Package ‘SADmoment’

July 22, 2019

Type Package

Title Extrapolate SAD and calculate parameters of best fitting distribution for subareas with several others

Version 1.2

Date 2019-05-14

Author Saeid Alirezazadeh

Maintainer Saeid Alirezazadeh <saeid.alirezazadeh@gmail.com>

Description Extrapolation of SAD and finding parameters of the best fitting distribution for subareas from the study of moments across scales. Moments of fitting distributions across scales by only using their parameters across scales.

Depends copula, BB, stats4, VGAM, sads, bbmle, poilog, splines, BBmisc

License GPL (>= 2)

LazyData true

R topics documented:

`SADmoment-package` *Extrapolate SAD and calculate parameters of best fitting distribution for subareas with several others*

Description

Extrapolate SAD and parameters of best fitting distribution for subareas.

Details

It is implemented by the techniques developed in Alirezazadeh S. (2019) and Alirezazadeh et. al. (2018). The most important functions of the package LLfun and SAD_Moment. The function SAD_Moment(X,Y,Splist,ext.rate,dplot) asks for positions (X,Y) of individuals of species Splist in the area of existing sample, an extrapolation rate that is the proportion of area we are interested to extrapolate to by the sample area, and dplot that identify whether the user want to visualise the SAD of the extrapolated area and compare it with the sample data or not. If the dplot=T then after running the function provides the histogram for species abundances of the sample and a line which is the estimation of species abundances of the extrapolated area. Note that the function SAD_Moment(X,Y,Splist,ext.rate,dplot) returns valuable information regarding the running time, information on the sample and also information regarding the extrapolated area. So, we suggest to run it as: name<-SAD_Moment(X,Y,Splist,ext.rate,dplot). Also, note that the length of X, Y and Splist must be the same. The function LLfun(X,splist), where X is a data.frame consists of species names and their (X,Y) positions. And splist is the name of existing species in the sample. And by pressing return it requests the user to enter a value for sub-area size and then with respect to the Log-Likelihood, finds the best fitting distribution of data and returns its parameters, see Alirezazadeh et.al. (2018)

Author(s)

Saeid Alirezazadeh

References

Alirezazadeh, S., 2019, Improvement on Extrapolation of Species Abundance Distribution Across Scales from Moments Across Scales (Unpublished)

Saeid Alirezazadeh and Luís Borda-de-Água and Paulo Borges and Rosalina Gabriel and F. Miguel Dionisio and Henrique Miguel Pereira and Pedro Cardoso, 2018, Theoretical Approach for how Species Abundance Distributions Change Across Scales, 2018 13th APCA International Conference on Control and Soft Computing (CONTROLO), pp. 131-136.

Examples

```
# Data from Barro Colorado Island (BCI) is used here for the example,
# within total 50 ha area all the individuals
# with at least 1cm (Diameter at Breast Height) (dbh) are considered.

SAD_Moment(dbci[,3],dbci[,4],dbci[,1],ext.rate=1.1,dplot=TRUE)

# In the following we used all the individuals
# with at least 10cm (Diameter at Breast Height) (dbh) from the BCI data.
#R> name<-SAD_Moment(X,Y,Spnames,ext.rate=1.2,dplot=FALSE)
#R> summary(name)
#           Length Class     Mode
#Data       6      -none-    list
#Initial    4      -none-    list
#Extrapolate 7     -none-    list
#Time       5      proc_time numeric
#R> summary(name$Initial)
#              Length Class  Mode
#Area           1     -none- numeric
#Moments        20    -none- numeric
#XY.Ratio       1     -none- numeric
#Number.of.bins  1    -none- numeric
#R> summary(name$Extrapolate)
#                                    Length Class  Mode
#Area                                 1     -none- numeric
#Moments                             20     -none- numeric
#Number.of.bins                       1     -none- numeric
#Coefficients.of.Tchebychev.Polynomials 49  -none- numeric
#Tchebychev.Polynomials.Values       98     -none- numeric
#Tchebychev.Moments.Values            7     -none- numeric
#Extrapolate.SAD                     12     -none- numeric
#R> summary(name$Time)
#   user  system elapsed
#  4.632   2.009   6.690
#R>print(name$Data$SAD)
```

```
# [1] 27 11 20 30 29 38 30 21 20  5  4  1
#R> print(name$Extrapolate$Extrapolate.SAD)
# [1] 32 13 15 24 31 33 30 24 17 10  5  2
#
#R> LLfun(X,Splist)
#Enter sub-area size: 9800
#     [,1]
#[1,] "For the sub-area ( 9800 ) the best fitting distribution is : Poisson Gamma"
#[2,] "alpha= 1.2089269105249 and beta= 1.2089269105249"
#R> LLfun(X,Y,Splist)
#Enter sub-area size: 200000
#     [,1]
#[1,] "For the sub-area ( 2e+05 ) the best fitting distribution is : Log-Normal"
#[2,] "mu= 3.51478121939837 and sigma= 2.2582385148143"
#
```

`alffinder` *Parameter alpha in the fitting with a Gamma distribution*

Description

Finds parameter alpha in the case of fitting with a Gamma distribution

Usage

```
alffinder(res, slope, ro, c, z, A)
```

Arguments

`res`	intercept of first moments acorss area
`slope`	slope of first moments acorss area
`ro`	slope of number of individuals across area
`c`	parameter c in the power-law species area relationship
`z`	parameter z in the power-law species area relationship
`A`	total area of consideration

`BellB` *Bell function*

Description

Returns the values of Bell function

Usage

```
BellB(n, k, x)
```

Arguments

`n`	total elements of the sets
`k`	number of partitions
`x`	specific element(s) of the set

bet.f *bet.f*

Description

returns beta(n,N) as a part of finding values of Tchebychev polynomials.

Usage

```
bet.f(n, N)
```

Arguments

n	order of polynomial
N	number of bins

betfinder *Parameter beta in the fitting with a Gamma distribution*

Description

Finds parameter beta in the case of fitting with a Gamma distribution

Usage

```
betfinder(res, slope, ro, c, z, A)
```

Arguments

res	slope of first moments acorss area
slope	slope of first moments acorss area
ro	slope of number of individuals across area
c	parameter c in the power-law species area relationship
z	parameter z in the power-law species area relationship
A	total area of consideration

calcalpha *Logseries parameter*

Description

Finds the parameter alpha in the case of fitting with a log-series distribution

Usage

```
calcalpha(n.orig, s.orig)
```

Arguments

n.orig	number of individuals
s.orig	number of species

cmoment *moment calculator, first step in the Log-Normal, values need to be transformed*

Description

First step for calculating moments of log-normal, will be used in another function for transformation.

Usage

```
cmoment(mu, sig, k)
```

Arguments

mu	the parameter mu in the log-normal
sig	the parameter sigma in the log-normal
k	maximum order of moments

cmul *symbolic constant multiplication*

Description

symbolic multiplication with a constant

Usage

```
cmul(u, c)
```

Arguments

u	matrix with two rows and second raw consists of 1 to number of columns (represent powers) and first raw is the coefficients
c	a constant

`Coef.Tcheb.pol.2nd` *Coefficients of the Tchebychev polynomials up to a certain degree*

Description

Coefficients of the Tchebychev polynomials up to a certain degree as a matrix

Usage

```
Coef.Tcheb.pol.2nd(N, no.m)
```

Arguments

`N`	Number of bins
`no.m`	maximum order of polynomials

`data_read` *get values from keyboard for future use.*

Description

Take a value from keyboard for future use. It is a part of LLfun which is to find the best fitting distribution for a smaller sub-areas.

Usage

```
data_read()
```

`datfun` *average number of species and individuals*

Description

Returns average number of species and individuals in a given sub-area, by randomly sampling that sub-area size from the total area.

Usage

```
datfun(df, x, spp, n, ratio)
```

Arguments

`df`	data frame, name of species, position over X and Y axis
`x`	sub-section size over Y axis
`spp`	list of all species
`n`	number of random sampling
`ratio`	ratio of X by Y

`dbfact` *double factorial*

Description

Finds the value of double factorial value.

Usage

```
dbfact(i)
```

Arguments

`i`	an integer

`Dist_Moment` *Moments of the most common fitting distributions*

Description

Returns the moments up to degree 20 of the fitting distribution of data across scales.

Usage

```
Dist_Moment(xpos, ypos, splist, DIST = "LS")
```

Arguments

`xpos`	position of individuals of species by X-axis
`ypos`	position of individuals of species by Y-axis
`splist`	Respective name of the species in the position (X,Y)
`DIST`	The fitting distribution under consideration ("LS","LN","PG","PLN")

`expnd` *symbolic expansion*

Description

expand symbolic polynomials to higher degree

Usage

```
expnd(u, n)
```

Arguments

`u`	matrix with two rows and second raw consists of 1 to number of columns (represent powers) and first raw is the coefficients
`n`	numeric value that we need to expand the polynomial.

`extpfun` *extrapolation of moments across scales*

Description

extrapolate moments.

Usage

```
extpfun(dat, m)
```

Arguments

`dat`	list of values (the list must have almost linear pattern).
`m`	the corresponding numeric index in the list that we try to extrapolate to.

`extrapolation.dist` *extrapolate distribution moments across scales*

Description

Returns the values of extrapolations of moments of different orders of the distribution moments

Usage

```
extrapolation.dist(momraw, momdist, ext.rate, momr)
```

Arguments

`momraw`	raw moments values
`momdist`	distribution moments values
`ext.rate`	extrapolation rate.
`momr`	values of extrapolation of raw moments to that rate.

`extrapolation.raw` *extrapolate raw moments across scales*

Description

Returns the values of extrapolations of moments of different orders of the raw moments

Usage

```
extrapolation.raw(momraw, ratio, ba, bba)
```

Arguments

`momraw`	values of raw moments
`ratio`	the ration of x by y.
`ba`	total area we try to extrapolate to.
`bba`	total area that we have informations.

`f` *individuals in a given positions*

Description

Lista of individuals exist in a given sub-area.

Usage

```
f(df, xcoor, ycoor, x, ratio)
```

Arguments

`df`	data frame, name of species, position over X and Y axis
`xcoor`	initial X-coordinate inside total area
`ycoor`	initial Y-coordinate inside total area
`x`	sub-section size over Y axis
`ratio`	ratio of X by Y

`fg` *list of number individuals of all species*

Description

Returns the abundances of species of interest from a given list of individuals.

Usage

```
fg(g, sppnames)
```

Arguments

`g`	list of all individuals in a sub-area
`sppnames`	list of all species

`fo` *fo*

Description

used as part of method to evaluate the parameter of log-series across scales

Usage

```
fo(a, n, s)
```

Arguments

`a`	area
`n`	number of individual
`s`	number of species

`fprime` *first derivative of funtion fo, as a part of the function calculate the parameter of log-series*

Description

first derivative of the fo function used for the Newton's method to find the parameter of log-series

Usage

```
fprime(a, n, s)
```

Arguments

`a`	area
`n`	numbero of individuals
`s`	number of species

`LLfun` *Parameters of the best fitting distribution*

Description

Returns the parameters of the best fitting distribution with respect to the log-likelihoods.

Usage

```
LLfun(Y, spp)
```

Arguments

`Y`	A matrix consists of x,y and species names respect all individuals in the community
`spp`	list of all species in the community

`LNmom` *Moments of log-normal*

Description

Returns moments up to degree k of the log-normal distribution.

Usage

```
LNmom(mu, sig, k)
```

Arguments

`mu`	mu value in the log-normal distribution
`sig`	sigma value in the log-normal distribution
`k`	the order of moment.

`LSmom` *Moments of the log-series distribution.*

Description

Returns moments up to degree k of the log-normal distribution.

Usage

```
LSmom(x, n)
```

Arguments

`x`	the parameter of log-series
`n`	order of moment

`M1` *M1, theoretical average*

Description

Returns average number of individuals (Theoretical)

Usage

```
M1(ro, c, z, A)
```

Arguments

`ro`	parameter ro in N=ro A
`c`	parameter c in S=c A^z
`z`	parameter z in S=c A^z
`A`	area or scale

`manage_data` *data across scales, number of individuals of species across scales*

Description

returns number of individuals of all species and number of species across scales.

Usage

```
manage_data(df, spp, n, x, ratio)
```

Arguments

`df`	data frame, name of species, position over X and Y axis
`spp`	list of all species
`n`	number of random sampling over Y axis
`x`	sub-section size over Y axis
`ratio`	ratio of X by Y

mmoment *first step for finding moment of Possion-Gamma, the values needed to be transformed*

Description

the values from this function is used as the first step to calculate the moments of Poisson-Gamma distribution and need to be transformed

Usage

```
mmoment(al, bet, k)
```

Arguments

`al`	parameter alpha in the gamma distribution
`bet`	parameter beta in the gamma distribution
`k`	maximum order of moments

mo2nd *2nd moment across scales*

Description

consider the linear pattern for the 2nd moment across scales.

Usage

```
mo2nd(res, slope, A)
```

Arguments

`res`	residual of second moment across scales
`slope`	slope of second moment across scales
`A`	area or scale

`moment1` *moment of log-series number of individuals are not log-transformed*

Description

return moments of different orders of log-series but in the case where the number of individuals are not log-transformed

Usage

```
moment1(x, n)
```

Arguments

`x`	parameter of log-series
`n`	maximum moment order

`momfun` *raw, distribution, and linear moments*

Description

all moments are calculated here. linear moments is when we do not log transform the number of individuals.

Usage

```
momfun(df, x, spp, n, no.m, ratio)
```

Arguments

`df`	data frame consists of name of species together with positions over X and Y axis
`x`	subsection size over Y axis
`spp`	list of all species
`n`	number of randomize sampling in the total area
`no.m`	maximum order of moments
`ratio`	ratio of X by Y

`mufinderLN` *mean finder for the Log-Normal*

Description

find the parameter mu in a log-normal

Usage

```
mufinderLN(res, slope, A)
```

Arguments

`res`	residual of the first raw moment across scales
`slope`	slope of the first raw moment across scales
`A`	area or scale

`mufinderPLN` *mean finder for the Poisson log-normal*

Description

returns the parameter of mu in the Poisson log-normal.

Usage

```
mufinderPLN(res, slope, A)
```

Arguments

`res`	residual of the first moment across scales
`slope`	slope of the first moment across scales
`A`	area or scale

`numsum` *weighted sum*

Description

Calculate the weighted sum for given weight and a list

Usage

```
numsum(n, x, k)
```

Arguments

`n`	maximum degree
`x`	a numeric parameter
`k`	a numeric value for the weight

PGmom *Moments of the Poisson Gamma*

Description

Returns moments up to degree k of the Poisson Gamma distribution.

Usage

```
PGmom(al, bet, k)
```

Arguments

`al`	parameter alpha of poisson gamma distribution
`bet`	parameter beta of poisson gamma distribution
`k`	order of moment

PLNmom *Moments of the Poisson log-normal distribution.*

Description

Returns moments up to degree k of the Poisson log-normal distribution.

Usage

```
PLNmom(mu, sig, k)
```

Arguments

`mu`	parameter mu of the distribution
`sig`	parameter sigma of the distribution
`k`	order of moment

provide_info *provide some info*

Description

inside function to print some information from LLfun function.

Usage

```
provide_info(L)
```

Arguments

`L`	A list of information provided by the LLfun function.

`ptransform` *polylog transformation*

Description

Polylog transformation used to calculate moments of log-series where number individuals are log transformed

Usage

```
ptransform(x, k)
```

Arguments

`x`	parameter for log series
`k`	integer as degree of the polylog function to expand to.

`r.mom` *raw moments*

Description

calculate raw moments

Usage

```
r.mom(dat, no.m)
```

Arguments

`dat`	data list of number of individuals of species
`no.m`	maximum order of moments

`relpwave` *wave function*

Description

fit wave function and use it for extrapolation

Usage

```
relpwave(l, t, ba)
```

Arguments

`l`	list of values that is considered to be wave
`t`	the larger scale we need to extrapolate to
`ba`	area or scale

ro.c.z *parameter finder*

Description

returns theoretical parameter ro, c, and z in N=ro A and S=cA^z

Usage

```
ro.c.z(No.Sp, No.Id, bba)
```

Arguments

No.Sp	number of species across scales
No.Id	number of individuals across scales
bba	area or scales

SAD_Moment *Extrapolate Species Abundance Distribution*

Description

Extrapolates SAD by extrapolating of moments.

Usage

```
SAD_Moment(xpos = NULL, ypos = NULL, splist = NULL, ext.rate = 1, dplot = T)
```

Arguments

xpos	List of position of individuals over X-axis
ypos	List of position of individuals over Y-axis
splist	Name of species where its individual is in place (X,Y)
ext.rate	extrapolation rate(>=1)
dplot	Values (T or F) respectively to inform whether the plot of the extrapolation should appears or not.

shrt *Symbolic shortenning*

Description

Shorten a polynomial with symbolic calculation

Usage

```
shrt(u)
```

Arguments

u	matrix with two rows and second raw consists of 1 to number of columns (represent powers) and first raw is the coefficients

sigfinderLN *standard deviation finder for Log Normal across scales*

Description

returns the parameter sigma in the Log-normal Distribion.

Usage

```
sigfinderLN(res, slope, ro, c, z, A)
```

Arguments

res	residual for the change of the first raw moments across scales
slope	slope of the change of the first raw moments across scales
ro	the parameter ro in the formula N=ro A, number of individuals across scales
c	the parameter c in S=cA^z
z	the parameter z in S=cA^z
A	area or scale

sigfinderPLN *standard deviation finder for Poisson Log Normal across scales*

Description

returns the parameter sigma in a Poisson log-normal.

Usage

```
sigfinderPLN(res1, slope1, res2, slope2, A)
```

Arguments

`res1`	residual for the change of means across scales
`slope1`	slope of the change of means across scales
`res2`	residual for the change of moments of second order across scales
`slope2`	slope of the change of moments of second order across scales
`A`	area or scale

smom *Log-normal moments*

Description

moments of log-normal of different orders are calculate here

Usage

```
smom(mu, sig, k)
```

Arguments

`mu`	mean
`sig`	standard deviation
`k`	maximum order of moment

st2k *Sterling number of second kind*

Description

Returns the Sterling number of second kind

Usage

```
st2k(n, k)
```

Arguments

n	number of elements of the set
k	number of partitions of the set

summ *symbolic addition*

Description

Adding polynomials symbolically

Usage

```
summ(u1, u2)
```

Arguments

u1	matrix with two rows and second raw consists of 1 to number of columns (represent powers) and first raw is the coefficients
u2	matrix with two rows and second raw consists of 1 to number of columns (represent powers) and first raw is the coefficients

Tcheb.pol.2nd *Values of Tchebychev polynomials up to certain degree at each bin.*

Description

Returns values of the Tchebychev polynomials up to a certain degree and by considering a certain number of bins.

Usage

```
Tcheb.pol.2nd(N, no.m)
```

Arguments

N	number of bins
no.m	maximum order of polynomials

tn *Thecbychev polynomials recursive formula*

Description

calculate recursively the Tchebychev polynomials up to given degree and in different bins

Usage

```
tn(n, N)
```

Arguments

`n` maximum degree of polynomial

`N` maximum number of bins

xmul *symbolic multiplication*

Description

Returns the symbolic multiplications of a polynomial by its variable

Usage

```
xmul(u)
```

Arguments

`u` matrix with two rows and second raw consists of 1 to number of columns (represent powers) and first raw is the coefficients

Index