Package: RCMinification (via r-universe)

August 28, 2024

Version 1.2

Title Random Coefficient Minification Time Series Models

Author L. Han [aut, cre]

Maintainer L. Han <lengyi.han@ubc.ca>

Description Data sets, and functions for simulating and fitting nonlinear time series with minification and nonparametric models.

Depends KernSmooth, locpol

LazyLoad true

LazyData true

ZipData no

License Unlimited

NeedsCompilation no

Date/Publication 2023-01-17 10:10:02 UTC

Repository https://spectrumhan.r-universe.dev

RemoteUrl https://github.com/cran/RCMinification

RemoteRef HEAD

RemoteSha 8325c1957b5fca341ec89d2a703bea0797037057

Contents

`ARlocpoly` *Fit a nonlinear AR1 model using local polynomial regression*

Description

This function uses local polynomial regression to nonparametrically estimate the autoregression function in a nonlinear AR1 model.

Usage

```
ARlocpoly(z, deg = 1, h, ...)
```

Arguments

z	numeric vector of time series observations.
deg	numeric, degree of local polynomial fit.
h	numeric, bandwidth for local polynomial estimate.
...	any other arguments taken by `locpoly`.

Value

A list containing

x	numeric vector of evaluation points.
y	numeric vector of nonparametric estimates at the values in `x`.
h	numeric, bandwidth

Author(s)

L. Han and S. Snyman

References

Fan, J. and Yao, Q. (2008) Nonlinear Time Series: Nonparametric and Parametric Methods. Springer.

Examples

```
x <- nonlinearAR1.sim(100, g = function(x) x*sin(x),  sd = 1.5) # simulated data
ARlocpoly(x, deg = 0, h = 0.5)
```

`BCfireArea` *BC Fire Area*

Description

The `BCfireArea` time series object consists of 13 observations on annual area burnt in the province of BC.

Usage

```
data(BCfireArea)
```

Format

A time series object

Examples

```
ts.plot(BCfireArea)
```

`BeerVolume` *Beer Volume Time Series*

Description

Weekly volumes (in litres) of produced at a large brewery for 137 weeks.

Usage

```
data(FWI)
```

Format

A time series object

Examples

```
acf(BeerVolume)
```

ChengTS — *Fit a nonlinear AR1 model using local polynomial regression via the method of Cheng et al.*

Description

This function uses local polynomial regression to nonparametrically estimate the autoregression function in a nonlinear AR1 model using Cheng's bias reduction method.

Usage

```
ChengTS(z, degree = 1, hopt, ...)
```

Arguments

`z`	numeric vector of time series observations.
`degree`	numeric, degree of local polynomial fit.
`hopt`	numeric, base bandwidth for local polynomial estimate.
`...`	any other arguments taken by `locpoly`.

Value

A list containing

`x`	numeric vector of evaluation points.
`y`	numeric vector of nonparametric estimates at the values in `x`.

Author(s)

L. Han and S. Snyman

References

Cheng, M., Huang, R., Liu, P. and Liu, H. (2018) Bias reduction for nonparametric and semiparametric regression models. Statistica Sinica 28(4):2749-2770.

Examples

```
x <- nonlinearAR1.sim(100, g = function(x) x*sin(x),  sd = 1.5) # simulated data
ChengTS(x, degree = 1, hopt = 0.5)

x <- nonlinearAR1.sim(100, g = function(x) x*sin(x),  sd = 0.5) # simulated data
degree <- 1; xrange <- c(-.5, .5); n <- length(x)
h <- thumbBw(x[-n], x[-1], deg = degree, kernel=gaussK)
x.lp <- ARlocpoly(x, deg = degree, h = h, range.x = xrange)
x.shp <- sharpARlocpoly(x, deg = degree, range.x = xrange, h = x.lp$h*n^(4/45))
x.cheng <- ChengTS(x, degree = degree,  hopt = h, range.x = xrange)
lag.plot(x, do.lines=FALSE)
```

```
lines(x.lp)
lines(x.shp, col=2)
lines(x.cheng, col=4)
```

FWI *Fire Weather Index Series*

Description

The FWI list consists of 4 vectors containing daily Fire Weather Index observations.

Usage

```
data(FWI)
```

Format

This list contains the following vectors:

PG2008 FWI observations from Prince George, BC for 2008

PG2009 FWI observations from Prince George, BC for 2009

ED2013 FWI observations from Edmonton, AB for 2013

ED2014 FWI observations from Edmonton, AB for 2014

Examples

```
RCMTmle(FWI$PG2009[c(100:300)])
```

Globaltemps *Global Average Temperature Changes*

Description

Global average temperatures are recorded in terms of number of Celsius degrees above a baseline temperature from 1880 to 2016. The baseline temperature is the average temperature for the year 1990.

Usage

```
data(Globaltemps)
```

Format

A numeric vector

Examples

```
temps <- ts(Globaltemps, start = 1880, end = 2016)
ts.plot(temps, ylab =  "Change in Temperature")
```

`longitudinalAcceleration` *Longitudinal Acceleration Measurements on an Air Tanker*

Description

Longitudinal acceleration measurements of an air tanker fighting a forest wildfire taken at 1 second intervals.

Usage

```
data(longitudinalAcceleration)
```

Format

A time series object

Examples

```
acf(longitudinalAcceleration)
```

`nickel` *Electroless nickel concentrations*

Description

Electroless nickel concentrations in a chrome plating process were measured at the beginning of each eight hour work shift for a period of 25 days. A concentration of 4.5 ounces per gallon is considered optimal in this application.

Usage

```
data(nickel)
```

Format

A time series object

Source

Farnum, N. (1994) Statistical Quality Control and Improvement. Belmont, Duxbury Press.

Examples

```
ts.plot(nickel)
```

nonlinearAR1.sim *Nonlinear AR1 Simulator*

Description

This function simulates sequences of variates follow a nonlinear autoregressive order 1 process of the form z_n = g(z_n-1) + epsilon. A normal distribution is assumed for the innovations.

Usage

```
nonlinearAR1.sim(n, g, ...)
```

Arguments

`n`	number of observations.
`g`	autoregression function.
`...`	any parameters that are taken by `rnorm`

Author(s)

L. Han and S. Snyman

Examples

```
x <- nonlinearAR1.sim(50, g = function(x) x*sin(x), sd = 2.5)
ts.plot(x)
```

RCMTmle *Tailed Exponential and Weibull Random Coefficient Minification Maximum Likelihood Estimation*

Description

This function estimates parameters for tailed exponential and Weibull random coefficient minification process models from a nonnegative time series.

Usage

```
RCMTmle(y)
```

Arguments

`y`	numeric vector of nonnegative observations.

Value

A list containing

`n`	the number of time series observations.
`p`	estimated power for transformation from exponential to Weibull.
`p.eps`	estimated tailed exponential probability parameter when preceding observation is nonzero.
`p.delta`	estimated tailed exponential probability parameter when preceding observation is 0
`mu`	estimated mu parameter for lognormal distribution used to simulated random coefficients.
`sigma`	estimated sigma parameter for lognormal distribution used to simulate random coefficients.
`lambda`	estimated tailed exponential rate parameter when preceding observation is nonzero.
`gamma`	estimated tailed exponential rate parameter when preceding observation is 0.
`like`	maximum value of likelihood.
`y`	original observations

Author(s)

L. Han

References

Han, L., Braun, W.J. and Loeppky (2018) Random Coefficient Minification Processes. Statistical Papers, pp 1-22.

`rET` *Tailed Exponential Random Number Generator*

Description

This function simulates sequences of tailed exponential variates which have survivor function P(X > x) = (1-p)exp(-lambda x), for x > 0 and P(X = 0) = p.

Usage

```
rET(n, prob, rate)
```

Arguments

`n`	number of observations.
`prob`	vector of probabilities.
`rate`	vector of exponential rate parameters.

Author(s)

L. Han

References

Littlejohn, R.P. (1994) A Reversibility Relationship for Two Markovian Time Series Models with Stationary Exponential Tailed Distribution. Journal of Applied Probability. 31 pp 575-581.

`robustSD` *Tatum's Robust Estimate of the Standard Deviation*

Description

Standard deviation estimate which is insensitive to outliers and random trends.

Usage

```
robustSD(x)
```

Arguments

`x`	A numeric vector.

Author(s)

L. Han

References

Tatum, L.G. (1997) Robust Estimation of the Process Standard Deviation for Control Charts. Journal of the American Statistical Association 39, pp 127-141.

Examples

```
robustSD(EuStockMarkets[,1])
```

`rRCMT` *Tailed Exponential and Weibull Random Coefficient Minification Process Simulator*

Description

This function simulates sequences of tailed exponential and Weibull random coefficient minification process variates. Random coefficients are lognormal distributed with parameters mu and sigma.

Usage

```
rRCMT(n, p, p.delta, p.eps, lambda, gamma, mu, sigma, RCMTobj)
```

Arguments

`n`	number of observations.
`p`	power for transformation from exponential to Weibull.
`p.delta`	tailed exponential probability parameter when preceding observation is 0
`p.eps`	tailed exponential probability parameter when preceding observation is nonzero.
`lambda`	tailed exponential rate parameter when preceding observation is nonzero.
`gamma`	tailed exponential rate parameter when preceding observation is 0.
`mu`	mu parameter for lognormal distribution used to simulated random coefficients.
`sigma`	sigma parameter for lognormal distribution used to simulate random coefficients.
`RCMTobj`	list containing elements n, p, p.delta, p.eps, lambda and gamma

Author(s)

L. Han

References

Han, L., Braun, W.J. and Loeppky (2018) Random Coefficient Minification Processes. Statistical Papers, pp 1-22.

`sharpARlocpoly` *Fit a nonlinear AR1 model using local polynomial regression and data sharpening*

Description

This function uses local polynomial regression to nonparametrically estimate the autoregression function in a nonlinear AR1 model, after employing data sharpening on the responses.

Usage

```
sharpARlocpoly(z, deg = 1, h, ...)
```

Arguments

`z`	numeric vector of time series observations.
`deg`	numeric, degree of local polynomial fit.
`h`	numeric, bandwidth for local polynomial estimate.
`...`	any other arguments taken by `ARlocpoly`.

Value

A list containing

`x`	numeric vector of evaluation points.
`y`	numeric vector of nonparametric estimates at the values in `x`.

Author(s)

L. Han and S. Snyman

References

Choi, E., Hall, P. and Rousson, V. (2000) Data Sharpening Methods for Bias Reduction in Nonparametric Regression. Annals of Statistics 28(5):1339-1355.

Examples

```
x <- nonlinearAR1.sim(100, g = function(x) x*sin(x),  sd = 1.5) # simulated data
sharpARlocpoly(x, deg = 0, h = 0.5)
```

Index