Package ‘aspace’

April 20, 2024

Type Package

Title Functions for Estimating Centrographic Statistics

Version 4.1.2

Date 2024-04-18

Maintainer Tarmo K. Remmel <remmelt@yorku.ca>

Description A collection of functions for computing centrographic statistics (e.g., standard distance, standard deviation ellipse, standard deviation box) for observations taken at point locations. Separate plotting functions have been developed for each measure. Users interested in writing results to ESRI shapefiles can do so by using results from 'aspace' functions as inputs to the convert.to.shapefile() and write.shapefile() functions in the 'shapefiles' library. We intend to provide 'terra' integration for geographic data in a future release. The 'aspace' package was originally conceived to aid in the analysis of spatial patterns of travel behaviour (see Buliung and Remmel 2008 <doi:10.1007/s10109-008-0063-7>).

License GPL-3

Depends R (>= 2.3.0), splancs, Hmisc

LazyData true

Repository CRAN

Date/Publication 2024-04-19 23:52:53 UTC

NeedsCompilation no

Author Tarmo K. Remmel [aut, cre] (<https://orcid.org/0000-0001-6251-876X>),
Randy Bui [aut],
Ron N. Buliung [aut]

R topics documented:

`aspace-package` *A collection of functions for estimating centrographic statistics and computational geometries for spatial point patterns*

Description

A collection of functions for computing centrographic statistics (e.g., standard distance, standard deviation ellipse, standard deviation box) for observations taken at point locations. The 'aspace' package was originally conceived to aid in the analysis of spatial patterns of travel behaviour (see Buliung and Remmel, 2008).

Details

Package:	aspace
Type:	Package
Version:	4.1.0
Date:	2023-09-05
License:	GPL-3

Author(s)

Randy Bui, Ron N. Buliung, Tarmo K. Remmel

References

Bachi, R. 1963. Standard distance measures and related methods for spatial analysis. Papers of the Regional Science Association 10: 83-132.

Buliung, R.N. and Remmel, T. (2008) Open source, spatial analysis, and activity travel behaviour research: capabilities of the aspace package. Journal of Geographical Systems, 10: 191-216.

Buliung, R.N. and Kanaroglou, P.S. (2006) Urban form and household activity-travel behaviour. Growth and Change, 37: 174-201.

Ebdon, D. 1988. Statistics in Geography 2nd Edition. Oxford UK: Blackwell.

Levine, N. 2002. CrimeStat II: A Spatial Statistics Program for the Analysis of Crime Incident Locations (version 2.0) Houston TX/National Institute of Justice, Washington DC: Ned Levine & Associates.

---

acos_d *Compute inverse cosine with angle given in degrees*

Description

Provides the functionality of acos, but for input angles measured in degrees (not radians).

Usage

```
acos_d(theta = 0)
```

Arguments

theta	A numeric angular measurement in degrees from north.

Details

Since the R default is to compute trigonometric functions on angular measurements stored in radians, this simple function performs the conversion from degrees, reducing the need to do so a priori, outside the function.

Value

Returns a numeric value for the inverse cosine of the specified angular measurement

Note

To reduce the need for unit conversions prior to calling trigonometric functions, this function accepts input in angular degrees rather than radians. Depending on the data source, this function may be preferred to the existing version requiring input in angular radians.

Author(s)

Tarmo K. Remmel

See Also

sin_d, cos_d, tan_d, asin_d, atan_d

Examples

```
acos_d(theta = 90)
```

---

activities — *Demo Data: x and y coordinates of 10 specified point locations*

Description

This is a simple two-column data frame (or matrix) containing x,y coordinates for a series of point locations. These data mimic UTM coordinates such that the first column contains Easting (x), and the second Northing (y) coordinates for the set of unique points.

Usage

```
data(activities)
```

Format

A data frame with 10 observations on the following 2 variables.

`col1` A numeric vector of x-coordinates

`col2` A numeric vector of y-coordinates

Details

The coordinates of the points must have the same units and projection as the specified center.

Source

This demonstration data has been manufactured for illustrative purposes only.

Examples

```
data(activities)
str(activities)
plot(activities)
```

`activities2` *Demo Data: x and y coordinates of 10 specified point locations*

Description

This is a simple two-column data frame (or matrix) containing x,y coordinates for a series of point locations. These data mimic UTM coordinates such that the first column contains Easting (x), and the second Northing (y) coordinates for the set of unique points.

Usage

```
data(activities2)
```

Format

A data frame with 10 observations on the following 2 variables.

`col1` A numeric vector of x-coordinates

`col2` A numeric vector of y-coordinates

Details

The coordinates of the points must have the same units and projection as the specified center.

Source

This demonstration data has been manufactured for illustrative purposes only.

Examples

```
data(activities2)
str(activities2)
plot(activities2)
```

`asin_d` *Compute inverse sine with angle given in degrees*

Description

Provides the functionality of asin, but for input angles measured in degrees (not radians).

Usage

```
asin_d(theta = 0)
```

Arguments

theta	A numeric angular measurement in degrees from north.

Details

Since the R default is to compute trigonometric functions on angular measurements stored in radians, this simple function performs the conversion from degrees, reducing the need to do so a priori, outside the function.

Value

Returns a numeric value for the inverse sine of the specified angular measurement.

Note

To reduce the need for unit conversions prior to calling trigonometric functions, this function accepts input in angular degrees rather than radians. Depending on the data source, this function may be preferred to the existing version requiring input in angular radians.

Author(s)

Tarmo K. Remmel

See Also

sin_d, cos_d, tan_d, acos_d, atan_d

Examples

```
asin_d(theta = 90)
```

---

as_radians *Converts degrees to radians*

Description

This function converts an angular measure stored in degrees to radians. This is an alternative to the rad function available in the package circular.

Usage

```
as_radians(theta = 0)
```

Arguments

theta	A numeric angular measurement in degrees from north.

Details

Achieves a very simple conversion with a convenient function call.

Value

Returns a numeric value for an angle in radians that is equivalent to the input theta in degrees.

Note

The purpose of this function is to reduce computer code clutter when using angular measuremnts in R. The simple function call ensures that degree to radian conversions are completed consistently and accurately. Since trigonometric functions in R require angular measures in radians rather than degrees, this simple function can be used for simple angular unit conversion.

Author(s)

Tarmo K. Remmel

See Also

sin_d, cos_d, tan_d, asin_d, acos_d, atan_d

Examples

```
as_radians(theta = 90)
```

---

atan_d *Compute inverse tangent with angle given in degrees*

Description

Provides the functionality of atan, but for input angles measured in degrees (not radians).

Usage

```
atan_d(theta = 0)
```

Arguments

theta	A numeric angular measurement in degrees from north.

Details

Since the R default is to compute trigonometric functions on angular measurements stored in radians, this simple function performs the conversion from degrees, reducing the need to do so a priori, outside the function.

Value

Returns a numeric value for the inverse tangent of the specified angular measurement.

Note

To reduce the need for unit conversions prior to calling trigonometric functions, this function accepts input in angular degrees rather than radians. Depending on data, this function may be preferred to the existing version requiring input in angular radians.

Author(s)

Tarmo K. Remmel

See Also

`sin_d`, `cos_d`, `tan_d`, `asin_d`, `acos_d`

Examples

```
atan_d(theta = 90)
```

---

calc_box *Calculate the Standard Deviation Box*

Description

The orthogonal dispersion of a set of points can be described using the standard deviation of the x- and y-coordinates of a set of point observations. The orthogonal dispersion can then be visualized with a Standard Deviation Box. This function computes the properties of the Standard Deviation Box (SD Box) from a set of point observations.

Usage

```
calc_box(id=1, centre.xy=NULL, calccentre=TRUE, weighted=FALSE,
weights=NULL, points=NULL, verbose=FALSE)
```

Arguments

`id`	A unique integer to identify a SD Box
`centre.xy`	A vector of length 2, containing the x- and y-coordinates of the geographic centre of the SD Box
`calccentre`	Boolean: Set to TRUE if the mean center is to be calculated
`weighted`	Boolean: Set to TRUE if the weighted mean center is to be computed with weighted coordinates
`weights`	Weights applied to point observations, number of weights should equal the number of observations

points	A 2-column matrix or data frame containing the set of point observations input to the calc_box function
verbose	Boolean: Set to TRUE if extensive feedback is desired on the standard output

Details

Use the LOCATIONS element in the output list object along with the ATTRIBUTES elements can be used to produce shapefiles or other vector point files for geographic data.

Value

The returned result is a list:

TYPE	The type of calculation results stored in the object: BOX, SDD, SDE, CMD, CF, or CF2PTS, MNC, MDC
DATE	The date and time that the function was run
ID	Identifier for the SD Box shape - it should be unique
LOCATIONS	Locations pertinent for the BOX that can be used with ATTRIBUTES if wishing to build a vector point file for geographic data outside of this pacakge.
FORPLOTTING	Coordinates and identifiers used for plotting by plot_box()
ATTRIBUTES	Attributes for the output BOX that can be used with LOCATIONS coordinates if wishing to build a vector point file for geographic data outside of this package.
id	Identifier for the SD Box shape - it should be unique
calccentre	Boolean: TRUE if the mean centre was estimated
weighted	Boolean: TRUE if the weighted mean centre was estimated
CENTRE.x	X-coordinate of the centre
CENTRE.y	Y-coordinate of the centre
SD.x	Orthogonal standard deviation in the x-axis
SD.y	Orthogonal standard deviation in the y-axis
Box.area	Area of the standard deviation box
NW.coord	North-west coordinates of SD Box
NE.coord	North-east coordinates of SD Box
SW.coord	South-west coordinates of SD Box
SE.coord	South-east coordinates of SD Box

Note

Results specific for plotting are stored in the FORPLOTTING element within the produced list object. Pass the entire object to plot_box() and the function automatically extracts this information. This function can be used on its own (once) or repetitively in a loop to process grouped point data stored in a larger table. When used repetitively, be sure to increment the id argument to ensure that each SD BOX has a unique identifier.

Author(s)

Tarmo K. Remmel, Randy Bui, Ron N. Buliung

See Also

calc_sdd, calc_sde, calc_cmd, calc_cf, calc_cf2pts, calc_mnc, calc_mdc, wtd.var

Examples

```
# BOX EXAMPLE
data(activities)
a <- calc_box(id=1, centre.xy=NULL, points=activities)
str(a)
print(a)

# IF THE RESULT OF THIS FUNCTION IS STORED TO AN OBJECT, THE plot_box()
# FUNCTION WILL TAKE THAT OBJECT AS INPUT FOR PLOTTING VIA THE datin ARGUMENT

# BOX TO SHAPEFILE EXAMPLE (REMOVE THE COMMENTS TO RUN)
# shp <- convert.to.shapefile(a$LOCATIONS, a$ATTRIBUTES, "id", 5)
# write.shapefile(shp, "BOX_Shape", arcgis=T)
```

---

calc_cf *Central Feature (CF) Calculator*

Description

Identifies the central feature within a set of point locations.

Usage

```
calc_cf(id=1, points=NULL, verbose=FALSE)
```

Arguments

id	A unique integer to identify the CF
points	A 2-column matrix or data frame containing the set of point observations
verbose	Boolean flag for verbose output to monitor

Details

Use the LOCATIONS element in the output list object along with the ATTRIBUTES elements can be used to produce shapefiles or other vector point files for geographic data.

Value

The returned result is a list:

TYPE	The type of calculation results stored in the object: BOX, SDD, SDE, CMD, CF, or CF2PTS, MNC, MDC
DATE	The date and time that the function was run
ID	Identifier for the central feature - it should be unique
LOCATIONS	Locations pertinent for the CF that can be used with ATTRIBUTES if wishing to build a vector point file for geographic data outside of this pacakge.
FORPLOTTING	Coordinates and identifiers used for plotting.
ATTRIBUTES	Attributes for the output CF that can be used with LOCATIONS coordinates if wishing to build a vector point file for geographic data outside of this package.
id	Identifier for the central feature - it should be unique
CF.x	X-coordinate of the central feature
CF.y	Y-coordinate of the central feature

Note

Results specific for plotting are stored in the FORPLOTTING element within the produced list object. Pass the entire object to plot_box() and the function automatically extracts this information. This function can be used on its own (once) or repetitively in a loop to process grouped point data stored in a larger table. When used repetitively, be sure to increment the id argument to ensure that each CF has a unique identifier.

Author(s)

Randy Bui, Ron Buliung, Tarmo K Remmel

See Also

calc_box, calc_sdd, calc_sde, calc_cmd, calc_cf2pts, calc_mnc, calc_mdc

Examples

```
# CF EXAMPLE
data(activities)
a <- calc_cf(id=1, points=activities)
str(a)
print(a)

# BOX TO SHAPEFILE EXAMPLE (REMOVE THE COMMENTS TO RUN)
# shp <- convert.to.shapefile(a$LOCATIONS, a$ATTRIBUTES, "id", 5)
# write.shapefile(shp, "CF_Shape", arcgis=T)
```

calc_cf2pts *Central feature between 2 point patterns (CF2PTS) Calculator*

Description

Central feature of point2 within point1. Identifies the central feature as the point location in the first pattern that has the smallest cummulative distance to features in a second point pattern.

Usage

```
calc_cf2pts(id=1, points1=NULL, points2=NULL, verbose=FALSE)
```

Arguments

id	A unique integer to identify the CF2PTS
points1	A 2-column matrix or data frame containing the first set of point observations
points2	A 2-column matrix or data frame containing the second set of point observations
verbose	A Boolean flag to control verbose reporting on monitor

Details

Use the LOCATIONS element in the output list object along with the ATTRIBUTES elements can be used to produce shapefiles or other vector point files for geographic data.

Value

The returned result is a list:

TYPE	The type of calculation results stored in the object: BOX, SDD, SDE, CMD, CF, or CF2PTS, MNC, MDC
DATE	The date and time that the function was run
ID	Identifier for the central feature - it should be unique
LOCATIONS	Locations pertinent for the CF2PTS that can be used with ATTRIBUTES if wishing to build a vector point file for geographic data outside of this pacakge.
FORPLOTTING	Coordinates and identifiers used for plotting.
ATTRIBUTES	Attributes for the output CF2PTS that can be used with LOCATIONS coordinates if wishing to build a vector point file for geographic data outside of this package.
id	Identifier for the central feature - it should be unique
CF2PTS.x	X-coordinate of the central feature
CF2PTS.y	Y-coordinate of the central feature

Note

Results specific for plotting are stored in the FORPLOTTING element within the produced list object. This function can be used on its own (once) or repetitively in a loop to process grouped point data stored in a larger table. When used repetitively, be sure to increment the id argument to ensure that each CF2PTS has a unique identifier.

Author(s)

Randy Bui, Ron Buliung

See Also

calc_box, calc_sdd, calc_sde, calc_cmd, calc_cf, calc_mnc, calc_mdc

Examples

```
# CF2PTS EXAMPLE
data(activities)
data(activities2)
a <- calc_cf2pts(id=1, points1=activities, points2=activities2)
str(a)
print(a)

# IF THE RESULT OF THIS FUNCTION IS STORED TO AN OBJECT, THE plot_box()
# FUNCTION WILL TAKE THAT OBJECT AS INPUT FOR PLOTTING VIA THE datin ARGUMENT

# CF2PTS TO SHAPEFILE EXAMPLE (REMOVE THE COMMENTS TO RUN)
# shp <- convert.to.shapefile(a$LOCATIONS, a$ATTRIBUTES, "id", 5)
# write.shapefile(shp, "CF2PTS_Shape", arcgis=T)
```

---

calc_cmd *Centre of Minimum Distance (CMD) Calculator*

Description

Compute the CMD within a set of point locations.

Usage

```
calc_cmd(id=1, dist=100, points=NULL, verbose=FALSE)
```

Arguments

id	A unique integer to identify the CMD
dist	Hold distance value between i and ith iterations
points	A 2-column matrix or data frame containing the set of point observations
verbose	A Boolean flag to control verbose feedback on screen

Details

Use the cmdloc (coordinates) and cmdatt(attributes) to produce shapefiles using the convert.to.shapefile and write.shapefile from the shapefiles library

Value

The returned result is a list:

`TYPE`	The type of calculation results stored in the object: BOX, SDD, SDE, CMD, CF, or CF2PTS, MNC, MDC
`DATE`	The date and time that the function was run
`ID`	Identifier for the CMD - it should be unique
`LOCATIONS`	Locations pertinent for the CMD that can be used with ATTRIBUTES if wishing to build a vector point file for geographic data outside of this pacakge.
`FORPLOTTING`	Coordinates and identifiers used for plotting by plot_cmd()
`ATTRIBUTES`	Attributes for the output CMD that can be used with LOCATIONS coordinates if wishing to build a vector point file for geographic data outside of this package.
`id`	Identifier for the CMD - it should be unique
`CMD.x`	X-coordinate of the CMD
`CMD.y`	Y-coordinate of the CMD
`distance`	Hold distance value between i and ith iterations (metres
`Number of Cells`	Hold number of cells in each grid created for each iteration

Note

Results specific for plotting are stored in the FORPLOTTING element within the produced list object. Pass the entire object to plot_box() and the function automatically extracts this information. This function can be used on its own (once) or repetitively in a loop to process grouped point data stored in a larger table. When used repetitively, be sure to increment the id argument to ensure that each SD BOX has a unique identifier.

Author(s)

Randy Bui, Ron Buliung, Tarmo K. Remmel

See Also

`calc_box`, `calc_sdd`, `calc_sde`, `calc_cf`, `calc_cf2pts`, `calc_mnc`, `calc_mdc`

Examples

```
# CMD EXAMPLE
a <- calc_cmd(id=1, dist=100, points=activities)
str(a)
print(a)

# CMD TO SHAPEFILE EXAMPLE (REMOVE THE COMMENTS TO RUN)
```

```
# shp <- convert.to.shapefile(a$LOCATIONS, a$ATTRIBUTES, "id", 5)
# write.shapefile(shp, "CMD_Shape", arcgis=T)
```

---

calc_mdc *Median Centre Calculator*

Description

Compute the median centre from a series of point locations.

Usage

```
calc_mdc(id=1, points=NULL, verbose=FALSE)
```

Arguments

id	A unique integer to identify the median centre
points	A 2-column matrix or data frame containing the set of point observations
verbose	A Boolean flag to control verbose content on the monitor

Details

Use the medianloc (coordinates) and medianatt(attributes) to produce shapefiles using the convert.to.shapefile and write.shapefile from the shapefiles library

Value

The returned result is a list:

TYPE	The type of calculation results stored in the object: BOX, SDD, SDE, CMD, CF, or CF2PTS, MNC, MDC
DATE	The date and time that the function was run
ID	Identifier for the median centre - it should be unique
LOCATIONS	Locations pertinent for the MDC that can be used with ATTRIBUTES if wishing to build a vector point file for geographic data outside of this pacakge.
FORPLOTTING	Coordinates and identifiers used for plotting.
ATTRIBUTES	Attributes for the output MDC that can be used with LOCATIONS coordinates if wishing to build a vector point file for geographic data outside of this package.
id	Identifier for the median centre - it should be unique
median.x	X-coordinate of the median centre
median.y	Y-coordinate of the median centre

Note

Results are stored in the r.median object and can be passed through plotting functions. This function can also be used repetitively within a loop to compute multiple median centres from different datasets.

Author(s)

Randy Bui, Ron Buliung

See Also

calc_box, calc_sdd, calc_sde, calc_cmd, calc_cf, calc_cf2pts, calc_mnc

Examples

```
# MEDIAN CENTRE EXAMPLE
a <- calc_mdc(id=1, points=activities)
str(a)
print(a)

# MEDIAN CENTRE TO SHAPEFILE EXAMPLE (REMOVE THE COMMENTS TO RUN)
# shp <- convert.to.shapefile(a$LOCATIONS, a$ATTRIBUTES, "id", 5)
# write.shapefile(shp, "Median_Shape", arcgis=T)
```

---

calc_mnc *Mean Centre Calculator*

Description

Compute the mean centre from a series of point locations.

Usage

```
calc_mnc(id=1, weighted=FALSE, weights=NULL,
 points=NULL, verbose=FALSE)
```

Arguments

id	A unique integer to identify the mean centre
weighted	Boolean: Set to TRUE if the weighted mean center is to be computed with weighted coordinates
weights	Weights applied to point observations, number of weights should equal the number of observations
points	A 2-column matrix or data frame containing the set of point observations
verbose	A Boolean flag that controls verbose feedback to the monitor

Details

Use the meanloc (coordinates) and meanatt(attributes) to produce shapefiles using the convert.to.shapefile and write.shapefile from the shapefiles library

Value

The returned result is a list:

TYPE	The type of calculation results stored in the object: BOX, SDD, SDE, CMD, CF, or CF2PTS, MNC, MDC
DATE	The date and time that the function was run
ID	Identifier for the mean centre - it should be unique
LOCATIONS	Locations pertinent for the MNC that can be used with ATTRIBUTES if wishing to build a vector point file for geographic data outside of this pacakge.
FORPLOTTING	Coordinates and identifiers used for plotting.
ATTRIBUTES	Attributes for the output MNC that can be used with LOCATIONS coordinates if wishing to build a vector point file for geographic data outside of this package.
id	Identifier for the mean centre - it should be unique
weighted	Boolean: TRUE if the weighted mean centre is to be used instead
weights	Weights applied to point observations
CENTRE.x	X-coordinate of the mean centre
CENTRE.y	Y-coordinate of the mean centre

Note

Results are stored in the r.mean object and can be passed through plotting functions. This function can also be used repetitively within a loop to compute multiple mean centres from different datasets.

Author(s)

Randy Bui, Ron Buliung

See Also

calc_box, calc_sdd, calc_sde, calc_cmd, calc_cf, calc_cf2pts, calc_mdc

Examples

```
# MEAN CENTRE EXAMPLE
a <- calc_mnc(id=1, points=activities)
str(a)
print(a)

# MEAN CENTRE TO SHAPEFILE EXAMPLE (REMOVE THE COMMENTS TO RUN)
# shp <- convert.to.shapefile(a$LOCATIONS, a$ATTRIBUTES, "id", 5)
# write.shapefile(shp, "Mean_Shape", arcgis=T)
```

calc_sdd *Calculate the Standard Distance Deviation (Standard Distance)*

Description

This function computes the Standard Distance Deviation (SDD) or Standard Distance from a set of points.

Usage

```
calc_sdd(id=1, centre.xy=NULL, calccentre=TRUE, weighted=FALSE,
 weights=NULL, points=NULL, verbose=FALSE)
```

Arguments

id	A unique integer to identify a SDD estimate
centre.xy	A vector of length 2, containing the x- and y-coordinates of the SDD centre
calccentre	Boolean: Set to TRUE if the mean center is to be calculated
weighted	Boolean: Set to TRUE if the weighted mean center is to be computed with weighted coordinates
weights	Weights applied to point observations, number of weights should equal the number of observations
points	A 2-column matrix or data frame containing the set of point observations input to the calc_sdd function
verbose	Boolean: Set to TRUE if extensive feedback is desired on the standard output

Details

Use the LOCATIONS element in the output list object along with the ATTRIBUTES elements can be used to produce shapefiles or other vector point files for geographic data.

Value

The returned result is a list:

TYPE	The type of calculation results stored in the object: BOX, SDD, SDE, CMD, CF, or CF2PTS, MNC, MDC
DATE	The date and time that the function was run
ID	Identifier for the SDD shape - it should be unique
LOCATIONS	Locations pertinent for the SDD that can be used with ATTRIBUTES if wishing to build a vector point file for geographic data outside of this pacakge.
FORPLOTTING	Coordinates and identifiers used for plotting by plot_sdd()
ATTRIBUTES	Attributes for the output SDD that can be used with LOCATIONS coordinates if wishing to build a vector point file for geographic data outside of this package.

id	Identifier for the SDD shape - it should be unique
calccentre	Boolean: TRUE if mean centre is computed
weighted	Boolean: TRUE if the weighted mean centre is to be used instead
CENTRE.x	X-coordinate of the centre
CENTRE.y	Y-coordinate of the centre
SDD.radius	SDD value, radius of the SDD
SDD.area	Area of the SDD circle

Note

Results specific for plotting are stored in the FORPLOTTING element within the produced list object. Pass the entire object to plot_sdd() and the function automatically extracts this information. This function can be used on its own (once) or repetitively in a loop to process grouped point data stored in a larger table. When used repetitively, be sure to increment the id argument to ensure that each SDD has a unique identifier.

Author(s)

Tarmo K. Remmel, Randy Bui, Ron Buliung

See Also

calc_box, calc_sde, calc_cmd, calc_cf, calc_cf2pts, calc_mnc, calc_mdc

Examples

```
# SDD EXAMPLE
data(activities)
a <- calc_sdd(id=1, centre.xy=NULL, calccentre=TRUE, points=activities)
str(a)
print(a)

# IF THE RESULT OF THIS FUNCTION IS STORED TO AN OBJECT, THE plot_box()
# FUNCTION WILL TAKE THAT OBJECT AS INPUT FOR PLOTTING VIA THE datin ARGUMENT

# SDD TO SHAPEFILE EXAMPLE (REMOVE THE COMMENTS TO RUN)
# shp <- convert.to.shapefile(a$LOCATIONS, a$ATTRIBUTES,"id",5)
# write.shapefile(shp, "SDD_Shape", arcgis=T)
```

---

calc_sde   *Calculate the Standard Deviation Ellipse*

Description

This function computes the Standard Deviation Ellipse (SDE) from a set of points. The SDE is a centrographic measure used to characterize the dispersion of point observations along two orthogonal axes. The SDE also captures directional bias in a spatial point pattern, the ellipse will be oriented in the direction of maximum dispersion.

Usage

```
calc_sde(id=1, centre.xy=NULL, calccentre=TRUE,
weighted=FALSE, weights=NULL, points=NULL, verbose=FALSE)
```

Arguments

`id`	A unique integer to identify the shape
`centre.xy`	A vector of length 2, containing the x- and y-coordinates of the SDE centre (Planar Coordinates Only!)
`calccentre`	Boolean: Set to TRUE if the mean center is to be calculated
`weighted`	Boolean: Set to TRUE if the weighted mean center is to be computed with weighted coordinates
`weights`	Weights applied to point observations, number of weights should equal the number of observations
`points`	A 2-column matrix or data frame containing point coordinates
`verbose`	Boolean: Set to TRUE if extensive feedback is desired on the standard output

Details

Use the LOCATIONS element in the output list object along with the ATTRIBUTES elements can be used to produce shapefiles or other vector point files for geographic data.

Value

The returned result is a list:

`TYPE`	The type of calculation results stored in the object: BOX, SDD, SDE, CMD, CF, or CF2PTS, MNC, MDC
`DATE`	The date and time that the function was run
`ID`	Identifier for the SDE shape - it should be unique
`LOCATIONS`	Locations pertinent for the SDE that can be used with ATTRIBUTES if wishing to build a vector point file for geographic data outside of this pacakge.
`FORPLOTTING`	Coordinates and identifiers used for plotting by plot_sde()
`ATTRIBUTES`	Attributes for the output SDE that can be used with LOCATIONS coordinates if wishing to build a vector point file for geographic data outside of this package.
`id`	Identifier for the SDE shape - it should be unique
`calccentre`	Boolean: TRUE if mean centre is computed
`weighted`	Boolean: TRUE if the weighted mean centre is to be used instead
`CENTRE.x`	X-coordinate of the centre
`CENTRE.y`	Y-coordinate of the centre
`Sigma.x`	Half-length of axis along x-axis
`Sigma.y`	Half-length of axis along y-axis
`Major`	String indicating which axis is the major elliptical axis

`Minor`	String indicating which axis is the minor elliptical axis
`Theta`	Rotation angle in degrees
`Eccentricity`	A measure of eccentricity (i.e., the flatness of the ellipse)
`Area.sde`	Area of the SDE
`TanTheta`	Trigonometric result
`SinTheta`	Trigonometric result
`CosTheta`	Trigonometric result
`SinThetaCosTheta`	Trigonometric result
`Sin2Theta`	Trigonometric result
`Cos2Theta`	Trigonometric result
`ThetaCorr`	Corrected theta angle for rotation of major axis from north

Note

Results specific for plotting are stored in the FORPLOTTING element within the produced list object. Pass the entire object to plot_box() and the function automatically extracts this information. This function can be used on its own (once) or repetitively in a loop to process grouped point data stored in a larger table. When used repetitively, be sure to increment the id argument to ensure that each SDE has a unique identifier.

Author(s)

Tarmo K. Remmel, Randy Bui, Ron N. Buliung

References

See chapter 4 of the documentation manual for CrimeStat at http://www.icpsr.umich.edu/CRIMESTAT/ and Ebdon, D. 1987. Statistics in geography. 2nd edition. New York, NY Basil Blackwell Ltd. 232 p.

See Also

`calc_box`, `calc_sde`, `calc_cmd`, `calc_cf`, `calc_cf2pts`, `calc_mnc`, `calc_mdc`, `gridpts`

Examples

```
# SDE EXAMPLE
data(activities)
a <- calc_sde(id=1, centre.xy=NULL, points=activities)
str(a)
print(a)

# IF THE RESULT OF THIS FUNCTION IS STORED TO AN OBJECT, THE plot_sde()
# FUNCTION WILL TAKE THAT OBJECT AS INPUT FOR PLOTTING VIA THE datin ARGUMENT

# SDE TO SHAPEFILE EXAMPLE (REMOVE THE COMMENTS TO RUN)
# shp <- convert.to.shapefile(a$LOCATIONS, a$ATTRIBUTES, "id", 5)
# write.shapefile(shp, "SDE_Shape", arcgis=T)
```

centre — *Demo Data: Coordinates of a single source, centre, location*

Description

This is a simple two-element vector containing x,y coordinates for a source or central location associated with a spatial point pattern. In this example, the center location represents a point of importance in an individuals daily activity pattern. Surrounding point locations are places physically contacted by an individual during a particular time interval. Demonstration data mimics UTM coordinates such that the first element represents Easting (x), and the second, Northing (y).

Usage

```
data(centre)
```

Format

The format is a two-element vector of numeric entries.

Details

The coordinates of the center must have the same units and projection as the remaining point observations.

Source

This demonstration data has been manufactured for illustrative purposes only.

Examples

```
  data(centre)
  str(centre)
  plot(centre)

## plot_centres by default takes as input the result produced from mean_centre,
## median centre, CF, CF2PTS, and CMD, read from the current workspace.
```

cos_d — *Compute cosine with angle given in degrees*

Description

Provides the functionality of cos, but for input angles measured in degrees (not radians).

Usage

```
cos_d(theta = 0)
```

Arguments

`theta`	A numeric angular measurement in degrees from north.

Details

Since the R default is to compute trigonometric functions on angular measurements stored in radians, this simple function performs the conversion from degrees, reducing the need to do so a priori, outside the function.

Value

Returns a numeric value for the cosine of the specified angular measurement

Note

To reduce the need for unit conversions prior to calling trigonometric functions, this function accepts input in angular degrees rather than radians. Depending on data, this function may be preferred to the existing version requiring input in angular radians.

Author(s)

Tarmo K. Remmel

See Also

`sin_d`, `tan_d`, `asin_d`, `acos_d`, `atan_d`

Examples

```
cos_d(theta = 90)
```

`distances` *Multiple Euclidean distance calculator*

Description

Compute distances from a source location (point) to a series of destination locations (points).

Usage

```
distances(centre.xy = NULL, destmat = NULL, verbose = FALSE)
```

Arguments

`centre.xy`	Two-element vector containing x,y coordinates of the source location
`destmat`	Two-column matrix or data frame containing x,y coordinates of the activity locations
`verbose`	Boolean: Set to T if verbose output is desired

Details

Distance computations are strictly Euclidean between the source point and each destination point.

Value

A vector of distances, where each element corresponds to one of the distance between the source point and a destination (one row) from the destinations matrix.

Note

The order of distances in the output vector corresponds to the order of destination points in the destinations object starting at row = 1 through row = n.

Author(s)

Tarmo K. Remmel

Examples

```
data(centre)
data(activities)
distances(centre.xy=centre, destmat=activities, verbose=FALSE)
```

`plot_box` *Plot the Standard Distance Box*

Description

This function plots the standard deviation of x- and y-coordinates as a box, with the edges set, respectively, to the standard deviation of the x- and y-coordinates.

Usage

```
plot_box(datin=NULL, plotnew=TRUE, plothv=FALSE, plotweightedpts=FALSE,
weightedpts.col='black', weightedpts.pch=19, plotpoints=TRUE,
points.col='black', points.pch=1, plotcentre=TRUE, centre.col='black',
centre.pch=19, titletxt="Title", xaxis="Easting (m)",
yaxis="Northing (m)", box.col='black', box.lwd=2, jpeg=FALSE, ...)
```

Arguments

`datin`	Input data object; the result from calc_box()
`plotnew`	Boolean: Set to TRUE to create a new plot. Set to FALSE to overlay current plot.
`plothv`	Boolean: Set to TRUE if the orthogonal N-S, E-W axes are to be plotted through the centre

plotweightedpts	Boolean: Set to TRUE if the weighted point observations are to be plotted
weightedpts.col	Specify a colour for the weighted point observations
weightedpts.pch	Specify a plotting symbol for the weighted point observations
plotpoints	Boolean: Set to TRUE if the point observations are to be plotted
points.col	Specify a colour for the point observations
points.pch	Specify a plotting symbol for the point observations
plotcentre	Boolean: Set to TRUE if the mean/weighted/user-defined centre is to be plotted
centre.col	Specify a colour for the centre
centre.pch	Specify a plotting symbol for the centre
titletxt	A string to indicate the title for the plot
xaxis	A string to label the x-axis of the plot
yaxis	A string to label the y-axis of the plot
box.col	Specify a line colour for the SD Box
box.lwd	Specify a line width for the SD Box
jpeg	Boolean: Set to TRUE if the plot should be saved in JPEG format
...	Arguments to be passed to graphical parameters

Details

The element FORPLOTTING contained within the calc_box() output object is required to plot an SD Box. Provide the whole plot_box() output object as the argment for datin.

Value

This function returns a plot in the graphics device.

Author(s)

Tarmo K. Remmel, Randy Bui, Ron N. Buliung

See Also

plot_sdd, plot_sde

Examples

```
# NEED TO RUN THE BOX GENERATOR FIRST AND PASS THAT TO THE NEXT LINE
a <- calc_box(id=1, points=activities)
plot_box(datin=a, plotnew=TRUE, plothv=FALSE, plotweightedpts=FALSE,
plotpoints=TRUE, plotcentre=TRUE, titletxt="Title",
xaxis="Easting (m)", yaxis="Northing (m)")

# plot_box() BY DEFAULT, TAKES AS INPUT THE RESULT PRODUCED BY calc_box()
```

plot_centres *Plot centres*

Description

This function plots various centre of a set of point observations.

Usage

```
plot_centres(datin=NULL, plotnew=FALSE, plotSDE=FALSE,
xaxis="Easting (m)", yaxis="Northing (m)", plotweightedpts=FALSE,
weightedpts.col='black', weightedpts.pch=19, plotpoints=TRUE,
points.col='black', points.pch=1, plotcentre=FALSE, centre.col='black',
centre.pch=19, plotcentral=FALSE, central.col='green', central.pch=19,
plotCF2PTS=FALSE, CF2PTS.col='orange', CF2PTS.pch=19, plotmedian=FALSE,
median.col='blue', median.pch=17, plotCMD=FALSE, CMD.col='red',
CMD.pch=17, TITLE="Title", ...)
```

Arguments

datin	List object of all calc_ function objects that you wish to plot.
plotnew	Boolean: Set to TRUE to create a new plot. Set to FALSE to overlay current plot.
plotSDE	Boolean: Set to TRUE if the centres for the SDE are to be plotted.
xaxis	A string to label the x-axis of the plot.
yaxis	A string to label the y-axis of the plot.
plotweightedpts	Boolean: Set to TRUE if the weighted point observations are to be plotted.
weightedpts.col	Specify a colour for the weighted point observations.
weightedpts.pch	Specify a plotting symbol for the weighted point observations.
plotpoints	Boolean: Set to TRUE if the point observations are to be plotted.
points.col	Specify a colour for the point observations.
points.pch	Specify a plotting symbol for the point observations.
plotcentre	Boolean: Set to TRUE if the mean/weighted/user-defined centre is to be plotted.
centre.col	Specify a colour for the centre.
centre.pch	Specify a plotting symbol for the centre.
plotcentral	Boolean: Set to TRUE if the central feature is to be highlighted
central.col	Specify a colour for the central feature.
central.pch	Specify a plotting symbol for the central feature.

plotCF2PTS	Boolean: Set to TRUE if the central feature between 2 point patterns is to be highlighted.
CF2PTS.col	Specify a colour for the central feature.
CF2PTS.pch	Specify a plotting symbol for the central feature.
plotmedian	Boolean: Set to TRUE if the median centre is to be plotted.
median.col	Specify a colour for the median centre.
median.pch	Specify a plotting symbol for the median centre.
plotCMD	Boolean: Set to TRUE if the centre of minimum distance is to be plotted.
CMD.col	Specify a colour for the centre of minimum distance.
CMD.pch	Specify a plotting symbol for the centre of minimum distance.
TITLE	A character string with the title for your plot.
...	Arguments to be passed to graphical parameters.

Details

The element FORPLOTTING contained within any of the calc function output lists is required as an argument for datin.

Value

This function returns a plot in the graphics device.

Author(s)

Tarmo K. Remmel, Randy Bui, Ron N. Buliung

See Also

plot_box, plot_sdd, plot_sde

Examples

```
# MNC (BLACK CIRCLE)
a <- calc_mnc(points=activities)

# MDC (BLUE TRIANGLE)
b <- calc_mdc(points=activities)

# CF (GREEN CIRCLE)
d <- calc_cf(points=activities)

# CMD (RED TRIANGLE)
e <- calc_cmd(points=activities)

# CF2PTS (ORANGE CIRCLE)
f <- calc_cf2pts(points1=activities, points2=activities2)

# BUILD LIST OF OBJECTS TO PASS AS THE datin ARGUMENT
```

```
robjects <- list(a,b,d,e,f)
# CALL THE PLOT FUNCTION
plot_centres(datin=robjects, plotnew=TRUE, plotcentre=TRUE, plotmedian=TRUE)

# A FULL CALL COULD LOOK LIKE THE FOLLOWING
plot_centres(datin=robjects, plotnew=TRUE, plotcentre=TRUE,
plotmedian=TRUE, plotcentral=TRUE, plotCMD=TRUE, plotCF2PTS=TRUE)
```

---

plot_sdd *Plot the Standard Distance Deviation (Standard Distance)*

Description

This function plots the SDD as a circle with radius (standard distance), centred on a mean/weighted-mean/user-defined centre of a set of point observations.

Usage

```
plot_sdd(datin=NULL, plotnew=TRUE, plothv=FALSE, plotweightedpts=FALSE,
weightedpts.col='black', weightedpts.pch=19, plotpoints=TRUE,
points.col='black', points.pch=1, plotcentre=TRUE, centre.col='black',
centre.pch=19, titletxt="Title", xaxis="Easting (m)",
yaxis="Northing (m)", sdd.col='black', sdd.lwd=2, jpeg=FALSE, ...)
```

Arguments

datin	Input data object; the result from calc_sdd()
plotnew	Boolean: Set to TRUE to create a new plot. Set to FALSE to overlay current plot.
plothv	Boolean: Set to TRUE if the orthogonal N-S, E-W axes are to be plotted through the centre
plotweightedpts	Boolean: Set to TRUE if the weighted point observations are to be plotted
weightedpts.col	Specify a colour for the weighted point observations
weightedpts.pch	Specify a plotting symbol for the weighted point observations
plotpoints	Boolean: Set to TRUE if the point observations are to be plotted
points.col	Specify a colour for the point observations
points.pch	Specify a plotting symbol for the point observations
plotcentre	Boolean: Set to TRUE if the mean/weighted/user-defined centre is to be plotted
centre.col	Specify a colour for the centre
centre.pch	Specify a plotting symbol for the centre
titletxt	A string to indicate the title on the plot

xaxis	A string to label the x-axis of the plot
yaxis	A string to label the y-axis of the plot
sdd.col	Specify a line colour for the SDD circle
sdd.lwd	Specify a line width for the SDD circle
jpeg	Boolean: Set to TRUE if the plot should be saved in JPEG format
...	Arguments to be passed to graphical parameters

Details

The element FORPLOTTING contained within the calc_box() output object is required to plot an SD Box. Provide the whole plot_box() output object as the argment for datin.

Value

This function returns a plot in the graphics device.

Author(s)

Tarmo K. Remmel, Randy Bui, Ron N. Buliung

See Also

plot_sde, plot_box

Examples

```
a <- calc_sdd(points=activities)
plot_sdd(datin=a, plotnew=TRUE, plothv=FALSE, plotweightedpts=FALSE,
plotpoints=TRUE, plotcentre=TRUE, titletxt="Title",
xaxis="Easting (m)", yaxis="Northing (m)")

# plot_sdd() BY DEFAULT, TAKES AS INPUT THE RESULT PRODUCED BY calc_sdd()
```

---

plot_sde *Plot the Standard Deviation Ellipse*

Description

This function plots the SDE as an ellipse centred on the mean/weighted/user-defined centre of a set of point observations. The plot characterizes the dispersion of point observations along two orthogonal axes.

Usage

```
plot_sde(datin=NULL, plotnew=TRUE, plotSDEaxes=FALSE, plotweightedpts=FALSE,
weightedpts.col='black', weightedpts.pch=19, plotpoints=TRUE,
points.col='black', points.pch=1, plotcentre=TRUE, centre.col='black',
centre.pch=19, titletxt="Title", xaxis="Easting (m)",
yaxis="Northing (m)", sde.col='black', sde.lwd=2, jpeg=FALSE, ...)
```

Arguments

datin	Input data object; the result from calc_sde()
plotnew	Boolean: Set to TRUE to create a new plot. Set to FALSE to overlay current plot.
plotSDEaxes	Boolean: Set to TRUE if the orthogonal axes through the centroid are to be plotted
plotweightedpts	Boolean: Set to TRUE if the weighted point observations are to be plotted
weightedpts.col	Specify a colour for the weighted point observations
weightedpts.pch	Specify a plotting symbol for the weighted point observations
plotpoints	Boolean: Set to TRUE if the point observations are to be plotted
points.col	Specify a colour for the point observations
points.pch	Specify a plotting symbol for the point observations
plotcentre	Boolean: Set to TRUE if the mean/weighted/user-defined centre is to be plotted
centre.col	Specify a colour for the centre
centre.pch	Specify a plotting symbol for the centre
titletxt	A string to indicate the title on the plot
xaxis	A string to label the x-axis of the plot
yaxis	A string to label the y-axis of the plot
sde.col	Specify a line colour for the SDE circle
sde.lwd	Specify a line width for the SDE circle
jpeg	Boolean: Set to TRUE if the plot should be saved in JPEG format
...	Arguments to be passed to graphical parameters

Details

The element FORPLOTTING contained within the calc_box() output object is required to plot an SD Box. Provide the whole plot_box() output object as the argment for datin.

Value

This function returns a plot in the graphics device.

Author(s)

Tarmo K. Remmel, Randy Bui, Ron N. Buliung

See Also

plot_sdd, plot_box

Examples

```
a <- calc_sde(points=activities)
plot_sde(datin=a, plotnew=TRUE, plotSDEaxes=FALSE, plotweightedpts=FALSE,
plotpoints=TRUE, plotcentre=TRUE, titletxt="Title",
xaxis="Easting (m)", yaxis="Northing (m)")

# plot_sde() BY DEFAULT, TAKES AS INPUT THE RESULT PRODUCED BY calc_sde()
```

---

sin_d *Compute sine with angle given in degrees*

Description

Provides the functionality of sin, but for input angles measured in degrees (not radians).

Usage

```
sin_d(theta = 0)
```

Arguments

theta	A numeric angular measurement in degrees from north.

Details

Since the R default is to compute trigonometric functions on angular measurements stored in radians, this simple function performs the conversion from degrees, reducing the need to do so a priori, outside the function.

Value

Returns a numeric value for the sine of the specified angular measurement

Note

To reduce the need for unit conversions prior to calling trigonometric functions, this function accepts input in angular degrees rather than radians. Depending on data, this function may be preferred to the existing version requiring input in angular radians.

Author(s)

Tarmo K. Remmel

See Also

cos_d, tan_d, asin_d, acos_d, atan_d

Examples

```
sin_d(theta = 90)
```

tan_d *Compute tangent with angle given in degrees*

Description

Provides the functionality of tan, but for input angles measured in degrees (not radians).

Usage

```
tan_d(theta = 0)
```

Arguments

theta	A numeric angular measurement in degrees from north.

Details

Since the R default is to compute trigonometric functions on angular measurements stored in radians, this simple function performs the conversion from degrees, reducing the need to do so a priori, outside the function.

Value

Returns a numeric value for the tangent of the specified angular measurement

Note

To reduce the need for unit conversions prior to calling trigonometric functions, this function accepts input in angular degrees rather than radians. Depending on data, this function may be preferred to the existing version requiring input in angular radians.

Author(s)

Tarmo K. Remmel

See Also

sin_d, cos_d, asin_d, acos_d, atan_d

Examples

```
tan_d(theta = 45)
```

`wts` *Weights vector*

Description

This is a single column vector for weighting the importance of point locations.

Usage

```
data(wts)
```

Format

A single column vector of numeric values.

Details

The weights can be specified according to any reasonable criteria specified by the user

Source

This demonstration data has been manufactured for illustrative purposes only.

Examples

```
data(wts)
str(wts)
plot(wts)
```

Index