix with each column containing a covariate that affects the time series response in terms of an ARMA filter of order (p,q), i.e. if Z is one such covariate, its effect on the time series is $(theta_0 + theta_1 B + ... + theta_{q-1} B^{q-1})/(1 - phi_1 B - ... - phi_p B^p) Z_t$ In particular, if $p = 0$ and $q = 1$, this specifies a simple regression relationship, which should be included in xreg and not here. Note that the filter starts with zero initial values. Hence, it is pertinent to mean-delete each distributed-lag covariate, and this is not done automatically.
- `transfer` — a list consisting of the ARMA orders for each transfer (distributed lag) covariate.

Value

An Arimax object contining the model fit.

Author(s)

Original author of the arima function in R stats: Brian Ripley. The arimax function is based on the stats:::arima function, with modifications by Kung-Sik Chan.

See Also

`arima`

Examples

```
data(hare)
arima(sqrt(hare),order=c(3,0,0))
```

---

arima.boot *Compute the Bootstrap Estimates of an ARIMA Model*

Description

This function bootstraps time series according to the fitted ARMA(p,d,q) model supplied by the fitted object arima.fit, and estimate the same model using the arima function. Any bootstrap sample that has problem when fitted with the ARIMA model will be omitted from the final results and all error messages will be suppressed. You can check if there is any fitting problem by running the command geterrmessage().

Usage

```
arima.boot(arima.fit, cond.boot = FALSE, is.normal = TRUE, B = 1000, init, ntrans = 100)
```

Arguments

`arima.fit`	a fitted object from the arima function (seasonal components not allowed)
`cond.boot`	whether or not the bootstrap is conditional on the (p+d) initial values; if it is set true. If false (default), the stationary bootstrap is used.
`is.normal`	if true (default), errors are normally distributed, otherwise errors are drawn randomly and with replacement from the residuals of the fitted model.
`B`	number of bootstrap replicates (1000, default)
`init`	initial values for the bootstrap; needed if cond.boot=True default values are the initial values of the time series of the fitted model.
`ntrans`	number of transient values for the stationary bootstrap. Default=100

Value

a matrix each row of which consists of the coefficient estimates of a bootstrap time-series.

Author(s)

Kung-Sik Chan

Examples

```
data(hare)
arima.hare=arima(sqrt(hare),order=c(3,0,0))
boot.hare=arima.boot(arima.hare,B=50,init=sqrt(hare)[1:3],ntrans=100)
apply(boot.hare,2,quantile, c(.025,.975))
period.boot=apply(boot.hare,1,function(x){
roots=polyroot(c(1,-x[1:3]))
min1=1.e+9
rootc=NA
for (root in roots) {
if( abs(Im(root))<1e-10) next
if (Mod(root)< min1) {min1=Mod(root); rootc=root}
}
if(is.na(rootc)) period=NA else period=2*pi/abs(Arg(rootc))
period
})
hist(period.boot)
quantile(period.boot,c(0.025,.975))
```

`arimax` *Fitting an ARIMA model with Exogeneous Variables*

Description

This function builds on and extends the capability of the arima function in R stats by allowing the incorporation of transfer functions, innovative and additive outliers. For backward compatitibility, the function is also named arima. Note in the computation of AIC, the number of parameters excludes the noise variance.

Usage

```
arimax(x, order = c(0, 0, 0), seasonal = list(order = c(0, 0, 0), period = NA),
 xreg = NULL, include.mean = TRUE, transform.pars = TRUE, fixed = NULL,
init = NULL, method = c("CSS-ML", "ML", "CSS"), n.cond, optim.control = list(),
 kappa = 1e+06, io = NULL, xtransf, transfer = NULL)
```

Arguments

`x`	time series response
`order`	regular ARIMA order
`seasonal`	seasonal ARIMA order
`xreg`	a dataframe containing covariates
`include.mean`	if true, an intercept term is incorporated in the model; applicable only to stationary model.
`transform.pars`	if true, the AR parameters are transformed to ensure stationarity

fixed	a vector indicating which coefficients are fixed or free
init	initial values
method	estimation method
n.cond	number of initial values to be conditioned on a conditional analysis
optim.control	control parameters for the optimization procedure
kappa	prior variance; used in dealing with initial values All of the above parameters have the same usage as those in the arima function. Please check the help manual of the arima function. Below are new options.
io	a list of time points at which the model may have an innovative outlier. The time point of the outlier can be given either as absolute time point or as c(a,b), i.e. at the b-th 'month' of the a-th 'year' where each year has frequency(x) months, assuming x is a time series.
xtransf	xtranf is a matrix with each column containing a covariate that affects the time series response in terms of an ARMA filter of order (p,q), i.e. if Z is one such covariate, its effect on the time series is $(theta_0+theta_1 B+...+theta_{q-1} B^{q-1})/(1-phi_1 B-...-phi_p B^p) Z_t$ In particular, if $p=0$ and $q=1$, this specifies a simple regression relationship, which should be included in xreg and not here. Note that the filter starts with zero initial values. Hence, it is pertinent to mean-delete each distributed-lag covariate, which is not done automatically.
transfer	a list consisting of the ARMA orders for each transfer (distributed lag) covariate.

Value

An Arimax object containing the model fit.

Author(s)

Original author of the arima function in R stats: Brian Ripley. The arimax function is based on the stats:::arima function, with modifications by Kung-Sik Chan.

See Also

arima

Examples

```
data(airmiles)
plot(log(airmiles),ylab='Log(airmiles)',xlab='Year', main='')
acf(diff(diff(window(log(airmiles),end=c(2001,8)),12)),lag.max=48,main='')
air.m1=arimax(log(airmiles),order=c(0,1,1),seasonal=list(order=c(0,1,1),
period=12),xtransf=data.frame(I911=1*(seq(airmiles)==69),
I911=1*(seq(airmiles)==69)),
transfer=list(c(0,0),c(1,0)),xreg=data.frame(Dec96=1*(seq(airmiles)==12),
Jan97=1*(seq(airmiles)==13),Dec02=1*(seq(airmiles)==84)),method='ML')
```

arma11.s — *A Simulated ARMA(1,1) Series/ time series*

Description

A simulated ARMA(1,1) series with the model given by: $y_t = 0.6 * y_{t-1} + e_t + 0.3 * e_{t-1}$ where the e's are iid standard normal random variables.

Usage

```
data(arma11.s)
```

Format

The format is: Time-Series [1:100] from 1 to 100: -0.765 1.297 0.668 -1.607 -0.626 ...

Examples

```
data(arma11.s)
## maybe str(arma11.s) ; plot(arma11.s) ...
```

ARMAspec — *Theoretical spectral density function of a stationary ARMA model*

Description

Computes and plots the theoretical spectral density function of a stationary ARMA model

Usage

```
ARMAspec(model, freq = seq(0, 0.5, 0.001), plot = TRUE, ...)
```

Arguments

`model`	an arma model
`freq`	vector of frequency over which the spectral density is computed
`plot`	if true, plot the spectral density function; default is true
`...`	other parameters to be passed to the plot function

Value

a list:

`spec`	spectral density values
`freq`	same as freq in the input
`model`	the arma model

Author(s)

Kung-Sik Chan

See Also

spec

Examples

```
theta=.9 # Reset theta for other MA(1) plots
ARMAspec(model=list(ma=-theta))
```

---

armasubsets *Selection of Subset ARMA Models*

Description

This function finds a number of subset ARMA models. A "long" AR model is fitted to the data y to compute the residuals which are taken as a proxy of the error process. Then, an ARMA model is approximated by a regression model with the the covariates being the lags of the time series and the lags of the error process. Subset ARMA models may then be selected using the subset regression technique by leaps and bounds, via the regsubsets function of the leaps package in R.

Usage

```
armasubsets(y, nar, nma, y.name = "Y", ar.method = "ols", ...)
```

Arguments

`y`	time-series data
`nar`	maximum AR order
`nma`	maximum MA order
`y.name`	label of the time series
`ar.method`	method used for fitting the long AR model; default is ols with the AR order determined by AIC
`...`	arguments passed to the plot.armasubsets function

Value

An object of the armasubsets class to be processed by the plot.armasubsets function.

Author(s)

Kung-Sik Chan

Examples

```
set.seed(92397)
test=arima.sim(model=list(ar=c(rep(0,11),.8),ma=c(rep(0,11),0.7)),n=120)
res=armasubsets(y=test,nar=14,nma=14,y.name='test',ar.method='ols')
plot(res)
```

---

beersales *Monthly beer sales / time series*

Description

Monthly beer sales in millions of barrels, 01/1975 - 12/1990.

Usage

```
data(beersales)
```

Format

The format is: Time-Series [1:192] from 1975 to 1991: 11.12 9.84 11.57 13.01 13.42 ...

Source

Frees, E. W., Data Analysis Using Regression Models, Prentice Hall, 1996.

Examples

```
data(beersales)
## maybe str(beersales) ; plot(beersales) ...
```

---

bluebird *Blue Bird Potato Chip Data*

Description

Weekly unit sales (log-transformed) of Bluebird standard potato chips (New Zealand) and their price for 104 weeks.

Usage

```
data(bluebird)
```

Format

The format is: mts [1:104, 1:2] 11.5 11.5 11.8 11.9 11.3 ... - attr(*, "dimnames")=List of 2 ..$: NULL ..$: chr [1:2] "log.sales" "price" - attr(*, "tsp")= num [1:3] 1 104 1 - attr(*, "class")= chr [1:2] "mts" "ts"

Source

www.stat.auckland.ac.nz/~balemi/Assn3.xls

Examples

```
data(bluebird)
## maybe str(bluebird) ; plot(bluebird) ...
```

`bluebirdlite` *Bluebird Lite potato chip data*

Description

Weekly unit sales (log-transformed) of Bluebird Lite potato chips (New Zealand) and their price for 104 weeks.

Usage

```
data(bluebirdlite)
```

Format

A data frame with 104 observations on the following 2 variables.

`log.sales` a numeric vector

`price` a numeric vector

Source

www.stat.auckland.ac.nz/~balemi/Assn3.xls

Examples

```
data(bluebirdlite)
## maybe str(bluebirdlite) ; plot(bluebirdlite) ...
```

boardings — *Monthly public transit boardings and gasoline price in Denver*

Description

Monthly public transit boardings (mostly buses and light rail) and gasoline price (both log-transformed), Denver, Colorado region, 08/2000 - 03/2006.

Source

Personal communication from Lee Cryer, Project Manager, Regional Transportation District, Denver, Colorado. Denver gasoline prices were obtained from the Energy Information Administration, U.S. Department of Energy, Washington, D.C.at www.eia.doe.gov

Examples

```
data(boardings)
plot(boardings)
## maybe str(boardings) ; plot(boardings) ...
```

BoxCox.ar — *Determine the power transformation for serially correlated data*

Description

Determine the appropriate power transformation for time-series data. The objective is to estimate the power transformation so that the transformed time series is approximately a Gaussian AR process.

Usage

```
BoxCox.ar(y, order, lambda = seq(-2, 2, 0.01), plotit = TRUE,
method = c("mle", "yule-walker", "burg", "ols", "yw"), ...)
```

Arguments

`y`	univariate time series (must be positive)
`order`	AR order for the data; if missing, the order is determined by AIC for the log-transformed data
`lambda`	a vector of candidate power transformation values; if missing, it is set to be from -2 to 2, with increment .01
`plotit`	logical value, if true, plot the profile log-likelihood for the power estimator
`method`	method of AR estimation; default is "mle"
`...`	other parameters to be passed to the ar function

Value

A list that contains the following:

`lambda`	candidate power transformation parameter values
`loglike`	profile log-likelihood
`mle`	maximum likelihood estimate of the power transformation value
`ci`	95% C.I. of the power transformation value

Note

The procedure is very computer intensive. Be patient for the outcome

Author(s)

Kung-Sik Chan

Examples

```
data(hare)
# hare.transf=BoxCox.ar(y=hare)
# hare.transf$ci
```

`co2` — *Levels of Carbon Dioxide at Alert, Canada / Time series*

Description

Monthly CO2 level at Alert, Northwest Territories, Canada, near the Artic Circle, 01/1994 - 12/2004.

Usage

```
data(co2)
```

Format

The format is: Time-Series [1:132] from 1994 to 2005: 363 364 365 364 364 ...

Source

`http://cdiac.ornl.gov/trends/co2/sio-alt.htm`

Examples

```
data(co2)
## maybe str(co2) ; plot(co2) ...
```

color — *Color property/time series*

Description

Color property from 35 consecutive batches in an industrial process.

Usage

```
data(color)
```

Format

The format is: Time-Series [1:35] from 1 to 35: 67 63 76 66 69 71 72 71 72 72 ...

Source

"The Estimation of Sigma for an X Chart", Journal of Quality Technology, Vol. 22, No. 3 (July 1990), by Jonathan D. Cryer and Thomas P. Ryan.

Examples

```
data(color)
## maybe str(color) ; plot(color) ...
```

CREF — *Daily CREF Values*

Description

Daily values of one unit of the CREF (College Retirement Equity Fund) Stock fund, 08/26/04 - 08/15/06.

Usage

```
data(CREF)
```

Format

The format is: Time-Series [1:501] from 1 to 501: 170 170 169 170 171 ...

Source

www.tiaa-cref.org/performance/retirement/data/index.html

Examples

```
data(CREF)
## maybe str(CREF) ; plot(CREF) ...
```

cref.bond *Daily CREF Bond Values*

Description

Daily values of one unit of the CREF (College Retirement Equity Fund) Bond fund, 08/26/04 - 08/15/06.

Usage

```
data(CREF)
```

Source

www.tiaa-cref.org/performance/retirement/data/index.html

Examples

```
data(CREF)
## maybe str(CREF) ; plot(CREF) ...
```

days *Number of days between payment to Winegard Corp. / time series*

Description

Accounts receivable data. Number of days until a distributor of Winegard Company products pays their account.

Usage

```
data(days)
```

Format

The format is: Time-Series [1:130] from 1 to 130: 39 39 41 26 28 28 25 26 24 38 ...

Source

Personal communication from Mark Selergren, Vice President, Winegard, Inc., Burlington, Iowa.

Examples

```
data(days)
## maybe str(days) ; plot(days) ...
```

deere1 — *Deviations of an industrial process at Deere \& Co. – Series 1*

Description

82 consecutive values for the amount of deviation (in 0.000025 inch units) from a specified target value in an industrial machining process at Deere \& Co.

Usage

```
data(deere1)
```

Format

The format is: Time-Series [1:82] from 1 to 82: 3 0 -1 -4 7 3 7 3 3 -1 ...

Source

Personal communication from William F. Fulkerson, Deere \& Co. Technical Center, Moline, Illinois.

Examples

```
data(deere1)
## maybe str(deere1) ; plot(deere1) ...
```

deere2 — *Deviations of an industrial process at Deere \& Co. – Series 2*

Description

102 consecutive values for the deviation (in 0.0000025 inch units) from a specified target value.

Usage

```
data(deere2)
```

Format

The format is: Time-Series [1:102] from 1 to 102: -18 -24 -17 -27 -37 -34 -8 14 18 7 ...

Source

Personal communication from William F. Fulkerson, Deere \& Co. Technical Center, Moline, Illinois.

Examples

```
data(deere2)
## maybe str(deere2) ; plot(deere2) ...
```

---

deere3 *Deviations of an industrial process at Deere \& Co. – Series 3*

Description

Fifty seven consecutive values for the deviation (in 0.0000025 inch units) from a specified target value.

Usage

```
data(deere3)
```

Format

The format is: Time-Series [1:57] from 1 to 57: -500 -1250 -500 -3000 -2375 ...

Source

Personal communication from William F. Fulkerson, Deere \& Co. Technical Center, Moline, Illinois.

Examples

```
data(deere3)
## maybe str(deere3) ; plot(deere3) ...
```

---

detectAO *Additive Outlier Detection*

Description

This function serves to detect whether there are any additive outliers (AO). It implements the test statistic $lambda_{2,t}$ proposed by Chang, Chen and Tiao (1988).

Usage

```
detectAO(object, alpha = 0.05, robust = TRUE)
```

Arguments

`object`	a fitted ARIMA model
`alpha`	family significance level (5% is the default) Bonferroni rule is used to control the family error rate.
`robust`	if true, the noise standard deviation is estimated by mean absolute residuals times sqrt(pi/2). Otherwise, it is the estimated by sqrt(sigma2) from the arima fit.

Value

A list containing the following components:

`ind`	the time indices of potential AO
`lambda2`	the corresponding test statistics

Author(s)

Kung-Sik Chan

References

Chang, I.H., Tiao, G.C. and C. Chen (1988). Estimation of Time Series Parameters in the Presence of Outliers. Technometrics, 30, 193-204.

See Also

`detectIO`

Examples

```
set.seed(12345)
y=arima.sim(model=list(ar=.8,ma=.5),n.start=158,n=100)
y[10]
y[10]=10
y=ts(y,freq=1,start=1)
plot(y,type='o')
acf(y)
pacf(y)
eacf(y)
m1=arima(y,order=c(1,0,0))
m1
detectAO(m1)
detectAO(m1, robust=FALSE)
detectIO(m1)
```

detectIO *Innovative Outlier Detection*

Description

This function serves to detect whether there are any innovative outliers (IO). It implements the test statistic $lambda_{2,t}$ proposed by Chang, Chen and Tiao (1988).

Usage

```
detectIO(object, alpha = 0.05, robust = TRUE)
```

Arguments

`object`	a fitted ARIMA model
`alpha`	family significance level (5% is the default) Bonferroni rule is used to control the family error rate.
`robust`	if true, the noise standard deviation is estimated by mean absolute residuals times sqrt(pi/2). Otherwise, it is the estimated by sqrt(sigma2) from the arima fit.

Value

A list containing the following components:

`ind`	the time indices of potential AO
`lambda1`	the corresponding test statistics

Author(s)

Kung-Sik Chan

References

Chang, I.H., Tiao, G.C. and C. Chen (1988). Estimation of Time Series Parameters in the Presence of Outliers. Technometrics, 30, 193-204.

See Also

`detectIO`

Examples

```
set.seed(12345)
y=arima.sim(model=list(ar=.8,ma=.5),n.start=158,n=100)
y[10]
y[10]=10
y=ts(y,freq=1,start=1)
plot(y,type='o')
acf(y)
pacf(y)
eacf(y)
m1=arima(y,order=c(1,0,0))
m1
detectAO(m1)
detectAO(m1, robust=FALSE)
detectIO(m1)
```

---

eacf *Compute the sample extended acf (ESACF)*

Description

Computes the sample extended acf (ESACF) for the time series stored in z. The matrix of ESACF with the AR order up to ar.max and the MA order up to ma.max is stored in the matrix EACFM.

Usage

```
eacf(z, ar.max = 7, ma.max = 13)
```

Arguments

`z`	the time series data
`ar.max`	maximum AR order; default=7
`ma.max`	maximum MA order; default=13

Value

A list containing the following two components:

`eacf`	a matrix of sample extended ACF
`symbol`	corresponding matrix of symbols indicating the significance of the ESACF

Side effect of the eacf function: The function prints a coded ESACF table with significant values denoted by * and nosignificant values by 0.

Author(s)

Kung-Sik Chan

References

Tsay, R. and Tiao, G. (1984). "Consistent Estimates of Autoregressive Parameters and Extended Sample Autocorrelation Function for Stationary and Nonstationary ARMA Models." Journal of the American Statistical Association, 79 (385), pp. 84-96.

Examples

```
data(arma11.s)
eacf(arma11.s)
```

`eeg` *EEG Data*

Description

An electroencephalogram (EEG) is a noninvasive test used to detect and record the electrical activity generated in the brain. These data were measured at a frequency of 256 per second and came from a patient suffering a seizure. This a portion of a series on the website of Professor Richard Smith, University of North Carolina. His source: Professors Mike West and Andrew Krystal, Duke University.

Usage

```
data(eeg)
```

Format

The format is: ts [1:13000, 1] -3.08 -20.15 -45.05 -69.95 -94.57 ... - attr(*, "dimnames")=List of 2 ..$: NULL ..$: chr "eeg" - attr(*, "tsp")= num [1:3] 2001 15000 1

Source

http://www.stat.unc.edu/faculty/rs/s133/Data/datadoc.html

Examples

```
data(eeg)
## maybe str(eeg) ; plot(eeg) ...
```

`electricity` *Monthly US electricity production / time series*

Description

Monthly U.S. electricity generation (in millions of kilowatt hours) of all types: coal, natural gas, nuclear, petroleum, and wind, 01/1973 - 12/2005.

Usage

```
data(electricity)
```

Format

The format is: 'ts' int [1:396, 1] 160218 143539 148158 139589 147395 161244 173733 177365 156875 154197 ... - attr(*, "dimnames")=List of 2 ..$: NULL ..$: chr "electricity" - attr(*, "tsp")= num [1:3] 1973 2006 12

Source

Source: www.eia.doe.gov/emeu/mer/elect.html

Examples

```
data(electricity)
## maybe str(electricity) ; plot(electricity) ...
```

`euph` *A digitized sound file of a B flat played on a euphonium*

Description

A digitized sound file of about 0.4 seconds of a B flat just below middle C played on a euphonium by one of the authors (JDC), a member of the group Tempered Brass.

Usage

```
data(euph)
```

Format

The format is: Time-Series [1:1105] from 1 to 1105: 0.244 0.635 0.712 0.608 0.317 ...

Examples

```
data(euph)
## maybe str(euph) ; plot(euph) ...
```

explode.s *A simulated explosive AR(1) series*

Description

A simulated AR(1) series with the AR(1) coefficient being 3.

Usage

```
data(explode.s)
```

Format

The format is: Time-Series [1:8] from 1 to 8: 0.63 0.64 3.72 12.67 39.57 ...

Examples

```
data(explode.s)
## maybe str(explode.s) ; plot(explode.s) ...
```

fitted.Arima *Fitted values of an arima model.*

Description

Computes the fitted values of an arima model.

Usage

```
## S3 method for class 'Arima'
fitted(object,...)
```

Arguments

object	a fitted model from the arima function.
...	other arguments; not used here but kept to be consistent with the generic method

Value

fitted values

Author(s)

Kung-Sik Chan

See Also

arima

Examples

```
data(hare)
hare.m1=arima(sqrt(hare),order=c(3,0,0))
fitted(hare.m1)
```

`flow` *Monthly River Flow for the Iowa River*

Description

Flow data (in cubic feet per second) for the Iowa river measured at Wapello, Iowa for the period 09/1958 - 08/2006.

Usage

```
data(flow)
```

Source

http://waterdata.usgs.gov/ia/nwis/sw

Examples

```
data(flow)
## maybe str(flow) ; plot(flow) ...
```

`garch.sim` *Simulate a GARCH process*

Description

Simulate a GARCH process.

Usage

```
garch.sim(alpha, beta, n = 100, rnd = rnorm, ntrans = 100,...)
```

Arguments

`alpha`	The vector of ARCH coefficients including the intercept term as the first element
`beta`	The vector of GARCH coefficients
`n`	sample size
`rnd`	random number generator for the noise; default is normal
`ntrans`	burn-in size, i.e. number of initial simulated data to be discarded
`...`	parameters to be passed to the random number generator

Details

Simulate data from the GARCH(p,q) model: $x_t = \sigma_{t|t-1} e_t$ where $\{e_t\}$ is iid, e_t independent of past $x_{t-s}, s = 1, 2, \ldots$, and

$$\sigma_{t|t-1} = \sum_{j=1}^{p} \beta_j \sigma_{t-j|t-j-1} + \alpha_0 + \sum_{j=1}^{q} \alpha_j x_{t-i}^2$$

Value

simulated GARCH time series of size n.

Author(s)

Kung-Sik Chan

Examples

```
set.seed(1235678)
garch01.sim=garch.sim(alpha=c(.01,.9),n=500)
plot(garch01.sim,type='l', main='',ylab=expression(r[t]),xlab='t')
```

---

gBox *Generalized Portmanteau Tests for GARCH Models*

Description

Perform a goodness-of-fit test for the GARCH model by checking whether the standardized residuals are iid based on the ACF of the absolute residuals or squared residuals.

Usage

```
gBox(model, lags = 1:20, x, method = c("squared", "absolute")[1], plot = TRUE)
```

Arguments

`model`	fitted model from the garch function of the tseries library
`lags`	a vector of maximum ACF lags to be used in the test
`x`	time series data to which the GARCH model is fitted
`method`	"squared": test is based on squared residuals; "absolute": test is based on absolute residuals
`plot`	logical variable, if TRUE, the p-values of the tests are plotted

Value

`lags`	lags in the input
`pvalue`	a vector of p-values of the tests
`method`	method used
`x`	x

Author(s)

Kung-Sik Chan

References

"Time Series Analysis, with Applications in R" by J.D. Cryer and K.S. Chan

Examples

```
require(tseries) # need to uncomment this line when running the example
data(CREF)
r.cref=diff(log(CREF))*100
m1=tseries::garch(x=r.cref,order=c(1,1))
summary(m1)
gBox(m1,x=r.cref,method='squared')
```

`gold` *Gold Price / time series*

Description

Daily price of gold (in \$ per troy ounce) for the 252 trading days of 2005

Usage

```
data(gold)
```

Format

The format is: Time-Series [1:252] from 1 to 252: 427 426 426 423 421 ...

Source

www.lbma.org.uk/2005dailygold.htm

Examples

```
data(gold)
## maybe str(gold) ; plot(gold) ...
```

---

google *Daily returns of the google stock*

Description

Daily returns of the google stock from 08/20/04 - 09/13/06.

Usage

```
data(google)
```

Format

The format is: Time-Series [1:521] from 1 to 521: 0.0764 0.0100 -0.0423 0.0107 0.0179 ...

Source

`http://finance.yahoo.com/q/hp?s=GOOG`

Examples

```
data(google)
## maybe str(google) ; plot(google) ...
```

hare *Canadian hare data/ time series*

Description

Annual number of hare data.

Usage

```
data(hare)
```

Format

The format is: Time-Series [1:31] from 1905 to 1935: 50 20 20 22 27 50 55 78 70 59 ...

Details

These are yearly hare abundances for the main drainage of the Hudson Bay, based on trapper questionnaires.

Source

MacLulich, D. A. (1937) Fluctuations in the Number of the Varying Hare (Lepus americanus) (Univ. of Toronto Press, Toronto)

References

Stenseth, N. C., Falck, W., Bjornstad, O. N. and Krebs. C. J. (1997) Population regulation in snowshoe hare and Canadian lynx: Asymmetric food web configurations between hare and lynx. Proc. Natl. Acad. Sci., 94, 5147-5152.

Examples

```
data(hare)
```

harmonic *Construct harmonic functions for fitting harmonic trend model*

Description

The function creates a matrix of the first m pairs of harmonic functions for fitting a harmonic trend (cosine-sine trend, Fourier regresssion) models with the response being x, a time series.

Usage

```
harmonic(x, m = 1)
```

Arguments

`x`	a time series
`m`	the number of pairs of harmonic functions to be created; 2m must be less than or equal to the frequency of x

Value

a matrix consisting of $\cos(2k\pi t), \sin(2k\pi t), k = 1, 2, ..., m,$ excluding any zero functions.

Author(s)

Kung-Sik Chan

See Also

`season`

Examples

```
data(tempdub)
# first creates the first pair of harmonic functions and then fit the model
har.=harmonic(tempdub,1)
model4=lm(tempdub~har.)
summary(model4)
```

`hours` *Average hours worked in US manufacturing sector / time series*

Description

Average hours worked (times 10) in U.S. manufacturing sector, from 07/1982 - 06/1987

Usage

```
data(hours)
```

Format

The format is: Time-Series [1:60] from 1983 to 1987: 389 390 389 390 393 397 392 388 396 398 ...

Source

Cryer, J. D. Time Series Analysis, Duxbury Press, 1986.

Examples

```
data(hours)
## maybe str(hours) ; plot(hours) ...
```

ima22.s — *Simulated IMA(2,2) series / time series*

Description

A simulated IMA(2,2) series with theta1=1 and theta2=-0.6

Usage

```
data(ima22.s)
```

Format

The format is: Time-Series [1:62] from 1 to 62: 0.00000 0.00000 -0.00569 2.12404 2.15337 ...

Examples

```
data(ima22.s)
## maybe str(ima22.s) ; plot(ima22.s) ...
```

JJ — *Quarterly earnings per share for the Johnson \& Johnson Company*

Description

Quarterly earnings per share for 1960Q1 to 1980Q4 of the U.S. company, Johnson \& Johnson, Inc.

Usage

```
data(JJ)
```

Format

The format is: Time-Series [1:84] from 1960 to 1981: 0.71 0.63 0.85 0.44 0.61 0.69 0.92 0.55 0.72 0.77 ...

Source

http://www.stat.pitt.edu/stoffer/tsa2/

Examples

```
data(JJ)
## maybe str(JJ) ; plot(JJ) ...
```

Keenan.test — *Keenan's one-degree test for nonlinearity*

Description

Carry out Keenan's 1-degree test for nonlinearity against the null hypothesis that the time series follows some AR process.

Usage

```
Keenan.test(x, order, ...)
```

Arguments

`x`	time series
`order`	working AR order; if missing, it is estimated by minimizing AIC via the ar function.
`...`	user-supplied options to the ar function.

Details

The test is designed to have optimal local power against depature from the linear autoregressive function in the direction of the square of the linear autoregressive function.

Value

A list containing the following components

`test.stat`	The observed test statistic
`p.value`	p-value of the test
`order`	working AR order

Author(s)

Kung-Sik Chan

References

Keenan, D. M. (1985), A Tukey nonadditivity-type test for time series Nonlinearity, Biometrika, 72, 39-44.

See Also

`Tsay.test`,`tlrt`

Examples

```
data(spots)
Keenan.test(sqrt(spots))
```

`kurtosis` *Kurtosis*

Description

Computes the Kurtosis.

Usage

```
kurtosis(x, na.rm = FALSE)
```

Arguments

`x`	data
`na.rm`	logical variable, if true, missing values are excluded from analysis

Details

Given data $x_1, x_2, \ldots, x_n$, the sample kurtosis is defined by the formula:

$$\frac{\sum_{i=1}^{n}(x_i-\bar{x})^4/n}{(\sum_{i=1}^{n}(x_i-\bar{x})^2/n)^2}-3.$$

Value

The function returns the kurtosis of the data.

Author(s)

Kung-Sik Chan

Examples

```
data(CREF)
r.cref=diff(log(CREF))*100
kurtosis(r.cref)
```

`lagplot` *Lagged Regression Plot*

Description

Computes and plots the nonparametric regression function of a time series against its various lags.

Usage

```
lagplot(x, lag.max = 6, deg = 1, nn = 0.7, method = c("locfit", "gam", "both")[1])
```

Arguments

`x`	time series
`lag.max`	maximum lag
`deg`	degree of local polynomial, needed only for the locfit method
`nn`	fraction of nearest data contained in a window, needed only for the locfit method
`method`	Two methods for nonparametric estimation: "locfit" is the default which uses the local polynomial approach via the locfit library to estimate the conditional mean function of $E(X_t \vert X_{t-k} = x)$ for $1 \leq k \leq lag.max$; Another method is GAM, via the mgcv library.

Value

Side effects: The nonparametric lagged regression functions are plotted lag by lag, with the raw data superimposed on the plots.

Author(s)

Kung-Sik Chan

References

"Time Series Analysis, with Applications in R" by J.D. Cryer and K.S. Chan

Examples

```
set.seed(2534567)
par(mfrow=c(3,2))
y=arima.sim(n=61,model=list(ar=c(1.6,-0.94),ma=-0.64))
# lagplot(y)
```

`larain` *Annual rainfall in Los Angeles / time series*

Description

Annual precipitation (in inches) in Los Angeles, 1878-1992.

Usage

```
data(larain)
```

Format

The format is: Time-Series [1:115] from 1778 to 1892: 20.86 17.41 18.65 5.53 10.74 ...

Source

Personal communication from Professor Donald Bentley, Pomona College, Claremont, California. For more data see `http://www.wrh.noaa.gov/lox/climate/cvc.php`

Examples

```
data(larain)
## maybe str(larain) ; plot(larain) ...
```

`LB.test` *Portmanteau Tests for Fitted ARIMA models*

Description

This function modifies the Box.test function in the stats package, and it computes the Ljung-Box or Box-Pierce tests checking whether or not the residuals appear to be white noise.

Usage

```
LB.test(model, lag = 12, type = c("Ljung-Box", "Box-Pierce"), no.error = FALSE,
 omit.initial = TRUE)
```

Arguments

`model`	model fit from the arima function
`lag`	number of lags of the autocorrelation of the residuals to be included in the test statistic. (default=12)
`type`	either Ljung-Box or Box-Pierce
`no.error`	a system variable; normally it is not changed
`omit.initial`	if true, (d+Ds) initial residuals are omitted from the test

Value

a list:

`statistics`	test statistic
`p.value`	p-value
`parameter`	d.f. of the Chi-square test
`lag`	no of lags

Author(s)

Kung-Sik Chan, based on A. Trapletti's work on the Box.test function in the stats package

References

Box, G. E. P. and Pierce, D. A. (1970), Distribution of residual correlations in autoregressive-integrated moving average time series models. Journal of the American Statistical Association, 65, 15091526.

Ljung, G. M. and Box, G. E. P. (1978), On a measure of lack of fit in time series models. Biometrika 65, 553564.

Examples

```
data(color)
m1.color=arima(color,order=c(1,0,0))
LB.test(m1.color)
```

`ma1.1.s` *A simulated MA(1) series / time series*

Description

A simulated MA(1) series with the MA(1) coefficient equal to 0.9.

Usage

```
data(ma1.1.s)
```

Format

The format is: Time-Series [1:120] from 1 to 120: 0.182 -0.748 -0.355 1.014 -2.363 ...

Details

The model is $Y(t) = e(t) - 0.9e(t-1)$ where the e's are iid standard normal.

Examples

```
data(ma1.1.s)
## maybe str(ma1.1.s) ; plot(ma1.1.s) ...
```

`ma1.2.s` *A simulated MA(1) series / time series*

Description

A simulated MA(1) series with the MA(1) coefficient equal to -0.9.

Usage

```
data(ma1.2.s)
```

Format

The format is: Time-Series [1:120] from 1 to 120: 1.511 1.821 0.957 -1.538 -2.888 ...

Details

The model is $Y(t) = e(t) + 0.9e(t-1)$ where the e's are iid standard normal.

Examples

```
data(ma1.2.s)
## maybe str(ma1.2.s) ; plot(ma1.2.s) ...
```

`ma2.s` *A simulated MA(2) series*

Description

A simulated MA(2) series with MA coefficients being 1 and -0.6.

Usage

```
data(ma2.s)
```

Format

The format is: Time-Series [1:120] from 1 to 120: -0.4675 0.0815 0.9938 -2.6959 2.8116 ...

Details

The model is $Y(t) = e(t) - e(t-1) + 0.6 * e(t-2)$ where the e's are iid standard normal random variables.

Examples

```
data(ma2.s)
## maybe str(ma2.s) ; plot(ma2) ...
```

McLeod.Li.test — *McLeod-Li test*

Description

Perform the McLeod-Li test for conditional heteroscedascity (ARCH).

Usage

```
McLeod.Li.test(object, y, gof.lag, col = "red", omit.initial = TRUE,
plot = TRUE, ...)
```

Arguments

`object`	a fitted Arima model, ususally the output from the arima function. If supplied, then the Mcleod-Li test is applied to the residuals of the model, and the y-argument is ignored.
`y`	time series data with which one wants to test for the presence of conditional heteroscedascity
`gof.lag`	maximum number of lags for which the test is carried out.
`col`	color of the reference line
`omit.initial`	suppress the initial (d+Ds) residuals if set to be TRUE
`plot`	suppress plotting if set to be FALSE
`...`	other arguments to be passed to the plot function

Details

The test checks for the presence of conditional heteroscedascity by computing the Ljung-Box (portmanteau) test with the squared data (if y is supplied and object suppressed) or with the squared residuals from an arima model (if an arima model is passed to the function via the object argument.)

Value

`pvlaues`	the vector of p-values for the Ljung-Box test statistics computed using the first m lags of the ACF of the squared data or residuals, for m ranging from 1 to gof.lag.

Author(s)

Kung-Sik Chan

References

McLeod, A. I. and W. K. Li (1983). Diagnostic checking ARMA time series models using squared residual autocorrelations. Journal of Time Series Analysis, 4, 269273.

Examples

```
data(CREF)
r.cref=diff(log(CREF))*100
McLeod.Li.test(y=r.cref)
```

---

milk *Monthly Milk Production*

Description

Average monthly milk production per cow in the US, 01/1994 - 12/2005

Usage

```
data(milk)
```

Format

The format is: 'ts' int [1:144, 1] 1343 1236 1401 1396 1457 1388 1389 1369 1318 1354 ... - attr(*, "dimnames")=List of 2 ..$: NULL ..$: chr "milk" - attr(*, "tsp")= num [1:3] 1994 2006 12

Examples

```
data(milk)
str(milk)
plot(milk)
```

---

oil.price *Monthly Oil Price / time series*

Description

Monthly spot price for crude oil, Cushing, OK (in U.S. dollars per barrel), 01/1986 - 01/2006.

Usage

```
data(oil.price)
```

Format

The format is: Time-Series [1:241] from 1986 to 2006: 22.9 15.4 12.6 12.8 15.4 ...

Source

tonto.eia.doe.gov/dnav/pet/hist/rwtcM.htm

Examples

```
data(oil.price)
## maybe str(oil.price) ; plot(oil.price) ...
```

---

oilfilters — *Monthly sales to dealers of a specialty oil filter/time series*

Description

Monthly wholesale specialty oil filters sales, Deere \& Co, 07/1983 - 06/1987.

Usage

```
data(oilfilters)
```

Format

The format is: Time-Series [1:48] from 1984 to 1987: 2385 3302 3958 3302 2441 3107 5862 4536 4625 4492 ... - attr(*, "freq")= num 12 - attr(*, "start")= num [1:2] 1987 7

Source

Data courtesy of William F. Fulkerson, Deere \& Company, Technical Center, Moline, Illinois.

Examples

```
data(oilfilters)
## maybe str(oilfilters) ; plot(oilfilters) ...
```

---

periodogram — *Computing the periodogram*

Description

This is a wrapper that computes the periodogram

Usage

```
periodogram(y,log='no',plot=TRUE,ylab="Periodogram", xlab="Frequency",lwd=2,...)
```

Arguments

`y`	A univariate time series
`log`	if set to "yes", the periodogram is plotted on the log-scale; default="no"
`plot`	The periodogram is plotted if it is set to be TRUE which is the default
`ylab`	label on the y-axis
`xlab`	label on the x-axis
`lwd`	thickness of the periodogram lines
`...`	other arguments to be passed to the plot function

Value

A list that contains the following elements:

`freq`	Vector of frequencies at which the spectral density is estimated. (Possibly approximate Fourier frequencies.
`spec`	Vector of estimates of the periodogram at frequencies corresponding to freq.

References

Bloomfield, P. (1976) Fourier Analysis of Time Series: An Introduction. Wiley.

Brockwell, P. J. and Davis, R. A. (1991) Time Series: Theory and Methods. Second edition. Springer.

Examples

```
data(star)
plot(star,xlab='Day',ylab='Brightness')
periodogram(star,ylab='Variable Star Periodogram');  abline(h=0)
```

---

plot.Arima — *Compute and Plot the Forecasts Based on a Fitted Time Series Model*

Description

Plots the time series data and its predictions with 95% prediction bounds.

Usage

```
## S3 method for class 'Arima'
plot(x, n.ahead = 12, col = "black", ylab = object$series,
lty = 2, n1, newxreg, transform, Plot=TRUE, ...)
```

Arguments

`x`	a fitted arima model
`n.ahead`	number of prediction steps ahead (default=12)
`col`	color of the prediction bounds
`ylab`	label of the y-axis
`lty`	line type of the point predictor; default=dashed lines
`n1`	starting time point of the plot (default=earliest time point)
`newxreg`	a matrix of covariate(s) over the period of prediction
`transform`	function used to transform the forecasts and their prediction bounds; if missing, no transformation will be carried out. This option is useful if the model was fitted to the transformed data and it is desirable to obtain the forecasts on the original scale. For example, if the model was fitted with the logarithm of the data, then transform = exp will plot the forecasts and their prediction bounds on the original scale.
`Plot`	Plotting will be suppressed if Plot is set to be FALSE; default is TRUE
`...`	additional parameters passed to the plot function

Value

Side effects of the function: plot the forecasts and their 95% prediction bounds, unless Plot is set to be FALSE. The part of the observed series is plotted with all data plotted as open circles and linked by a smooth line. By default the predicted values are plotted as open circles joined up by a dashed line. The plotting style of the predicted values can be altered by supplying relevant plotting options, e.g specifying the options type='o', pch=19 and lty=1 will plot the predicted values as solid circles that are overlaid on the connecting smooth solid line. The prediction limits are plotted as dotted lines, with default color being black. However, the prediction limits can be drawn in other colors. For example, setting col='red' paints the prediction limits in red. An interesting use of the col argument is setting col=NULL which has the effect of not drawing the prediction limits.

The function returns an invisible list containing the following components.

`pred`	the time series of predicted values
`lpi`	the corresponding lower 95% prediction limits
`upi`	the corresponding upper 95% prediction limits

Author(s)

Kung-Sik Chan

Examples

```
data(oil.price)
oil.IMA11alt=arima(log(oil.price),order=c(0,1,1),
# create the design matrix of the covariate for prediction
xreg=data.frame (constant=seq(oil.price)))
n=length(oil.price)
n.ahead=24
```

```
newxreg=data.frame(constant=(n+1):(n+n.ahead))
# do the prediction and plot the results
plot(oil.IMA11alt,n.ahead=n.ahead,newxreg=newxreg,
ylab='Log(Oil Price)',xlab='Year',n1=c(2000,1))
# do the same thing but on the orginal scale
plot(oil.IMA11alt,n.ahead=n.ahead,newxreg=newxreg,
ylab='Oil Price',xlab='Year',n1=c(2000,1),transform=exp,pch=19, lty=1,type='o')
# Setting pch=19 plots the predicted values as solid circles.
res=plot(oil.IMA11alt,n.ahead=n.ahead,newxreg=newxreg,
ylab='Oil Price',xlab='Year',n1=c(2000,1),transform=exp,pch=19,col=NULL)
# Setting col=NULL will make the prediction bands invisible. Try col='red'.
res
# prints the predicted values and their 95% prediction limits.
```

---

plot.armasubsets *Plot the Best Subset ARMA models*

Description

This function is adapted from the plot.regsubsets function of the leaps package, and its main use is to plot the output from the armasubsets function.

Usage

```
## S3 method for class 'armasubsets'
plot(x, labels = obj$xnames, main = NULL,
scale = c("BIC", "AICc", "AIC", "Cp", "adjR2", "R2"),
col = gray(c(seq(0.4, 0.7, length = 10), 0.9)), draw.grid = TRUE,
axis.at.3 = TRUE, ...)
```

Arguments

`x`	an object of class armasubsets
`labels`	variable names
`main`	title for plot
`scale`	which summary statistic to use for ordering plots
`col`	the last color should be close to but distinct from white
`draw.grid`	a logical argument; if it is true (default), gray grid lines are superimposed on the graph.
`axis.at.3`	a logical argument; if if it is true (default), the x-labels are drawn on the upper horizontal axis.
`...`	other arguments

Value

Plot the few best subset ARMA models.

Author(s)

Kung-Sik Chan, based on previoud work by Thomas Lumley and Merlise Clyde

See Also

armasubsets

Examples

```
set.seed(53331)
test=arima.sim(model=list(ar=c(rep(0,11),.8),ma=c(rep(0,11),0.7)),n=120)
res=armasubsets(y=test,nar=14,nma=14,y.name='test',ar.method='ols')
plot(res)
```

---

plot1.acf *Plot1*

Description

A workhorse function for the acf function in the TSA pacakage.

Author(s)

Kung-Sik Chan

---

predict.TAR *Prediction based on a fitted TAR model*

Description

Predictions based on a fitted TAR model. The errors are assumed to be normally distributed. The predictive distributions are approximated by simulation.

Usage

```
## S3 method for class 'TAR'
predict(object, n.ahead = 1, n.sim = 1000,...)
```

Arguments

`object`	a fitted TAR model from the tar function
`n.ahead`	number of prediction steps ahead
`n.sim`	simulation size
`...`	other arguments; not used here but kept for consistency with the generic method

Value

fit
: a vector of medians of the 1-step to n.ahead-step predictive distributions

pred.interval
: a matrix whose i-th row consists of the 2.5 and 97.5 precentiles of the i-step predictive distribution

pred.matrix
: a matrix whose j-th column consists of all simulated values from the j-step predictive distribution

Author(s)

Kung-Sik Chan

References

"Time Series Analysis, with Applications in R" by J.D. Cryer and K.S. Chan

See Also

tar

Examples

```
data(prey.eq)
prey.tar.1=tar(y=log(prey.eq),p1=4,p2=4,d=3,a=.1,b=.9,print=TRUE)
set.seed(2357125)
pred.prey=predict(prey.tar.1,n.ahead=60,n.sim=1000)
yy=ts(c(log(prey.eq),pred.prey$fit),frequency=1,start=1)
plot(yy,type='n',ylim=range(c(yy,pred.prey$pred.interval)),ylab='Log Prey',
xlab=expression(t))
lines(log(prey.eq))
lines(window(yy, start=end(prey.eq)[1]+1),lty=2)
lines(ts(pred.prey$pred.interval[2,],start=end(prey.eq)[1]+1),lty=2)
lines(ts(pred.prey$pred.interval[1,],start=end(prey.eq)[1]+1),lty=2)
```

---

prescrip — *Cost per prescription / time series*

Description

Monthly U.S. average prescription costs for the months 08/1986 - 03/1992.

Usage

```
data(prescrip)
```

Format

The format is: Time-Series [1:68] from 1987 to 1992: 14.5 14.7 14.8 14.6 14.3 ...

Source

Frees, E. W., Data Analysis Using Regression Models, Prentice Hall, 1996.

Examples

```
data(prescrip)
## maybe str(prescrip) ; plot(prescrip) ...
```

`prewhiten` *Prewhiten a Bivariate Time Series, and Compute and Plot Their Sample Cross-Correlation Function*

Description

The bivariate time series are prewhitened according to an AR model fitted to the x-component of the bivariate series. Alternatively, if an ARIMA model is provided, it will be used to prewhiten both series. The CCF of the prewhitened bivariate series is then computed and plotted.

Usage

```
prewhiten(x, y, x.model = ar.res,ylab="CCF", ...)
```

Arguments

`x`	first component series
`y`	second component series
`x.model`	an ARIMA model; if provided, it is used to prewhiten both series. Otherwise, an AR model is fitted to the x-series and used to prewhiten both series.The AR order is chosen by minimizing the AIC and the fit carried out by the ar.ols function.
`ylab`	label of y-axis; default is "CCF"
`...`	additional parameters to be passed to the ar.ols and the ccf function.

Value

A list containing the following components:

`ccf`	Output from the ccf function on the prewhitened data.
`ar`	The AR model fit to the x-series, or x.model if it is provided.

Author(s)

Kung-Sik Chan

Examples

```
data(milk)
data(electricity)
milk.electricity=ts.intersect(milk,log(electricity))
plot(milk.electricity,yax.flip=TRUE,main='')
ccf(as.numeric(milk.electricity[,1]),as.numeric(milk.electricity[,2]),
main='milk & electricity',ylab='CCF')
me.dif=ts.intersect(diff(diff(milk,12)),diff(diff(log(electricity),12)))
prewhiten(as.numeric(me.dif[,1]),as.numeric(me.dif[,2]),
,ylab='CCF' )
```

---

prey.eq *Prey series / time series*

Description

The stationary part of the Didinium series in the veilleux data frame.

Usage

```
data(prey.eq)
```

Format

The format is: Time-Series [1:57] from 7 to 35: 26.9 53.2 65.6 81.2 143.9 ...

See Also

veilleux

Examples

```
data(prey.eq)
## maybe str(prey.eq) ; plot(prey.eq) ...
```

---

qar.sim *Simulate a first-order quadratic AR model*

Description

Simulates a first-order quadratic AR model with normally distributed noise.

Usage

```
qar.sim(const = 0, phi0 = 0, phi1 = 0.5, sigma = 1, n = 20, init = 0)
```

Arguments

`const`	intercept
`phi0`	coefficient of the lag 1
`phi1`	coefficient of the squared lag 1
`sigma`	noise standard deviation
`n`	sample size
`init`	number of burn-in values

Details

The quadratic AR(1) model specifies that

$$Y_t = \mathrm{const} + \phi_0 Y_{t-1} + \phi_1 Y_{t-1}^2 + e_t$$

where e_t are iid normally distributed with zero mean and standard deviation σ. If $\sigma = 0$, the model is deterministic.

Value

A simulated series from the quadratic AR(1) model, as a vector

Author(s)

Kung-Sik Chan

See Also

`tar.sim`

Examples

```
set.seed(1234567)
plot(y=qar.sim(n=15,phi1=.5,sigma=1),x=1:15,type='l',ylab=expression(Y[t]),xlab='t')
y=qar.sim(n=100,const=0.0,phi0=3.97, phi1=-3.97,sigma=0,init=.377)
plot(y,x=1:100,type='l',ylab=expression(Y[t]),xlab='t')
acf(y,main='')
```

`retail` *U.K. retail sales / time series*

Description

Monthly total UK (United Kingdom) retail sales (non-food stores in billions of pounds), 01/1983 - 12/1987.

Usage

```
data(retail)
```

Format

The format is: Time-Series [1:60] from 1983 to 1988: 81.3 78.9 93.8 94 97.8 1.6 99.6 1.2 98 1.7 ...

Source

www.statistics.gov.uk/statbase/TSDdownload1.asp

Examples

```
data(retail)
## maybe str(retail) ; plot(retail) ...
```

---

robot *The distance of a robot from a desired position / time series*

Description

Final position in the x direction of an industrial robot put through a series of planned exercises many times.

Usage

```
data(robot)
```

Format

The format is: Time-Series [1:324] from 1 to 324: 0.0011 0.0011 0.0024 0 -0.0018 0.0055 0.0055 -0.0015 0.0047 -1e-04 ...

Source

Personal communication from William F. Fulkerson, Deere \& Co. Technical Center, Moline, Illinois.

Examples

```
data(robot)
## maybe str(robot) ; plot(robot) ...
```

`rstandard.Arima` *Compute the Standardized Residuals from a Fitted ARIMA Model*

Description

Computes the internally standardized residuals from a fitted ARIMA model.

Usage

```
## S3 method for class 'Arima'
rstandard(model,...)
```

Arguments

`model`	model fitted by the arima function
`...`	not used; kept here for consistency with the generic method

Details

residuals/(error std. dev.)

Value

time series of standarized residuals

Examples

```
data(oil.price)
m1.oil=arima(log(oil.price),order=c(0,1,1))
plot(rstandard(m1.oil),ylab='Standardized residuals',type='l')
abline(h=0)
```

`runs` *Runs test*

Description

Test the independence of a sequence of random variables by checking whether there are too many or too few runs above (or below) the median.

Usage

```
runs(x,k=0)
```

Arguments

`x`	time series
`k`	the value above or below which runs are counted; default is zero, so data is assumed to have zero median

Details

The runs test examines the data in sequence to look for patterns that would give evidence against independence. Runs above or below k are counted. A small number of runs would indicate that neighboring values are positively dependent and tend to hang together over time. On the other hand, too many runs would indicate that the data oscillate back and forth across their median of zero. Then neighboring residuals are negatively dependent. So either too few or too many runs lead us to reject independence. When applied to residuals, the runs test is useful for model diagnostics.

Value

`pvalue`	p-value of the test
`observed.runs`	observed number of runs
`expected.runs`	expected number of runs
`n1`	number of data less than or equal to k
`n2`	number of data above k

Author(s)

Kung-Sik Chan

Examples

```
data(tempdub)
month.=season(tempdub) # the period sign is included to make the printout from
# the following command clearer.
model3=lm(tempdub~month.) # intercept is automatically included so one month (Jan) is dropped
summary(model3)
runs(rstudent(model3))
```

`rwalk` *A simulated random walk / Time series*

Description

A simulated random walk with standard normal increments

Usage

```
data(rwalk)
```

Examples

```
data(rwalk)
## maybe str(rwalk) ; plot(rwalk) ...
```

---

season — *Extract the season info from a time series*

Description

Extract the season info from a equally spaced time series and create a vector of the season info. For example for monthly data, the function outputs a vector containing the months of the data.

Usage

```
season(x, labels)
```

Arguments

`x`	a time series
`labels`	the user supplied labels for the seasons

Details

The time series must have frequency greater than 1, otherwise the function will stop and issue an error message. If labels is missing, labels will be set as follows: It is set to be c("1Q","2Q","3Q","4Q) if the frequency of x equals 4, c("January",...,"December") if the frequency equals 12, and c("Monday",...,"Sunday") if frequency equals 7. Otherwise, it is set to be c("S1",...)

Value

An invisible vector containing the seasons of the data

Author(s)

Kung-Sik Chan

See Also

harmonic

Examples

```
data(tempdub)
month.=season(tempdub) # the period sign is included to make the printout from
# the commands two line below clearer; ditto below.
model2=lm(tempdub~month.-1) # -1 removes the intercept term
summary(model2)
```

`skewness` *Skewness*

Description

Computes the skewness of the data

Usage

```
skewness(x, na.rm = FALSE)
```

Arguments

`x`	data
`na.rm`	logical variable, if true, missing values are excluded from analysis

Details

Given data $x_1, x_2, \ldots, x_n$, the sample skewness is defined by the formula:

$$\frac{\sum_{i=1}^{n}(x_i - \bar{x})^3/n}{(\sum_{i=1}^{n}(x_i - \bar{x})^2/n)^{3/2}}.$$

Value

The function returns the skewness of the data.

Author(s)

Kung-Sik Chan

Examples

```
data(CREF)
r.cref=diff(log(CREF))*100
skewness(r.cref)
```

`SP` *Quarterly Standard \& Poor's Composite Index of stock price values / time series*

Description

Quarterly S&P Composite Index, 1936Q1 - 1977Q4.

Usage

```
data(SP)
```

Format

The format is: Time-Series [1:168] from 1936 to 1978: 149 148 160 172 179 ...

Source

Frees, E. W., Data Analysis Using Regression Models, Prentice Hall, 1996.

Examples

```
data(SP)
## maybe str(SP) ; plot(SP) ...
```

`spec` *Computing the spectrum*

Description

This is a wrapper that allows the user to invoke either the spec.pgram function or the spec.ar function in the stats pacakge. Note that the seasonal attribute of the data, if it exists, will be removed, for our preferred way of presenting the output.

Usage

```
spec(x, taper = 0, detrend = FALSE, demean = TRUE, method = c("pgram",
  "ar")[1], ci.plot = FALSE, ylim = range(c(lower.conf.band, upper.conf.band)),
   ...)
```

Arguments

A list that contains the following:

A univariate or multivariate time series

`xtaper`	amount of taper; 0 is the default
`detrend`	logical; if True, the data are detrended; default is False
`demean`	logical; if True, the data are centered; default is True
`method`	String specifying the method used to estimate the spectral density. Allowed methods are "pgram" (the default) and "ar".
`ci.plot`	logical; if True, the 95% confidence band will be plotted.
`ylim`	Plotting parameter vector specifying the minimum and maximum of the y-axis.
`...`	other arguments

Value

The output is from the spec.pgram function or spec.ar function, and the following description of the output is taken from the help manual of the spec function in the stats package. An object of class "spec", which is a list containing at least the following components:

`freq`	Vector of frequencies at which the spectral density is estimated. (Possibly approximate Fourier frequencies.) The units are the reciprocal of cycles per unit time (and not per observation spacing): see Details below.
`spec`	Vector (for univariate series) or matrix (for multivariate series) of estimates of the spectral density at frequencies corresponding to freq. coh NULL for univariate series. For multivariate time series, a matrix containing the squared coherency between different series. Column i + (j - 1) * (j - 2)/2 of coh contains the squared coherency between columns i and j of x, where i < j.
`phase`	NULL for univariate series. For multivariate time series a matrix containing the cross-spectrum phase between different series. The format is the same as coh.
`series`	The name of the time series.
`snames`	For multivariate input, the names of the component series.
`method`	The method used to calculate the spectrum.

The result is returned invisibly if plot is true.

References

Bloomfield, P. (1976) Fourier Analysis of Time Series: An Introduction. Wiley.

Brockwell, P. J. and Davis, R. A. (1991) Time Series: Theory and Methods. Second edition. Springer.

Venables, W. N. and Ripley, B. D. (2002) Modern Applied Statistics with S-PLUS. Fourth edition. Springer. (Especially pages 3927.)

Examples

```
set.seed(271435); n=200; phi=-0.6
y=arima.sim(model=list(ar=phi),n=n)
k=kernel('daniell',m=15)
sp=spec(y,kernel=k,main='',sub='',xlab='Frequency',
ylab='Log(Smoothed Sample Spectrum)',ci.plot=TRUE,ci.col='black')
lines(sp$freq,ARMAspec(model=list(ar=phi),sp$freq,plot=FALSE)$spec,lty=4)
abline(h=0)
```

---

spots *Relative annual sunspot number / time series*

Description

Annual American (relative) sunspot numbers collected from 1945 to 2007. The annual (relative) sunspot number is a weighted average of solar activities measured from a network of observatories.

Usage

```
data(spots)
```

Format

The format is: Time-Series [1:61] from 1945 to 2005: 32.3 99.9 170.9 166.6 174.1 ...

Source

http://www.ngdc.noaa.gov/stp/SOLAR/ftpsunspotnumber.html#american

References

"Time Series Analysis, with Applications in R" by J.D. Cryer and K.S. Chan

Examples

```
data(spots)
## maybe str(spots) ; plot(spots) ...
```

spots1 — *Annual international sunspot numbers*

Description

Annual international sunspot numbers, NOAA National Geophysical Data Center, 1700 - 2005.

Usage

```
data(spots1)
```

Format

The format is: ts [1:306, 1] 5 11 16 23 36 58 29 20 10 8 ... - attr(*, "dimnames")=List of 2 ..$: NULL ..$: chr "spots" - attr(*, "tsp")= num [1:3] 1700 2005 1

Source

`ftp://ftp.ngdc.noaa.gov/STP/SOLAR_DATA/SUNSPOT_NUMBERS/YEARLY.PLT`

Examples

```
data(spots1)
## maybe str(spots1) ; plot(spots1) ...
```

star — *Star Brightness*

Description

Brightness (magnitude) of a particular star at midnight on 600 consecutive nights.

Usage

```
data(star)
```

Source

Whittaker, E. T. and Robinson, G., (1924). The Calculus of Observations. London: Blackie and Son.

Examples

```
data(star)
## maybe str(star) ; plot(star) ...
data(star)
plot(star,xlab='Day',ylab='Brightness')
```

summary.armasubsets *Summary of output from the armasubsets function*

Description

Add the calculation of AIC and AICc. See the help manual of regsubsets function of the leaps package

Usage

```
## S3 method for class 'armasubsets'
summary(object, all.best = TRUE, matrix = TRUE, matrix.logical = FALSE,
 df = NULL, ...)
```

Arguments

`object`	armasubsets object
`all.best`	Show all the best subsets or just one of each size
`matrix`	Show a matrix of the variables in each model or just summary statistics
`matrix.logical`	With matrix=TRUE, the matrix is logical TRUE/FALSE or string "*"/code" "
`df`	Specify a number of degrees of freedom for the summary statistics. The default is n-1
`...`	Other arguments for future methods

Author(s)

Kung-Sik Chan, based on previous work of Thomas Lumley

tar *Estimation of a TAR model*

Description

Estimation of a two-regime TAR model.

Usage

```
tar(y, p1, p2, d, is.constant1 = TRUE, is.constant2 = TRUE, transform = "no",
 center = FALSE, standard = FALSE, estimate.thd = TRUE, threshold,
method = c("MAIC", "CLS")[1], a = 0.05, b = 0.95, order.select = TRUE, print = FALSE)
```

Arguments

y	time series
p1	AR order of the lower regime
p2	AR order of the upper regime
d	delay parameter
is.constant1	if True, intercept included in the lower regime, otherwise the intercept is fixed at zero
is.constant2	similar to is.constant1 but for the upper regime
transform	available transformations: "no" (i.e. use raw data), "log", "log10" and "sqrt"
center	if set to be True, data are centered before analysis
standard	if set to be True, data are standardized before analysis
estimate.thd	if True, threshold parameter is estimated, otherwise it is fixed at the value supplied by threshold
threshold	known threshold value, only needed to be supplied if estimate.thd is set to be False.
method	"MAIC": estimate the TAR model by minimizing the AIC; "CLS": estimate the TAR model by the method of Conditional Least Squares.
a	lower percent; the threshold is searched over the interval defined by the a*100 percentile to the b*100 percentile of the time-series variable
b	upper percent
order.select	If method is "MAIC", setting order.select to True will enable the function to further select the AR order in each regime by minimizing AIC
print	if True, the estimated model will be printed

Details

The two-regime Threshold Autoregressive (TAR) model is given by the following formula:

$$Y_t = \phi_{1,0} + \phi_{1,1} Y_{t-1} + \ldots + \phi_{1,p} Y_{t-p_1} + \sigma_1 e_t, \text{ if } Y_{t-d} \le r$$

$$Y_t = \phi_{2,0} + \phi_{2,1} Y_{t-1} + \ldots + \phi_{2,p_2} Y_{t-p} + \sigma_2 e_t, \text{ if } Y_{t-d} > r.$$

where r is the threshold and d the delay.

Value

A list of class "TAR" which can be further processed by the by the predict and tsdiag functions.

Author(s)

Kung-Sik Chan

References

Tong, H. (1990) "Non-linear Time Series, a Dynamical System Approach," Clarendon Press Oxford

"Time Series Analysis, with Applications in R" by J.D. Cryer and K.S. Chan

See Also

predict.TAR, tsdiag.TAR, tar.sim, tar.skeleton

Examples

```
data(prey.eq)
prey.tar.1=tar(y=log(prey.eq),p1=4,p2=4,d=3,a=.1,b=.9,print=TRUE)
```

---

tar.sim *Simulate a two-regime TAR model*

Description

Simulate a two-regime TAR model.

Usage

```
tar.sim(object, ntransient = 500, n = 500, Phi1, Phi2, thd, d, p, sigma1,
sigma2, xstart = rep(0, max(p,d)), e)
```

Arguments

object	a TAR model fitted by the tar function; if it is supplied, the model parameters and initial values are extracted from it
ntransient	the burn-in size
n	sample size of the simulated series
Phi1	the coefficient vector of the lower-regime model
Phi2	the coefficient vector of the upper-regime model
thd	threshold
d	delay
p	maximum autoregressive order
sigma1	noise std. dev. in the lower regime
sigma2	noise std. dev. in the upper regime
xstart	initial values for the simulation
e	standardized noise series of size equal to length(xstart)+ntransient+n; if missing, it will be generated as some normally distributed errors

Details

The two-regime Threshold Autoregressive (TAR) model is given by the following formula:

$$Y_t = \phi_{1,0} + \phi_{1,1} Y_{t-1} + \ldots + \phi_{1,p} Y_{t-p_1} + \sigma_1 e_t, \text{ if } Y_{t-d} \le r$$

$$Y_t = \phi_{2,0} + \phi_{2,1} Y_{t-1} + \ldots + \phi_{2,p_2} Y_{t-p} + \sigma_2 e_t, \text{ if } Y_{t-d} > r.$$

where r is the threshold and d the delay.

Value

A list containing the following components:

- `y` simulated TAR series
- `e` the standardized errors

...

Author(s)

Kung-Sik Chan

References

Tong, H. (1990) "Non-linear Time Series, a Dynamical System Approach," Clarendon Press Oxford
"Time Series Analysis, with Applications in R" by J.D. Cryer and K.S. Chan

See Also

`tar`

Examples

```
set.seed(1234579)
y=tar.sim(n=100,Phi1=c(0,0.5),
Phi2=c(0,-1.8),p=1,d=1,sigma1=1,thd=-1,
sigma2=2)$y
plot(y=y,x=1:100,type='b',xlab="t",ylab=expression(Y[t]))
```

`tar.skeleton` *Find the asympotitc behavior of the skeleton of a TAR model*

Description

The skeleton of a TAR model is obtained by suppressing the noise term from the TAR model.

Usage

```
tar.skeleton(object, Phi1, Phi2, thd, d, p, ntransient = 500, n = 500,
xstart, plot = TRUE,n.skeleton = 50)
```

Arguments

`object`	a TAR model fitted by the tar function; if it is supplied, the model parameters and initial values are extracted from it
`ntransient`	the burn-in size
`n`	sample size of the skeleton trajectory
`Phi1`	the coefficient vector of the lower-regime model
`Phi2`	the coefficient vector of the upper-regime model
`thd`	threshold
`d`	delay
`p`	maximum autoregressive order
`xstart`	initial values for the iteration of the skeleton
`plot`	if True, the time series plot of the skeleton is drawn
`n.skeleton`	number of last n.skeleton points of the skeleton to be plotted

Details

The two-regime Threshold Autoregressive (TAR) model is given by the following formula:

$$Y_t = \phi_{1,0} + \phi_{1,1} Y_{t-1} + \ldots + \phi_{1,p} Y_{t-p_1} + \sigma_1 e_t, \text{ if } Y_{t-d} \le r$$

$$Y_t = \phi_{2,0} + \phi_{2,1} Y_{t-1} + \ldots + \phi_{2,p_2} Y_{t-p} + \sigma_2 e_t, \text{ if } Y_{t-d} > r.$$

where r is the threshold and d the delay.

Value

A vector that contains the trajectory of the skeleton, with the burn-in discarded.

Author(s)

Kung-Sik Chan

References

Tong, H. (1990) "Non-linear Time Series, a Dynamical System Approach," Clarendon Press Oxford. "Time Series Analysis, with Applications in R" by J.D. Cryer and K.S. Chan

See Also

`tar`

Examples

```
data(prey.eq)
prey.tar.1=tar(y=log(prey.eq),p1=4,p2=4,d=3,a=.1,b=.9,print=TRUE)
tar.skeleton(prey.tar.1)
```

`tbone` *A digitized sound file of a B flat played on a tenor trombone*

Description

A digitized sound file of about 0.4 seconds of a B flat just below middle C played on a tenor trombone by Chuck Kreeb, a member of Tempered Brass and a friend of one of the authors.

Usage

```
data(tbone)
```

Format

The format is: Time-Series [1:17689] from 1 to 17689: 0.0769 0.0862 0.0961 0.1050 0.1129 ...

Examples

```
data(tbone)
## maybe str(tbone) ; plot(tbone) ...
```

`tempdub` *Monthly average temperature in Dubuque/time series*

Description

Monthly average temperature (in degrees Fahrenheit) recorded in Dubuque 1/1964 - 12/1975.

Usage

```
data(tempdub)
```

Format

The format is: Time-Series [1:144] from 1964 to 1976: 24.7 25.7 30.6 47.5 62.9 68.5 73.7 67.9 61.1 48.5 ...

Source

http://mesonet.agron.iastate.edu/climodat/index.phtml?station=ia2364&report=16

Examples

```
data(tempdub)
## maybe str(tempdub) ; plot(tempdub) ...
```

`tlrt` *Likelihood ratio test for threshold nonlinearity*

Description

Carry out the likelihood ratio test for threshold nonlinearity, with the null hypothesis being a normal AR process and the alternative hypothesis being a TAR model with homogeneous, normally distributed errors.

Usage

```
tlrt(y, p, d = 1, transform = "no", a = 0.25, b = 0.75,...)
```

Arguments

`y`	time series
`p`	working AR order
`d`	delay
`transform`	available transformations: "no" (i.e. use raw data), "log", "log10" and "sqrt"
`a`	lower percent; the threshold is searched over the interval defined by the a*100 percentile to the b*100 percentile of the time-series variable
`b`	upper percent
`...`	other arguments to be passed to the ar function which determines the Ar order, if p is missing

Details

The search for the threshold parameter may be narrower than that defined by the user as the function attempts to ensure adequate sample size in each regime of the TAR model. The p-value of the test is based on large-sample approximation and also is more reliable for small p-values.

Value

`p.value`	p-value of the test
`test.statistic`	likelihood ratio test statistic
`a`	the actual lower fraction that defines the interval of search for the threshold; it may differ from the a specified by the user
`b`	the actual upper fraction that defines the interval of search for the threshold

Author(s)

Kung-Sik Chan

References

Chan, K.S. (1990). Percentage points of likelihood ratio tests for threshold autoregression. Journal of Royal Statistical Society, B 53, 3, 691-696.

See Also

`Keenan.test`, `Tsay.test`

Examples

```
data(spots)
pvaluem=NULL
for (d in 1:5){
res=tlrt(sqrt(spots),p=5,d=d,a=0.25,b=0.75)
pvaluem= cbind( pvaluem, round(c(d,signif(c(res$test.statistic,
res$p.value))),3))
}
rownames(pvaluem)=c('d','test statistic','p-value')
pvaluem
```

`Tsay.test` *Tsay's Test for nonlinearity*

Description

Carry out Tsay's test for quadratic nonlinearity in a time series.

Usage

```
Tsay.test(x, order, ...)
```

Arguments

`x`	time series
`order`	working linear AR order; if missing, it will be estimated via the ar function by minimizing AIC
`...`	options to be passed to the ar function

Details

The null hypothesis is that the true model is an AR process. The AR order, if missing, is estimated by minimizing AIC via the ar function, i.e. fitting autoregressive model to the data. The default fitting method of the ar function is "yule-walker."

Value

A list containing the following components

`test.stat`	The observed test statistic
`p.value`	p-value of the test
`order`	working AR order

Author(s)

Kung-Sik Chan

References

Tsay, R. S. (1986), Nonlinearity test for time series, Biometrika, 73, 461-466.

See Also

`Tsay.test`,`tlrt`

Examples

```
data(spots)
Tsay.test(sqrt(spots))
```

`tsdiag.Arimax` *Model Diagnostics for a Fitted ARIMAX Model*

Description

This function is modified from the tsdiag function of the stats package.

Usage

```
## S3 method for class 'Arimax'
tsdiag(object, gof.lag, tol = 0.1, col = "red", omit.initial = TRUE,...)
```

Arguments

`object`	a fitted ARIMAX model
`gof.lag`	maximum lag used in ACF and Ljung-Box tests for the residuals
`tol`	tolerance (default=0.1); see below
`col`	color of some warning lines in the figures (default=red)
`omit.initial`	suppress the initial (d+Ds) residuals if true
`...`	other arguments to be passed to the acf function

Value

Side effects: Plot the time plot of the standardized residuals. Red dashed line at +/-qnorm(0.025/no of data) are added to the plot. Data beyond these lines are deemed outliers, based on the Bonferronni rule. The ACF of the standardized residuals is plotted. The p-values of the Ljung-Box test are plotted using a variety of the first K residuals. K starts at the lag on and beyond which the moving-average weights (in the MA(infinity) representation) are less than tol.

Author(s)

Kung-Sik Chan, based on the tsdiag function of the stats pacakage

Examples

```
data(color)
m1.color=arima(color,order=c(1,0,0))
tsdiag(m1.color,gof=15,omit.initial=FALSE)
```

`tsdiag.TAR` *Model diagnostics for a fitted TAR model*

Description

The time series plot and the sample ACF of the standardized residuals are plotted. Also, a portmanteau test for detecting residual correlations in the standardized residuals are carried out.

Usage

```
## S3 method for class 'TAR'
tsdiag(object, gof.lag, col = "red",xlab = "t", ...)
```

Arguments

`object`	a fitted TAR model output from the tar function
`gof.lag`	number of lags of ACF to be examined
`col`	color of the lines flagging outliers, etc.
`xlab`	x labels for the plots
`...`	any additional user-supplied options to be passed to the acf function

Value

Side effects: plot the time-series plot of the standardized residuals, their sample ACF and portmanteau test for residual autocorrelations in the standardized errors.

Author(s)

Kung-Sik Chan

References

"Time Series Analysis, with Applications in R" by J.D. Cryer and K.S. Chan

See Also

tar

Examples

```
data(prey.eq)
prey.tar.1=tar(y=log(prey.eq),p1=4,p2=4,d=3,a=.1,b=.9,print=TRUE)
tsdiag(prey.tar.1)
```

---

tuba *A digitized sound file of a B flat played on a BB flat tuba*

Description

A digitized sound file of about 0.4 seconds of a B flat an octave and one whole step below middle C played on a BB flat tuba by Linda Fisher, a member of Tempered Brass and a friend one of the authors.

Usage

```
data(tuba)
```

Format

The format is: Time-Series [1:4402] from 1 to 4402: 0.217 0.209 0.200 0.195 0.196 ...

Examples

```
data(tuba)
## maybe str(tuba) ; plot(tuba) ...
```

`units` *Annual sales of certain large equipment*

Description

Annual sales of certain large equipment, 1983 - 2005.

Usage

```
data(units)
```

Format

The format is: ts [1:24, 1] 71.7 78.6 111.1 125.6 133.0 ... - attr(*, "tsp")= num [1:3] 1982 2005 1 - attr(*, "dimnames")=List of 2 ..$: NULL ..$: chr "Units"

Source

Proprietary sales data from a large international company

Examples

```
data(units)
## maybe str(units) ; plot(units) ...
```

`usd.hkd` *Daily US Dollar to Hong Kong Dollar Exchange Rates*

Description

Daily USD/HKD (US dollar to Hong Kong dollar) exchange rate from January 1, 2005 to March 7, 2006

Usage

```
data(usd.hkd)
```

Format

A data frame with 431 observations on the following 6 variables.

`r` daily returns of USD/HKD exchange rates

`v` estimated conditional variances based on an AR(1)+GARCH(3,1) model

`hkrate` daily USD/HKD exchange rates

`outlier1` dummy variable of day 203, corresponding to July 22, 2005

`outlier2` dummy variable of day 290, another possible outlier

`day` calendar day

Source

http://www.oanda.com/convert/fxhistory

References

"Time Series Analysis, with Applications in R" by J.D. Cryer and K.S. Chan

Examples

```
data(usd.hkd)
## maybe str(usd.hkd) ; plot(usd.hkd) ...
```

`veilleux` *An experimental prey-predator time series*

Description

A data frame consisting of bivariate time series from an experiment for studying prey-predator dynamics. The first time series consists of the numbers of prey individuals (Didinium natsutum) per ml measured every twelve hours over a period of 35 days; the second time series consists of the corresponding number of predators (Paramecium aurelia) per ml.

Usage

```
data(veilleux)
```

Format

The format is: mts [1:71, 1:2] 15.7 53.6 73.3 93.9 115.4 ... - attr(*, "dimnames")=List of 2 ..$: NULL ..$: chr [1:2] "Didinium" "Paramecium" - attr(*, "tsp")= num [1:3] 0 35 2 - attr(*, "class")= chr [1:2] "mts" "ts"

Source

http://www-personal.buseco.monash.edu.au/~hyndman/TSDL/

References

Veilleux (1976) "The analysis of a predatory interaction between Didinium and Paramecium", Masters thesis, University of Alberta.

Jost & Ellner (2000) "Testing for predator dependence in predator-prey dynamics: a non-parametric approach", Proceedings of the Royal Society of London B, 267, 1611-1620.

Examples

```
data(veilleux)
## maybe str(veilleux) ; plot(veilleux) ...
```

`wages` *Average hourly wages in the apparel industry / time series*

Description

Average hourly wages in the apparel industry, from 07/1981 - 06/1987.

Usage

```
data(wages)
```

Format

The format is: Time-Series [1:72] from 1982 to 1987: 4.92 4.96 5.04 5.05 5.04 5.04 5.18 5.13 5.15 5.18 ...

Source

Cryer, J. D. Time Series Analysis, Duxbury Press, 1986.

Examples

```
data(wages)
## maybe str(wages) ; plot(wages) ...
```

`winnebago` *Monthly unit sales of recreational vehicles / time series*

Description

Monthly unit sales of recreational vehicles from Winnebago, Inc., Forrest City, Iowa, from 11/1966 - 02/1972.

Usage

```
data(winnebago)
```

Format

The format is: Time-Series [1:64] from 1967 to 1972: 61 48 53 78 75 58 146 193 124 120 ...

Source

Roberts, H. V., Data Analysis for Managers with Minitab, second edition, The Scientific Press, 1991.

Examples

```
data(winnebago)
## maybe str(winnebago) ; plot(winnebago) ...
```

---

zlag *Compute the lag of a vector.*

Description

Computes the lag of a vector, with missing elements replaced by NA

Usage

```
zlag(x, d= 1)
```

Arguments

x	vector
d	compute the lag d of x

Value

A vector whose k-th element equals x[k-d] with x[t]=NA for t<=0

Author(s)

Kung-Sik Chan

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.
x=1:5
zlag(x,2)
```

Index

∗ **datasets**
airmiles, 5
airpass, 6
ar1.2.s, 6
ar1.s, 7
ar2.s, 8
arma11.s, 13
beersales, 15
bluebird, 15
bluebirdlite, 16
boardings, 17
co2, 18
color, 19
CREF, 19
cref.bond, 20
days, 20
deere1, 21
deere2, 21
deere3, 22
eeg, 26
electricity, 27
euph, 27
explode.s, 28
flow, 29
gold, 31
google, 32
hare, 33
hours, 34
ima22.s, 35
JJ, 35
larain, 39
ma1.1.s, 40
ma1.2.s, 41
ma2.s, 41
milk, 43
oil.price, 43
oilfilters, 44
prescrip, 49
prey.eq, 51
retail, 52
robot, 53
rwalk, 55
SP, 58
spots, 60
spots1, 61
star, 61
tbone, 67
tempdub, 67
tuba, 72
units, 73
usd.hkd, 73
veilleux, 74
wages, 75
winnebago, 75

∗ **methods**
acf, 4
arima, 8
arima.boot, 10
arimax, 11
ARMAspec, 13
armasubsets, 14
BoxCox.ar, 17
detectAO, 22
detectIO, 24
eacf, 25
fitted.Arima, 28
garch.sim, 29
gBox, 30
harmonic, 33
Keenan.test, 36
kurtosis, 37
lagplot, 38
LB.test, 39
McLeod.Li.test, 42
periodogram, 44
plot.Arima, 45
plot.armasubsets, 47
predict.TAR, 48

prewhiten, 50
qar.sim, 51
rstandard.Arima, 54
runs, 54
season, 56
skewness, 57
spec, 58
summary.armasubsets, 62
tar, 62
tar.sim, 64
tar.skeleton, 65
tlrt, 68
Tsay.test, 69
tsdiag.Arimax, 70
tsdiag.TAR, 71
zlag, 76
∗ **misc**
plot1.acf, 48
∗ **package**
TSA-package, 3

acf, 4
airmiles, 5
airpass, 6
ar1.2.s, 6
ar1.s, 7
ar2.s, 8
arima, 8, *9*, *12*, *29*
arima.boot, 10
arimax, 11
arma11.s, 13
ARMAacf, *5*
ARMAspec, 13
armasubsets, 14

beersales, 15
bluebird, 15
bluebirdlite, 16
boardings, 17
BoxCox.ar, 17

co2, 18
color, 19
CREF, 19
cref.bond, 20

days, 20
deere1, 21
deere2, 21
deere3, 22
detectAO, 22
detectIO, *23, 24*, 24

eacf, 25
eeg, 26
electricity, 27
euph, 27
explode.s, 28

fitted.Arima, 28
flow, 29

garch.sim, 29
gBox, 30
gold, 31
google, 32

hare, 33
harmonic, 33, *56*
hours, 34

ima22.s, 35

JJ, 35

Keenan.test, 36, *69*
kurtosis, 37

lagplot, 38
larain, 39
LB.test, 39

ma1.1.s, 40
ma1.2.s, 41
ma2.s, 41
McLeod.Li.test, 42
milk, 43

oil.price, 43
oilfilters, 44

periodogram, 44
plot.acf, *5*
plot.Arima, 45
plot.armasubsets, 47
plot1.acf, 48
predict.TAR, 48, *64*
prescrip, 49
prewhiten, 50
prey.eq, 51

qar.sim, 51

retail, 52
robot, 53
rstandard.Arima, 54
runs, 54
rwalk, 55

season, *34*, 56
skewness, 57
SP, 58
spec, *14*, 58
spots, 60
spots1, 61
star, 61
summary.armasubsets, 62

tar, *49*, 62, *65*, *66*, *72*
tar.sim, *52*, *64*, 64
tar.skeleton, *64*, 65
tbone, 67
tempdub, 67
tlrt, *36*, 68, *70*
TSA *(*TSA-package*)*, 3
TSA-package, 3
Tsay.test, *36*, *69*, 69, *70*
tsdiag.Arimax, 70
tsdiag.TAR, *64*, 71
tuba, 72

units, 73
usd.hkd, 73

veilleux, *51*, 74

wages, 75
winnebago, 75

zlag, 76