The aspace Package

May 30, 2006

Type Package

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

Version 0.1

Date 2006-05-25

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 (Version 2 or later)

Depends R (>= 2.0.1), adehabitat, ade4, gpclib

LazyData yes

R topics documented:

acos.d ................................. 2
activities .................................. 3
asin.d .................................. 3
aspace-package .............................. 4
as.radians ................................. 5
atan.d .................................. 6
calc.mcp .................................. 7
calc.sdd .................................. 8
calc.sde .................................. 10
centre ................................... 12
cos.d ................................. 13
distances .................................. 14
ellipse3 ................................... 15
makeshapes .................................. 16
sin.d .................................. 17
tan.d .................................. 18
wts .................................. 19
acos.d

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: 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.

<table>
<thead>
<tr>
<th>col1</th>
<th>col2</th>
</tr>
</thead>
<tbody>
<tr>
<td>A numeric vector of x-coordinates</td>
<td>A numeric vector of y-coordinates</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>data(activities)</td>
</tr>
<tr>
<td>str(activities)</td>
</tr>
<tr>
<td>plot(activities)</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>asin.d(theta = 0)</td>
</tr>
</tbody>
</table>

Arguments

<table>
<thead>
<tr>
<th>Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>theta</td>
</tr>
</tbody>
</table>
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

Author(s)
Tarmo K. Remmel, Ron N. Buliung

References


as.radians (theta = 0)

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(theta = 90)

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, pct = 100, plot = TRUE, plotdest = TRUE)
```

**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 <= pct <=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
calc.sdd

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, plot = TRUE, plotdest = TRUE)

calc.sdd  

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, useWMC = FALSE, weightpoints = ..., weights = ..., destmat = activities, verbose = FALSE, plot = TRUE, plothv = TRUE, plotdest = TRUE, plotcenter = TRUE, box = 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, calcCentre = TRUE, useWMC = FALSE, weightpoints = FALSE, destmat = activities, verbose = FALSE, plot = TRUE, plothv = TRUE, plotdest = TRUE, plotcenter = TRUE, box = TRUE)
```
**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 centrographic measure used to characterize the dispersion of point observations along two orthogonal axes. The SDE also captures directional bias in aspatial 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", calccentre = FALSE, useWMC = FALSE, 
          centre.xy = centre, destmat = destmat, titletxt = titletxt, verbose = FALSE, 
          plot = TRUE, calcSDxy = TRUE, plotSDEaxes = TRUE, plotdest = TRUE, 
          plotcentroid = TRUE, plotSDxy = TRUE, weightpoints = FALSE, weights = wts, 
          jpeg = FALSE)
```

**Arguments**

- **id**
  - An identifier for a given SDE. When running calc.sde in a loop for multiple sets of points, increment id such that it is unique!

- **filename**
  - The name of an ASCII (text) file where SDE coordinates will be written

- **calccentre**
  - Boolean: Set to TRUE if the mean center of the points is to be used as the SDE centroid

- **useWMC**
  - Boolean: Set to TRUE if the weighted mean center is to be used as the SDE centroid

- **centre.xy**
  - A numeric vector of length 2, containing specified x- and y-coordinates to use as the centroid

- **destmat**
  - 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.

- **titletxt**
  - A string to use as the title on the plot

- **verbose**
  - Boolean: Set to TRUE if extensive standard output feedback is desired

- **plot**
  - Boolean: Set to TRUE if the SDE is to be plotted

- **calcSDxy**
  - Boolean: Set to TRUE if the standard deviations in the orthogonal (x and y) directions are to be computed

- **plotSDEaxes**
  - Boolean: Set to TRUE if the orthogonal axes through the centroid are to be plotted

- **plotdest**
  - Boolean: Set to TRUE if input point observations are to be plotted along with the SDE

- **plotcentroid**
  - Boolean: Set to TRUE if the centroid is to be plotted along with the SDE

- **plotSDxy**
  - Boolean: Set to TRUE if the orthogonal standard deviation box should be plotted along with the SDE

- **weightpoints**
  - Boolean: Set to TRUE if the point observations are to be weighted

- **weights**
  - A matrix or data frame of weights for the points

- **jpeg**
  - 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:

- CALCCENTRE (Boolean): Indicates whether the mean centre was computed
- WeightPoints (Boolean): Indicates whether the points were weighted
- UseWMC (Boolean): Indicates whether the weighted mean center is to be used
- Orig.x (Original x-coordinate of center)
- Orig.y (Original y-coordinate of center)
- 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
centre

References


See Also
calc.sdd, calc.mcp, makeshapes

Examples
calc.sde(id = 1, filename = "SDE_Output.txt", calccentre = FALSE, useWMC = FALSE, centre.xy = centre, destmat = ... = TRUE, plotdest = TRUE, plotcentroid = TRUE, plotSDxy = TRUE, weightpoints = FALSE, weights = wts, jpeg = FALSE)

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

\[ \text{cos.d}(\theta = 0) \]

Arguments

\begin{itemize}
  \item \text{theta} \quad \text{A numeric angular measurement in degrees from north.}
\end{itemize}

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

\text{sind, tan.d, asin.d, acos.d, atan.d}

Examples

\[ \text{cos.d}(\theta = 90) \]
**Description**

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

**Usage**

```r
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. Remmel

**Examples**

```r
data(centre)
data(activities)
distances(centre.xy = centre, destmat = activities, verbose=FALSE)
```
ellipse3

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

```r
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 `par(pty="s")` to eliminate the need for this parameter.
- `pointsonly`: 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 calc.sde function utilizes these coordinates to build a textfile of coordinates which the makeshapes function uses to build ESRI Shapefiles.
- `fill`: Boolean: If TRUE, the plotted ellipse will be shaded in
- `...`: Any additional parameters suitable for plotting

Details

This function plots an ellipse with center (cx, cy). The rotation angle in radians (form 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.

- `x`: A numeric vector of x-coordinates
- `y`: 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 = FALSE, fill = FALSE, col=6)

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 centrographic 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 = TRUE)

Arguments

ascaliname The name of the ASCII file containing coordinate info. input to the shape building procedure
headerskip An integer to indicate how many lines of the ASCII file to skip at the top if a header has been added
outname The name of the output Shapefile. Do not use spaces or illegal filename characters.
verbose 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(ascliname="SDD_Output.txt", headerskip=0, outname="Test", verbose=TRUE)

______________________________
sin.d
______________________________

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
**tan.d**

**See Also**

`cos.d`, `tan.d`, `asin.d`, `acos.d`, `atan.d`

**Examples**

```r
sin.d(theta = 90)
```

tan.d(theta = 0)

**Description**

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

**Usage**

```r
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**

```r
tan.d(theta = 45)
```
**Weights vector**

**Description**

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

**Usage**

```r
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**

```r
data(wts)
str(wts)
plot(wts)
```
Index

※Topic **arith**
  aspace-package, 4
calc.mcp, 6
calc.sdd, 7
calc.sde, 9
distances, 13

※Topic **array**
  acos.d, 1
  as.radians, 4
  asin.d, 3
  atan.d, 5
  cos.d, 12
  sin.d, 16
  tan.d, 17

※Topic **datasets**
  activities, 2
  centre, 11
  wts, 18

※Topic **graphs**
  ellipse3, 14

※Topic **manip**
  makeshapes, 15

acos.d, 1, 3, 5, 6, 12, 17
activities, 2
asin.d, 4, 15
asin.d, 2, 3, 5, 6, 12, 17
aspace (aspace-package), 4
aspace-package, 4
atan.d, 2, 3, 5, 5, 12, 17
calc.mcp, 6, 9, 11, 16
calc.sdd, 7, 7, 11, 16
calc.sde, 7, 9, 9, 15, 16
centre, 11
cos.d, 2, 3, 5, 6, 12, 17
distances, 13
ellipse3, 9, 14
makeshapes, 7, 9, 11, 15
mcp, 7
sin.d, 2, 3, 5, 6, 12, 16, 17

tan.d, 2, 3, 5, 6, 12, 17, 17
wts, 18