

# Package ‘deldir’

January 2, 2012

**Version** 0.0-16

**Date** 2011-11-04

**Title** Delaunay Triangulation and Dirichlet (Voronoi) Tessellation.

**Author** Rolf Turner <r.turner@auckland.ac.nz>

**Maintainer** Rolf Turner <r.turner@auckland.ac.nz>

**Depends** R (>= 0.99)

**Description** Calculates the Delaunay triangulation and the Dirichlet or Voronoi tessellation (with respect to the entire plane) of a planar point set.

**License** GPL (>= 2)

**URL** <http://www.math.unb.ca/~rolf/>

**Repository** CRAN

**Date/Publication** 2011-11-04 07:32:52

## R topics documented:

|                            |    |
|----------------------------|----|
| deldir . . . . .           | 2  |
| plot.deldir . . . . .      | 5  |
| plot.tile.list . . . . .   | 7  |
| plot.triang.list . . . . . | 9  |
| tile.centroids . . . . .   | 10 |
| tile.list . . . . .        | 11 |
| triang.list . . . . .      | 13 |

|              |           |
|--------------|-----------|
| <b>Index</b> | <b>14</b> |
|--------------|-----------|

---

deldir

*Delaunay triangulation and Dirichlet tessellation*


---

### Description

This function computes the Delaunay triangulation (and hence the Dirichlet or Voronoi tessellation) of a planar point set according to the second (iterative) algorithm of Lee and Schacter — see REFERENCES. The triangulation is made to be with respect to the whole plane by suspending it from so-called ideal points  $(-\text{Inf}, -\text{Inf})$ ,  $(\text{Inf}, -\text{Inf})$ ,  $(\text{Inf}, \text{Inf})$ , and  $(-\text{Inf}, \text{Inf})$ . The triangulation is also enclosed in a finite rectangular window. A set of dummy points may be added, in various ways, to the set of data points being triangulated.

### Usage

```
deldir(x, y, dpl=NULL, rw=NULL, eps=1e-09, sort=TRUE, plotit=FALSE,
       digits=6, ...)
```

### Arguments

- |     |   |
|-----|---|
| x,y | The coordinates of the point set being triangulated. These can be given by two arguments x and y which are vectors or by a single argument x which is a list with components x and y.   |
| dpl | <p>A list describing the structure of the dummy points to be added to the data being triangulated. The addition of these dummy points is effected by the auxilliary function <code>dumpts()</code>. The list may have components:</p> <ul style="list-style-type: none"> <li>• <code>ndx</code>: The x-dimension of a rectangular grid; if either <code>ndx</code> or <code>ndy</code> is null, no grid is constructed.</li> <li>• <code>ndy</code>: The y-dimension of the aforementioned rectangular grid.</li> <li>• <code>nrad</code>: The number of radii or “spokes”, emanating from each data point, along which dummy points are to be added.</li> <li>• <code>nper</code>: The number of dummy points per spoke.</li> <li>• <code>fctr</code>: A factor determining the length of each spoke; each spoke is of length equal to <code>fctr</code> times the mean nearest neighbour distance of the data. (This distance is calculated by the auxilliary function <code>mnnd()</code>.)</li> <li>• <code>x</code>: A vector of x-coordinates of “ad hoc” dummy points</li> <li>• <code>y</code>: A vector of the corresponding y-coordinates of “ad hoc” dummy points</li> </ul> |
| rw  | The coordinates of the corners of the rectangular window enclosing the triangulation, in the order (xmin, xmax, ymin, ymax). Any data points (including dummy points) outside this window are discarded. If this argument is omitted, it defaults to values given by the range of the data, plus and minus 10 percent.  |
| eps | A value of epsilon used in testing whether a quantity is zero, mainly in the context of whether points are collinear. If anomalous errors arise, it is possible that these may averted by adjusting the value of <code>eps</code> upward or downward.   |

|        |  |
|--------|--|
| sort   | Logical argument; if TRUE (the default) the data (including dummy points) are sorted into a sequence of “bins” prior to triangulation; this makes the algorithm slightly more efficient. Normally one would set sort equal to FALSE only if one wished to observe some of the fine detail of the way in which adding a point to a data set affected the triangulation, and therefore wished to make sure that the point in question was added last. Essentially this argument would get used only in a de-bugging process. |
| plotit | Logical argument; if TRUE a plot is produced. The nature of the plot may be controlled by using the ... argument to pass appropriate arguments to plot.deldir(). Without “further instruction” a plot of the points being triangulated and of both the triangulation and the tessellation is produced;   |
| digits | The number of decimal places to which all numeric values in the returned list should be rounded. Defaults to 6.  |
| ...    | Auxilliary arguments add, wlines, wpoints, number, nex, col, lty, pch, xlim, and ylim (and possibly other plotting parameters) may be passed to plot.deldir through ... if plotit=TRUE.  |

## Details

This package is a (straightforward) adaptation of the Splus library section “delaunay” to R. That library section is an implementation of the Lee-Schacter algorithm, which was originally written as a stand-alone Fortran program in 1987/88 by Rolf Turner, while with the Division of Mathematics and Statistics, CSIRO, Sydney, Australia. It was re-written as an Splus function (using dynamically loaded Fortran code), by Rolf Turner while visiting the University of Western Australia, May, 1995.

Further revisions were made December 1996. The author gratefully acknowledges the contributions, assistance, and guidance of Mark Berman, of D.M.S., CSIRO, in collaboration with whom this project was originally undertaken. The author also acknowledges much useful advice from Adrian Baddeley, formerly of D.M.S., CSIRO (now of CMIS, CSIRO and Adjunct Professor of Statistics at the University of Western Australia). Daryl Tingley of the Department of Mathematics and Statistics, University of New Brunswick provided some helpful insight. Special thanks are extended to Alan Johnson, of the Alaska Fisheries Science Centre, who supplied two data sets which were extremely valuable in tracking down some errors in the code.

Don MacQueen, of Lawrence Livermore National Lab, wrote an Splus driver function for the old stand-alone version of this software. That driver, which was available on Statlib, is now deprecated in favour of the current package “delaunay” package. Don also collaborated in the preparation of that package.

See the ChangeLog for information about further revisions and bug-fixes.

## Value

A list (of class `deldir`), invisible if `plotit=TRUE`, with components:

|        |   |
|--------|---|
| delsgs | a matrix with 6 columns. The first 4 entries of each row are the coordinates of the points joined by an edge of a Delaunay triangle, in the order (x1,y1,x2,y2). The last two entries are the indices of the two points which are joined. |
| dirsgs | a data frame with 8 columns. The first 4 entries of each row are the coordinates of the endpoints of one the edges of a Dirichlet tile, in the order (x1,y1,x2,y2).   |

The fifth and sixth entries are the indices of the two points, in the set being triangulated, which are separated by that edge. The seventh and eighth entries are logical values. The seventh indicates whether the first endpoint of the corresponding edge of a Dirichlet tile is a boundary point (a point on the boundary of the rectangular window). Likewise for the eighth entry and the second endpoint of the edge.

|                       |  |
|-----------------------|--|
| summary               | a matrix with 9 columns and $n.data + n.dumrows$ (see below). The rows correspond to the points in the set being triangulated. The column names are <code>x</code> (the x-coordinate of the point), <code>y</code> (the y-coordinate), <code>n.tri</code> (the number of Delaunay triangles emanating from the point), <code>del.area</code> (1/3 of the total area of all the Delaunay triangles emanating from the point), <code>del.wts</code> (the corresponding entry of the <code>del.area</code> column divided by the sum of this column); <code>n.tside</code> (the number of sides — within the rectangular window — of the Dirichlet tile surrounding the point), <code>nbpt</code> (the number of points in which the Dirichlet tile intersects the boundary of the rectangular window), <code>dir.area</code> (the area of the Dirichlet tile surrounding the point), and <code>dir.wts</code> (the corresponding entry of the <code>dir.area</code> column divided by the sum of this column).<br>Note that the factor of 1/3 associated with the <code>del.area</code> column arises because each triangle occurs three times — once for each corner. |
| <code>n.data</code>   | the number of real (as opposed to dummy) points in the set which was triangulated, with any duplicate points eliminated. The first <code>n.data</code> rows of <code>summary</code> correspond to real points.   |
| <code>n.dum</code>    | the number of dummy points which were added to the set being triangulated, with any duplicate points (including any which duplicate real points) eliminated. The last <code>n.dum</code> rows of <code>summary</code> correspond to dummy points.  |
| <code>del.area</code> | the area of the convex hull of the set of points being triangulated, as formed by summing the <code>del.area</code> column of <code>summary</code> .   |
| <code>dir.area</code> | the area of the rectangular window enclosing the points being triangulated, as formed by summing the <code>dir.area</code> column of <code>summary</code> .  |
| <code>rw</code>       | the specification of the corners of the rectangular window enclosing the data, in the order ( <code>xmin</code> , <code>xmax</code> , <code>ymin</code> , <code>ymax</code> ).   |

### Side Effects

If `plotit==TRUE` a plot of the triangulation and/or tessellation is produced or added to an existing plot.

### Warning

The process for determining if points are duplicated changed between versions 0.1-9 and 0.1-10. Previously there was an argument `frac` for this function, which defaulted to 0.0001. Points were deemed to be duplicates if the difference in x-coordinates was less than `frac` times the in y-coordinates was less than `frac` times the height of `rw`. This process has been changed to one which uses `duplicated()` on the data frame whose columns are `x` and `y`.

As a result it may happen that points which were previously eliminated as duplicates will no longer be eliminated.

**Note**

If  $ndx \geq 2$  and  $ndy \geq 2$ , then the rectangular window IS the convex hull, and so the values of `del.area` and `dir.area` (if the latter is not NULL) are identical.

**Author(s)**

Rolf Turner <r.turner@auckland.ac.nz> <http://www.math.unb.ca/~rolf>

**References**

Lee, D. T., and Schacter, B. J. Two algorithms for constructing a Delaunay triangulation, Int. J. Computer and Information Sciences, Vol. 9, No. 3, 1980, pp. 219 – 242.

Ahuja, N. and Schacter, B. J. (1983). Pattern Models. New York: Wiley.

**See Also**

plot.deldir

**Examples**

```
x <- c(2.3,3.0,7.0,1.0,3.0,8.0)
y <- c(2.3,3.0,2.0,5.0,8.0,9.0)
try <- deldir(x,y,list(ndx=2,ndy=2),c(0,10,0,10))
# Puts dummy points at the corners of the rectangular
# window, i.e. at (0,0), (10,0), (10,10), and (0,10)
## Not run:
try <- deldir(x,y,list(ndx=2,ndy=2),c(0,10,0,10),plot=TRUE,wl='tr')

## End(Not run)
# Plots the triangulation which was created (but not the tessellation).
```

---

plot.deldir

*Plot objects produced by deldir*

---

**Description**

This is a method for plot.

**Usage**

```
## S3 method for class 'deldir'
plot(x,add=FALSE,wlines=c('both','triang','tess'),
      wpoints=c('both','real','dummy','none'),
      number=FALSE,cex=1,nex=1,col=NULL,lty=NULL,
      pch=NULL,xlim=NULL,ylim=NULL,xlab='x',ylab='y',
      showrect=FALSE,...)
```

**Arguments**

|          |   |
|----------|---|
| x        | An object of class "deldir" as constructed by the function deldir.  |
| add      | logical argument; should the plot be added to an existing plot?   |
| wlines   | "which lines?". I.e. should the Delaunay triangulation be plotted (wlines='triang'), should the Dirichlet tessellation be plotted (wlines='tess'), or should both be plotted (wlines='both', the default) ?   |
| wpoints  | "which points?". I.e. should the real points be plotted (wpoints='real'), should the dummy points be plotted (wpoints='dummy'), should both be plotted (wpoints='both', the default) or should no points be plotted (wpoints='none')?   |
| number   | Logical argument, defaulting to FALSE; if TRUE then the points plotted will be labelled with their index numbers (corresponding to the row numbers of the matrix "summary" in the output of deldir).  |
| cex      | The value of the character expansion argument cex to be used with the plotting symbols for plotting the points.   |
| nex      | The value of the character expansion argument cex to be used by the text function when numbering the points with their indices. Used only if number=TRUE.   |
| col      | the colour numbers for plotting the triangulation, the tessellation, the data points, the dummy points, and the point numbers, in that order; defaults to c(1,1,1,1,1). If fewer than five numbers are given, they are recycled. (If more than five numbers are given, the redundant ones are ignored.) |
| lty      | the line type numbers for plotting the triangulation and the tessellation, in that order; defaults to 1:2. If only one value is given it is repeated. (If more than two numbers are given, the redundant ones are ignored.)   |
| pch      | the plotting symbols for plotting the data points and the dummy points, in that order; may be either integer or character; defaults to 1:2. If only one value is given it is repeated. (If more than two values are given, the redundant ones are ignored.)   |
| xlim     | the limits on the x-axis. Defaults to rw[1:2] where rw is the rectangular window specification returned by deldir().  |
| ylim     | the limits on the y-axis. Defaults to rw[3:4] where rw is the rectangular window specification returned by deldir().  |
| xlab     | label for the x-axis. Defaults to x. Ignored if add=TRUE.   |
| ylab     | label for the y-axis. Defaults to y. Ignored if add=TRUE.   |
| showrect | logical scalar; show the enclosing rectangle rw (see <a href="#">deldir()</a> ) be plotted?   |
| ...      | Further plotting parameters to be passed to plot() segments() or points(). Unlikely to be used.   |

**Details**

The points in the set being triangulated are plotted with distinguishing symbols. By default the real points are plotted as circles (pch=1) and the dummy points are plotted as triangles (pch=2).

**Side Effects**

A plot of the points being triangulated is produced or added to an existing plot. As well, the edges of the Delaunay triangles and/or of the Dirichlet tiles are plotted. By default the triangles are plotted with solid lines (lty=1) and the tiles with dotted lines (lty=2).

**Author(s)**

Rolf Turner <r.turner@auckland.ac.nz> <http://www.math.unb.ca/~rolf>

**See Also**

`deldir()`

**Examples**

```
## Not run:
try <- deldir(x,y,list(ndx=2,ndy=2),c(0,10,0,10))
plot(try)
#
deldir(x,y,list(ndx=4,ndy=4),plot=TRUE,add=TRUE,wl='te',
       col=c(1,1,2,3,4),num=TRUE)
# Plots the tessellation, but does not save the results.
try <- deldir(x,y,list(ndx=2,ndy=2),c(0,10,0,10),plot=TRUE,wl='tr',
             wp='n')
# Plots the triangulation, but not the points, and saves the
# returned structure.

## End(Not run)
```

---

plot.tile.list

*Plot Dirchlet (Voronoi) tiles*

---

**Description**

A method for plot. Plots (sequentially) the tiles associated with each point in the set being tessellated.

**Usage**

```
## S3 method for class 'tile.list'
plot(x, verbose = FALSE, close=FALSE, pch=1,
     polycol=NA, showpoints=TRUE, showrect=FALSE,
     add=FALSE, asp=1, xlab = "x", ylab = "y",
     main = "", ...)
```

**Arguments**

|            |   |
|------------|---|
| x          | A list of the tiles in a tessellation, as produced the function <code>tile.list()</code> .  |
| verbose    | Logical scalar; if TRUE the tiles are plotted one at a time (with a “Go?” prompt after each) so that the process can be watched.  |
| close      | Logical scalar; if TRUE the outer edges of of the tiles (i.e. the edges of the enclosing rectangle) are drawn. Otherwise tiles on the periphery of the tessellation are left “open”.  |
| pch        | The plotting character for plotting the points of the pattern which was tessellated. Ignored if <code>showpoints</code> is FALSE.   |
| polycol    | Optional vector of integers (or NAs); the <i>i</i> -th entry indicates with which colour to fill the <i>i</i> -th tile. Note that an NA indicates the use of no colour at all.  |
| showpoints | Logical scalar; if TRUE the points of the pattern which was tessellated are plotted.  |
| showrect   | Logical scalar; show the enclosing rectangle <code>rw</code> (see <code>deldir()</code> ) be plotted?   |
| add        | Logical scalar; should the plot of the triangles be added to an existing plot?  |
| asp        | The aspect ratio of the plot; integer scalar or NA. Set this argument equal to NA to allow the data to determine the aspect ratio and hence to make the plot occupy the complete plotting region in both x and y directions. This is inadvisable; see the <b>Warnings</b> . |
| xlab       | Label for the x-axis (used only if <code>add</code> is FALSE).  |
| ylab       | Label for the y-axis (used only if <code>add</code> is FALSE).  |
| main       | A title for the plot (used only if <code>add</code> is FALSE).  |
| ...        | Optional arguments; not used. There for consistency with the generic plot function.   |

**Value**

NULL; side effect is a plot.

**Warnings**

The default value for `verbose` was formerly TRUE; it is now FALSE.

The user is *strongly advised* not to set the value of `asp` but rather to leave `asp` equal to its default value of 1. Any other value distorts the tessellation and destroys the perpendicular appearance of lines which are indeed perpendicular. (And conversely can cause lines which are not perpendicular to appear as if they are.)

The argument `asp` is present “just because it can be”.

**Author(s)**

Rolf Turner <r.turner@auckland.ac.nz> <http://www.math.unb.ca/~rolf>

**See Also**

[tile.list\(\)](#)

**Examples**

```
x <- runif(20)
y <- runif(20)
z <- deldir(x,y,rw=c(0,1,0,1))
w <- tile.list(z)
plot(w)
ccc <- heat.colors(20) # Or topo.colors(20), or terrain.colors(20)
# or cm.colors(20), or rainbow(20).
plot(w,polycol=ccc,close=TRUE)
```

---

plot.triang.list      *Plot Delaunay triangles*

---

**Description**

A method for plot. Plots the triangles of a Delaunay triangulation of a set of points in the plane.

**Usage**

```
## S3 method for class 'triang.list'
plot(x, showrect = FALSE, add = FALSE,
      xlab = "x", ylab = "y", main = "", asp = 1, ...)
```

**Arguments**

|          |   |
|----------|---|
| x        | An object of class “triang.list” as produced by <code>triang.list()</code> .  |
| showrect | Logical scalar; show the enclosing rectangle rw (see <code>deldir()</code> ) be plotted?  |
| add      | Logical scalar; should the plot of the triangles be added to an existing plot?  |
| xlab     | Label for the x-axis.   |
| ylab     | Label for the y-axis.   |
| main     | A title for the plot (used only if add is FALSE).   |
| asp      | The aspect ratio of the plot; integer scalar or NA. Set this argument equal to NA to allow the data to determine the aspect ratio and hence to make the plot occupy the complete plotting region in both x and y directions. This is inadvisable; see the <b>Warnings</b> . |
| ...      | Arguments passed to <code>polygon()</code> which does the actual plotting of the triangles.   |

**Value**

None. This function has the side effect of producing (or adding to) a plot.

## Warnings

The user is *strongly advised* not to set the value of `asp` but rather to leave `asp` equal to its default value of 1. Any other value distorts the tessellation and destroys the perpendicular appearance of lines which are indeed perpendicular. (And conversely can cause lines which are not perpendicular to appear as if they are.)

The argument `asp` is present “just because it can be”.

## Author(s)

Rolf Turner <r.turner@auckland.ac.nz> <http://www.math.unb.ca/~rolf>

## See Also

[deldir\(\)](#) [plot.triang.list\(\)](#) [tile.list\(\)](#) [plot.tile.list\(\)](#)

## Examples

```
set.seed(42)
x <- runif(20)
y <- runif(20)
d <- deldir(x,y)
ttt <- triang.list(d)
plot(ttt,border="red",showrect=TRUE)
sss <- tile.list(d)
plot(sss)
plot(ttt,add=TRUE,border="blue")
```

---

tile.centroids

*Compute centroids of Dirichlet (Voronoi) tiles*

---

## Description

Given a list of Dirichlet tiles, as produced by `tile.list()`, produces a data frame consisting of the centroids of those tiles.

## Usage

```
tile.centroids(xxx)
```

## Arguments

`xxx` A list of the tiles (produced by `tile.list()`) in a Dirichlet tessellation of a set of planar points.

## Value

A data frame with two columns named `x` and `y`. Each row of this data frame constitutes the centroid of one of the Dirichlet tiles.

**Author(s)**

Rolf Turner <r.turner@auckland.ac.nz> <http://www.math.unb.ca/~rolf>

**References**

URL <http://en.wikipedia.org/wiki/Centroid>

**See Also**

[tile.list\(\)](#)

**Examples**

```
set.seed(42)
x <- runif(20)
y <- runif(20)
d <- deldir(x,y)
l <- tile.list(d)
g <- tile.centroids(l)
## Not run:
plot(l,close=TRUE)
points(g,pch=20,col="red")

## End(Not run)
```

---

tile.list

*Create a list of tiles in a tessellation*

---

**Description**

For each point in the set being tessellated produces a list entry describing the Dirichlet/Voronoi tile containing that point.

**Usage**

```
tile.list(object)
```

**Arguments**

object            An object of class `deldir` as produced by the function `deldir()`.

**Value**

A list with one entry for each of the points in the set being tessellated. Each entry is in turn a list with components

pt                The coordinates of the point whose tile is being described.  
x                 The x coordinates of the vertices of the tile, in anticlockwise order.

|    |  |
|----|--|
| y  | The y coordinates of the vertices of the tile, in anticlockwise order.   |
| bp | Vector of logicals indicating whether the tile vertex is a “real” vertex, or a <i>boundary point</i> , i.e. a point where the tile edge intersects the boundary of the enclosing rectangle |

### Acknowledgement

The author expresses sincere thanks to Majid Yazdani who found and pointed out a serious bug in `tile.list` in a previous version (0.0-5) of the `deldir` package.

### Warning

The set of vertices of each tile may be “incomplete”. Only vertices which lie within the enclosing rectangle, and “boundary points” are listed.

Note that the enclosing rectangle may be specified by the user in the call to `deldir()`.

In contrast to some earlier versions of `deldir`, the corners of the enclosing rectangle are now include as vertices of tiles. I.e. a tile which in fact extends beyond the rectangular window and contains a corner of that window will have that corner added to its list of vertices. Thus when the corresponding polygon is plotted, the result is the intersection of the tile with the enclosing

### Author(s)

Rolf Turner <[r.turner@auckland.ac.nz](mailto:r.turner@auckland.ac.nz)> <http://www.math.unb.ca/~rolf>

### See Also

`deldir()`, `plot.tile.list()`

### Examples

```
x <- runif(20)
y <- runif(20)
z <- deldir(x,y)
w <- tile.list(z)

z <- deldir(x,y,rw=c(0,1,0,1))
w <- tile.list(z)

z <- deldir(x,y,rw=c(0,1,0,1),dpl=list(ndx=2,ndy=2))
w <- tile.list(z)
```

---

`triang.list`*Create a list of Delaunay triangles*

---

**Description**

From an object of class “deldir” produces a list of the Delaunay triangles in the triangulation of a set of points in the plane.

**Usage**

```
triang.list(object)
```

**Arguments**

`object` An object of class “deldir” as produced by `deldir()`.

**Value**

A list each of whose components is a 3-x-2 data frame corresponding to one of the Delaunay triangles specified by “object”. The rows of each such data frame consist of the coordinates of the vertices of the corresponding Delaunay triangle.

The returned value has an attribute “rw” consisting of the enclosing rectangle of the triangulation.

**Note**

The code of this function was taken more-or-less directly from code written by Adrian Baddeley for the “delaunay()” function in the “spatstat” package.

**Author(s)**

Rolf Turner <[r.turner@auckland.ac.nz](mailto:r.turner@auckland.ac.nz)> <http://www.math.unb.ca/~rolf>

**See Also**

`deldir()` `plot.triang.list()` `tile.list()` `plot.tile.list()`

**Examples**

```
set.seed(42)
x <- runif(20)
y <- runif(20)
d <- deldir(x,y)
ttt <- triang.list(d)
```

# Index

\*Topic **hplot**

plot.deldir, 5

plot.tile.list, 7

\*Topic **spatial**

deldir, 2

plot.triang.list, 9

tile.centroids, 10

tile.list, 11

triang.list, 13

deldir, 2, 6–13

  duplicated, 4

plot.deldir, 5

plot.tile.list, 7, 10, 12, 13

plot.triang.list, 9, 10, 13

  polygon, 9

tile.centroids, 10

tile.list, 8, 10, 11, 11, 13

triang.list, 9, 13