Package 'deldir'

March 6, 2016

Version 0.1-12

Date 2016-06-03

Title Delaunay Triangulation and Dirichlet (Voronoi) Tessellation

Author Rolf Turner

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

Depends R (>= 0.99)

Suggests polyclip

Imports graphics, grDevices

Description Calculates the Delaunay triangulation and the Dirichlet or Voronoi tessellation (with respect to the entire plane) of a planar point set. Plots triangulations and tessellations in various ways. Clips tessellations to sub-windows. Calculates perimeters of tessellations.

License GPL (>= 2)

NeedsCompilation yes

Repository CRAN

Date/Publication 2016-03-06 09:37:13

R topics documented:

`deldir` *Delaunay triangulation and Dirichlet tessellation*

Description

This function computes the Delaunay triangulation (and hence the Dirichlet or Voronoi tesselation) 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 (-Inf,-Inf), (Inf,-Inf) (Inf,Inf), and (-Inf,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, z=NULL, zdum=NULL, suppressMsge=FALSE, ...)
```

Arguments

`x,y` These arguments specify the coordinates of the point set being triangulated or tessellated. These can be given by two separate arguments `x` and `y` which are vectors or by a single argument `x` which is either a data frame or a generic list, possibly one of class `ppp`. (See package `spatstat`.)

If `x` is a data frame then the `x` coordinates of the points to be triangulated or tessellated are taken to be the column of this data frame which is named "x" if there is one, else the first column of the data frame which is not named either "y" or "z". The `y` coordinates are taken to be the column of this data frame which is named "y" if there is one. If there is no column named "y" but there are columns named "x" and "z" then the `y` coordinates are taken to be the first "other" column. If there no columns named either "x" or "y", then the `x` coordinates are taken to be the first column not named "z" and the `y` coordinates are taken to be the *second* column not named "z".

If there is a column named "z" and if the argument `z` (see below) is NULL, then this the column named "z" is taken to be the value of `z`.

If `x` is a list (but not a data frame) then it must have components named `x` and `y`, and possibly a component named `z`. The `x` and `y` components give the `x` and `y` coordinates respectively of the points to be triangulated or tessellated. If `x` is *not* of class `ppp`, if it has a component `z` and if argument `z` is `NULL`, then the `z` argument is set equal to this component `z`. If `x` *is* of class "ppp", if the argument `z` is `NULL`, if `x` is "marked" (see package `spatstat`) and if the marks of `x` are a vector or a factor (as opposed to a data frame) then the `z` argument is set equal to these marks. In this case `x` should *not* have a component `z`, and at any rate such a component would be ignored.

`dpl` 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 auxiliary function dumpts(). The list may have components:

- `ndx`: The x-dimension of a rectangular grid; if either ndx or ndy is null, no grid is constructed.
- `ndy`: The y-dimension of the aforementioned rectangular grid.
- `nrad`: The number of radii or "spokes", emanating from each data point, along which dummy points are to be added.
- `nper`: The number of dummy points per spoke.
- `fctr`: A numeric "multiplicative factor" determining the length of each spoke; each spoke is of length equal to fctr times the mean nearest neighbour distance of the data. (This distance is calculated by the auxiliary function mnnd().)
- `x`: A vector of x-coordinates of "ad hoc" dummy points
- `y`: A vector of the corresponding y-coordinates of "ad hoc" dummy points

`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 eps 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.

`z` An optional vector of "auxiliary" values or "weights" associated with the respective points. (**NOTE:** These "weights" are values associated with the points and hence with the tiles of the tessellation produced. They **DO NOT** affect the tessellation, i.e. the tessellation produced is the same as is it would be if there were no weights. The `deldir` package **DOES NOT** do weighted tessellation. The so-called weights in fact need not be numeric.)

If `z` is left `NULL` then it is taken to be the third column of `x`, if `x` is a data frame or to be the `z` component of `x` if `x` is a generic list. If `z` is left `NULL` and if `x` is of class "ppp" and is "marked" (see package `spatstat`) and if in addition the marks are atomic (i.e. *not* a data frame) then `z` is taken to be the marks of `x`.

zdum	Values of z to be associated with any dummy points that are created. See **Warnings**.
suppressMsge	Logical scalar indicating whether a message (alerting the user to changes from previous versions of deldir) should be suppressed.
...	Auxiliary 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 Professor of Statistics at Curtin University). 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 data frame 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 10 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, in the columns named ind1 and ind2, are the indices of the two points, in the set being triangulated, which are separated by that edge. The seventh and eighth entries, in the columns named bp1 and bp2 are logical values. The entry in column bp1 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 entry in column bp2 and the second endpoint of the edge.

The nineth and tenth entries, in columns named thirdv1 and thirdv2 are the indices of the third vertex of the Delaunay triangle whose circumcentre constitutes the corresponding vertex of the edge under consideration. (The other two vertices are indexed by the entries of columns ind1 and ind2.)

The entries of columns thirdv1 and thirdv2 may (also) take the values $-1, -2, -3$, and -4. This will be the case if the circumcentre in question lies outside of the rectangular window rw. In these circumstances the corresponding endpoint of the tile edge is the intersection of the line joining the two circumcentres with the boundary of rw, and the numeric value of the entry of column "vi3" indicates which side. The numbering follows the convention for numbering the sides of a plot region in R: 1 for the bottom side, 2 for the left side, 3 for the top side and 4 for the right side.

Note that the entry in column thirdv1 will be negative if and only if the corresponding entry in column bp1 is TRUE. Similarly for columns thirdv2 and bp2.

summary — a data frame with 9, 10 or 11 columns and n.data + n.dum rows (see below). The rows correspond to the points in the set being triangulated. The columns are:

- x (the x-coordinate of the point)
- y (the y-coordinate of the point)
- pt.type (a character vector with entries "data" and "dummy"; present only if n.dum > 0)
- z (the auxiliary values or weights; present only if these were specified)
- n.tri (the number of Delaunay triangles emanating from the point)
- del.area (1/3 of the total area of all the Delaunay triangles emanating from the point)
- del.wts (the corresponding entry of the del.area column divided by the sum of this column)
- n.tside (the number of sides — within the rectangular window — of the Dirichlet tile surrounding the point)
- nbpt (the number of points in which the Dirichlet tile intersects the boundary of the rectangular window)
- dir.area (the area of the Dirichlet tile surrounding the point)
- dir.wts (the corresponding entry of the dir.area column divided by the sum of this column).

Note that the factor of 1/3 associated with the del.area column arises because each triangle occurs three times — once for each corner.

n.data — the number of real (as opposed to dummy) points in the set which was triangulated, with any duplicate points eliminated. The first n.data rows of summary correspond to real points.

n.dum — 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 n.dum rows of summary correspond to dummy points.

del.area — the area of the convex hull of the set of points being triangulated, as formed by summing the del.area column of summary.

dir.area	the area of the rectangular window enclosing the points being triangulated, as formed by summing the dir.area column of summary.
rw	the specification of the corners of the rectangular window enclosing the data, in the order (xmin, xmax, ymin, ymax).
ind.orig	A vector of the indices of the points (x,y) in the set of coordinates initially supplied (as data points or as dummy points) to deldir() before duplicate points (if any) were removed. These indices are used by triang.list().

Remark:

If ndx >= 2 and ndy >= 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.

Side Effects

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

Warnings

1. 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 width of rw and 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. (And possibly vice-versa.)

2. The components delsgs and summary of the value returned by deldir() are now *data frames* rather than matrices. The component summary was changed to allow the "auxiliary" values z to be of arbitrary mode (i.e. not necessarily numeric). The component delsgs was then changed for consistency. Note that the other "matrix-like" component dirsgs has been a data frame since time immemorial.

   A message alerting the user to the foregoing two items is printed out the first time that deldir() is called with suppressMsge=FALSE in a given session. In succeeding calls to deldir() in the same session, no message is printed. (I.e. the "alerting" message is printed *at most once* in any given session.)

   The "alerting" message is *not* produced via the warning() function, so suppressWarnings() will *not* suppress its appearance. To effect such suppression (necessary only on the first call to deldir() in a given session) one must set the suppressMsge argument of deldir equal to TRUE.

3. If any dummy points are created, and if a vector z, of "auxiliary" values or "weights" associated with the points being triangulated, is supplied, then it is up to the user to supply the corresponding auxiliary values or weights associated with the dummy points. These values should be supplied as zdum. If zdum is not supplied then the auxiliary values or weights associated with the dummy points are all taken to be missing values (i.e. NA).

Author(s)

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

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(), tile.list(), triang.list()

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)
# An "alerting note" is printed.; let deldir() choose the
# rectangular window.
dxy1 <- deldir(x,y)

# User chooses the rectangular window.
dxy2 <- deldir(x,y,rw=c(0,10,0,10))

# Put dummy points at the corners of the rectangular
# window, i.e. at (0,0), (10,0), (10,10), and (0,10)
dxy3 <- deldir(x,y,dpl=list(ndx=2,ndy=2),rw=c(0,10,0,10))

# Plot the triangulation created (but not the tesselation).
## Not run:
dxy2 <- deldir(x,y,rw=c(0,10,0,10),plot=TRUE,wl='tr')

## End(Not run)

# Auxiliary values associated with points; 4 dummy points to be
# added so 4 dummy "z-values" provided.
z    <- c(1.63,0.79,2.84,1.56,0.22,1.07)
zdum <- rep(42,4)
dxy4 <- deldir(x,y,dpl=list(ndx=2,ndy=2),rw=c(0,10,0,10),z=z,zdum=zdum)
```

---

divchain *Dividing chain; generic.*

Description

Generic function for creating the "dividing chain" of a Dirchlet tesselation. The tessellation must have been created from a set of points having associated categorical "weights". The dividing chain consists of those edges of Dirichlet tiles which separate points having different values of the given weights.

Usage

```
divchain(x, ...)
```

Arguments

x	Either an object specifying coordinates or an object of class "deldir".
...	Arguements to be passed to the appropriate method for this generic function.

Details

If x is a (numeric) vector it will be taken to be the "x" coordinates of the points being tessellated. In this case the ...\ arguments must contain a vector y specifying the "y" coordinates. The ...\ arguments must also contain a *factor* z specifying the relevant "weights" argument. The argument x may also be a data frame or list from which the coordinates and the weights will be extracted. See divchain.default() for details.

If x is an object of class class "deldir" then it must have been created with an appropriate (factor) "weights" argument, otherwise an error is given.

Value

An object of class "divchain". See divchain.deldir() for details.

Note

This function was created in response to a question asked on stackoverflow.com by a user named "Dan".

Author(s)

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

See Also

divchain.default() divchain.deldir() deldir() plot.divchain()

Examples

```
set.seed(42)
x   <- runif(50)
y   <- runif(50)
z   <- factor(kmeans(cbind(x,y),centers=4)$cluster)
dc1 <- divchain(x,y,z,rw=c(0,1,0,1))
dxy <- deldir(x,y,z=z,rw=c(0,1,0,1))
dc2 <- divchain(dxy)
```

divchain.default *Dividing chain; default method.*

Description

Create the "dividing chain" of the Dirchlet tesselation of a given set of points having distinguishing (categorical) "weights". This dividing chain consists of those edges of Dirichlet tiles which separate points having different values of the given weights.

Usage

```
  ## Default S3 method:
divchain(x, y, z, ...)
```

Arguments

x,y	These provide the coordinates of the set of points being tesselated. Argument x may be a data frame or a list, in particular one of class ppp. (See the spatstat package.) For a full description see the discussion of these arguments in the help for deldir().
z	A *factor* specifying "auxiliary" values or "weights" If this argument is left NULL then it is extracted, if possible, from the components of x. See deldir() for further details.
...	Other arguments to be passed to deldir.

Value

An object of class divchain. See divchain.deldir() for details.

Note

This function was created in response to a question asked on stackoverflow.com by a user named "Dan".

Author(s)

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

See Also

divchain.deldir() deldir() plot.divchain()

Examples

```
set.seed(42)
x    <- runif(50)
y    <- runif(50)
z    <- factor(kmeans(cbind(x,y),centers=4)$cluster)
dcxy <- divchain(x,y,z,rw=c(0,1,0,1))
```

`divchain.deldir` *Dividing chain; "deldir" method.*

Description

Create the "dividing chain" of a Dirchlet tesselation. The tessellation must have been created from a set of points having associated categorical "weights". The dividing chain consists of those edges of Dirichlet tiles which separate points having different values of the given weights.

Usage

```
   ## S3 method for class 'deldir'
divchain(x, ...)
```

Arguments

`x`	An object of class "deldir". This object must have been created in such a way that the points of the set being tessellated have associate categorical "weights". That is, `deldir()` must have been called with a *factor* valued `z` argument or the `x` argument to `deldir()` must have had an appropriate component which could be taken to be `z`.
`...`	Not used.

Value

An object of class "divchain" consisting of a data frame with columns named "x0", "y0", "x1","y1", "v01", "v02", "v03", "v11", "v12" and "v13".

The columns named "x0" and "y0" consist of the coordinates of one endpoint of an edge of a Dirichlet tile and the columns named "x1" and "y1" consist of the coordinates of the other endpoint.

The columns named "vij", i = 0, 1, j = 1, 2, 3, consist of the indices of the vertices of the Delaunay triangles whose circumcentres constitute the respective endpoints of the corresponding edge of a Dirichlet tile. The entries of column "vi3" may (also) take the values $-1, -2, -3$, and -4. This will be the case if the circumcentre in question lay outside of the rectangular window `rw` (see `deldir()`) enclosing the points being tessellated. In these circumstances the corresponding endpoint of the tile edge is the intersection of the line joining the two circumcentres with the boundary of `rw`, and the numeric value of the entry of column "vi3" indicates which side. The numbering follows the convention for numbering the sides of a plot region in `R`: 1 for the bottom side, 2 for the left side, 3 for the top side and 4 for the right side.

Note that the triple of vertices uniquely identify the endpoint of the tile edge.

The object has an attribute `rw` which is equal to the specification of the rectangular window within which the class "deldir" object `x` was constructed. (See `deldir()`.)

Note

This function was created in response to a question asked on `stackoverflow.com` by a user named "Dan".

Author(s)

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

See Also

divchain.default() deldir() plot.divchain()

Examples

```
set.seed(42)
x   <- runif(50)
y   <- runif(50)
z   <- factor(kmeans(cbind(x,y),centers=4)$cluster)
dxy <- deldir(x,y,z=z,rw=c(0,1,0,1))
dc  <- divchain(dxy)
```

---

duplicatedxy *Determine duplicated points.*

Description

Find which points among a given set are duplicates of others.

Usage

```
duplicatedxy(x, y)
```

Arguments

x	Either a vector of x coordinates of a set of (two dimensional) points, or a list (or data frame) with columns x and y giving the coordinates of a set of such points.
y	A vector of y coordinates of a set of (two dimensional) points. Ignored if x is a list or data frame.

Details

Often it is of interest to associate each Dirichlet tile in a tessellation of a planar point set with the point determining the tile. This becomes problematic if there are *duplicate* points in the set being tessellated/triangulated. Duplicated points are automatically eliminated "internally" by deldir(). The association between tiles and the indices of the original set of points is now preserved by the component ind.orig of the object returned by deldir(). However confusion could still arise.

If it is of interest to associate Dirichlet tiles with the points determining them, then it is better to proceed by eliminating duplicate points to start with. This function (duplicatedxy()) provides a convenient way of doing so.

Value

A logical vector of length equal to the (original) number of points being considered, with entries TRUE if the corresponding point is a duplicate of a point with a smaller index, and FALSE otherwise.

Warning

Which indices will be considered to be indices of duplicated points (i.e. get TRUE values) will of course depend on the order in which the points are presented.

Note

The real work is done by the base **R** function duplicated().

Author(s)

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

See Also

duplicated(), deldir()

Examples

```
set.seed(42)
xy  <- data.frame(x=runif(20),y=runif(20))
# Lots of duplicated points.
xy  <- rbind(xy,xy[sample(1:20,20,TRUE),])
# Scramble.
ii  <- sample(1:40,40)
x   <- xy$x[ii]
y   <- xy$y[ii]
# Unduplicate!
iii <- !duplicatedxy(x,y)
xu  <- x[iii]
yu  <- y[iii]
# The i-th tile is determined by (xu[i],yu[i]):
dxy <- deldir(xu,yu)
```

---

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 tesselation, 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 tesselation, 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 deldir()) 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>

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 tesselation, 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.divchain` *Plot a dividing chain.*

Description

Plot the dividing chain of a Dirchlet tesselation. The tessellation must have been created from a set of points having associated categorical "weights". The dividing chain consists of those edges of Dirichlet tiles which separate points having different values of the given weights.

Usage

```
   ## S3 method for class 'divchain'
plot(x, add = FALSE, ...)
```

Arguments

x	An object of class "divchain". See divchain.deldir() for details.
add	Logical scalar. It add=TRUE the plot of the dividing chain is added to an existing plot.
...	Graphical parameters such as main, xlab, col.main, col.lab. In particular if bty is supplied (as a value other than n) a "box" will be drawn around the plot that is formed when add=FALSE. Also a non-standard graphical parameter boxcol may be supplied which will be taken to be the colour with which the box is drawn. If a col argument is supplied, this determines the colour for plotting the segments constituting the dividing chain.

Value

None.

Note

This function was created in response to a question asked on stackoverflow.com by a user named "Dan".

Author(s)

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

See Also

divchain() divchain.default() divchain.deldir() deldir()

Examples

```
set.seed(42)
x   <- runif(50)
y   <- runif(50)
z   <- factor(kmeans(cbind(x,y),centers=4)$cluster)
dc  <- divchain(x,y,z,rw=c(0,1,0,1))
plot(dc,lwd=2,col="blue",bty="o")
```

`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,
                         fillcol = getCol(x,warn=warn),col.pts = NULL,
                         border=NULL, showpoints = TRUE, add = FALSE,
                         asp = 1, clipp = NULL, xlab = "x", ylab = "y",
                         main = "", warn = FALSE, ...)
```

Arguments

`x`	A list of the tiles in a tessellation, as produced the function `tile.list()`.
`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 which are contained in the enclosing rectangle) are drawn. Otherwise tiles on the periphery of the tessellation are left "open".
`pch`	The plotting character (or vector of plotting characters) with which to plot the points of the pattern which was tessellated. Ignored if `showpoints` is `FALSE`.
`fillcol`	Optional vector (possibly of length 1, i.e. a scalar) whose entries can be interpreted as colours by `col2rgb()`. The i-th entry indicates with which colour to fill the i-th tile. Note that an `NA` entry indicates the use of no colour at all. This argument will be replicated to have length equal to the number of tiles.
`col.pts`	Optional vector like unto `fillcol` whose entries can be interpreted as colours by `col2rgb()`. The i-th entry indicates with which colour to plot the i-th point. This argument will be replicated to have length equal to the number of tiles. Ignored if `showpoints` is `FALSE`.
`border`	A *scalar* indicating the colour with which to plot the tile boundaries. Defaults to black unless all of the fill colours specified by `fillcol` are black, in which case it defaults to white. If `length(border) > 1` then an error is given.
`showpoints`	Logical scalar; if `TRUE` the points of the pattern which was tesselated are plotted.
`add`	Logical scalar; should the plot of the tiles 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 **Warnings**.

`clipp` An object specifying a polygon to which the tessellation being plotted should be clipped. It should consist either of:

- a list containing two components x and y giving the coordinates of the vertices of a single polygon. The last vertex should not repeat the first vertex. Or:
- a list of list(x,y) structures giving the coordinates of the vertices of several polygons.

If this argument is provided then the plot of the tessellation is "clipped" to the polygon specified by `clipp`.

`xlab` Label for the x-axis (used only if `add` is `FALSE`).

`ylab` Label for the y-axis (used only if `add` is `FALSE`).

`main` A title for the plot (used only if `add` is `FALSE`).

`warn` Logical scalar passed to the internal function `getCol()`. Should a warning be issued if the z components of the entries of x cannot all be interpreted as colours. See **Details**.

`...` Optional arguments; not used. There for consistency with the generic `plot` function.

Value

NULL; side effect is a plot.

Warnings

- The behaviour of this function with respect to "clipping" has changed substantially since the previous release of `deldir`, i.e. 1.1-0. The argument `clipwin` has been re-named `clipp` ("p" for "polygon"). Clipping is now effected via the new package `polyclip`. The `spatstat` package is no longer used. The argument `use.gpclib` has been eliminated, since `gpclib` (which used to be called upon by `spatstat` has been superceded by `polyclip` which has an unrestrictive license.
- As of release 0.1-1 of the `deldir` package, the argument `fillcol` to this function *replaces* the old argument `polycol`, but behaves somewhat differently.
- The argument `showrect` which was present in versions of this function prior to release 0.1-1 has been eliminated. It was redundant.
- As of release 0.1-1 the `col.pts` argument *might* behave somewhat differently from how it behaved in the past.
- The arguments `border`, `clipp`, and `warn` are new as of release 0.1-1.
- Users, unless they *really* understand what they are doing and why they are doing it, are *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 tesselation 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.)

Notes

- If `clipp` is not `NULL` and `showpoints` is `TRUE` then it is possible that some of the points "shown" will not fall inside any of the plotted tiles. (This will happen if the parts of the tiles in which they fall have been "clipped" out.) If a tile is clipped out *completely* then the point which determines that tile is *not* plotted irrespective of the value of `showpoints`.
- The new behaviour in respect of the colours with which to fill the plotted tiles, and the argument `clipp` were added at the request of Chris Triggs.
- The argument `asp` was added at the request of Zubin Dowlaty.

Author(s)

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

See Also

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

Examples

```
set.seed(42)
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,fillcol=ccc,close=TRUE)
if(require(polyclip)) {
   CP <- list(x=c(0.49,0.35,0.15,0.20,0.35,0.42,
                      0.43,0.62,0.46,0.63,0.82,0.79),
                    y=c(0.78,0.86,0.79,0.54,0.58,0.70,
                      0.51,0.46,0.31,0.20,0.37,0.54))
   plot(w,clipp=CP,showpoints=FALSE,fillcol=topo.colors(20))
}
```

---

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 triang.list().
showrect	Logical scalar; show the enclosing rectangle rw (see deldir()) 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 **Warnings**.
...	Arguments passed to polygon() 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 tesselation 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 was added at the request of Zubin Dowlaty.

Author(s)

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

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 Dirchlet (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 consitutes the centroid of one of the Dirichlet tiles.

Author(s)

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

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 tesselated. Each entry is in turn a list with components

`pt`	The coordinates of the point whose tile is being described.
`ptType`	The "type" of the `pt`, either "data" or "dummy". Present only if any dummy points were specified in the call to `deldir()`.
`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 *boundary point*, i.e. a point where the tile edge intersects the boundary of the enclosing rectangle.
`z`	The "auxiliary value" or "weight" associated with the `pt`; present only if such values were supplied in the call to `deldir()`.

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 the corresponding polygon is the intersection of the tile with the enclosing rectangle.

Author(s)

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

See Also

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

---

tilePerim *Calculate tile perimeters.*

Description

Calculates the perimeters of all of the Dirichlet (Voronoi) tiles in a tessellation of a set of planar points. Also calculates the sum and the mean of these perimeters.

Usage

```
tilePerim(object,inclbdry=TRUE)
```

Arguments

object	An object of class tile.list (as produced by tile.list() specifying the Dirichlet (Voronoi) tiles in a tessellation of a set of planar points.
inclbdry	Logical scalar. Should boundary segments (edges of tiles at least one of whose endpoints lies on the enclosing rectangle rw (see deldir()) be included in the perimeter?

Value

A list with components

perimeters	A vector consisting of the values of the perimeters of the Dirichlet tiles in the tessellation.
totalPerim	The sum of perimeters.
meanPerim	The mean of perimeters.

Note

Function added at the request of Haozhe Zhang.

Author(s)

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

See Also

tile.list(), plot.tile.list()

Examples

```
x <- runif(20)
y <- runif(20)
z <- deldir(x,y,rw=c(0,1,0,1))
w <- tile.list(z)
p1 <- tilePerim(w)
p0 <- tilePerim(w,inclbdry=FALSE)
p1$totalPerim - p0$totalPerim # Get 4 = the perimeter of rw.
ss <- apply(as.matrix(z$dirsgs[,1:4]),1,
            function(x){(x[1]-x[3])^2 + (x[2]-x[4])^2})
2*sum(sqrt(ss)) - p0$totalPerim # Get 0; in tilePerim() each interior
                               # edge is counted twice.
```

---

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 3×3, 3×4 or 3×5 data frame corresponding to one of the Delaunay triangles specified by "object". The rows of each such data frame correspond to the vertices of the corresponding Delaunay triangle. The columns are:

- ptNum (the point number of the vertex in the original list of points being triangulated)

- ptType (the type of the vertex; "data" or "dummy"; present only if there were any dummy points specified)
- x (the x-coordinate of the vertex)
- y (the x-coordinate of the vertex)
- z (the "auxiliary value" or "weight" z associated with the vertex; present only if such values were supplied in the call to deldir())

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>

See Also

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

Examples

```
set.seed(42)
x   <- runif(20)
y   <- runif(20)
z   <- sample(1:100,20)
d   <- deldir(x,y,z=z)
ttt <- triang.list(d)
```

---

triMat *Produce matrix of triangle vertex indices.*

Description

Lists the indices of the vertices of each Delaunay triangle in the triangulation of a planar point set. The indices are listed (in increasing numeric order) as the rows of an $n \times 3$ matrix where n is the number of Delaunay triangles in the triangulation.

Usage

```
triMat(object)
```

Arguments

object	An object of class deldir (as produced by the funtion deldir()) specifying the Delaunay triangulation and Dirichlet (Voronoi) tesselation of a planar point set.

Details

This function was suggested by Robin Hankin of the School of Mathematical and Computing Sciences at Auckland University of Technology.

Value

An $n \times 3$ matrix where n is the number of Delaunay triangles in the triangulation specified by `object`. The i^{th} row consists of the indices (in the original list of points being triangulated) of vertices of the i^{th} Delaunay triangle. The indices are listed in increasing numeric order in each row.

Author(s)

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

See Also

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

Examples

```
set.seed(42)
x    <- runif(10)
y    <- runif(10)
ddxy <- deldir(x,y)
M    <- triMat(ddxy)
```

Index