

Package ‘aspace’

April 17, 2009

Type Package

Title A collection of functions for estimating centrographic statistics and computational geometries from spatial point patterns

Version 1.2

Date 2007-05-08

Author Tarmo K. Remmel, Ron N. Buliung

Maintainer Ron N. Buliung <ron.buliung@utoronto.ca>

Description A collection of functions for computing centrographic statistics (e.g., standard distance, standard deviation ellipse), and minimum convex polygons (MCP) for observations taken at point locations. A tool is also provided for converting geometric objects associated with the centrographic statistics, and MCPs into ESRI Shapefiles.

License GPL (>= 2)

Depends R (>= 2.0.1), adehabitat, ade4, gpcplib, sp, maptools

LazyData yes

Repository CRAN

Date/Publication 2007-05-08 19:28:11

R topics documented:

aspace-package	2
acos_d	3
activities	4
asin_d	4
as_radians	5
atan_d	6
calc_mcp	7
calc_sdd	8
calc_sde	10

centre	13
cos_d	13
distances	14
ellipse3	15
makeshapes	17
sin_d	18
tan_d	19
wts	20

Index	21
--------------	-----------

aspace-package	<i>A collection of functions for estimating centrophraphic statistics and computational geometries from spatial point patterns</i>
----------------	--

Description

A collection of functions for computing centrophraphic statistics (e.g., standard distance, standard deviation ellipse), and minimum convex polygons (MCP) from point data. A tool is also provided for converting geometric objects associated with the centrophraphic statistics, and MCPs into ESRI Shapefiles. This library was initially conceived to aid in the analysis of spatial patterns of travel behaviour (see Buliung and Kanaroglou, 2006).

Details

Package:	aspace
Type:	Package
Version:	1.2
Date:	2007-05-08
License:	GPL (>= 2.0)

Author(s)

Tarmo K. Rimmel, Ron N. Buliung

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 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.

Mohr, C.O. 1947. Table of equivalent populations of North American mammals. American Midland Naturalist, 37: 223-249.

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. Rimmel

See Also

[sin_d](#), [cos_d](#), [tan_d](#), [asin_d](#), [atan_d](#)

Examples

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

`activities`*Demo Data: Coordinates of 10 specified activity locations*

Description

This is a simple two-column data frame (or matrix) containing x,y coordinates for a series of activity point locations. These are meant to represent locations physically contacted by an individual during a specific time interval. Demonstration data mimic UTM coordinates such that the first column contains Easting (x), and the second Northing (y) coordinates for unique destinations (one destination per row).

Usage

```
data(activities)
```

Format

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

co11 A numeric vector of x-coordinates

co12 A numeric vector of y-coordinates

Details

The coordinates of the activities 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)
```

`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 measurements 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_mcp

Compute and plot a Minimum Convex Polygon (MCP)

Description

The geographical extent of a set of points on a Cartesian plane can be described using a Minimum Convex Polygon (MCP). The MCP is the minimum area polygon containing a set of point locations.

Usage

```
calc_mcp(id=1, destmat = activities, filename="MCP_Output.txt", verbose = FALSE, po
```

Arguments

id	Provide a unique integer to identify this MCP from others you may construct with other data points
destmat	Two-column matrix or data frame of point coordinates
filename	A character name for an ASCII output file
verbose	Boolean: set to TRUE if extended processing feedback is wanted
pct	Integer $0 \leq \text{pct} \leq 100$, the percentage of the MCP for which area is provided
plot	Boolean: the MCP will be plotted if set to TRUE
plotdest	Boolean: all points will be plotted if set to TRUE

Details

This function is most powerful when used repetitively within a loop to compute the MCP for subsets of points stored in a large data table.

Value

The result is a LIST

MCP.area	The area of the MCP in hectares
MCP.pct	The desired percentage of the MCP for which area is computed
MCP.coords	A matrix containing MCP vertices. Each row represents a unique point, the first column contains x-coordinates, and the second, y-coordinates

Note

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 parameter to ensure that each MCP has a unique identifier. The output ASCII coordinate file can be further processed using the makeshapes function to generate an ESRI Shapefile for MCP polygons.

Author(s)

Tarmo K. Remmel

References

Builds upon MCP functions available in the adehabitat package

See Also

[mcp](#), [calc_sdd](#), [calc_sde](#), [makeshapes](#)

Examples

```
plot.new()
calc_mcp(id=1, destmat = activities, filename="MCP_Output.txt", verbose = FALSE, pct = 100)
```

calc_sdd

Calculate and plot the Standard Distance Deviation (Standard Distance), and a Standard Deviation Box

Description

The dispersion of a set of points on a Cartesian plane can be described using the Standard Distance Deviation (SDD) or Standard Distance. For the purpose of geographic visualization, the SDD is typically portrayed as a circle with radius SDD centered on the mean center of a set of point observations. The orthogonal dispersion of a set of points can also be described using the standard deviation of the x- and y-coordinates of a set of point observations. The standard deviation of x- and y-coordinates can be geographically visualized using a box, with the edges set, respectively, to the standard deviation of the x- and y-coordinates.

Usage

```
calc_sdd(id = 1, filename = "SDD_Output.txt", centre.xy = centre, calccentre = TRUE)
```

Arguments

id	A unique integer to identify the shape
filename	A string indicating the ASCII textfile where shape coordinates will be written
centre.xy	A vector of length 2, containing the x- and y-coordinates of the SDD centroid
calccentre	Boolean: Set to TRUE if the mean center is to be calculated
useWMC	Boolean: Set to TRUE if the mean center is to be computed with weighted coordinates
weightpoints	Boolean: Set to TRUE if the point observations are to be weighted
weights	Weights applied to point observations
destmat	A 2-column matrix or data frame containing point coordinates
verbose	Boolean: Set to TRUE if extensive feedback is desired on the standard output
plot	Boolean: Set to TRUE if the SDD is to be plotted
plothv	Boolean: Set to TRUE if the orthogonal N-S, E-W axes are to be plotted through the center
plotdest	Boolean: Set to TRUE if the point observations are to be plotted
plotcenter	Boolean: Set to TRUE if the mean center is to be plotted
box	Boolean: Set to TRUE if the standard deviation of the x- and y-coordinates are to be plotted as a box

Details

This function is most powerful when used repetitively within a loop to compute the SDD for subsets of points stored in a large table.

Value

The result is a list of terms:

id	Identifier for the SDD shape - it should be unique
calccentre	True if mean centre is computed
Orig.x	Original x-coordinate of center before mean center calculation

Orig.y	Original y-coordinate of center before mean center calculation
CENTRE.x	Actual, used x-coordinate of centre
CENTRE.y	Actual, used y-coordinate of centre
SD.x	Standard deviation of the x-coordinates
SD.y	Standard deviation of the y-coordinates
SDD.radius	SDD value, radius of the SDD
Box.area	Area of the box formed by the standard deviation of the x- and y-coordinates
SDD.area	Area of the SDD circle
useWMC	Boolean: TRUE if the weighted mean center is used
WeightPoints	Boolean: TRUE if point observations are weighted

Note

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 parameter to ensure that each SDD has a unique identifier. The output ASCII coordinate file can be further processed using the `makeshapes` function to generate an ESRI Shapefile for SDD polygons.

Author(s)

Tarmo K. Remmel, Ron Buliung

See Also

[ellipse3](#), [calc_mcp](#), [calc_sde](#), [makeshapes](#)

Examples

```
calc_sdd(id = 1, filename = "SDD_Output.txt", centre.xy = centre, calcentre = TRUE, useWMC
```

calc_sde

Calculate and plot a Standard Deviation Ellipse

Description

This function computes the Standard Deviation Ellipse (SDE) for a set of points. The SDE is a centrophraphic measure used to characterize the dispersion of point observations along two orthogonal axes. The SDE also captures directional bias in a spatial point pattern and will be oriented in the direction of maximum dispersion. The function provides options for weighting observations, and centring the ellipse on a user defined point, mean centre, or weighted mean centre of the input point locations. Output includes plotting of the SDE and an ASCII (text) file containing elliptical coordinates.

Usage

```
calc_sde(id = 1, filename = "SDE_Output.txt", calcentre = FALSE, useWMC = FALSE, c
```

Arguments

<code>id</code>	An identifier for a given SDE. When running <code>calc.sde</code> in a loop for multiple sets of points, increment <code>id</code> such that it is unique!
<code>filename</code>	The name of an ASCII (text) file where SDE coordinates will be written
<code>calccentre</code>	Boolean: Set to TRUE if the mean center of the points is to be used as the SDE centroid
<code>useWMC</code>	Boolean: Set to TRUE if the weighted mean center is to be used as the SDE centroid
<code>centre.xy</code>	A numeric vector of length 2, containing specified x- and y-coordinates to use as the centroid
<code>destmat</code>	A numeric matrix or data frame with two columns. The first column represents x-coordinates, the second, y-coordinates. Each row corresponds to a single point location.
<code>title.txt</code>	A string to use as the title on the plot
<code>verbose</code>	Boolean: Set to TRUE if extensive standard output feedback is desired
<code>plot</code>	Boolean: Set to TRUE if the SDE is to be plotted
<code>calcSDxy</code>	Boolean: Set to TRUE if the standard deviations in the orthogonal (x and y) directions are to be computed
<code>plotSDEaxes</code>	Boolean: Set to TRUE if the orthogonal axes through the centroid are to be plotted
<code>plotdest</code>	Boolean: Set to TRUE if input point observations are to be plotted along with the SDE
<code>plotcentroid</code>	Boolean: Set to TRUE if the centroid is to be plotted along with the SDE
<code>plotSDxy</code>	Boolean: Set to TRUE if the orthogonal standard deviation box should be plotted along with the SDE
<code>weightpoints</code>	Boolean: Set to TRUE if the point observations are to be weighted
<code>weights</code>	A matrix or data frame of weights for the points
<code>jpeg</code>	Boolean: Set to TRUE if the plot should be saved in JPEG format

Details

This function is most powerful when used repetitively within a loop to compute the SDE for subsets of points stored in a large data table.

Value

The returned result is a list:

<code>CALCCENTRE</code>	Boolean: Indicates whether the mean centre was computed
<code>WeightPoints</code>	Boolean: Indicates whether the points were weighted
<code>UseWMC</code>	Boolean: Indicates whether the weighted mean centre is to be used
<code>Orig.x</code>	Original x-coordinate of centre
<code>Orig.y</code>	Original y-coordinate of centre

CENTRE.x	x-coordinate after computation of mean centre
CENTRE.y	y-coordinate after computation of mean 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
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
WMC.x	Weighted mean center x-coordinate
WMC.y	Weighted mean center y-coordinate

Note

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 parameter to ensure that each SDE has a unique identifier. The output ASCII coordinate file can be further processed using the `makeshapes` function to generate an ESRI Shapefile for SDE polygons.

Author(s)

Tarmo K. Remmel, Ron 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_sdd](#), [calc_mcp](#), [makeshapes](#)

Examples

```
calc_sde(id = 1, filename = "SDE_Output.txt", calcentre = FALSE, useWMC = FALSE, centre.xy
```

`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)
```

`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 = centre, destmat = activities, 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. Rimmel

Examples

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

ellipse3

Ellipse drawing tool

Description

A convenient tool for plotting circles or ellipses with the functionality of rotation about an angle theta (in radians). The function can also be used to capture coordinates of the perimeter of the shape for further usage outside the function.

Usage

```
ellipse3(cx, cy, rx, ry, theta = 0, yaxis = TRUE, pointsonly = FALSE, fill = FALSE,
```

Arguments

cx	x-coordinate of ellipse center
cy	y-coordinate of ellipse center
rx	radius along x-axis
ry	radius along y-axis
theta	rotation angle in radians from north
yaxis	Deprecated. This parameter adjusts the size correct in the y- or x-axis, as plotting is generally not square. Therefore, use <code>par(pty="s")</code> to eliminate the need for this parameter.

<code>pointsonly</code>	Boolean: If TRUE, ellipse will not be plotted, but rather the coordinates of the perimeter are returned in a list object for further use. The <code>calc.sde</code> function utilizes these coordinates to build a textfile of coordinates which the <code>makeshapes</code> function uses to build ESRI Shapefiles.
<code>fill</code>	Boolean: If TRUE, the plotted ellipse will be shaded in
<code>...</code>	Any additional parameters suitable for plotting

Details

This function plots an ellipse with center (`cx`, `cy`). The rotation angle in radians (from north) is given as `theta`. Note that a circle can be obtained by `rx=ry`, in which case `theta` is not very useful. Additional parameters (e.g., colour and fill density) can be provided as indicated by the `'...'` in the function call.

Value

This function returns a plot of an ellipse when `pointsonly = FALSE`. When `pointsonly = TRUE`, the result is a list of `x,y` coordinates

<code>x</code>	A numeric vector of x-coordinates
<code>y</code>	A numeric vector of y-coordinates

Note

This function is an an adjustment by Brad Biggerstaff (CDC) and Tarmo K. Remmel of the function `circle()` written by John Wallace (University of Washington) and obtained from the S-News listserv.

Author(s)

Tarmo K. Remmel

See Also

[calc_sde](#), [as_radians](#)

Examples

```
plot.new()
plot(10,10, type="n")
ellipse3(cx = 10, cy = 8, rx = 2, ry = 1, theta = as_radians(45), yaxis = TRUE, pointsonly
```

`makeshapes`*Builder of ESRI Shapefiles*

Description

This function is a basic ESRI Shapefile builder. The SDD, SDE, and MCP functions included in this library produce ASCII output files that represent the input required for this function. However, any similarly formatted file will also work. This function provides capabilities for converting the centographic and geometric summary measures of geographical extent and dispersion into GIS databases for further cartographic rendering and analysis.

Usage

```
makeshapes(asciiname = "SDD_Output.txt", headerskip = 0, outname = "Test", verbose
```

Arguments

<code>asciiname</code>	The name of the ASCII file containing coordinate info. input to the shape building procedure
<code>headerskip</code>	An integer to indicate how many lines of the ASCII file to skip at the top if a header has been added
<code>outname</code>	The name of the output Shapefile. Do not use spaces or illegal filename characters.
<code>verbose</code>	Boolean: Set to TRUE if extended feedback to the standard output is required

Details

The level of detail recorded in the Shapefile will be determined by the weed tolerance of points defining the shapes in the input ASCII file.

Value

The result is an ESRI format Shapefile containing 3 files (.shp, .dbf, .shx). The base filename will be that specified by the parameter `outname`.

Note

Currently, the unique identifier is the only attribute that separates polygon objects within the Shapefile. Once the Shapefile is built, a collection of other attributes can be joined to the database via the unique identifier.

Author(s)

Tarmo K. Remmel with significant help from Rick Reeves - NCEAS

See Also

[calc_sdd](#), [calc_sde](#), [calc_mcp](#)

Examples

```
calc_sdd(weights=NULL)
makeshapes(asciiiname="SDD_Output.txt", headerskip=0, outname="Test", verbose=TRUE)
```

`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. Rimmel

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 activity 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

*Topic **arith**

- aspace-package, 2
- calc_mcp, 7
- calc_sdd, 8
- calc_sde, 10
- distances, 14

*Topic **array**

- acos_d, 3
- as_radians, 5
- asin_d, 4
- atan_d, 6
- cos_d, 13
- sin_d, 18
- tan_d, 19

*Topic **datasets**

- activities, 4
- centre, 13
- wts, 20

*Topic **graphs**

- ellipse3, 15

*Topic **manip**

- makeshapes, 17

acos_d, 3, 5–7, 14, 18, 19
activities, 4
as_radians, 5, 16
asin_d, 3, 4, 6, 7, 14, 18, 19
aspace (*aspace-package*), 2
aspace-package, 2
atan_d, 3, 5, 6, 6, 14, 18, 19

calc_mcp, 7, 10, 12, 17
calc_sdd, 8, 8, 12, 17
calc_sde, 8, 10, 10, 16, 17
centre, 13
cos_d, 3, 5–7, 13, 18, 19

distances, 14

ellipse3, 10, 15

makeshapes, 8, 10, 12, 17

mcp, 8

sin_d, 3, 5–7, 14, 18, 19

tan_d, 3, 5–7, 14, 18, 19

wts, 20