Package 'tsapp'

October 30, 2021

Type Package

Title Time Series, Analysis and Application

Version 1.0.4

Author Rainer Schlittgen

Maintainer Rainer Schlittgen <R.Schlittgen@t-online.de>

Description Accompanies the book Rainer Schlittgen and Cristina Sattarhoff (2020) <https://www.degruyter.com/view/title/575978> ``Angewandte Zeitreihenanalyse mit R, 4. Auflage'' . The package contains the time series and functions used therein. It was developed over many years teaching courses about time series analysis.

License GPL

Encoding UTF-8

LazyData true

Depends R (>= 3.6.0), Matrix , vars, fftwtools, hdm

SystemRequirements under Linux, fftwtools needs libfftw3-dev

RoxygenNote 7.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2021-10-30 11:30:02 UTC

R topics documented:

ACCIDENT — *Monthly numbers of road traffic accidents with personal injury in BRD*

Description

Monthly numbers of road traffic accidents with personal injury in BRD

Usage

```
ACCIDENT
```

Format

ACCIDENT is a univariate time series of length 528, start January 1974, frequency = 12

ACCIDENT Monthly numbers of road traffic accidents with personal injury

Source

< https://www-genesis.destatis.de/genesis//online?operation=table&code=46241-0002&levelindex=0&levelid=1583749114977>

Examples

```
data(ACCIDENT)
## maybe  tsp(ACCIDENT) ; plot(ACCIDENT)
```

acfmat — `acfmat` *computes a sequence of autocorrelation matrices for a multivariate time series*

Description

`acfmat` computes a sequence of autocorrelation matrices for a multivariate time series

Usage

```
acfmat(y, lag.max)
```

Arguments

`y`	multivariate time series
`lag.max`	maximum number of lag

Value

out list with components:

- `M` — array with autocovariance matrices
- `M1` — array with indicators if autocovariances are significantly greater (+), lower (-) than the critical value or insignificant (.) at 95 percent level

Examples

```
data(ICECREAM)
out <- acfmat(ICECREAM,7)
```

---

acfpacf — `acfpacf` *produces a plot of the acf and the pacf of a time series*

Description

`acfpacf` produces a plot of the acf and the pacf of a time series

Usage

```
acfpacf(x, lag, HV = "H")
```

Arguments

- `x` — the series, a vector or a time series
- `lag` — scalar, maximal lag to be plotted
- `HV` — character, controls division of graphic window: "H" horizontal, "V" vertical, default is "H"

Examples

```
data(LYNX)
acfpacf(log(LYNX),15,HV="H")
```

ALCINCOME — *Alcohol Demand, UK, 1870-1938.*

Description

Alcohol Demand, UK, 1870-1938.

Usage

```
ALCINCOME
```

Format

ALCINCOME is a threevariate time series of length 69 and 3 variables; start 1870, frequency = 1

Y log consumption per head

Z log real income per head

X log real price

Source

Durbin & Watson (1951) <https://doi.org/10.1093/biomet/38.1-2.159>

Examples

```
data(ALCINCOME)
## maybe  tsp(ALCINCOME) ; plot(ALCINCOME)
```

armathspec — `armathspec` *determines the theoretical spectrum of an arma process*

Description

`armathspec` determines the theoretical spectrum of an arma process

Usage

```
armathspec(a, b, nf, s = 1, pl = FALSE)
```

Arguments

`a`	ar-coefficients
`b`	ma-coefficients
`nf`	scalar, the number of equally spaced frequencies
`s`	variance of error process
`pl`	logical, if TRUE, the spectrum is plotted, FALSE for no plot

Value

out (nf+1,2) matrix, the frequencies and the spectrum

Examples

```
out <-armathspec(c(0.3,-0.5),c(-0.8,0.7),50,s=1,pl=FALSE)
```

---

aspectratio — `aspectratio` *determines the aspect ratio to plot a time series*

Description

`aspectratio` determines the aspect ratio to plot a time series

Usage

```
aspectratio(y)
```

Arguments

y	time series

Value

a scalar, the aspect ratio

Examples

```
data(GDP)
a <- aspectratio(GDP)
```

---

bandfilt — `bandfilt` *does a bandpass filtering of a time series*

Description

`bandfilt` does a bandpass filtering of a time series

Usage

```
bandfilt(y, q, pl, pu)
```

Arguments

y	the series, a vector or a time series
q	scalar, half of length of symmetric weights
pl	scalar, lower periodicity (>= 2)
pu	scalar, upper periodicity (> pl)

Value

yf (n,1) vector, the centered filtered time series with NA's at beginning and ending

Examples

```
data(GDP)
yf <- bandfilt(GDP,5,2,6)
plot(GDP); lines(yf+mean(GDP),col="red")
```

BEER — *Monthly beer production in Australia: megalitres. Includes ale and stout. Does not include beverages with alcohol percentage less than 1.15.*

Description

Monthly beer production in Australia: megalitres. Includes ale and stout. Does not include beverages with alcohol percentage less than 1.15.

Usage

```
BEER
```

Format

BEER is a univariate time series of length 476, start January 1956, end Aug 1995, frequency = 12

BEER Monthly production of beer in Australia

Source

R package tsdl <https://github.com/FinYang/tsdl>

Examples

```
data(BEER)
## maybe  tsp(BEER) ; plot(BEER)
```

`bispeces` *`bispeces` performs indirect bivariate spectral estimation of two series y1, y2 using lagwindows*

Description

`bispeces` performs indirect bivariate spectral estimation of two series y1, y2 using lagwindows

Usage

```
bispeces(y1, y2, q, win = "bartlett")
```

Arguments

`y1`	vector, the first time series
`y2`	vector, the second time series
`q`	number of covariances used for indirect spectral estimation
`win`	lagwindow (possible: "bartlett", "parzen", "tukey")

Value

out data frame with columns:

`f`	frequencies 0, 1/n, 2/n, ... (<= 1/2)
`coh`	estimated coherency at Fourier frequencies 0,1/n, ...
`ph`	estimated phase at Fourier frequencies 0,1/n, ...

Examples

```
data(ICECREAM)
y <- ICECREAM
out <- bispeces(y[,1],y[,2],8,win="bartlett")
```

`BLACKOUT` *Weekly number of births in New York*

Description

Weekly number of births in New York

Usage

```
BLACKOUT
```

Format

BLACKOUT is a univariate time series of length 313, 1961 – 1966

BLACKOUT Weekly numbers of births in New York

Source

Izenman, A. J., and Zabell, S. L. (1981) <https://www.sciencedirect.com/science/article/abs/pii/0049089X81900181>

Examples

```
data(BLACKOUT)
## maybe  tsp(BLACKOUT) ; plot(BLACKOUT)
```

BoxCox — `BoxCox` *determines the power of a Box-Cox transformation to stabilize the variance of a time series*

Description

`BoxCox` determines the power of a Box-Cox transformation to stabilize the variance of a time series

Usage

```
BoxCox(y, seg, Plot = FALSE)
```

Arguments

`y`	the series, a vector or a time series
`seg`	scalar, number of segments
`Plot`	logical, should a plot be produced?

Value

l scalar, the power of the Box-Cox transformation

Examples

```
data(INORDER)
lambda <-BoxCox(INORDER,6,Plot=FALSE)
```

COFFEE — *U.S. annual coffee consumption*

Description

U.S. annual coffee consumption

Usage

```
COFFEE
```

Format

COFFEE is a univariate time series of length 61; start 1910, frequency = 1

COFFEE annual coffee-consumption USA, logarithmic transformed

Source

R package tsdl <https://github.com/FinYang/tsdl>

Examples

```
data(COFFEE)
## maybe  tsp(COFFEE) ; plot(COFFEE)
```

DAX — *Market value of DAX*

Description

Market value of DAX

Usage

```
DAX
```

Format

DAX is a multivariate time series of length 12180 and 4 variables

DAY Day of the week

MONTH Month

Year Year

DAX30 Market value

Examples

```
data(DAX)
## maybe  tsp(DAX) ; plot(DAX)
```

DIABETES — *Incidences of insulin-dependent diabetes mellitus*

Description

Incidences of insulin-dependent diabetes mellitus

Usage

```
DIABETES
```

Format

DIABETES is a univariate time series of length 72, start January 1979, frequency = 12

DIABETES Incidences of insulin-dependent diabetes mellitus

Source

Waldhoer, T., Schober, E. and Tuomilehto, J. (1997) <https://www.sciencedirect.com/science/article/abs/pii/S0895435696003344>

Examples

```
data(DIABETES)
## maybe  tsp(DIABETES) ; plot(DIABETES)
```

DOMINANCE — *Running yield of public bonds in Austria and Germany*

Description

Running yield of public bonds in Austria and Germany

Usage

```
DOMINANCE
```

Format

DOMINANCE is a bivariate time series of length 167:

X Interest rate Germany

Y Interest rate Austria

Source

Jaenicke, J. and Neck, R. (1996) <https://doi.org/10.17713/ajs.v25i2.555>

Examples

```
data(DOMINANCE)
## maybe  tsp(DOMINANCE) ; plot(DOMINANCE)
```

`dynspecest` *`dynspecest` performs a dynamic spectrum estimation*

Description

`dynspecest` performs a dynamic spectrum estimation

Usage

```
dynspecest(y, nseg, nf, e, theta = 0, phi = 15, d, Plot = FALSE)
```

Arguments

`y`	time series or vector
`nseg`	number of segments for which the spectrum is estimated
`nf`	number of equally spaced frequencies
`e`	equal bandwidth
`theta`	azimuthal viewing direction, see R function persp
`phi`	colatitude viewing direction, see R function persp
`d`	a value to vary the strength of the perspective transformation, see R function persp
`Plot`	logical, schould a plot be generated?

Value

out list with components

`f`	frequencies, vector of length nf
`t`	time, vector of length nseg
`spec`	the spectral estimates, (nf,nt)-matrix

Examples

```
data(IBM)
y <- diff(log(IBM))
out <- dynspecest(y,60,50,0.2,theta=0,phi=15,d=1,Plot=FALSE)
```

ENGINES *ENGINES is an alias for MACHINES*

Description

ENGINES is an alias for MACHINES

Usage

```
ENGINES
```

Format

ENGINES is a univariate time series of length 188, start January 1972 frequency = 12

ENGINES Incoming orders for engines

Examples

```
data(ENGINES)
## maybe  tsp(ENGINES) ; plot(ENGINES)
```

FINANCE *Portfolio-Insurance-Strategies*

Description

Portfolio-Insurance-Strategies

Usage

```
FINANCE
```

Format

FINANCE is a multivariate time series of length 7529:

CPPI first Portfolio-Insurance-Strategy

TIPP second Portfolio-Insurance-Strategy

StopLoss third Portfolio-Insurance-Strategy

SyntheticPut fourth Portfolio-Insurance-Strategy

CASH money market investment

Source

Dichtl, H. and Drobetz, W. (2011) <doi:10.1016/j.jbankfin.2010.11.012>

Examples

```
data(FINANCE)
## maybe  tsp(FINANCE) ; plot(FINANCE)
```

GDP *Germany's gross domestic product adjusted for price changes*

Description

Germany's gross domestic product adjusted for price changes

Usage

```
GDP
```

Format

GDP is a univariate time series of length 159, start January 1970, frequency = 4

GDP Gross domestic product adjusted for price changes

Source

<https://www-genesis.destatis.de/genesis//online?operation=table&code=81000-0002&levelindex =0&levelid=1583750132341>

Examples

```
data(GDP)
## maybe  tsp(GDP) ; plot(GDP)
```

GDPORIG *Germany's gross domestic product, values of Laspeyres index to base 2000*

Description

Germany's gross domestic product, values of Laspeyres index to base 2000

Usage

```
GDPORIG
```

Format

GDPORIG is a univariate time series of length 159, start January 1970, frequency = 4

GDPORIG gross domestic product, values of Laspeyres index to the base 2000

Source

<https://www-genesis.destatis.de/genesis//online?operation=table&code=81000-0002&levelindex=0&levelid=1583750132341>

Examples

```
data(GDPORIG)
## maybe  tsp(GDPORIG) ; plot(GDPORIG)
```

`Grangercaus` — `Grangercaus` *determines three values of BIC from a twodimensional VAR process*

Description

`Grangercaus` determines three values of BIC from a twodimensional VAR process

Usage

```
Grangercaus(x, y, p)
```

Arguments

`x`	first time series
`y`	second time series
`p`	maximal order of VAR process

Value

out list with components

`BIC` vector of length 3:

BIC1	minimum aic value for all possible lag structures
BIC2	minimum aic value when Y is not included as regressor in the equation for X
BIC3	minimum aic value when X is not included as regressor in the equation for Y

`out1`	output of function lm for regression equation for x-series
`out2`	output of function lm for regression equation for y-series

Examples

```
data(ICECREAM)
out <- Grangercaus(ICECREAM[,1],ICECREAM[,2],3)
```

`HAC` *HAC Covariance Matrix Estimation* `HAC` *computes the central quantity (the meat) in the HAC covariance matrix estimator, also called sandwich estimator. HAC is the abbreviation for "heteroskedasticity and autocorrelation consistent".*

Description

HAC Covariance Matrix Estimation `HAC` computes the central quantity (the meat) in the HAC covariance matrix estimator, also called sandwich estimator. HAC is the abbreviation for "heteroskedasticity and autocorrelation consistent".

Usage

```
HAC(mcond, method = "Bartlett", bw)
```

Arguments

`mcond`	a q-dimensional multivariate time series. In the case of OLS regression with q regressors mcond contains the series of the form regressor*residual (see example below).
`method`	kernel function, choose between "Truncated", "Bartlett", "Parzen", "Tukey-Hanning", "Quadratic Spectral".
`bw`	bandwidth parameter, controls the number of lags considered in the estimation.

Value

mat a (q,q)-matrix

Source

Heberle, J. and Sattarhoff, C. (2017) <doi:10.3390/econometrics5010009> "A Fast Algorithm for the Computation of HAC Covariance Matrix Estimators"

Examples

```
data(MUSKRAT)
y <- ts(log10(MUSKRAT))
n <- length(y)
t <- c(1:n)
t2 <- t^2
out2 <- lm(y ~ t +t2)
mat_xu <- matrix(c(out2$residuals,t*out2$residuals, t2*out2$residuals),nrow=62,ncol=3)
hac <- HAC(mat_xu, method="Bartlett", 4)

mat_regr<- matrix(c(rep(1,62),t,t2),nrow=62,ncol=3)
mat_q <- t(mat_regr)%*%mat_regr/62
```

```
vcov_HAC <- solve(mat_q)%*%hac%*%solve(mat_q)/62
# vcov_HAC is the HAC covariance matrix estimation for the OLS coefficients.
```

HEARTBEAT *Cardiac frequency of a patient*

Description

Cardiac frequency of a patient

Usage

```
HEARTBEAT
```

Format

HEARTBEAT is a univariate time series of length 30:

HEARTBEAT cardiac frequency of a patient

Examples

```
data(HEARTBEAT)
## maybe  tsp(HEARTBEAT) ; plot(HEARTBEAT)
```

HSV *HSV's position in the first German soccer league*

Description

HSV's position in the first German soccer league

Usage

```
HSV
```

Format

HSV is a univariate time series of length 47:

HSV HSV's position in the first German soccer league

Source

<https://www.transfermarkt.de/hamburger-sv/platzierungen/verein/41>

Examples

```
data(HSV)
## maybe  tsp(HSV) ; plot(HSV)
```

IBM — *IBM's stock price*

Description

IBM's stock price

Usage

```
IBM
```

Format

IBM is a univariate time series of length 369, start 17 May 1961

IBM IBM's daily stock price

Source

Box, G. E. P. and Jenkins, G. M. (1970, ISBN: 978-0816210947) "Time series analysis: forecasting and control"

Examples

```
data(IBM)
## maybe  tsp(IBM) ; plot(IBM)
```

ICECREAM — *Temperature and consumption of ice cream*

Description

Temperature and consumption of ice cream

Usage

```
ICECREAM
```

Format

ICECREAM is a bivariate time series of length 160:

ICE consumption of ice cream

TEMP Temperature in Fahrenheit degrees

Source

Hand, D. J., et al. (1994, ISBN: 9780412399206) "A Handbook of Small Data Sets"

Examples

```
data(ICECREAM)
## maybe  tsp(ICECREAM) ; plot(ICECREAM)
```

`init_values` *`init_values` is an auxiliary function for rlassoHAC, for fitting linear models with the method of least squares where only the variables in X with highest correlations are considered; taken from package hdm.*

Description

`init_values` is an auxiliary function for rlassoHAC, for fitting linear models with the method of least squares where only the variables in X with highest correlations are considered; taken from package hdm.

Usage

```
init_values(X, y, number = 5, intercept = TRUE)
```

Arguments

`X`	Regressors (matrix or object can be coerced to matrix).
`y`	Dependent variable(s).
`number`	How many regressors in X should be considered.
`intercept`	Logical. If TRUE, intercept is included which is not penalized.

Value

init_values returns a list containing the following components:

`residuals`	Residuals.
`coefficients`	Estimated coefficients.

Source

Victor Chernozhukov, Chris Hansen, Martin Spindler (2016). hdm: High-Dimensional Metrics, R Journal, 8(2), 185-199. URL https://journal.r-project.org/archive/2016/RJ-2016-040/index.html.

INORDER — *Income orders of a company*

Description

Income orders of a company

Usage

```
INORDER
```

Format

INORDER is a univariate time series of length 237, start January 1968, frequency =12

INORDER Income orders of a company

Examples

```
data(INORDER)
## maybe  tsp(INORDER) ; plot(INORDER)
```

interpol — `interpol` *help function for missls*

Description

`interpol` help function for missls

Usage

```
interpol(rho, xcent)
```

Arguments

`rho`	autocorrelation function
`xcent`	centered time series

Value

z new version of xcent

kweightsHAC — kweightsHAC *help function for HAC*

Description

kweightsHAC help function for HAC

Usage

```
kweightsHAC(
  kernel = c("Truncated", "Bartlett", "Parzen", "Tukey-Hanning", "Quadratic Spectral"),
   dimN,
   bw
)
```

Arguments

kernel	kernel function, choose between "Truncated", "Bartlett", "Parzen", "Tukey-Hanning", "Quadratic Spectral".
dimN	number of observations
bw	bandwidth parameter

Value

ww weights

L921 — *Subsoil water level and precipitation at pilot well L921*

Description

Subsoil water level and precipitation at pilot well L921

Usage

```
L921
```

Format

L921 is a trivariate time series of length 335:

T Day

Y Water level

Z Supplemented water level

Examples

```
data(L921)
## maybe  tsp(L921) ; plot(L921)
```

`lagwinba` — `lagwinba` *Bartlett's Lag-window for indirect spectrum estimation*

Description

`lagwinba` Bartlett's Lag-window for indirect spectrum estimation

Usage

```
lagwinba(NL)
```

Arguments

`NL`	number of lags used for estimation

Value

win vector, one-sided weights

Examples

```
win <-lagwinba(5)
```

`lagwinpa` — `lagwinpa` *Parzen's Lag-window for indirect spectrum estimation*

Description

`lagwinpa` Parzen's Lag-window for indirect spectrum estimation

Usage

```
lagwinpa(NL)
```

Arguments

`NL`	number of lags used for estimation

Value

win vector, one-sided weights

Examples

```
win <- lagwinpa(5)
```

lagwintu — `lagwintu` *Tukey's Lag-window for indirect spectrum estimation*

Description

`lagwintu` Tukey's Lag-window for indirect spectrum estimation

Usage

```
lagwintu(NL)
```

Arguments

`NL`	number of lags used for estimation

Value

win vector, one-sided weights

Examples

```
win <- lagwintu(5)
```

lambdaCalculationHAC — `lambdaCalculationHAC` *is an auxiliary function for rlassoHAC; it calculates the penalty parameters.*

Description

`lambdaCalculationHAC` is an auxiliary function for rlassoHAC; it calculates the penalty parameters.

Usage

```
lambdaCalculationHAC(
  X.dependent.lambda = FALSE,
  c = 2,
  gamma = 0.1,
  kernel,
  bands,
  bns,
  lns,
  nboot,
  y = NULL,
  x = NULL
)
```

Arguments

`X.dependent.lambda`
: Logical, TRUE, if the penalization parameter depends on the design of the matrix x. FALSE, if independent of the design matrix (default).

`c`
: Constant for the penalty with default c = 2 .

`gamma`
: Constant for the penalty with default gamma=0.1.

`kernel`
: String kernel function, choose between "Truncated", "Bartlett", "Parzen", "Tukey-Hanning", "Quadratic Spectral".

`bands`
: Constant bandwidth parameter.

`bns`
: Block length.

`lns`
: Number of blocks.

`nboot`
: Number of bootstrap iterations.

`y`
: Residual which is used for calculation of the variance or the data-dependent loadings.

`x`
: Regressors (vector, matrix or object can be coerced to matrix).

Value

`lambda0`
: Penalty term

`Ups0`
: Penalty loadings, vector of length p (no. of regressors)

`lambda`
: This is lambda0 * Ups0

`penalty`
: Summary of the used penalty function.

Source

Victor Chernozhukov, Chris Hansen, Martin Spindler (2016). hdm: High-Dimensional Metrics, R Journal, 8(2), 185-199. URL https://journal.r-project.org/archive/2016/RJ-2016-040/index.html.

`lambdaCalculationLoad` `lambdaCalculationLoad` *is an auxiliary function for rlassoLoad; it calculates the penalty parameters with predefined loadings.*

Description

`lambdaCalculationLoad` is an auxiliary function for rlassoLoad; it calculates the penalty parameters with predefined loadings.

Usage

```
lambdaCalculationLoad(
  X.dependent.lambda = FALSE,
  c = 2,
  gamma = 0.1,
  load,
  bns,
  lns,
  nboot,
  y = NULL,
  x = NULL
)
```

Arguments

`X.dependent.lambda`	Logical, TRUE, if the penalization parameter depends on the design of the matrix x. FALSE, if independent of the design matrix (default).
`c`	Constant for the penalty with default c = 2 .
`gamma`	Constant for the penalty with default gamma=0.1.
`load`	Penalty loadings, vector of length p (no. of regressors).
`bns`	Block length.
`lns`	Number of blocks.
`nboot`	Number of bootstrap iterations.
`y`	Residual which is used for calculation of the variance or the data-dependent penalty.
`x`	Regressors (vector, matrix or object can be coerced to matrix).

Value

`lambda0`	Penalty term
`Ups0`	Penalty loadings, vector of length p (no. of regressors)
`lambda`	This is lambda0 * Ups0
`penalty`	Summary of the used penalty function

Source

Victor Chernozhukov, Chris Hansen, Martin Spindler (2016). hdm: High-Dimensional Metrics, R Journal, 8(2), 185-199. URL https://journal.r-project.org/archive/2016/RJ-2016-040/index.html.

ldrec — ldrec *does Levinson-Durbin recursion for determing all coefficients a(i,j)*

Description

`ldrec` does Levinson-Durbin recursion for determing all coefficients a(i,j)

Usage

```
ldrec(a)
```

Arguments

a	(p+1,1)-vector of acf of a time series: acov(0),...,acov(p) or 1,acor(1),..,acor(p)

Value

mat (p,p+2)-matrix, coefficients in lower triangular, pacf in colum p+2 and Q(p) in colum p+1

Examples

```
data(HEARTBEAT)
a <- acf(HEARTBEAT,5,plot=FALSE)
mat <- ldrec(a$acf)
```

LITH — *Daily subsoil water level and precipitation at pilot well Lith*

Description

Daily subsoil water level and precipitation at pilot well Lith

Usage

```
LITH
```

Format

LITH is a bivariate time series of length 1347:

N precipitation amount

G water level

Examples

```
data(LITH)
## maybe  tsp(LITH) ; plot(LITH)
```

LjungBoxPierceTest — *`LjungBoxPierceTest` determines the test statistic and p values for several lags for a residual series*

Description

`LjungBoxPierceTest` determines the test statistic and p values for several lags for a residual series

Usage

```
LjungBoxPierceTest(y, n.par = 0, maxlag = 48)
```

Arguments

`y`	the series of residuals, a vector or a time series
`n.par`	number of parameters which had been estimated
`maxlag`	maximal lag up to which the test statistic is computed, default is maxlag = 48

Value

BT matrix with columns: lags, degrees of freedom, test statistic, p-value

Examples

```
data(COFFEE)
out <- arima(COFFEE,order=c(1,0,0))
BT <- LjungBoxPierceTest(out$residuals,1,20)
```

LUHORMONE — *Level of Luteinzing hormone of a cow*

Description

Level of Luteinzing hormone of a cow

Usage

```
LUHORMONE
```

Format

LUHORMONE is a bivariate time series of length 29:

T Time in minutes

X Level of the Luteinzing-hormone

`LYNX` *Annual lynx trappings in a region of North-West Canada. Taken from Andrews and Herzberg (1985).*

Description

Annual lynx trappings in a region of North-West Canada. Taken from Andrews and Herzberg (1985).

Usage

```
LYNX
```

Format

LYNX is a univariate time series of length 114; start 1821 frequency = 1

LYNX annual lynx trappings in a region of North-west Canada

Source

Andrews, D. F. and Herzberg, A. M. (1985) "Data" <https://www.springer.com/gp/book/9781461295631>

Examples

```
data(LYNX)
## maybe  tsp(LYNX) ; plot(LYNX)
```

`LYNXHARE` *Size of populations of lynxes and snow hares*

Description

Size of populations of lynxes and snow hares

Usage

```
LYNXHARE
```

Format

LYNXHARE is a simulated bivariate time series from a VAR[1]-model of length 100:

X Number of lynxes

Y Number of snow hares

Examples

```
data(LYNXHARE)
```

MACHINES *Number of incoming orders for machines*

Description

Number of incoming orders for machines

Usage

```
MACHINES
```

Format

MACHINES is a univariate time series of length 188, start January 1972 frequency = 12

MACHINES Incoming orders for machines

Examples

```
data(MACHINES)
## maybe  tsp(MACHINES) ; plot(MACHINES)
```

MAUNALOA *Atmospheric CO2 concentrations (ppmv) derived from in situ air samples collected at Mauna Loa Observatory, Hawaii*

Description

Atmospheric CO2 concentrations (ppmv) derived from in situ air samples collected at Mauna Loa Observatory, Hawaii

Usage

```
MAUNALOA
```

Format

MAUNALOA is a univariate time series of length 735; start March 1958, frequency = 12

MAUNALOA CO2-concentration at Mauna Loa

Source

Keeling, C. D. , Piper, S. C., Bacastow, R. B., Wahlen, M. , Whorf, T. P., Heimann, M., and Meijer, H. A. (2001) <https://library.ucsd.edu/dc/object/bb3859642r>

Examples

```
data(MAUNALOA)
## maybe  tsp(MAUNALOA) ; plot(MAUNALOA)
```

MDAX *Stock market price of MDAX*

Description

Stock market price of MDAX

Usage

```
MDAX
```

Format

MDAX is a multivariate time series of length 6181 and 4 variables

DAY Day of the week

MONTH Month

YEAR Year

MDAX Opening stock market price

Source

<https://www.onvista.de/index/MDAX-Index-323547>

Examples

```
data(MDAX)
## maybe  tsp(MDAX) ; plot(MDAX[,3])
```

MELANOM *Melanoma incidence in Connecticut*

Description

Melanoma incidence in Connecticut

Usage

```
MELANOM
```

Format

MELANOM is a multivariate time series of length 45 and 3 variables

POP Population

RATE Incidence

SUN Sunspots

Source

Andrews, D. F. and Herzberg, A. M. (1985) "Data" <https://www.springer.com/gp/book/9781461295631>

Examples

```
data(MELANOM)
## maybe  tsp(MELANOM) ; plot(MELANOM[,-1])
```

`mfraccheck` *multifractal check `mfraccheck` computes the absolute empirical moments of the differenced series for various lags and moment orders. E.g. for lag = 3 and moment order = 1 the average absolute value of the differences with lag 3 will be computed. By default, the maximum lag is determined so that the differenced series contains at lest 50 observations.*

Description

multifractal check `mfraccheck` computes the absolute empirical moments of the differenced series for various lags and moment orders. E.g. for lag = 3 and moment order = 1 the average absolute value of the differences with lag 3 will be computed. By default, the maximum lag is determined so that the differenced series contains at lest 50 observations.

Usage

```
mfraccheck(p, q_max)
```

Arguments

`p`	the series
`q_max`	maximum moment order

Value

out list with components:

`moments`	matrix with lagmax raws and q_max columns containing the values of the absolute empirical moments
`lagmax`	the maximum lag for differencing

Examples

```
data(NIKKEI)
p <- NIKKEI
out <- mfraccheck(log(p),5)
mom <- ts(out$moments,start=1)
ts.plot(mom, log ="xy",xlab="lag",ylab="abs. empirical moments", lty=c(1:5))
```

---

missar — missar *Substitution of missing values in a time series by conditional exspectations of AR(p) models*

Description

`missar` Substitution of missing values in a time series by conditional exspectations of AR(p) models

Usage

```
missar(x, p, iterout = 0)
```

Arguments

`x`	vector, the time series
`p`	integer, the maximal order of ar polynom $0 < p < 18$,
`iterout`	if = 1, iteration history is printed

Value

out list with elements

`a`	(p,p)-matrix, estimated ar coefficients for ar-models
`y`	(n,1)-vector, completed time series
`iterhist`	matrix, NULL or the iteration history

Source

Miller R.B., Ferreiro O. (1984) <doi.org/10.1007/978-1-4684-9403-7_12> "A Strategy to Complete a Time Series with Missing Observations"

Examples

```
data(HEARTBEAT)
x <- HEARTBEAT
x[c(20,21)] <- NA
out <- missar(x,2)
```

missls — missls *substitutes missing values in a time series using the LS approach with ARMA models*

Description

missls substitutes missing values in a time series using the LS approach with ARMA models

Usage

```
missls(x, p = 0, tol = 0.001, theo = 0)
```

Arguments

x	vector, the time series
p	integer, the order of polynom alpha(B)/beta(B)
tol	tolerance that can be set; it enters via tol*sd(x,na.rm=TRUE)
theo	(k,1)-vector, prespecified Inverse ACF, IACF (starting at lag 1)

Value

y completed time series

Source

S. R. Brubacher and G. Tunnicliffe Wilson (1976) <https://www.jstor.org/stable/2346678> "Interpolating Time Series with Application to the Estimation of Holiday Effects on Electricity Demand Journal of the Royal Statistical Society"

Examples

```
data(HEARTBEAT)
x <- HEARTBEAT
x[c(20,21)] <- NA
out <-  missls(x,p=2,tol=0.001,theo=0)
```

moveav — moveav *smoothes a time series by moving averages*

Description

moveav smoothes a time series by moving averages

Usage

```
moveav(y, q)
```

Arguments

y	the series, a vector or a time series
q	scalar, span of moving average

Value

g vector, smooth component

Examples

```
data(GDP)
g <- moveav(GDP,12)
 plot(GDP) ; lines(g,col="red")
```

movemed — movemed *smoothes a time series by moving medians*

Description

movemed smoothes a time series by moving medians

Usage

```
movemed(y, q)
```

Arguments

y	the series, a vector or a time series
q	scalar, span of moving median

Value

g vector, smooth component

Examples

```
data(BIP)
g <- movemed(GDP,12)
 plot(GDP) ; t <- seq(from = 1970, to = 2009.5,by=0.25) ; lines(t,g,col="red")
```

MUSKRAT *Annual trade of muskrat pelts*

Description

Annual trade of muskrat pelts

Usage

```
MUSKRAT
```

Format

MUSKRAT is a univariate time series of length 62; start 1848, frequency = 1

MUSKRAT annual trade of muskrat pelts

Source

<https://archive.uea.ac.uk/~gj/book/data/mink.dat>

Examples

```
data(MUSKRAT)
## maybe  tsp(MUSKRAT) ; plot(MUSKRAT)
```

NIKKEI *Daily values of the Japanese stock market index Nikkei 225 between 02.02.2000 and 20.10.2020*

Description

Daily values of the Japanese stock market index Nikkei 225 between 02.02.2000 and 20.10.2020

Usage

```
NIKKEI
```

Format

NIKKEI is a univariate time series of length 5057

NIKKEI Daily values of Nikkei

Source

Heber, G., Lunde, A., Shephard, N. and Sheppard, K. (2009) "Oxford-Man Institute's realized library, version 0.3", Oxford-Man Institute, University of Oxford, Oxford <https://realized.oxford-man.ox.ac.uk/data>

Examples

```
data(NIKKEI)
## maybe plot(NIKKEI)
```

`outidentify` — `outidentify` *performs one iteration of Wei's iterative procedure to identify impact, locations and type of outliers in arma processes*

Description

`outidentify` performs one iteration of Wei's iterative procedure to identify impact, locations and type of outliers in arma processes

Usage

```
outidentify(x, object, alpha = 0.05, robust = FALSE)
```

Arguments

`x`	vector, the time series
`object`	output of a model fit with the function arima (from stats)
`alpha`	the level of the tests for deciding which value is to be considered an outlier
`robust`	logical, should the standard error be computed robustly?

Value

out list with elements

`outlier`	matrix with time index (ind), type of outlier (1 = AO, 2 = IO) and value of test statistic (lambda)
`arima.out`	output of final arima model where the outliers are incorporated as fixed regressors

Examples

```
data(SPRUCE)
out <- arima(SPRUCE,order=c(2,0,0))
out2 <- outidentify(SPRUCE,out,alpha=0.05, robust = FALSE)
```

OXYGEN *Amount of an Oxygen isotope*

Description

Amount of an Oxygen isotope

Usage

```
OXYGEN
```

Format

OXYGEN is a matrix with 164 rows and 2 columns

T Time

D DELTA18O

Source

Belecher, J., Hampton, J. S., and Tunnicliffe Wilson, T. (1994, ISSN: 1369-7412) "Parameterization of Continuous Time Autoregressive Models for Irregularly Sampled Time Series Data"

Examples

```
data(OXYGEN)
## maybe   plot(OXYGEN[,1],OXYGEN[,2],type="l"); rug(OXYGEN[,1])
```

pacfmat `pacfmat` *sequence of partial autocorrelation matrices and related statistics for a multivariate time series*

Description

`pacfmat` sequence of partial autocorrelation matrices and related statistics for a multivariate time series

Usage

```
pacfmat(y, lag.max)
```

Arguments

`y`	multivariate time series
`lag.max`	maximum number of lag

Value

out list with components:

`M`	array with matrices of partial autocovariances divided by their standard error
`M1`	array with indicators if partial autocovariances are significantly greater (+), lower (-) than the critical value or insignificant (.)
`R`	array with matrices of partial autocovariances
`S`	matrix of diagonals of residual covariances (row-wise)
`Test`	test statistic
`pval`	p value of test

Examples

```
data(ICECREAM)
out <- pacfmat(ICECREAM,7)
```

PAPER *Two measurements at a paper machine*

Description

Two measurements at a paper machine

Usage

```
PAPER
```

Format

PAPER is a bivariate time series of length 160

H High

W Weight

Source

Janacek, G. J. & Swift, L. (1993, ISBN: 978-0139184598) "Time Series: Forecasting, Simulation, Applications"

Examples

```
data(PAPER)
## maybe  tsp(PAPER) ; plot(PAPER)
```

periodogram — `periodogram` *determines the periodogram of a time series*

Description

`periodogram` determines the periodogram of a time series

Usage

```
periodogram(y, nf, ACF = FALSE, type = "cov")
```

Arguments

`y`	(n,1) vector, the time series or an acf at lags 0,1,...,n-1
`nf`	scalar, the number of equally spaced frequencies; not necessay an integer
`ACF`	logical, FALSE, if y is ts, TRUE, if y is acf
`type`	c("cov","cor"), area under spectrum, can be variance or normed to 1.

Value

out (floor(nf/2)+1,2) matrix, the frequencies and the periodogram

Examples

```
data(WHORMONE)
## periodogram at Fourier frequencies and frequencies 0 and 0.5
out <-periodogram(WHORMONE,length(WHORMONE)/2,ACF=FALSE,type="cov")
```

periodotest — `periodotest` *computes the p-value of the test for a hidden periodicity*

Description

`periodotest` computes the p-value of the test for a hidden periodicity

Usage

```
periodotest(y)
```

Arguments

`y`	vector, the time series

Value

pval the p-value of the test

Examples

```
data(PIGPRICE)
y <- PIGPRICE
out <- stl(y,s.window=6)
e <- out$time.series[,3]
out <- periodotest(e)
```

`perwinba` `perwinba` *Bartlett-Priestley window for direct spectral estimation*

Description

`perwinba` Bartlett-Priestley window for direct spectral estimation

Usage

```
perwinba(e, n)
```

Arguments

`e`	equal bandwidth (at most n frequencies are used for averaging)
`n`	length of time series

Value

w weights (symmetric)

Examples

```
data(WHORMONE)
w <- perwinba(0.1,length(WHORMONE))
```

`perwinda` `perwinda` *Daniell window for direct spectral estimation*

Description

`perwinda` Daniell window for direct spectral estimation

Usage

```
perwinda(e, n)
```

Arguments

e	equal bandwidth (at most n frequencies are used for averaging)
n	length of time series

Value

w weights (symmetric)

Examples

```
data(WHORMONE)
w <- perwinda(0.1,length(WHORMONE))
```

---

perwinpa   perwinpa *Parzen's window for direct spectral estimation*

Description

perwinpa Parzen's window for direct spectral estimation

Usage

```
perwinpa(e, n)
```

Arguments

e	equal bandwidth (at most n frequencies are used for averaging)
n	length of time series

Value

w weights (symmetric)

Examples

```
data(WHORMONE)
w <- perwinpa(0.1,length(WHORMONE))
```

pestep — `pestep` *help function for missar*

Description

`pestep` help function for missar

Usage

```
pestep(f, xt)
```

Arguments

`f`	IACF, inverse ACF
`xt`	segment of the time series

Value

xt new version of xt

PIGPRICE — *Monthly prices for pigs*

Description

Monthly prices for pigs

Usage

```
PIGPRICE
```

Format

PIGPRICE is a univariate time series of length 240; start January 1894, frequency =12

PIGPRICE Monthly prices for pigs

Source

Hanau, A. (1928) "Die Prognose der Schweinepreise"

Examples

```
data(PIGPRICE)
## maybe  tsp(PIGPRICE) ; plot(PIGPRICE)
```

`polymake`	`polymake` *generates the coefficients of an AR process given the zeros of the characteristic polynomial. The norm of the roots must be greater than one for stationary processes.*

Description

`polymake` generates the coefficients of an AR process given the zeros of the characteristic polynomial. The norm of the roots must be greater than one for stationary processes.

Usage

```
polymake(r)
```

Arguments

`r`	vector, the zeros of the characteristic polynomial

Value

C coefficients (a[1],a[2],...,a[p]) of the polynomial 1 - a[1]z -a[2]z^2 -...- a[p]z^p

Examples

```
C <- polymake(c(2,-1.5,3))
```

`PPDEMAND`	*Peak power demand in Berlin*

Description

Peak power demand in Berlin

Usage

```
PPDEMAND
```

Format

PPDEMAND is a univariate time series of length 37; start 1955, frequency = 1

PPDEMAND annual peak power demand in Berlin, Megawatt

Source

Fiedler, H. (1979) "Verschiedene Verfahren zur Prognose des des Stromspitzenbedarfs in Berlin (West)"

Examples

```
data(PPDEMAND)
## maybe  tsp(PPDEMAND) ; plot(PPDEMAND)
```

PRODINDEX — *Production index of manufacturing industries*

Description

Production index of manufacturing industries

Usage

```
PRODINDEX
```

Format

PRODINDEX is a univariate time series of length 119:

PRODINDEX Production index of manufacturing industries

Source

Statistisches Bundesamt (2009) <https://www-genesis.destatis.de/genesis/online>

Examples

```
data(PRODINDEX)
## maybe  tsp(PRODINDEX) ; plot(PRODINDEX)
```

---

psifair — *`psifair` is a psi-function for robust estimation*

Description

`psifair` is a psi-function for robust estimation

Usage

```
psifair(u)
```

Arguments

`u`	vector

Value

out transformed vector

Examples

```
out <- psifair(c(3.3,-0.7,2.1,1.8))
```

---

psihuber — *psihuber is a psi-function for robust estimation*

Description

`psihuber` is a psi-function for robust estimation

Usage

```
psihuber(u)
```

Arguments

`u`	vector

Value

out transformed vector

Examples

```
out <- psihuber(c(3.3,-0.7,2.1,1.8))
```

RAINFALL — *Annual amount of rainfall in Los Angeles*

Description

Annual amount of rainfall in Los Angeles

Usage

```
RAINFALL
```

Format

RAINFALL is a univariate time series of length 119; start 1878, frequency = 1

RAINFALL Amount of rainfall in Los Angeles

Source

LA Times (January 28. 1997)

Examples

```
data(RAINFALL)
## maybe  tsp(RAINFALL) ; plot(RAINFALL)
```

REDWINE *Monthly sales of Australian red wine (1000 l)*

Description

Monthly sales of Australian red wine (1000 l)

Usage

```
REDWINE
```

Format

REDWINE is a univariate time series of length 187; start January 1980, frequency =12

REDWINE Monthly sales of Australian red wine

Source

R package tsdl <https://github.com/FinYang/tsdl>

Examples

```
data(REDWINE)
## maybe  tsp(REDWINE) ; plot(REDWINE)
```

`rlassoHAC` *rlassoHAC performs Lasso estimation under heteroscedastic and autocorrelated non-Gaussian disturbances.*

Description

rlassoHAC performs Lasso estimation under heteroscedastic and autocorrelated non-Gaussian disturbances.

Usage

```
rlassoHAC(
  x,
  y,
  kernel = "Bartlett",
  bands = 10,
  bns = 10,
  lns = NULL,
  nboot = 5000,
  post = TRUE,
  intercept = TRUE,
  model = TRUE,
  X.dependent.lambda = FALSE,
  c = 2,
  gamma = NULL,
  numIter = 15,
  tol = 10^-5,
  threshold = NULL,
  ...
)
```

Arguments

x	Regressors (vector, matrix or object can be coerced to matrix).
y	Dependent variable (vector, matrix or object can be coerced to matrix).
kernel	Kernel function, choose between "Truncated", "Bartlett" (by default), "Parzen", "Tukey-Hanning", "Quadratic Spectral".
bands	Bandwidth parameter with default bands=10.
bns	Block length with default bns=10.
lns	Number of blocks with default lns = floor(T/bns).
nboot	Number of bootstrap iterations with default nboot=5000.
post	Logical. If TRUE (default), post-Lasso estimation is conducted, i.e. a refit of the model with the selected variables.
intercept	Logical. If TRUE, intercept is included which is not penalized.

model	Logical. If TRUE (default), model matrix is returned.
X.dependent.lambda	Logical, TRUE, if the penalization parameter depends on the design of the matrix x. FALSE (default), if independent of the design matrix.
c	Constant for the penalty, default value is 2.
gamma	Constant for the penalty, default gamma=0.1/log(T) with T=data length.
numIter	Number of iterations for the algorithm for the estimation of the variance and data-driven penalty, ie. loadings.
tol	Constant tolerance for improvement of the estimated variances.
threshold	Constant applied to the final estimated lasso coefficients. Absolute values below the threshold are set to zero.
...	further parameters

Value

rlassoHAC returns an object of class "rlasso". An object of class "rlasso" is a list containing at least the following components:

coefficients	Parameter estimates.
beta	Parameter estimates (named vector of coefficients without intercept).
intercept	Value of the intercept.
index	Index of selected variables (logical vector).
lambda	Data-driven penalty term for each variable, product of lambda0 (the penalization parameter) and the loadings.
lambda0	Penalty term.
loadings	Penalty loadings, vector of lenght p (no. of regressors).
residuals	Residuals, response minus fitted values.
sigma	Root of the variance of the residuals.
iter	Number of iterations.
call	Function call.
options	Options.
model	Model matrix (if model = TRUE in function call).

Source

Victor Chernozhukov, Chris Hansen, Martin Spindler (2016). hdm: High-Dimensional Metrics, R Journal, 8(2), 185-199. URL https://journal.r-project.org/archive/2016/RJ-2016-040/index.html.

Examples

```
set.seed(1)
T = 100 #sample size
p = 20 # number of variables
b = 5 # number of variables with non-zero coefficients
beta0 = c(rep(10,b), rep(0,p-b))
rho = 0.1 #AR parameter
Cov = matrix(0,p,p)
for(i in 1:p){
  for(j in 1:p){
     Cov[i,j] = 0.5^(abs(i-j))
  }
}
C <- chol(Cov)
X <- matrix(rnorm(T*p),T,p)%*%C
eps <- arima.sim(list(ar=rho), n = T+100)
eps <- eps[101:(T+100)]
Y = X%*%beta0 + eps
reg.lasso.hac1 <- rlassoHAC(X, Y,"Bartlett") #lambda is chosen independent of regressor
                                             #matrix X by default.

bn = 10 # block length
bwNeweyWest = 0.75*(T^(1/3))
reg.lasso.hac2 <- rlassoHAC(X, Y,"Bartlett", bands=bwNeweyWest, bns=bn, nboot=5000,
                            X.dependent.lambda = TRUE, c=2.7)
```

`rlassoLoad` *`rlassoLoad` performs Lasso estimation under heteroscedastic and autocorrelated non-Gaussian disturbances with predefined penalty loadings.*

Description

`rlassoLoad` performs Lasso estimation under heteroscedastic and autocorrelated non-Gaussian disturbances with predefined penalty loadings.

Usage

```
rlassoLoad(
  x,
  y,
  load,
  bns = 10,
  lns = NULL,
  nboot = 5000,
  post = TRUE,
```

```
  intercept = TRUE,
  model = TRUE,
  X.dependent.lambda = FALSE,
  c = 2,
  gamma = NULL,
  numIter = 15,
  tol = 10^-5,
  threshold = NULL,
  ...
)
```

Arguments

`x`	Regressors (vector, matrix or object can be coerced to matrix).
`y`	Dependent variable (vector, matrix or object can be coerced to matrix).
`load`	Penalty loadings, vector of length p (no. of regressors).
`bns`	Block length with default bns=10.
`lns`	Number of blocks with default lns = floor(T/bns).
`nboot`	Number of bootstrap iterations with default nboot=5000.
`post`	Logical. If TRUE (default), post-Lasso estimation is conducted, i.e. a refit of the model with the selected variables.
`intercept`	Logical. If TRUE, intercept is included which is not penalized.
`model`	Logical. If TRUE (default), model matrix is returned.
`X.dependent.lambda`	Logical, TRUE, if the penalization parameter depends on the design of the matrix x. FALSE (default), if independent of the design matrix.
`c`	Constant for the penalty default is 2.
`gamma`	Constant for the penalty default gamma=0.1/log(T) with T=data length.
`numIter`	Number of iterations for the algorithm for the estimation of the variance and data-driven penalty.
`tol`	Constant tolerance for improvement of the estimated variances.
`threshold`	Constant applied to the final estimated lasso coefficients. Absolute values below the threshold are set to zero.
`...`	further parameters

Value

rlassoLoad returns an object of class "rlasso". An object of class "rlasso" is a list containing at least the following components:

`coefficients`	Parameter estimates.
`beta`	Parameter estimates (named vector of coefficients without intercept).
`intercept`	Value of the intercept.
`index`	Index of selected variables (logical vector).

lambda	Data-driven penalty term for each variable, product of lambda0 (the penalization parameter) and the loadings.
lambda0	Penalty term.
loadings	Penalty loadings, vector of lenght p (no. of regressors).
residuals	Residuals, response minus fitted values.
sigma	Root of the variance of the residuals.
iter	Number of iterations.
call	Function call.
options	Options.
model	Model matrix (if model = TRUE in function call).

Source

Victor Chernozhukov, Chris Hansen, Martin Spindler (2016). hdm: High-Dimensional Metrics, R Journal, 8(2), 185-199. URL https://journal.r-project.org/archive/2016/RJ-2016-040/index.html.

Examples

```
set.seed(1)
T = 100 #sample size
p = 20 # number of variables
b = 5 # number of variables with non-zero coefficients
beta0 = c(rep(10,b), rep(0,p-b))
rho = 0.1 #AR parameter
Cov = matrix(0,p,p)
for(i in 1:p){
  for(j in 1:p){
     Cov[i,j] = 0.5^(abs(i-j))
  }
}
C <- chol(Cov)
X <- matrix(rnorm(T*p),T,p)%*%C
eps <- arima.sim(list(ar=rho), n = T+100)
eps <- eps[101:(T+100)]
Y = X%*%beta0 + eps

fit1 =  rlasso(X, Y, penalty = list(homoscedastic = "none",
              lambda.start = 2*0.5*sqrt(T)*qnorm(1-0.1/(2*p))), post=FALSE)
beta = fit1$beta
intercept = fit1$intercept
res = Y - X %*% beta - intercept * rep(1, length(Y))

load = rep(0,p)
for(i in 1:p){
  load[i] = sqrt(lrvar(X[,i]*res)*T)
  }
reg.lasso.load1 <- rlassoLoad(X,Y,load) #lambda is chosen independent of regressor
                                             #matrix X by default.
```

```
bn = 10 # block length
reg.lasso.load2 <- rlassoLoad(X, Y,load, bns=bn, nboot=5000,
                            X.dependent.lambda = TRUE, c=2.7)
```

---

robsplinedecomp — `robsplinedecomp` *decomposes a vector into trend, season and irregular component by robustified spline approach; a time series attribute is lost*

Description

`robsplinedecomp` decomposes a vector into trend, season and irregular component by robustified spline approach; a time series attribute is lost

Usage

```
robsplinedecomp(y, d, alpha, beta, Plot = FALSE)
```

Arguments

`y`	the series, a vector or a time series
`d`	seasonal period
`alpha`	smoothing parameter for trend component (the larger alpha is, the smoother will the smooth component g be)
`beta`	smoothing parameter for seasonal component
`Plot`	logical, should a plot be produced?

Value

out list with the elements trend, season, residual

Examples

```
data(GDP)
out  <- robsplinedecomp(GDP,4,2,10,Plot=FALSE)
```

RS — *RS rescaled adjusted range statistic*

Description

RS rescaled adjusted range statistic

Usage

```
RS(x, k)
```

Arguments

x	univariate time series
k	length of the segments for which the statistic is computed. Starting with t=1, the segments do not overlap.

Value

(l,3)-matrix, 1. column: k, second column: starting time of segment, third column: value of RS statistic.

Examples

```
data(TREMOR)
R <- RS(TREMOR,10)
```

SALES — *Monthly sales of a company*

Description

Monthly sales of a company

Usage

```
SALES
```

Format

SALES is a univariate time series of length 77:

y monthly sales of a company

Source

Newton, H. J. (1988, ISBN: 978-0534091989): "TIMESLAB: A time series analysis laboraty"

Examples

```
data(SALES)
## maybe  tsp(SALES) ; plot(SALES)
```

SCHAUINSLAND — *CO2-Concentration obtained in Schauinsland, Germany*

Description

CO2-Concentration obtained in Schauinsland, Germany

Usage

```
SCHAUINSLAND
```

Format

SCHAUINSLAND is a univariate time series of length 72:

SCHAUINSLAND CO2-Concentration obtained in Schauinsland

Source

<http://cdiac.ornl.gov/trends/co2/uba/uba-sc.html>

Examples

```
data(SCHAUINSLAND)
## maybe  tsp(SCHAUINSLAND) ; plot(SCHAUINSLAND)
```

---

simpledecomp — `simpledecomp` *decomposes a vector into trend, season and irregular component by linear regression approach*

Description

`simpledecomp` decomposes a vector into trend, season and irregular component by linear regression approach

Usage

```
simpledecomp(y, trend = 0, season = 0, Plot = FALSE)
```

Arguments

`y`	the series, a vector or a time series
`trend`	order of trend polynomial
`season`	period of seasonal component
`Plot`	logical, should a plot be produced?

Value

out: (n,3) matrix

`1. column`	smooth component
`2. column`	seasonal component
`3. column`	irregular component

Examples

```
data(GDP)
out  <- simpledecomp(GDP,trend=3,season=4,Plot=FALSE)
```

---

smoothls — `smoothls` *smoothes a time series by Whittaker graduation. The function depends on the package Matrix.*

Description

`smoothls` smoothes a time series by Whittaker graduation. The function depends on the package Matrix.

Usage

```
smoothls(y, beta = 0)
```

Arguments

`y`	the series, a vector or a time series
`beta`	smoothing parameter >=0 (the larger beta is, the smoother will g be)

Value

g vector, smooth component

Examples

```
data(GDP)
g <- smoothls(GDP,12)

 plot(GDP)
 t <- seq(from = tsp(GDP)[1], to = tsp(GDP)[2],by=1/tsp(GDP)[3]) ; lines(t,g,col="red")
```

smoothrb — *smoothrb smoothes a time series robustly by using Huber's psi-function. The initialisation uses a moving median.*

Description

smoothrb smoothes a time series robustly by using Huber's psi-function. The initialisation uses a moving median.

Usage

```
smoothrb(y, beta = 0, q = NA)
```

Arguments

y	the series, a vector or a time series
beta	smoothing parameter (The larger beta is, the smoother will the smooth component g be.)
q	length of running median which is used to get initial values

Value

g vector, the smooth component

Examples

```
data(GDP)
g  <- smoothrb(GDP,8,q=8)

 plot(GDP) ; t <- seq(from = 1970, to = 2009.5,by=0.25) ; lines(t,g,col="red")
```

specest — *specest direct spectral estimation of series y using periodogram window win*

Description

specest direct spectral estimation of series y using periodogram window win

Usage

```
specest(
  y,
  nf,
  e,
  win = c("perwinba", "perwinpa", "perwinda"),
  conf = 0,
  type = "cov"
)
```

Arguments

`y`	(n,1) vector, the ts
`nf`	number of equally spaced frequencies
`e`	equal bandwidth, must be 0 <= e <0.5
`win`	string, name of periodogram window (possible: "perwinba", "perwinpa", "perwinda")
`conf`	scalar, the level for confidence intervals
`type`	c("cov","cor"), area under spectrum is variance or is normed to 1.

Value

est (nf+1,2)- or (nf+1,4)-matrix:

column 1:	frequencies 0, 1/n, 2/n, ..., m/n
column 2:	the estimated spectrum
column 3+4:	the confidence bounds

Examples

```
data(WHORMONE)
est <- specest(WHORMONE,50,0.05,win = c("perwinba","perwinpa","perwinda"),conf=0,type="cov")
```

---

specplot — `specplot` *plot of spectral estimate*

Description

`specplot` plot of spectral estimate

Usage

```
specplot(s, Log = FALSE)
```

Arguments

`s`	(n,2) or (n,4) matrix, output of specest
`Log,`	logical, if TRUE, the logs of the spectral estimates are shown

Examples

```
data(WHORMONE)
est <- specest(WHORMONE,50,0.05,win = c("perwinba","perwinpa"),conf=0,type="cov")
specplot(est,Log=FALSE)
```

`splinedecomp` — `splinedecomp` *decomposes a time series into trend, season and irregular component by spline approach.*

Description

`splinedecomp` decomposes a time series into trend, season and irregular component by spline approach.

Usage

```
splinedecomp(x, d, alpha, beta, Plot = FALSE)
```

Arguments

`x`	the series, a vector or a time series
`d`	seasonal period
`alpha`	smoothing parameter for trend component (The larger alpha is, the smoother will the smooth component g be.)
`beta`	smoothing parameter for seasonal component
`Plot`	logical, should a plot be produced?

Value

out (n,3) matrix:

`1. column`	smooth component
`2. column`	seasonal component
`3. column`	irregular component

Examples

```
data(GDP)
out  <- splinedecomp(GDP,4,2,4,Plot=FALSE)
```

`SPRUCE` *Annual logging of spruce wood.*

Description

Annual logging of spruce wood.

Usage

```
SPRUCE
```

Format

SPRUCE is a univariate time series of length 42:

SPRUCE Annual logging of spruce wood

Examples

```
data(SPRUCE)
## maybe  tsp(SPRUCE) ; plot(SPRUCE)
```

`statcheck` `statcheck` *determines the means, standard deviations and acf's of segmets of a time series and plots the acf's for the segments.*

Description

`statcheck` determines the means, standard deviations and acf's of segmets of a time series and plots the acf's for the segments.

Usage

```
statcheck(y, d)
```

Arguments

`y`	the series, a vector or a time series
`d`	scalar, number of segments

Value

out list with components:

`ms`	matrix with means and standard deviations of the segments
`ac`	matrix with acf's, the first column: acf of the series, the others: acf's of the segments

Examples

```
data(COFFEE)
out <- statcheck(COFFEE,4)
```

---

subsets — `subsets` *determines all subsets of a set of n elements (labelled by 1,2,...,n).*

Description

`subsets` determines all subsets of a set of n elements (labelled by 1,2,...,n).

Usage

```
subsets(n)
```

Arguments

`n`	scalar, integer >= 1

Value

mat (2^n,n)-matrix, each row gives the membership indicators of the elements 1,2,...,n

Examples

```
out <- subsets(4)
```

---

symplot — `symplot` *produces a symmetry plot*

Description

`symplot` produces a symmetry plot

Usage

```
symplot(y)
```

Arguments

`y`	the series, a vector or a time series

Examples

```
data(LYNX)
symplot(LYNX)
```

`taper` *taper modification of a time series*

Description

`taper` taper modification of a time series

Usage

```
taper(y, part)
```

Arguments

y	the time series
part	scalar, 0 <= part <= 0.5, part of modification (at each end of y)

Value

tp tapered time series

Examples

```
data(WHORMONE)
out <-taper(WHORMONE,0.3)

plot(WHORMONE)
lines(out,col="red")
```

`TAXES` *Monthly community taxes in Germany (billions EURO)*

Description

Monthly community taxes in Germany (billions EURO)

Usage

```
TAXES
```

Format

TAXES is a univariate time series of length 246; start January 1999, frequency = 12

TAXES monthly community taxes in Germany

Source

<https://www-genesis.destatis.de/genesis/online?operation=previous&levelindex=1&step=1&titel=Tabellenaufbau&levelid=1583748637039>

Examples

```
data(TAXES)
## maybe  tsp(TAXES) ; plot(TAXES)
```

`TREERING` *Mean thickness of annual tree rings*

Description

Mean thickness of annual tree rings

Usage

```
TREERING
```

Format

TREERING is a multivariate time series of length 66 with 3 variables:

THICK mean thickness of annual tree rings

TEMP mean temperature of the year

RAIN amount of rain of the year

Source

<https://ltrr.arizona.edu/>

Examples

```
data(TREERING)
## maybe  tsp(TREERING) ; plot(TREERING)
```

TREMOR *Measurements of physiological tremor*

Description

Measurements of physiological tremor

Usage

```
TREMOR
```

Format

TREMOR is a univariate time series of length 400.

TREMOR Tremor

Examples

```
data(TREMOR)
## maybe  tsp(TREMOR) ; plot(TREMOR)
```

tsmat `tsmat` *constructs a (n-p+1,p) matrix from a time series where the first column is the shortened series y[p],...,y[n], the second is y[p-1],...,y[n-1], etc.*

Description

`tsmat` constructs a (n-p+1,p) matrix from a time series where the first column is the shortened series y[p],...,y[n], the second is y[p-1],...,y[n-1], etc.

Usage

```
tsmat(y, p)
```

Arguments

y	the series, a vector or a time series of length n
p	desired number of columns

Value

mat (n-p+1,p) matrix

Examples

```
out <- tsmat(c(1:20),4)
```

USAPOP — *Population of USA*

Description

Population of USA

Usage

```
USAPOP
```

Format

USAPOP is a univariate time series of length 39; start 1630, frequency = 0.1

USAPOP Population of USA

Source

<https://www.worldometers.info/world-population/us-population/>

Examples

```
data(USAPOP)
## maybe  tsp(USAPOP) ; plot(USAPOP)
```

vartable — `vartable` *determines table of variate differences*

Description

`vartable` determines table of variate differences

Usage

```
vartable(y, season)
```

Arguments

`y`	the series, a vector or a time series (no NA's)
`season`	scalar, period of seasonal component

Value

d matrix with ratios of variances for differend numbers of simple and seasonal differencing

Examples

```
data(GDP)
out <- vartable(GDP,4)
```

WHORMONE *Concentration of growth hormone of a bull*

Description

Concentration of growth hormone of a bull

Usage

```
WHORMONE
```

Format

WHORMONE is a univariate time series of length 97:

WHORMONE Concentration of growth hormone of a bull

Source

Newton, H. J. (1988, ISBN: 978-0534091989): "TIMESLAB: A time series analysis laboraty"

Examples

```
data(WHORMONE)
## maybe  tsp(WHORMONE) ; plot(WHORMONE)
```

---

wntest `wntest` *graphical test for white noise for a time series or a series of regression residuals*

Description

`wntest` graphical test for white noise for a time series or a series of regression residuals

Usage

```
wntest(e, a, k = 0)
```

Arguments

`e`	vector, the time series (k = 0) or residuals (k > 0)
`a`	scalar, level of significance
`k`	scalar >= 0, number of regressors used to compute e as residuals

Value

tp vector, value of test statistic and p-value

Examples

```
data(WHORMONE)
out <- wntest(WHORMONE,0.05,0)
```

Index