Package ‘GeoRange’

October 12, 2022

Type Package

Title Calculating Geographic Range from Occurrence Data

Version 0.1.0

Description Calculates and analyzes six measures of geographic range from a set of longitudinal and latitudinal occurrence data. Measures included are minimum convex hull area, minimum spanning tree distance, longitudinal range, latitudinal range, maximum pairwise great circle distance, and number of X by X degree cells occupied.

License GPL-3

LazyData TRUE

Imports sp,proj4,raster,moments,stats,graphics,grDevices,velociraptr

RoxygenNote 6.0.1

NeedsCompilation no

Author James Boyle Developer [aut, cre]

Maintainer James Boyle Developer <jamesboy@buffalo.edu>

Repository CRAN

Date/Publication 2017-06-15 15:07:19 UTC

R topics documented:

`BivalvePBDB` *Occurrence data on bivalves from the Ypresian Data from the Paleobiology Database via the velociraptr package on May 17th, 2017*

Description

Occurrence data on bivalves from the Ypresian Data from the Paleobiology Database via the velociraptr package on May 17th, 2017

Usage

```
data(BivalvePBDB)
```

Format

A csv file

Examples

```
data(BivalvePBDB)
```

CellCount *Calculates degree x degree cell counts of a specified size*

Description

Calculates degree x degree cell counts of a specified size

Usage

```
CellCount(longs, lats, CellSize = 5, longBounds = c(-180, 180),
  latBounds = c(-90, 90))
```

Arguments

longs	- Array of longitudinal occurrence values in decimal degrees
lats	- Array of latitudinal occurrence values in decimal degrees
CellSize	- Size of each cell in degree X degree
longBounds	- Array of longitudinal boundaries in decimal degrees
latBounds	- Array of latitudinal boundaries in decimal degrees

Value

Returns the number of cells occupied, specified cell size, and coordinate list

Note

This method uses grids cells constructed by equal degrees not area. So high latitude cells will have smaller areas than those at low latitude

Examples

```
longs<-c(22,55,-144)
lats<-c(-12,22,-12)
CellCount(longs,lats,CellSize=5,longBounds=c(-180,180),latBounds=c(-90,90))
```

CHullArea *Performs Convex Hull area calculation*

Description

Performs Convex Hull area calculation

Usage

```
CHullArea(longs, lats)
```

Arguments

`longs`	- Array of longitudinal occurrence values in decimal degrees
`lats`	- Array of latitudinal occurrence values in decimal degrees

Details

Uses the cylindrical equal area projection in order to check if the minimum convex hull wraps around the prime meridian

Value

Returns area of a set of coordinates

Note

Relies on the 'sp' package for the Polygon and chull function

Examples

```
longs<-c(-12,23,55)
lats<-c(34,22,30)
CHullArea(longs,lats)
```

CHullAreaEarth *Performs convex hull area calculation from coordinate sets on the Earth's surface*

Description

Performs convex hull area calculation from coordinate sets on the Earth's surface

Usage

```
CHullAreaEarth(longs, lats)
```

Arguments

`longs`	- Longitudinal coordinates of occurrences in decimal degrees
`lats`	- Latitudinal coordinates of occurrences in decimal degrees

Details

Uses the cylindrical equal area projection in order to check if the minimum convex hull wraps around the prime meridian

Value

Returns the convex hull area is square kilometers

Note

Relies on the ’sp’ package for the Polygon and chull function. Assumes latitude and longitude coordinates use the WGS84 datum

Examples

```
longs<-c(-133,-101,56)
lats<-c(33,12,-2)
CHullAreaEarth(longs,lats)
```

`CoordCollapse` *Removes duplicate geographic locations and binds coordinates into a single element*

Description

Removes duplicate geographic locations and binds coordinates into a single element

Usage

```
CoordCollapse(longs, lats)
```

Arguments

`longs`	- Longitudinal coordinates of occurrences in decimal degrees
`lats`	- Latitudinal coordinates of occurrences in decimal degrees

Value

Returns a 2-column array of coordinates without any duplicate locations

Note

Points are truncated to the hundredths place before checking for duplicates

Examples

```
longs<-c(34,133,-45)
lats<-c(-12,44,76)
CoordCollapse(longs,lats)
```

`CoordList_PBDB` *Creates an occurrence matrix of taxa by coordinates from the Paleobiology Database*

Description

Creates an occurrence matrix of taxa by coordinates from the Paleobiology Database

Usage

```
CoordList_PBDB(pbdb_data)
```

Arguments

`pbdb_data`	- Matrix of occurrence records from the Paleobiology Database, see downloadPBDB function in velociraptr package

Details

Cuts out records for which there is no paleogeographic information known

Value

Returns a taxa by coordinates matrix of occurrences

See Also

See the velociraptr package for more details on downloading PBDB data

Examples

```
data(BivalvePBDB)
CoordList_PBDB(BivalvePBDB)
```

`deg2rad` *Converts degrees to radians*

Description

Converts degrees to radians

Usage

```
deg2rad(deg)
```

Arguments

`deg` - decimal degree to be converted to radians

Value

Returns the degree in radians

References

[1]Originally from http://www.r-bloggers.com/great-circle-distance-calculations-in-r/

Examples

```
deg2rad(45)
```

`EqualAreaRectangle` *Create a rectangular shaped distribution with equal area to a given area*

Description

Create a rectangular shaped distribution with equal area to a given area

Usage

```
EqualAreaRectangle(center = c(0, 0), TargetArea, error = 0.001)
```

Arguments

`center` - Arrary containing the coordinates of the center of circular portion of the rectangle in decimal degree

`TargetArea` - Area in square kilometers desired for the rectangle

`error` - The tolerable proportion of error between the rectangular shape and the TargetArea

Value

Returns a 2-dimensional array of decimal degree coordinates outlining a rectangular shaped distribution

Note

This returns 100 evenly spaced points along each corner of the rectangle, in addition to the corners themselves

Examples

```
HorseShoeTest<-PtsAlgHorseShoe(z=2000,spacing=1,endAngles=c(-90,90))
HorseShoePts<-RandHorseShoe(center=c(0,0),npts=100,HorseShoeShape=HorseShoeTest)
EqualAreaRectangle(TargetArea=as.numeric(HorseShoePts$TotalArea_km2),error=0.001)
```

GCD *Calculates the maximum pairwise great circle distance from a set of decimal degree coordinates*

Description

Calculates the maximum pairwise great circle distance from a set of decimal degree coordinates

Usage

```
GCD(longs, lats)
```

Arguments

`longs`	- Longitudinal coordinates of occurrences in decimal degrees
`lats`	- Latitudinal coordinates of occurrences in decimal degrees

Details

Because this function does not account for the possibility that a taxa may wrap around more than half the Earth the maximum value is half the circumference of the Earth, approximately 20,038 kilometers.

Value

Returns the maximum great circle distance in kilometers

Note

The great circle distance can be extracted from the result of a minium spanning tree calcualation MSTDist() if available to avoid redundant calculations

Examples

```
longs<-c(34,156,-78)
lats<-c(45,12,9)
GCD(longs,lats)
```

---

gcd_hf *Calculates the geodesic distance between two points specified by latitude and longitude using the Haversine formula*

Description

Calculates the geodesic distance between two points specified by latitude and longitude using the Haversine formula

Usage

```
gcd_hf(long1, lat1, long2, lat2)
```

Arguments

`long1`	- Longitudinal value of first point in decimal degrees
`lat1`	- Latitudinal value of first point in decimal degrees
`long2`	- Longitudinal value of second point in decimal degrees
`lat2`	- Latitudinal value of second point in decimal degrees

Details

The Haversine formula can be inaccurate depending on coordinates

Value

Returns the distance between two points on the Earth in kilometers

Note

The haversine method is inaccuarate and should only be used when the vicenty formula fails or over very small distances

References

[1] Adapted from http://www.r-bloggers.com/great-circle-distance-calculations-in-r/

Examples

```
long1<-22
lat1<-44
long2<-52
lat2<-51
gcd_hf(long1,lat1,long2,lat2)
```

gcd_vif *Calculates the geodesic distance between two points specified by latitude/longitude using Vincenty inverse formula for ellipsoids*

Description

Calculates the geodesic distance between two points specified by latitude/longitude using Vincenty inverse formula for ellipsoids

Usage

```
gcd_vif(long1, lat1, long2, lat2)
```

Arguments

long1	- Longitudinal value of first point in decimal degrees
lat1	- Latitudinal value of first point in decimal degrees
long2	- Longitudinal value of second point in decimal degrees
lat2	- Latitudinal value of second point in decimal degrees

Value

Returns the distance between two points on the Earth in kilometers

References

[1] Adapted from http://www.r-bloggers.com/great-circle-distance-calculations-in-r/

Examples

```
long1<-22
lat1<-44
long2<-52
lat2<-51
gcd_vif(long1,lat1,long2,lat2)
```

`GeoCor` *Function to calculate the correlation coefficient for pairwise comparisons between geographic range measures*

Description

Function to calculate the correlation coefficient for pairwise comparisons between geographic range measures

Usage

```
GeoCor(GeoRange, Start = 1, method = "pearson")
```

Arguments

`GeoRange`	- A matrix of taxa by geographic range calculations, as from the GeoRange_MultiTaxa function
`Start`	- The column index value where geographic range measures to be compared starts
`method`	- The correlation method to be used. See the cor() function for available inputs

Value

Returns a sparse pairwsie matrix of correlation coefficients

Note

The correlation calculation uses the "pairwise.complete.obs" option from the cor function so that only complete pairs of observations are used, pairs containing an NA are ignored

See Also

See the velociraptr package for details of the downloadPBDB() function

Examples

```
## Not run:
data(BivalvePBDB)
BivalveMatrix<-CoordList_PBDB(BivalvePBDB)
testBivalve<-GeoRange_MultiTaxa(OccMatrix=BivalveMatrix,TaxaStart=3)
GeoCor(testBivalve,Start=1,method="kendall")

## End(Not run)
```

GeoPerformance_SkewCV *Function to calculate the skewness and coefficient of variance for a set of geographic range calculations*

Description

Function to calculate the skewness and coefficient of variance for a set of geographic range calculations

Usage

```
GeoPerformance_SkewCV(GeoRange)
```

Arguments

GeoRange	- Data matrix containing the geographic range calculations for a set of taxa, as from the GeoRange_MultiTaxa() function

Value

Returns a list of the skewness and coefficient of variance for each geographic range measure

Note

The coefficient of variance returned is standard deviation/mean

See Also

See the raster and moments packages for more details on the calculation of skewness and coefficient of variance

Examples

```
## Not run:
data(BivalvePBDB)
BivalveMatrix<-CoordList_PBDB(BivalvePBDB)
BivalveGeo<-GeoRange_MultiTaxa(OccMatrix=BivalveMatrix,TaxaStart=3)
GeoPerformance_SkewCV(BivalveGeo)

## End(Not run)
```

GeoRange_MultiTaxa *Function to tabulate number of occurrences/locations, six geographic range measures, minimum and maximum latitude and longitude for each taxon in a dataset*

Description

Function to tabulate number of occurrences/locations, six geographic range measures, minimum and maximum latitude and longitude for each taxon in a dataset

Usage

```
GeoRange_MultiTaxa(OccMatrix, TaxaStart, LongPos = 1, LatPos = 2,
  CellSize = 5, longBounds = c(-180, 180), latBounds = c(-90, 90))
```

Arguments

OccMatrix	- A matrix where columns are taxon occurrences and also having at least longitude and latitude values
TaxaStart	- The column index value where taxon records start
LongPos	- The column index value of longitudinal coordinates in the OccMatrix
LatPos	- The column index value of latitudinal coordinates in the OccMatrix
CellSize	- The size of each cell in degree X degree
longBounds	- Array of longitudinal boundaries in decimal degrees
latBounds	- Array of latitudinal boundaries in decimal degrees

Value

Returns a matrix of taxa by geographic range measures, including number of observations, number of unique locations observed at, minimum spanning tree distance, minimum convex hull area, maximum pairwise great circle distance, latitudinal range, longitudinal range, and number of degree X degree cells occupied

Note

Calculates the number of observations, localities, minimum spanning tree distance, convex hull area, longitudinal range, latitudinal range, and cell count

See Also

See the velociraptr package for details of the downloadPBDB() function

Examples

```
## Not run:
data(BivalvePBDB)
BivalveMatrix<-CoordList_PBDB(BivalvePBDB)
GeoRange_MultiTaxa(OccMatrix=BivalveMatrix,TaxaStart=3)

## End(Not run)
```

GeoRarefaction_MultiTaxa

Calculates six geographic range measures at resampled a number of time from specified sample sizes

---

Description

Calculates six geographic range measures at resampled a number of time from specified sample sizes

Usage

```
GeoRarefaction_MultiTaxa(nLocCut = 3, OccMatrix, TaxaStart, LongPos = 1,
  LatPos = 2, iter = 10, CellSize = 5, longBounds = c(-180, 180),
  latBounds = c(-90, 90), steps = c(1, 50, 40, 30, 20, 10, 5),
  replacePts = FALSE)
```

Arguments

`nLocCut`	- The minimum number of locations a taxon must be seen at to have geographic range measures calculated
`OccMatrix`	- A matrix where columns are taxon occurrences and also having at least longitude and latitude values
`TaxaStart`	- The column index value where taxon records start
`LongPos`	- The column index value of longitudinal coordinates in the OccMatrix
`LatPos`	- The column index value of latitudinal coordinates in the OccMatrix
`iter`	- The number of times a taxon's locations are resampled at each step size
`CellSize`	- The size of each cell in degree X degree
`longBounds`	- Array of longitudinal boundaries in decimal degrees
`latBounds`	- Array of latitudinal boundaries in decimal degrees
`steps`	- Array of the values representing the number of points to be subsampled for each taxon
`replacePts`	- A logical value indicating whether points are allowed to be sampled more than once during each subsampling iteration

Details

The nLocCut parameter is the minimum number of distinct geographic locations a taxon must be observed at to have geographic range measures calcualted, if below retruns NA. The steps parameter typically begins with a 1 representing that all points should be used in calculations but this is not required. The replacePts parameter must be set to TRUE if any of the steps require a greater number of points to be sampled than there are actual locations for a taxon, otherwise the function will fail.

Value

Returns a vector where each element is a taxon with a list of each geographic range measure as a separate matrix of values. Measures include minimum spanning tree distance, minimum convex hull area, maxmimum pairwise great circle distance, latitudinal range, longitudinal range, and number of degree X degree cells occupied.

Note

If PEE values are to be calculated as a next step the steps parameter needs to have a value of 1 as its first value

See Also

See the velociraptr package for details of the downloadPBDB() function

Examples

```
## Not run:
data(BivalvePBDB)
BivalveMatrix<-CoordList_PBDB(BivalvePBDB)
GeoRarefaction_MultiTaxa(nLocCut=20,OccMatrix=BivalveMatrix,TaxaStart=3,replacePts=TRUE)

## End(Not run)
```

GeoRarefaction_SingleTaxon

Calculates six geographic range measures at specified sample sizes for a single taxon

Description

Calculates six geographic range measures at specified sample sizes for a single taxon

Usage

```
GeoRarefaction_SingleTaxon(TName = "Bird1", OccMatrix, LongPos = 1,
  LatPos = 2, iter = 10, CellSize = 5, longBounds = c(-180, 180),
  latBounds = c(-90, 90), steps = c(1, 50, 40, 30, 20, 10, 5),
  replacePts = FALSE)
```

Arguments

`TName` - The name of the taxon of interest, as labeled in OccMatrix, as a string

`OccMatrix` - A matrix where columns are taxon occurrences and also having at least longitude and latitude values

`LongPos` - The column index value of longitudinal coordinates in the OccMatrix

`LatPos` - The column index value of latitudinal coordinates in the OccMatrix

`iter` - The number of times a taxon's locations are resampled at each step size

`CellSize` - The size of each cell in degree X degree

`longBounds` - Array of longitudinal boundaries in decimal degrees

`latBounds` - Array of latitudinal boundaries in decimal degrees

`steps` - Array of the values representing the number of points to be subsampled for each taxon

`replacePts` - A logical value indicating whether points are allowed to be sampled more than once during each subsampling iteration

Details

The nLocCut parameter is the minimum number of distinct geographic locations a taxon must be observed at to have geographic range measures calcualted, if below retruns NA. The steps parameter typically begins with a 1 representing that all points should be used in calculations but this is not required. The replacePts parameter must be set to TRUE if any of the steps require a greater number of points be locations be sampled than are available for a taxon, otherwise the function will fail.

Value

Returns a list of each geographic range measure as a separate matrix of values. Measures include minimum spanning tree distance, minimum convex hull area, maximum pairwise great circle distance, latitudinal range, longitudinal range, and number of degree X degree cells occupied.

Note

If PEE values are to be calculated as a next step the steps parameter needs to have a value of 1 as its first value

Examples

```
## Not run:
data(BivalvePBDB)
BivalveMatrix<-CoordList_PBDB(BivalvePBDB)
GeoRarefaction_SingleTaxon(TName=names(BivalveMatrix)[3],OccMatrix=BivalveMatrix,replacePts=TRUE)

## End(Not run)
```

`hello` *Prints Hello, world!*

Description

Prints Hello, world!

Usage

```
hello()
```

Value

Prints 'Hello, world!

`HorseShoeArea` *Function to calcuate the area of a given horseshoe shape*

Description

Function to calcuate the area of a given horseshoe shape

Usage

```
HorseShoeArea(HorseShoeShape)
```

Arguments

`HorseShoeShape`	- Object containing the outline of a horseshoe shape, output from PtsAlgHorseShoe function

Value

Returns the area of the horseshoe shape with the circular and rectangular portions separated

Note

Currently forces use of a height 5/4 width and r2=2*r1 of origin (center)

References

[1] From http://mathforum.org/library/drmath/view/51816.html

Examples

```
HorseShoeTest<-PtsAlgHorseShoe(z=2000,spacing=1,endAngles=c(-90,90))
HorseShoeArea(HorseShoeTest)
```

LatRg — *Calculates the latitudinal range in degrees and kilometers*

Description

Calculates the latitudinal range in degrees and kilometers

Usage

```
LatRg(lats)
```

Arguments

`lats`	- Latitudinal occurrences in decimal degrees

Value

Returns the outermost coordinates of the set, the latitudinal span in degrees and in kilometers

Examples

```
lats<-c(-23,56,-2,45,66)
LatRg(lats)
```

LongRg — *Calculates the longitudinal range in degrees and kilometers, assuming a latitude of 45 degrees for all points by default. Accounts for the possibility of wrapping around the globe.*

Description

Calculates the longitudinal range in degrees and kilometers, assuming a latitude of 45 degrees for all points by default. Accounts for the possibility of wrapping around the globe.

Usage

```
LongRg(longs, lats = 45)
```

Arguments

`longs`	- Longitudinal occurrences in decimal degrees
`lats`	- A single value representing the latitude to calculate longitudinal distance from or a list of latitudinal coordinates in decimal degrees

Details

Calculates the longitudinal range as 360-largest longitudinal gap and accounts for the possbility that a taxon's range wraps around the prime meridian

Value

Returns the outermost coordinates of the set, the longitudinal span in degrees and in kilometers

Examples

```
longs<-c(133,76,-77,7,-80)
lats<-c(45)
LongRg(longs)
```

`MSTDist` *Calculates the minimum spanning tree distance, in kilometers, using Prim's Algorithm [1]*

Description

Calculates the minimum spanning tree distance, in kilometers, using Prim's Algorithm [1]

Usage

```
MSTDist(longs, lats)
```

Arguments

`longs`	- Longitudinal occurrences in decimal degrees
`lats`	- Latitudinal occurrences in decimal degrees

Details

Uses Prim's algorithm for finding the minimum spanning tree, time-consuming calculation as the number of locations increases past 1000

Value

Returns the minimum spanning tree distance in kilometers, the pairwise distance matrix of occurrences, the order points were connected in, and a 2-column array of coordinates

References

[1] Prim, R.C. 1957. Shortest Connection Networks and Some Generalizations. The Bell System Technical Journal 36:1389-1401.

Examples

```
longs<-c(12,34,-55)
lats<-c(-41,3,56)
MSTDist(longs,lats)
```

MSTDist_FromMat *Calculates the minimum spanning tree distance, in kilometers, using Prim's Algorithm [1] and a previously calculated pairwsie distance matrix*

Description

Calculates the minimum spanning tree distance, in kilometers, using Prim's Algorithm [1] and a previously calculated pairwsie distance matrix

Usage

```
MSTDist_FromMat(longs, lats, DistMat)
```

Arguments

`longs`	- Longitudinal occurrences in decimal degrees
`lats`	- Latitudinal occurrences in decimal degrees
`DistMat`	- Pairwsie distance matrix of coordinates, from the PWMatrix() function

Details

Uses Prim's algorithm for finding the minimum spanning tree

Value

Returns the minimum spanning tree distance in kilometers, the pairwise distance matrix of occurrences, the order points were connected in, and a 2-column array of coordinates

References

[1] Prim, R.C. 1957. Shortest Connection Networks and Some Generalizations. The Bell System Technical Journal 36:1389-1401.

Examples

```
MSTCalc<-MSTDist(longs=c(22,44,-12,67),lats=c(-77,56,22,56))
MSTDist_FromMat(MSTCalc$Longitude,MSTCalc$Latitude,MSTCalc$MST_DistMat)
```

PEE_MultiTaxa *Function to compile the PBDB_PEE_SingleTaxon output for a list of taxa*

Description

Function to compile the PBDB_PEE_SingleTaxon output for a list of taxa

Usage

```
PEE_MultiTaxa(GeoRare_Multi)
```

Arguments

GeoRare_Multi	- The list of geographic range measures calculated from the GeoRarefaction_MultiTaxa function

Value

Returns a vector list of six geographic range measures matrix with percent error of estimates [1] for each value

References

[1] Russell, M.P. & D.R. Lindberg. 1988. Real and Random Patterns Associated with Molluscan Spatial and Temporal Distributions. Paleobiology 14:322-330.

See Also

See the velociraptr package for details of the downloadPBDB() function

Examples

```
## Not run:
data(BivalvePBDB)
BivalveMatrix<-CoordList_PBDB(BivalvePBDB)
BivalveGeo<-GeoRarefaction_MultiTaxa(nLocCut=20,OccMatrix=BivalveMatrix,TaxaStart=3,replacePts=TRUE)
PEE_MultiTaxa(BivalveGeo)

## End(Not run)
```

PEE_SingleTaxon *Function to calculate PEE [1] matrices for all the geographic range measures*

Description

Function to calculate PEE [1] matrices for all the geographic range measures

Usage

```
PEE_SingleTaxon(GeoRare, TName = "Brach 1")
```

Arguments

GeoRare	- The list of geographic range measures calculated from the GeoRarefaction_SingleTaxon or GeoRarefaction_MultiTaxa functions
TName	- Name of the target taxon for analysis as a string

Value

Returns a list of six geographic range measures matrix with percent error of estimates for each value

References

[1] Russell, M.P. & D.R. Lindberg. 1988. Real and Random Patterns Associated with Molluscan Spatial and Temporal Distributions. Paleobiology 14:322-330.

See Also

See the velociraptr package for details of the downloadPBDB() function

Examples

```
## Not run:
data(BivalvePBDB)
BivalveMatrix<-CoordList_PBDB(BivalvePBDB)
BivalveGeo<-GeoRarefaction_MultiTaxa(nLocCut=50,OccMatrix=BivalveMatrix,TaxaStart=3,iter=20)
PEE_SingleTaxon(GeoRare=BivalveGeo,TName=names(BivalveGeo)[3])

## End(Not run)
```

PlotConvexHull *Plots the minimum convex hull of a set of coordinates*

Description

Plots the minimum convex hull of a set of coordinates

Usage

```
PlotConvexHull(xcoord, ycoord, lcolor = "blue")
```

Arguments

`xcoord`	- Array of x-coordinates or longitudinal values
`ycoord`	- Array of y-coordinates or latitudinal values
`lcolor`	- String or integer value indicating the color of the convex hull boundary lines

Value

Plots a minimum convex hull

Note

This function does not account for the possibility of points crossing the prime meridian and in cases where this occurs the convex hull shown will be incorrect

Examples

```
longs<-c(20,20,40,40)
lats<-c(-5,5,-5,5)
PlotConvexHull(xcoord=longs,ycoord=lats)
```

PlotMST *Plots the minimum spanning tree of a set of coordinates*

Description

Plots the minimum spanning tree of a set of coordinates

Usage

```
PlotMST(MSTCalc, color = "black", symbol = 16, xlimit = "NA",
  ylimit = "NA")
```

Arguments

`MSTCalc`	- Output from the MSTDist function
`color`	- Color of the lines connecting point
`symbol`	- Symbol value for the pch graphical parameter for plotting coordinates
`xlimit`	- Array of values denoting the x-axis limits
`ylimit`	- Array of values denoting the y-axis limits

Value

Plots a minimum spanning tree

Note

If the xlimit and ylimit parameters are left to their default values the axis ranges are based on the minimum and maximum values of the coordinates This function does not account for the possibility of points crossing the prime meridian and in cases where this occurs lines will cut across the entire plot

Examples

```
w<-MSTDist(longs=c(23,78,-23,56),lats=c(21,4,55,-3))
PlotMST(MSTCalc=w)
```

`Plot_Rarefaction` *Plots the measured value versus rarefied samples or percent error of estimates (PEE) of a rarefied samples for a geographic range measure*

Description

Plots the measured value versus rarefied samples or percent error of estimates (PEE) of a rarefied samples for a geographic range measure

Usage

```
Plot_Rarefaction(Mes1_AllTaxa, Mes2_AllTaxa, symbol = 20, measure = 2,
  SampSize = 1, color = "black")
```

Arguments

`Mes1_AllTaxa`	- Vector list of measured values for multiple taxa, from the GeoRarefaction_MultiTaxa function, or PEE calculations for multiple taxa, from from PEE_MultiTaxa function
`Mes2_AllTaxa`	- Vector list of measured values for multiple taxa, output from GeoRarefaction_MultiTaxa function
`symbol`	- Symbol used for plotting, as per pch graphical parameter

`measure`	- Specifies which measure to be plotted, 2=MST 3=CH 4=GCD 5=LatRg 6=LongRg 7=CellCount
`SampSize`	- Specifies the index value of the rarefaction sample size from the PEE_AllTaxa parameter
`color`	- Specifies the color of symbols being plotted

Details

For each taxon for a specific geographic range measure using 95 Measure paramter is the ordinal position of the measure of interest 2=MST,3=CH,4=GCD,5=LatRg,6=LongRg,7=CellCount SampSize parameter indicates the index column position of steps size, default is first column (all points)

Value

Returns a plot of measured geographic range values or PEE versus true value for a specific geographic range measure at varying sample sizes

Examples

```
## Not run:
data(BivalvePBDB)
BivalveMatrix<-CoordList_PBDB(BivalvePBDB)
BivalveGeo<-GeoRarefaction_MultiTaxa(nLocCut=20,OccMatrix=BivalveMatrix,TaxaStart=3,replacePts=TRUE)
BivalvePEE<-PEE_MultiTaxa(BivalveGeo)
Plot_Rarefaction(BivalvePEE,BivalveGeo,symbol=20,measure=2,SampSize=2)

## End(Not run)
```

`PtsAlgHorseShoe` *Function to find points along a horseshoe shape*

Description

Function to find points along a horseshoe shape

Usage

```
PtsAlgHorseShoe(z, spacing = 1, endAngles = c(-90, 90))
```

Arguments

`z`	- Distance in kilometers from the center of the horseshoe to its lower boundary
`spacing`	- Degrees between successive point outlining the horseshoe shape
`endAngles`	- Array of values denoting the degree of rotation for the circular part of the horseshoe distribution

Details

Currently forces use 0,0 as the center, a height 5/4 width, and r2=2*r1 of origin (center)

Value

Returns a 2-dimensional array of decimal degree coordinates outlining a horseshoe shape

Note

Currently only works when endAngles are multiples of -90,90; otherwise rectangle point positions are off

References

[1] From http://mathforum.org/library/drmath/view/51816.html

Examples

```
PtsAlgHorseShoe(z=2000,spacing=1,endAngles=c(-90,90))
```

PWMatrix *Creates a sparse pairwise distance matrix of a coordinate set*

Description

Creates a sparse pairwise distance matrix of a coordinate set

Usage

```
PWMatrix(Coords)
```

Arguments

Coords	- Two-dimensional array of longitudinal and latitudinal coordinates output from CoordCollapse() function

Value

Returns a sparse pairwise distance matrix of great circle distances between pairs of points

Examples

```
longs<-runif(10,-22,33)
lats<-runif(10,-22,33)
Coords<-CoordCollapse(longs,lats)
PWMatrix(Coords)
```

`RandHorseShoe` *Function to randomly generate points within a horseshoe-shape*

Description

Function to randomly generate points within a horseshoe-shape

Usage

```
RandHorseShoe(center = c(0, 0), npts = 100, HorseShoeShape, HRatio = 1.5,
  RadRatio = 0.5)
```

Arguments

`center` - Arrary containint the coordinates of the center of circular portion of the horseshoe in decimal degrees

`npts` - Integer value indicating the number of points to generate within the horseshoe shape

`HorseShoeShape` - Object containing the outline of a horseshoe shape, output from PtsAlgHorseShoe function

`HRatio` - The ratio of the lower rectangle portions of the horseshoe to the outer radius of the circular portion of the horseshoe

`RadRatio` - The ration of the of the outer to inner radius of the circular part of the horseshoe shape

Value

Returns a 2-dimensional array of decimal degree coordinates within the horseshoe shape and the total area of the shape

Note

HRatio Currently defaults to 3/2 per the PtsAlgHorseShoe() function RadRatio currently defaults to 1/2 per the PtsAlgHorseShoe() function Center currently defaults to c(0,0) as this is always the center of the PtsAlgHorseShoe() function currently Function currently does not take acount of the decreasing surface area moving toward the poles so points closer to the poles will be overrepresented relative to the actual surface area they represent

Examples

```
HorseShoeTest<-PtsAlgHorseShoe(z=2000,spacing=1,endAngles=c(-90,90))
RandHorseShoe(center=c(0,0),npts=100,HorseShoeShape=HorseShoeTest)
```

RandRec *Function to randomly generate points within a given rectangular shaped distribution*

Description

Function to randomly generate points within a given rectangular shaped distribution

Usage

```
RandRec(RecShape, npts = 100)
```

Arguments

`RecShape`	- The outline of a rectangular distribution output from the EqualAreaRectangle() function
`npts`	- The number of randomly generated points within the rectangular shape

Value

Returns a 2-dimensional array of decimal degree coordinates within a rectangular shape

Note

Function currently does not take acount of the decreasing surface area moving toward the poles so points closer to the poles will be overrepresented relative to the actual surface area they represent

Examples

```
HorseShoeTest<-PtsAlgHorseShoe(z=2000,spacing=1,endAngles=c(-90,90))
HorseShoePts<-RandHorseShoe(center=c(0,0),npts=100,HorseShoeShape=HorseShoeTest)
RecOutline<-EqualAreaRectangle(TargetArea=as.numeric(HorseShoePts$TotalArea_km2),error=0.001)
RandRec(RecShape=RecOutline,npts=100)
```

Index