Package ‘Fractal’

28 September 2019

Title Fractal Analysis

Version 1.4-5

Date 2019-09-22

Author David Harte

Maintainer David Harte <d.s.harte@gmail.com>

Description Contains functions for the simulation of various fractal processes; and estimation of Renyi dimensions of multifractal measures.

URL http://www.statsresearch.co.nz/dsh/sslib/

R topics documented:

`Fractal-package` *Overview of Fractal Package*

Description

The package **Fractal** contains functions for the simulation of various fractal processes; and estimation of Renyi dimensions of multifractal measures.

Cited references, anywhere in the manual, are only listed within this topic.

References

Davies, R.B. & Harte, D.S. (1987). Tests for Hurst effect. *Biometrika* **74(1)**, 95–101. doi: 10.1093/biomet/74.1.95

Falconer, K.J. (1990). *Fractal Geometry, Mathematical Foundations and Applications.* John Wiley & Sons, Chichester.

Harte, D. (1998). Dimension estimates of earthquake epicentres and hypocentres. *Journal of Nonlinear Science* **8**, 581–618. doi: 10.1007/s003329900060

Harte, D. (2001). *Multifractals: Theory and Applications.* Chapman & Hall/CRC, Boca Raton.

Lorenz, E.N. (1963). Deterministic nonperiodic flow. *Journal of the Atmospheric Sciences* **20**, 130–141. doi: 10.1175/15200469(1963)020<0130:DNF>2.0.CO;2

Takens, F. (1981). Detecting strange attractors in turbulence. In: *Dynamical Systems and Turbulence, Warwick 1980. Proceedings of a Symposium held at the University of Warwick 1979/80. Lecture Notes in Mathematics 898.* Pages 366–381. (Edited by D.A. Rand and L.-S. Young). Springer-Verlag, Berlin. doi: 10.1007/bfb0091924

Weisstein, E.W. (2003). Logistic Map. *Eric Weisstein's World of Mathematics.* `http://mathworld.wolfram.com/LogisticMap.html`.

Whitney, H. (1936). Differentiable manifolds. *Annals of Mathematics* **37(3)**, 645–680. doi: 10.2307/1968482

`cantor` *Simulations from Cantor Measure*

Description

Generates a sample of points with probability distribution given by the Cantor measure. This is a probability measure on the middle third Cantor set.

Usage

```
cantor(n, k, p=0.5, seed=5)
```

Arguments

`n`	number of points to be simulated.
`k`	an integer giving the level of resolution. See Details below.
`p`	probability giving the way that weight is reallocated at each iteration.
`seed`	a seed for the random number generator. Any integer between 0 and 1000.

Details

Each realisation is simulated by generating a diadic sequence of zeros and twos (i.e. base three number, where no ones occur) of length k, where the proportion of twos is p; and then converting to a decimal number between zero and one.

Value

A numeric vector of length n containing the simulated sequence.

See Also

`hill`

Examples

```
n <- 10000
y <- cantor(n, 35, p=1/3)

par(mfrow=c(2, 2))
for (k in 1:4)
    hist(y, breaks=seq(0, 1, 1/3^k), right=FALSE,
         main=paste("Scale = 1/", 3^k, sep=""))

par(mfrow=c(1, 1))
plot(sort(y), seq(1,n)/n, type="l",
     ylab="Empirical Distribution Function", xlab="x",
     main=expression(paste("Cantor Measure with ", p[0]==1/3)))
```

`Change Log` *Changes Made to the Package*

Description

This page contains a listing of recent changes made to functions.

Details

1. hill.loop: Occurrences of F have been changed to FALSE. (October 2003)
2. "Standard" keywords have been used on manual pages. However, the "models" keyword is not completely satisfactory. (October 2003)
3. cantor, hill, lorenz: Minor changes to examples. (October 2003)
4. dimension.plot: Log base on x axis formatted as a subscript. (October 2003)
5. Minor formatting changes to manual pages. (January 2004)
6. dimension.plot: Call to X11 graphics device removed. (March 2004)
7. multinomial.theta, moran.theta, moran.sim: New functions. (March 2004)
8. Changes to the DESCRIPTION file. (7 May 2004)
9. multinomial.theta, moran.sim: Fixed documentation errors on manual pages. (19 April 2005)
10. dimension.plot: Reference to X11 graphics device removed from manual page. (21 April 2005)
11. multinomial.theta: Unbalanced bracket in documentation. (29 May 2008)
12. Removal of LaTeX markups from DESCRIPTION file and moran.sim documentation. (31 May 2009)
13. Add CITATION file. (24 Sep 2010)
14. Implement very basic NAMESPACE. (5 Nov 2011)
15. fgn: In call to function complex, change argument imag to imaginary. (03 Apr 2012)
16. Rebuild package. (03 Jul 2015)
17. Additions to file NAMESPACE. (09 Sep 2016)
18. Revise manual pages. (27 Sep 2017)
19. Rebuild package in R version 3.6.1. (22 Sep 2019)

---

dimension.plot *Plot of Renyi Dimension Estimates*

Description

Plots the Renyi dimension estimates, D_q. The estimates will have already been calculated using the function hill. The Hill estimate will be on the y axis and a measure of the interpoint distance (delta) on the x axis.

Usage

```
dimension.plot(DimEst, ylim, xscale="m", criteria=TRUE,
               reflines=NA, logbase=exp(1))
```

Arguments

- `DimEst` — a list object with the same format and information as that output by the function `hill`.
- `ylim` — a vector of length 2. The elements represent the lower and upper limits of the y axis.
- `xscale` — determines the way that the interpoint distances on the x axis are represented. Options are: `"m"` if by the order statistic number m (default), `"delta"` if by the mean value of delta (i.e. mean over all bootstraps for a given value of m), and `"logdelta"` if logarithm of mean delta. The default logarithm is the natural logarithm. See argument `logbase`.
- `criteria` — boolean. If `TRUE` (default), various parameters of the estimation procedure are listed at the bottom of the plot.
- `reflines` — a numeric vector. Each element in the vector represents the intersection with the y axis of a horizontal references line that will be drawn onto the plot. Default is `NA`.
- `logbase` — only has an effect if `xscale` is `"logdelta"`. Determines the base of the logarithm used, the default being the natural logarithm, i.e. $\exp(1)$.

Value

`NULL`

See Also

`hill`

Examples

```
#   See examples in function hill.
```

---

fgn — *Simulate Fractional Gaussian Noise*

Description

Simulates fractional Gaussian noise (*fGn*), i.e. the increments of fractional Brownian motion. Fractional Brownian motion can be simulated by taking the cumulative sum of fractional Gaussian noise. Fractional Brownian motion (*fBm*) is self similar if the process starts at zero.

Usage

```
fgn(H, n, sigma2=1, seed=5)
```

Arguments

- `H` — Hurst parameter, a number between zero and one. When H is one half, then the *fGn* is simply white noise. If H is greater than one half, the process has long range (positive) dependence. If H is less than one half, then the autocorrelation is relatively short term and negative.
- `n` — positive integer being the required length of the simulated series.
- `sigma2` — positive number being the required variance of the simulated series. Default is one.
- `seed` — a seed for the random number generator. It should be an integer between 0 and 1000. Default is 5.

Details

The simulation method and other properties of *fGn* and *fBm* are given by Davies & Harte (1987). The simulation method is exact, and does not use autoregressive type approximations. It can be used for any Gaussian series where the covariance function can be explicitly calculated.

Value

A vector of length n containing a simulated sequence of fractional Gaussian noise with parameter H and variance σ^2.

References

Cited references are listed on the Fractal manual page.

Examples

```
#   Simulations of fGn and fBm for H = 0.25, 0.50, 0.75
savpar <- par(mfcol=c(3,2))
x1 <- fgn(0.25, 1000)
x2 <- rnorm(1000)
x3 <- fgn(0.75, 1000)

plot.ts(x1, ylab="")
title(main="fGn with H = 0.25")
plot.ts(x2, ylab="")
title(main="fGn with H = 0.50")
plot.ts(x3, ylab="")
title(main="fGn with H = 0.75")

plot.ts(c(0, cumsum(x1)), ylab="")
title(main="fBm with H = 0.25")
plot.ts(c(0, cumsum(x2)), ylab="")
title(main="fBm with H = 0.50")
plot.ts(c(0, cumsum(x3)), ylab="")
title(main="fBm with H = 0.75")

par(savpar)
```

hill *Hill Estimation of Renyi Dimensions Dq*

Description

Calculates the Hill estimates of the point centred Renyi dimensions, D_q, for q an integer greater than or equal to 2. This is done for a specified multidimensional series. The correlation dimension, denoted by D_2, is the second order Renyi point centred dimension.

Usage

```
hill(bootstraps, mmin, mmax, increment, numpairs, x, epsilon,
     rounderr, scaled=FALSE, lnorm=Inf, qth=2)
```

Arguments

`bootstraps`	required number of bootstrap samples.
`mmin`	minimum value of m, the number of order statistics for which the Hill estimate of D_q is to be evaluated.
`mmax`	maximum value of m, the number of order statistics for which the Hill estimate of D_q is to be evaluated.
`increment`	the incremental value for m. That is, the Hill estimate is returned for the m smallest order statistics, where m is in `seq(mmin,mmax,increment)`.
`numpairs`	number of interpoint distances evaluated at each bootstrap.
`x`	data matrix. Each point in the phase space is represented by a row. Dimension estimates are made for each dimension in `seq(1,ncol(x),1)`.
`epsilon`	number between zero and one half. Fraction to be trimmed from the edge of the rectangle for edge effect correction.
`rounderr`	rounding error in the data. Should be a vector of length `ncol(x)`, each element representing the rounding error of the respective column of the matrix `x`.
`scaled`	boolean. `TRUE` if the data are scaled into the unit cube, `FALSE` (default) otherwise.
`lnorm`	an integer greater than 0. Default is `Inf`.
`qth`	an integer greater than or equal to 2. Denotes the order of the Renyi dimension to be estimated. The default value is two, thus the function estimates the correlation dimension D_2.

Details

Produces Hill estimates of D_q for dimensions up to `ncol(x)`, in the order of the columns in the data matrix `x`. If higher dimension estimates are to be made for a one dimensional time series, one needs to embed into a higher dimensional phase space before using this function.

At each iteration, events are selected and `numpairs` of realisations of Y, where

$$Y = \max\{\text{dist}(X_1 - X_2), \text{dist}(X_1 - X_3), \ldots, \text{dist}(X_1 - X_q)\}$$

are calculated. The distance is the L^p norm, where p is given by the function argument lnorm. The sample of events is then used to calculate the dimension in the 1st dimension (1st column of matrix), 2nd dimension (1st 2 columns of matrix), etc, up to the `ncol(x)` th column of the matrix. For each, the Hill estimate is evaluated for the m smallest interpoint distances, where m is in `seq(mmin,mmax,increment)`. This procedure is repeated for the number of times given by the parameter bootstraps. Further details can be found in Harte (1998, 2001).

Value

A list object is return with the original values as defined above of: `bootstraps`, `mmin`, `mmax`, `numpairs`, `epsilon`, `increment`, `qth`. Other variables included are:

`estimates`	a matrix containing the dimension estimates. `ncol(estimates) = ncol(x)` and `nrow(estimates) = (mmax -mmin + 1)/increment`.
`n`	an integer equal to `nrow(x)`. Number of points represented by data matrix `x`.
`nrestrict`	an integer. Number of points in the data matrix `x` that are within the trimmed rectangle, where a distance epsilon has been removed from each boundary.
`meanzeros`	vector of length `ncol(x)`. The average number of zero pair distances out of the `numpairs` for each dimension.
`meandelta`	a matrix of the same size as estimates. It is the average interpoint distance for a given value of m (row) and given dimension (column).
`stddev`	a matrix of the same size as estimates. It is the standard deviation of the estimates for a given value of m (row) and given dimension (column).
`Hest.mmin`	a matrix where `nrow(Hest.mmin) = bootstraps` and `ncol(Hest.mmin) = ncol(x)`. It contains the all individual dimension estimates for m equal to `mmin`.

References

Cited references are listed on the Fractal manual page.

Examples

```
#   Generate a sample of 10000 points that have a probability
#   distribution given by the Cantor measure
set.seed(4)
temp <- matrix(cantor(10000, 35, p=1/3), ncol=1)


#   Dq is the lattice Renyi dimension for Cantor measure
Dq <- function(p,q)
-log(p^q + (1-p)^q)/log(3)/(q-1)

savpar <- par(mfrow=c(2,2))
for (i in 2:5){
    #  Estimate D2, D3, D4, and D5
    set.seed(4)
    tempest <- hill(100, 4, 5000, 1, 5000, temp, 0,
                    rep(1/(10^16), ncol(temp)), qth=i)
    #  Plot the dimension estimates.
```

```
    dimension.plot(tempest, ylim=c(0.3,0.6),
                   xscale='logdelta', logbase=3)
    abline(h=Dq(1/3,i), lty=3)
    title(main=substitute(paste(D[ii], " for Cantor Measure with ",
          p[0]==1/3), list(ii=i)))
}
par(savpar)

#--------------------------------------------
#   Simulate uniform random numbers in a cube
set.seed(4)
temp <- matrix(runif(30000),ncol=3)

#   Estimate the correlation dimension
set.seed(4)
tempest1 <- hill(100,4,5000,1,5000,temp,0,rep(1/(10^16), ncol(temp)),
                 lnorm=2)
tempest2 <- hill(100,4,5000,1,5000,temp,0,rep(1/(10^16), ncol(temp)),
                 lnorm=Inf)

#   Plot the dimension estimates, comparing max and Euclidean norms
savpar <- par(mfrow=c(2,1))
matplot(tempest1$meandelta, tempest1$estimates, xlab= "mean delta",
        ylab = "Hill Estimate", type = "l", ylim = c(0,3))
matlines(tempest2$meandelta, tempest2$estimates, lty=3)
matplot(log(tempest1$meandelta), tempest1$estimates, xlab= "log(mean delta)",
        ylab = "Hill Estimate", type = "l", ylim = c(0,3))
matlines(log(tempest2$meandelta), tempest2$estimates, lty=3)
par(savpar)
```

`logistic` *Simulations from Logistic Map*

Description

Uses the logistic map to generate a sample of points.

Usage

```
logistic(n, beta = 3.569945672, N = 1000, x0 = 0.8)
```

Arguments

`n`	number of points to be simulated.
`beta`	logistic map parameter, see Details below.
`N`	number of initial points to exclude.
`x0`	starting point.

Details

The logistic map has recurrence relation

$$x(n+1) = \beta x(n)(1 - x(n))$$

where $0 \le x \le 4$ and $0 \le \beta \le 4$. The attractor is invariant under the mapping when $\beta = \beta_\infty \approx 3.569945672$. The attractor under goes a sequence of bifurcations as the parameter β increases to β_∞. For further details, see Falconer (1990), Harte (2001) or Weisstein (2003).

Value

A numeric vector of length n containing the simulated sequence.

Author(s)

David Harte, 2000

References

Cited references are listed on the Fractal manual page.

Examples

```
#   Plot iterations of the logistic map
x <- seq(0, 1, 0.01)
beta <- 3.569945672
plot(x, beta*x*(1-x), type="l", lty=3, ylab="x(n+1)",
     xlim=c(0,1), ylim=c(0,1), xlab="x(n)")
title(main=expression(paste("Logistic Map:  ", beta, " = 3.569945672")))
abline(0, 1, lty=3)

x <- logistic(500)
for (k in 1:499){
points(c(x[k], x[k]), c(x[k], x[k+1]),type="l",lty=1)
points(c(x[k], x[k+1]), c(x[k+1], x[k+1]),type="l",lty=1)
}

#-------------------------------------------------
#    Display scaling properties of Logistic Map

x <- logistic(200000)

par(mar=c(2.1, 4.1, 1.1, 2.1), oma=c(0,0,3,0), mfrow=c(4,1))
y <- c(0.6, 0.4, 0.35, 0.344)
for (k in 1:4){
   hist(x[x<y[k]], xlab="", main="", nclass=100)
   box()
}
mtext(expression(paste("Scaling Characteristics of Logistic Map with ",
      beta %~~% beta[infinity])), line=0, cex=par()$cex.main, outer=TRUE)
```

`lorenz` *Simulations from Lorenz Attractor*

Description

Simulates a trajectory path from the Lorenz attractor, see Lorenz (1963).

Usage

```
lorenz(N, n, h, sigma=10, r=28, b=8/3)
```

Arguments

`N`	required length of simulated trajectory.
`n`	initial number of points to discard.
`h`	increment size for numerical differentiation.
`sigma`	equation parameter. Default value is 10.
`r`	equation parameter. Default value is 28.
`b`	equation parameter. Default value is 8/3.

Details

The Lorenz equations are:

$$\frac{dx}{dt} = \sigma(y - x)$$

$$\frac{dy}{dt} = rx - y - xz$$

$$\frac{dz}{dt} = xy - bz$$

A numerical approximation is made by approximating the derivative with small increments h.

Value

An N by 3 matrix. Each column represents the x, y, and z coordinate respectively.

References

Cited references are listed on the Fractal manual page.

Examples

```
temp <- lorenz(10000,100,0.01)
plot(temp[,"x"], temp[,"z"], type="l", xlab="x", ylab="z")
```

`moran.sim` *Simulate a Moran Cascade Process*

Description

Simulates point locations sampled from a Moran cascade process.

Usage

```
moran.sim(n, p, ti, x0, y0, k = 35, seed = 5)
```

Arguments

`n`	number of points to be simulated.
`p`	vector of construction probabilities of length b.
`ti`	vector of similarity ratios of length b.
`x0`	vector of x locations of length b, giving the x mid-points of the scaled copies.
`y0`	vector of y locations of length b, giving the y mid-points of the scaled copies.
`k`	resolution, default is 35.
`seed`	seed for the random number generator, default is 5.

Details

The Moran fractal set consists of b similarities of the generating set. The method used to simulate points sampled from a Moran cascade probability distribution involves simulating a base b number between zero and one of k digits (resolution). Each digit gives increased accuracy of the location of the point. Further details can be found in Harte (2001, Section 10.3.1).

Value

A list object with the following components is returned:

`x`	vector containing x locations of simulated points.
`y`	vector containing y locations of simulated points.

Author(s)

David Harte, 2004

References

Cited references are listed on the Fractal manual page.

See Also

`moran.theta`, `cantor`

Examples

```
#   Simulate the same processes as in Harte (2001, Figure 10.4)

par(mfrow=c(1,2), pty="s")

w <- moran.sim(1000, p=rep(1/4, 4), ti=c(1/2, 1/3, 1/2, 1/3),
               x0= c(0.5, 0, -0.5, 0), y0= c(0, -2/3, 0, 2/3),
               k = 35, seed = 5)
plot(w$x, w$y, pch=16, cex=0.2, xlab="", ylab="",
     xlim=c(-1,1), ylim=c(-1,1))

w <- moran.sim(1000, p=c(0.4, 0.1, 0.4, 0.1),
               ti=c(1/2, 1/3, 1/2, 1/3),
               x0= c(0.5, 0, -0.5, 0), y0= c(0, -2/3, 0, 2/3),
               k = 35, seed = 5)
plot(w$x, w$y, pch=16, cex=0.2, xlab="", ylab="",
     xlim=c(-1,1), ylim=c(-1,1))
```

`moran.theta` *Moran Cascade Correlation Exponents*

Description

Calculates the correlation exponents, denoted by $\theta(q)$, of a Moran cascade process. These are often denoted as $\tau(q)$ in the physics literature.

Usage

```
moran.theta(p, ti, q = seq(-8, 8, 1), tol = 1e-12, iterlim = 200)
```

Arguments

`p`	vector of construction probabilities of length b.
`ti`	vector of similarity ratios of length b.
`q`	vector of q values to be calculated.
`tol`	convergence criteria, being the difference between successive iterates.
`iterlim`	maximum number of iterations.

Details

The Moran fractal set consists of b similarities of the generating set. The correlation exponents, $\theta(q)$, are the power law exponents of the generalised correlation integrals of order q; see Harte (2001, Theorem 6.2.7). They are often denoted as $\tau(q)$; see for example, Falconer (1990, Proposition 17.2).

Value

A list object with the following components is returned:

theta	values of $\theta(q)$.
q	vector of q values calculated.

Author(s)

David Harte, 2004

References

Cited references are listed on the Fractal manual page.

See Also

moran.sim

Examples

```
#  Consider the Moran fractal set with "division rule" shown in
#  Harte (2001, Figure 6.1). It has Hausdorff dimension:

D0 <- -moran.theta(p=rep(0.25, 4), ti=c(1/3, 1/2, 1/3, 1/2), q=0)$theta
print(D0)

#   Note the values of p above do not matter for D0
#   Now allocate the measure "uniformly" over the set
#   Renyi dimensions (Dq) should be equal
#   Calculate relative "size" of the circles and probs

alpha <- (2/3)^D0
p0 <- 1/(2*alpha + 2)
p.uniform <- c(alpha*p0, p0, alpha*p0, p0)

x <- moran.theta(p=p.uniform, ti=c(1/3, 1/2, 1/3, 1/2))
Dq <- x$theta/(x$q-1)
print(Dq)

#----------------------------------------------------

#   Divide a square into 2 by 2 squares, ad infinitum
#   The similarity ratios are all 0.5
#   Reallocate the probabilities between the squares as
#   p0=0.1, p1=0.4, p2=0.1, p3=0.4
#   Hausdorff dimension of set is 2, measure is multifractal

x <- moran.theta(p=c(0.1, 0.4, 0.1, 0.4), ti=rep(1/2, 4),
                 q=seq(-8, 8, 0.5))

plot(x$q[x$q!=1], x$theta[x$q!=1]/(x$q[x$q!=1]-1),
```

```
    ylab=expression(D[q]), xlab="q",
    main="Renyi Dimensions", type="l")
abline(h=2, lty=3)
```

`multinomial.theta` *Correlation Exponents of the Multinomial Measure*

Description

Calculates the correlation exponents of the multinomial measure of order b. The unit interval is divided into b subintervals of equal length at each division. The Cantor measure is a special case of the multinomial measure. For further details, see Harte (2001, Chapter 3) or Falconer (1990, Chapter 17).

Usage

```
multinomial.theta(p, q = seq(-8, 8, 1))
```

Arguments

`p`	vector of construction probabilities of length b.
`q`	vector of q values to be calculated.

Value

A list object with the following components is returned:

`theta`	values of $\theta(q)$.
`q`	vector of q values calculated.

Author(s)

David Harte, 2004

References

Cited references are listed on the Fractal manual page.

See Also

`moran.theta`, `cantor`

Examples

```
#   Calculate the correlation exponents as in Harte (2001, Fig 3.1)

x <- multinomial.theta(p=rep(0.1, 10), q=seq(-4, 6, 0.25))
plot(x$q, x$theta, ylim=c(-6, 3), xlim=c(-3,6), lty=1, type="l",
     xlab="q", ylab=expression(tilde(theta)(q)),
     main=expression(paste(tilde(theta)(q), " for Multinomial Measure with ", b==10)))

x <- multinomial.theta(p=c(rep(0.111, 9), 0.001), q=seq(-4, 6, 0.25))
points(x$q, x$theta, lty=2, type="l")

x <- multinomial.theta(p=c(rep(0.07, 9), 0.37), q=seq(-4, 6, 0.25))
points(x$q, x$theta, lty=3, type="l")

#--------------------------------------------------------

#   Correlation exponents for Cantor Measure with p0=2/3 and p2=1/3

x <- multinomial.theta(p=c(2/3, 0, 1/3), q=seq(-4, 6, 0.25))
plot(x$q, x$theta, lty=1, type="l",
     xlab="q", ylab=expression(tilde(theta)(q)),
     main=expression(paste(tilde(theta)(q), " for Cantor Measure")))
abline(v=0, lty=3)
abline(h=x$theta[x$q==0], lty=3)
```

---

phasecon *Reconstruct Phase Space*

Description

Reconstruct the phase space from a univariate time series, see Takens (1981).

Usage

```
phasecon(maxphase, delay, x)
```

Arguments

`maxphase`	scalar. Maximum dimension of reconstructed phase space.
`delay`	time difference delay.
`x`	numeric vector representing univariate time series.

Value

a vector with `length(x)-maxphase*delay+1` rows and `maxphase` columns. The first column is the original time series, 2nd column being that embedded into 2D, up to the last column being that embedded into maxphase dimensions.

Author(s)

Wang Qiang, 1994

References

Cited references are listed on the Fractal manual page.

Examples

```
series <- rnorm(100)
y <- phasecon(5, 3, series)
print(y)
```

Index