Package 'fNonlinear'

November 16, 2017

Title Rmetrics - Nonlinear and Chaotic Time Series Modelling

Date 2017-11-12

Version 3042.79

Author Diethelm Wuertz [aut],
Tobias Setz [cre],
Yohan Chalabi [ctb]

Maintainer Tobias Setz <tobias.setz@live.com>

Description Provides a collection of functions for testing various aspects of
univariate time series including independence and neglected
nonlinearities. Further provides functions to investigate the chaotic
behavior of time series processes and to simulate different types of chaotic
time series maps.

Depends R (>= 2.15.1), timeDate, timeSeries, fBasics

Imports methods, stats

Suggests RUnit, tcltk

LazyData yes

License GPL (>= 2)

URL http://www.rmetrics.org

NeedsCompilation yes

Repository CRAN

Date/Publication 2017-11-16 22:02:44 UTC

R topics documented:

NonLinModelling   *Chaotic Time Series Modelling*

Description

A collection and description of functions to simulate different types of chaotic time series maps.

Chaotic Time Series Maps:

tentSim	Simulates data from the Tent Map,
henonSim	simulates data from the Henon Map,
ikedaSim	simulates data from the Ikeda Map,
logisticSim	simulates data from the Logistic Map,
lorentzSim	simulates data from the Lorentz Map,
roesslerSim	simulates data from the Roessler Map.

Usage

```
tentSim(n = 1000, n.skip = 100, parms = c(a = 2), start = runif(1),
    doplot = FALSE)
henonSim(n = 1000, n.skip = 100, parms = c(a = 1.4, b = 0.3),
    start = runif(2), doplot = FALSE)
ikedaSim(n = 1000, n.skip = 100, parms = c(a = 0.4, b = 6.0, c = 0.9),
    start = runif(2), doplot = FALSE)
logisticSim(n = 1000, n.skip = 100, parms = c(r = 4), start = runif(1),
    doplot = FALSE)
lorentzSim(times = seq(0, 40, by = 0.01), parms = c(sigma = 16, r = 45.92,
    b = 4), start = c(-14, -13, 47), doplot = TRUE, ...)
roesslerSim(times = seq(0, 100, by = 0.01), parms = c(a = 0.2, b = 0.2, c = 8.0),
    start = c(-1.894, -9.920, 0.0250), doplot = TRUE, ...)
```

Arguments

doplot	a logical flag. Should a plot be displayed?
n, n.skip	[henonSim][ikedaSim][logisticSim] - the number of chaotic time series points to be generated and the number of initial values to be skipped from the series.
parms	the named parameter vector characterizing the chaotic map.
start	the vector of start values to initiate the chaotic map.
times	[lorentzSim][roesslerSim] - the sequence of time series points at which to generate the map.
...	arguments to be passed.

Value

[*Sim] -
All functions return invisible a vector of time series data.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

References

Brock, W.A., Dechert W.D., Sheinkman J.A. (1987); *A Test of Independence Based on the Correlation Dimension*, SSRI no. 8702, Department of Economics, University of Wisconsin, Madison.

Eckmann J.P., Oliffson Kamphorst S., Ruelle D. (1987), *Recurrence plots of dynamical systems*, Europhys. Letters 4, 973.

Hegger R., Kantz H., Schreiber T. (1999); *Practical implementation of nonlinear time series methods: The TISEAN package*, CHAOS 9, 413–435.

Kennel M.B., Brown R., Abarbanel H.D.I. (1992); *Determining embedding dimension for phase-space reconstruction using a geometrical construction*, Phys. Rev. A45, 3403.

Rosenstein M.T., Collins J.J., De Luca C.J. (1993); *A practical method for calculating largest Lyapunov exponents from small data sets*, Physica D 65, 117.

See Also

RandomInnovations.

Examples

```
## logisticSim -
   set.seed(4711)
   x = logisticSim(n = 100)
   plot(x, main = "Logistic Map")
```

NonLinStatistics   *Chaotic Time Series Statistics*

Description

A collection and description of functions to investigate the chaotic behavior of time series processes.

Functions to Analyse Chaotic Time Series:

mutualPlot	Returns mutual information,
falsennPlot	returns false nearest neigbours,
recurrencePlot	returns a recurrence plot,
separationPlot	returns a space-time separation plot,
lyapunovPlot	computes maximum lyapunov exponent.

Usage

```
mutualPlot(x, partitions = 16, lag.max = 20, doplot = TRUE, ...)
falsennPlot(x, m, d, t, rt = 10, eps = NULL, doplot = TRUE, ...)
recurrencePlot(x, m, d, end.time, eps, nt = 10, doplot = TRUE, ...)
separationPlot(x, m, d, mdt, idt = 1, doplot = TRUE, ...)
lyapunovPlot(x, m, d, t, ref, s, eps, k = 1, doplot = TRUE, ...)
```

Arguments

d	an integer value setting the value of the time delay.
eps	[falsennPlot] - a numeric value setting the value of the neighbour diameter. If NULL, which is the default value, then the value will be automatically setted to eps=sd(x)/10. [lyapunovPlot] - the radius where to find nearest neighbours. [recurrencePlot] - the neighbourhood threshold.
doplot	a logical flag. Should a plot be displayed?
end.time	[recurrencePlot] - ending time as number of observations.
idt	[separationPlot] - an integer value setting the number of observation steps in each iterations. By default 1.
k	[lyapunovPlot] - an integer setting th enumber of considered neighbours. By default 1.
lag.max	[mutualPlot] - an integer value setting the number of maximum lags, by default 20.
m	[*Plot] - an integer value setting the value of the maximum embedding dimension.
mdt	[separationPlot] - an integer value setting the number of iterations.
nt	[recurrencePlot] - observations in each step which will be plotted, by default 10. Increasing nt reduces number of points plotted which is usefule especially with highly sampled data.
rt	[falsennPlot] - an integer value setting the value for the escape factor. By default 10.
partitions	[mutualPlot] - an integer value setting the number of bins, by default 16.
ref	[lyapunovPlot] - the number of points to take into account.
s	[lyapunovPlot] - the iterations along which follow the neighbours of each point.

t	[*Plot] -	
	an integer value setting the value for the Theiler window.	
x	[*Plot] -	
	a numeric vector, or an object either of class 'ts' or of class 'timeSeries'.	
...	arguments to be passed.	

Details

Phase Space Representation:

The function `mutualPlot` estimates and plots the mutual information index of a given time series for a specified number of lags. The joint probability distribution function is estimated with a simple bi-dimensional density histogram.
The function `falsennPlot` uses the Method of false nearest neighbours to help deciding the optimal embedding dimension.

Non-Stationarity:

The funcdtion `recurrencePlot` creates a recurrence plot as proposed by Eckmann et al. [1987].
The function `separationPlot` creates a space-time separation plot qs introduced by Provenzale et al. [1992]. It plots the probability that two points in the reconstructed phase-space have distance smaller than epsilon in function of epsilon and of the time between the points, as iso-lines at levels 10, 20, ..., 100 percent levels. The plot can be used to decide the Theiler time window.

Lyapunov Exponents:

The function `lyapunovPlot` evaluates and plots the largest Lyapunov exponent of a dynamic system from a univariate time series. The estimate of the Lyapunov exponent uses the algorithm of Kantz. In addition, the function computes the regression coefficients of a user specified segment of the sequence given as input.

Dimensions and Entropies:

The function `C2` computes the sample correlation integral on the provided time series for the specified length scale and Theiler window. It uses a naiv algorithm: simply returns the fraction of points pairs nearer than eps. It is prefarable to use the function `d2`, which takes roughly the same time, but computes the correlation sum for multiple length scales and embedding dimensions at once.
The function `d2` computes the sample correlation integral over given length scales `neps` for embedding dimensions `1:m` for a given Theiler window. The slope of the linear segment in the log-log plot gives an estimate of the correlation dimension.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

References

Brock, W.A., Dechert W.D., Sheinkman J.A. (1987); *A Test of Independence Based on the Correlation Dimension*, SSRI no. 8702, Department of Economics, University of Wisconsin, Madison.

Eckmann J.P., Oliffson Kamphorst S., Ruelle D. (1987), *Recurrence plots of dynamical systems*, Europhys. Letters 4, 973.

Hegger R., Kantz H., Schreiber T. (1999); *Practical implementation of nonlinear time series methods: The TISEAN package*, CHAOS 9, 413–435.

Kennel M.B., Brown R., Abarbanel H.D.I. (1992); *Determining embedding dimension for phase-space reconstruction using a geometrical construction*, Phys. Rev. A45, 3403.

Rosenstein M.T., Collins J.J., De Luca C.J. (1993); *A practical method for calculating largest Lyapunov exponents from small data sets*, Physica D 65, 117.

See Also

RandomInnovations.

Examples

```
## mutualPlot -
   mutualPlot(logisticSim(1000))

## recurrencePlot -
   lorentz = lorentzSim(
     times = seq(0, 40, by = 0.01),
     parms = c(sigma = 16, r = 45.92, b = 4),
     start = c(-14, -13, 47),
     doplot = FALSE)
   recurrencePlot(lorentz[, 2], m = 3, d = 2, end.time = 800, eps = 3,
     nt = 5, pch = '.', cex = 2)
```

NonLinTests   *Time Series Tests*

Description

A collection and description of functions for testing various aspects of univariate time series, including independence, and neglected nonlinearities.

The family of time series tests includes the following hypothesis tests:

```
    bdsTest        Brock--Dechert--Scheinkman test for iid series
    tnnTest        Teraesvirta NN test for neglected nonlinearity
    wnnTest        White NN test for neglected nonlinearity
    runsTest       Runs test for detecting non-randomness
```

Usage

```
bdsTest(x, m = 3, eps = NULL, title = NULL, description = NULL)
tnnTest(x, lag = 1, title = NULL, description = NULL)
wnnTest(x, lag = 1, qstar = 2, q = 10, range = 4, title = NULL, description = NULL)
runsTest(x)
```

Arguments

description	optional description string, or a vector of character strings.
eps	[bdsTest] - a numeric vector of epsilon values for close points. The BDS test is computed for each element of eps. It should be set in terms of the standard deviation of x. If eps is NULL, then the four default values seq(0.5*sd(x), 2*sd(x), length = 4) are used.
lag	[tnnTest][wnnTest] - an integer which specifies the model order in terms of lags.
m	[bdsTest] - an integer indicating that the BDS test statistic is computed for embedding dimensions 2, ..., m.
q	[wnnTest] - an integer representing the number of phantom hidden units used to compute the test statistic.
qstar	[wnnTest] - the test is conducted using qstar principal components of the phantom hidden units. The first principal component is omitted since in most cases it appears to be collinear with the input vector of lagged variables. This strategy preserves power while still conserving degrees of freedom.
range	[wnnTest] - the input to hidden unit weights are initialized uniformly over [-range/2, range/2].
title	an optional title string, if not specified the inputs data name is deparsed.
x	a numeric vector or an object of class "timeseries".

Details

Brock–Dechert–Sheinkman Test:

The bdsTest test examines the *spatial dependence* of the observed series. To do this, the series is embedded in m-space and the dependence of x is examined by counting *near* points. Points for which the distance is less than eps are called near. The BDS test statistic is asymptotically standard Normal. Note, that missing values are not allowed. There is a special print method for objects of class "bdsTest" which by default uses 4 digits to format real numbers.
[tseries:bds.test]

Teraesvirta Neural Network Test:

The null is the hypotheses of linearity in mean. This test uses a Taylor series expansion of the

activation function to arrive at a suitable test statistic. If `type` equals `"F"`, then the F-statistic instead of the Chi-Squared statistic is used in analogy to the classical linear regression. Missing values are not allowed.
`[tseries:teraesvirta.test]`

White Neural Network Test:

The null is the hypotheses of linearity in "mean". This type of test is consistent against arbitrary nonlinearity in mean. If `type` equals `"F"`, then the F-statistic instead of the Chi-Squared statistic is used in analogy to the classical linear regression.
`[tseries:white.test]`

Runs Test:

The runs test can be used to decide if a data set is from a random process. A run is defined as a series of increasing values or a series of decreasing values. The number of increasing, or decreasing, values is the length of the run. In a random data set, the probability that the *(i+1)-th* value is larger or smaller than the *i-th* value follows a binomial distribution, which forms the basis of the runs test.
`[tseries:runs.test]`

Value

In contrast to R's output report from S3 objects of class `"htest"` a different output report is produced. The tests here return an S4 object of class `"fHTEST"`. The object contains the following slots:

`@call`	the function call.
`@data`	the data as specified by the input argument(s).
`@test`	a list whose elements contail the results from the statistical test. The information provided is similar to a list object of `"htest"`.
`@title`	a character string with the name of the test. This can be overwritten specifying a user defined input argument.
`@description`	a character string with an optional user defined description. By default just the current date when the test was applied will be returned.

The slot `@test` returns an object of class `"list"` containing the following (otionally empty) elements:

`statistic`	the value(s) of the test statistic.
`p.value`	the p-value(s) of the test.
`parameters`	a numeric value or vector of parameters.
`estimate`	a numeric value or vector of sample estimates.
`conf.int`	a numeric two row vector or matrix of 95
`method`	a character string indicating what type of test was performed.
`data.name`	a character string giving the name(s) of the data.

Note

The output of the various hypothesis tests is an object of class `htest`. The associated `print` method gives an unique report about the test results.

Author(s)

Adrian Trapletti for the tests from R's tseries package,
Blake LeBaron for the bds C program,
Diethelm Wuertz for the Rmetrics R-port.

References

Brock, W.A., Dechert W.D., Sheinkman J.A. (1987); *A Test of Independence Based on the Correlation Dimension*, SSRI no. 8702, Department of Economics, University of Wisconsin, Madison.

Conover, W.J. (1980); *Practical Nonparametric Statistics*, New York, Wiley.

Cromwell J.B., Labys W.C., Terraza M. (1994); *Univariate Tests for Time Series Models*, Sage, Thousand Oaks, CA, pages 32–36.

Lee T.H., White H., Granger C.W.J. (1993); *Testing for neglected nonlinearity in time series models*, Journal of Econometrics 56, 269–290.

Teraesvirta T., Lin C.F., Granger C.W.J. (1993); *Power of the Neural Network Linearity Test*, Journal of Time Series Analysis 14, 209–220.

Examples

```
## bdsTest -
   # iid Time Series:
   par(mfrow = c(3, 1))
   x = rnorm(100)
   plot(x, type = "l", main = "iid Time Series")
   bdsTest(x, m = 3)
   # Non Identically Distributed Time Series:
   x = c(rnorm(50), runif(50))
   plot(x, type = "l", main = "Non-iid Time Series")
   bdsTest(x, m = 3)
   # Non Independent Innovations from Quadratic Map:
   x = rep(0.2, 100)
   for (i in 2:100) x[i] = 4*(1-x[i-1])*x[i-1]
   plot(x, type = "l", main = "Quadratic Map")
   bdsTest(x, m = 3)

## tnnTest -
   # Time Series Non-linear in "mean" regression
   par(mfrow = c(2, 1))
   n = 1000
   x = runif(1000, -1, 1)
   tnnTest(x)
   # Generate time series which is nonlinear in "mean"
   x[1] = 0.0
   for (i in (2:n)) {
```

```
      x[i] = 0.4*x[i-1] + tanh(x[i-1]) + rnorm (1, sd = 0.5) }
    plot(x, main = "Teraesvirta Test", type = "l")
    tnnTest(x)

## wnnTest -
    # Time Series Non-Linear in "mean" Regression
    par(mfrow = c(2, 1))
    n = 1000
    x = runif(1000, -1, 1)
    wnnTest(x)
    # Generate time series which is nonlinear in "mean"
    x[1] = 0.0
    for (i in (2:n)) {
      x[i] = 0.4*x[i-1] + tanh(x[i-1]) + rnorm (1, sd = 0.5) }
    plot(x, main = "White Test", type = "l")
    wnnTest(x)
```

Index