Package 'polylabelr'

May 9, 2026

Title Find the Pole of Inaccessibility (Visual Center) of a Polygon

Version 1.0.0

Description A wrapper around the C++ library 'polylabel' from 'Mapbox', providing an efficient routine for finding the approximate pole of inaccessibility of a polygon, which usually serves as an excellent candidate for labeling of a polygon.

License MIT + file LICENSE

Copyright see file COPYRIGHTS

Encoding UTF-8

URL https://github.com/jolars/polylabelr, https://jolars.github.io/polylabelr/

BugReports https://github.com/jolars/polylabelr/issues

Depends R (>= 3.3.0)

LinkingTo Rcpp

Imports Rcpp

RoxygenNote 7.3.3

Suggests covr, testthat, spelling, sf

Language en-US

NeedsCompilation yes

Author Johan Larsson [aut, cre] (ORCID: <https://orcid.org/0000-0002-4029-5945>), Kent Johnson [ctb], Mapbox [cph] (polylabel, variant, and geometry libraries)

Maintainer Johan Larsson <johanlarsson@outlook.com>

Repository CRAN

Date/Publication 2026-01-19 06:50:02 UTC

Contents

`poi` *Pole of Inaccessibility (Visual Center) of a Polygon*

Description

This function computes and returns the approximate pole of inaccessibility for a polygon using a quadtree-based algorithm developed by the people from Mapbox.

Usage

```
poi(x, y = NULL, precision = 1)
```

Arguments

`x`	a vector of x coordinates or a matrix or data.frame of x and y coordinates, a list of components x and y, a time series (see `grDevices::xy.coords()` for details), or a simple features object from package `sf`.
`y`	a vector of y coordinates. Only needs to be provided if `x` is vector.
`precision`	the precision to use when computing the center

Details

If there are any `NA` values in the input, they will be treated as separators for multiple paths (rings) of the polygon, mimicking the behavior of `graphics::polypath()`.

Value

A list with items

`x`	x coordinate of the center
`y`	y coordinate of the center
`dist`	distance to the enclosing polygon

Source

Garcia-Castellanos & Lombardo, 2007. Poles of inaccessibility: A calculation algorithm for the remotest places on earth Scottish Geographical Journal, Volume 123, 3, 227-233. doi:10.1080/14702540801897809

`https://github.com/mapbox/polylabel`

`https://blog.mapbox.com/a-new-algorithm-for-finding-a-visual-center-of-a-polygon-7c77e6492fbc`

See Also

grDevices::xy.coords(), graphics::polypath()

Examples

```
plot_path <- function(x, y, ...) {
  plot.new()
  plot.window(range(x, na.rm = TRUE), range(y, na.rm = TRUE))
  polypath(x, y, ...)
}

x <- c(5, 10, 10, 5, 5, 6, 6, 7, 7, 6, 8, 8, 9, 9, 8)
y <- c(5, 5, 10, 10, 5, 6, 7, 7, 6, 6, 8, 9, 9, 8, 8)

plot_path(x, y, col = "grey", border = NA)

points(poi(x, y))

## Not run:
# Find visual centers for North Carolina counties
library(sf)
nc <- st_read(system.file("shape/nc.shp", package = "sf"))
locations <- do.call(rbind, poi(nc, precision = 0.01))
plot(st_geometry(nc))
points(locations)

## End(Not run)
```

---

poi_multi *Pole of Inaccessibility for Multi-Polygons*

Description

This function is a convenience function to be used with multi-polygons (lists of polygons). The function simply calls poi() for each polygon in the list and returns the point wit the maximum distance to the boundaries of the enclosing polygon.

Usage

```
poi_multi(polygons, ...)
```

Arguments

polygons	a list, each element containing entries x and y. See poi() for details about what these need to be.
...	Arguments passed on to poi
	x a vector of x coordinates or a matrix or data.frame of x and y coordinates, a list of components x and y, a time series (see grDevices::xy.coords() for details), or a simple features object from package sf.

`y` a vector of y coordinates. Only needs to be provided if `x` is vector.
`precision` the precision to use when computing the center

Value

A list with items

`x`	x coordinate of the center
`y`	y coordinate of the center
`dist`	distance to the enclosing polygon

See Also

`poi()`

Examples

```
p1 <- rbind(
  c(0, 0),
  c(1, 0),
  c(3, 2),
  c(2, 4),
  c(1, 4),
  c(0, 0),
  c(NA, NA),
  c(1, 1),
  c(1, 2),
  c(2, 2),
  c(1, 1)
)
p2 <- rbind(
  c(3, 0),
  c(4, 0),
  c(4, 1),
  c(3, 1),
  c(3, 0),
  c(NA, NA),
  c(3.3, 0.8),
  c(3.8, 0.8),
  c(3.8, 0.3),
  c(3.3, 0.3),
  c(3.3, 0.3)
)
p3 <- rbind(
  c(3, 3),
  c(4, 2),
  c(4, 3),
  c(3, 3)
)
mpol <- list(p1, p2, p3)

plot.new()
```

```
plot.window(c(0, 4), c(0, 4), asp = 1)
for (i in seq_along(mpol)) {
  polypath(mpol[[i]])
}

res <- poi_multi(mpol, precision = 1e-5)
points(res, pch = 19, col = "steelblue4")

res
```

Index