Package ‘Apollonius’

December 13, 2023

Title 2D Apollonius Graphs

Version 1.0.1

Description Computation of the Apollonius diagram of given 2D points and its dual the Apollonius graph, also known as the additively weighted Voronoi diagram, and which is a generalization of the classical Voronoi diagram. For references, see the bibliography in the CGAL documentation at <https://doc.cgal.org/latest/Apollonius_graph_2/citelist.html>.

License GPL-3

URL https://github.com/stla/Apollonius

BugReports https://github.com/stla/Apollonius/issues

Imports abind, colorsGen, graphics, grDevices, gyro (>= 1.3.0), plotrix, Polychrome, Rcpp, stats

LinkingTo BH, Rcpp, RcppCGAL, RcppEigen

Encoding UTF-8

RoxygenNote 7.2.3

SystemRequirements C++17, gmp, mpfr

NeedsCompilation yes

Author Stéphane Laurent [aut, cre]

Maintainer Stéphane Laurent <laurent_step@outlook.fr>

Repository CRAN

Date/Publication 2023-12-13 17:20:10 UTC

R topics documented:

Apollonius ................................................................. 2
plotApolloniusGraph .................................................. 3

Index 5
Apollonius

Description

Computation of the Apollonius diagram and the Apollonius graph of some weighted 2D points. The Apollonius graph is the dual of the Apollonius diagram. It is also called the additively weighted Voronoi diagram.

Usage

Apollonius(sites, radii, tmax = 30, nsegs = 100L, nrays = 300L)

Arguments

- **sites**: the 2D points, a numeric matrix with two columns (one point per row)
- **radii**: the weights, a numeric vector of length equal to the number of points (i.e. the number of rows of `sites`)
- **tmax**: a positive number passed to `gyroray`, controlling the length of the infinite edges (i.e. the hyperbolic rays) of the Apollonius graph
- **nsegs**: a positive integer, the desired number of points of each finite edge of the Apollonius graph
- **nrays**: a positive integer, the desired number of points of each infinite edge of the Apollonius graph

Details

See the CGAL documentation.

Value

A list with two fields `diagram` and `graph`. The `diagram` field is a list providing the sites and the faces of the Apollonius diagram. The `graph` field is a list providing the sites and the edges of the Apollonius graph.

Examples

```r
library(Apollonius)
sites <- rbind(
  c(0, 0),
  c(4, 1),
  c(2, 4),
  c(7, 4),
  c(8, 0),
  c(5, -2),
  c(-4, 4),
  c(-2, -1),
)```
plotApolloniusGraph

Description
Plot an Apollonius graph.

Usage
plotApolloniusGraph(
apo,
  limits = NULL,
  circles = TRUE,
  fill = TRUE,
  centers = TRUE,
  colors = "distinct",
  distinctArgs = list(seedcolors = c("ff0000", "00ff00", "0000ff")),
  randomArgs = list(hue = "random", luminosity = "dark"),
  ...
)

Arguments
apo an output of Apollonius
plotApolloniusGraph

- **limits**: either NULL or a vector of length two passed to the arguments `xlim` and `ylim` of `plot`; if NULL, automatic limits are calculated
- **circles**: Boolean, whether to plot the original sites as circles with the given radii
- **fill**: Boolean, whether to fill the circles if `circles=TRUE` or to plot only their border
- **centers**: when `circles=TRUE` and `fill=FALSE`, whether to plot the centers of the circles
- **colors**: a character string controlling the colors of the sites; "random" to get multiple colors with `randomColor`, "distinct" to get multiple colors with `createPalette`, or a color name or a hexadecimal color code
- **distinctArgs**: if `colors = "distinct"`, a list of arguments passed to `createPalette`
- **randomArgs**: if `colors = "random"`, a list of arguments passed to `randomColor`
- **...**: arguments passed to `plot`, such as `xlab` and `ylab`

**Value**

No returned value, called for plotting.

**Examples**

```r
library(Apollonius)
sites <- rbind(
  c(0, 0),
  c(4, 1),
  c(2, 4),
  c(7, 4),
  c(8, 0),
  c(5, -2),
  c(-4, 4),
  c(-2, -1),
  c(11, 4),
  c(11, 0)
)
radii <- c(1, 1.5, 1.25, 2, 1.75, 0.5, 0.4, 0.6, 0.7, 0.3)
apo <- Apollonius(sites, radii)
opar <- par(mar = c(3, 3, 1, 1))
plotApolloniusGraph(
  apo, fill = FALSE, colors = "random", xlab = NA, ylab = NA
)
par(opar)
```
Index

Apollonius, 2, 3
createPalette, 4
gyroray, 2
plot, 4
plotApolloniusGraph, 3
randomColor, 4