Package 'DonutMap'

June 8, 2026

Title Donut Maps with 'sf', 'ggplot2', and 'leaflet'

Version 0.1.0

Description Create donut charts positioned on maps from tidy data. The package provides helpers to compute donut polygon geometries, optional origin-destination flow lines, ready-to-use 'ggplot2' map layers, and interactive 'leaflet' widgets.

License MIT + file LICENSE

Encoding UTF-8

Language en-US

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Imports dplyr, ggplot2, htmltools, leaflet, rlang, scales, sf, tibble

Suggests ragg, knitr, rmarkdown, rnaturalearth, rnaturalearthdata, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

URL https://aureliennicosiaulaval.github.io/DonutMap/, https://github.com/AurelienNicosiaULaval/DonutMap

BugReports https://github.com/AurelienNicosiaULaval/DonutMap/issues

Config/Needs/website pkgdown

NeedsCompilation no

Author Aurélien Nicosia [aut, cre]

Maintainer Aurélien Nicosia <aurelien.nicosia@mat.ulaval.ca>

Repository CRAN

Date/Publication 2026-06-08 14:40:02 UTC

Contents

`donut_leaflet` *Draw an interactive donut map*

Description

`donut_leaflet()` returns a `leaflet` htmlwidget with clickable donut segments, optional origin-destination links or curved trajectories, popups, labels, a legend, and layer controls.

Usage

```
donut_leaflet(
  data,
  id,
  category,
  value,
  map = NULL,
  lon = NULL,
  lat = NULL,
  input_crs = 4326,
  crs = 3857,
  radius_range = NULL,
  inner_radius = 0.55,
  n = 96,
  colours = NULL,
  flows = NULL,
  from = NULL,
  to = NULL,
  flow_value = NULL,
  flow_group = NULL,
  flow_min = NULL,
  flow_weight_range = c(1, 8),
  flow_curvature = 0.18,
  flow_n = 30,
  flow_arrow = TRUE,
  flow_arrow_size = NULL,
  flow_colour = "grey35",
  flow_colours = NULL,
  flow_legend = TRUE,
  flow_legend_position = "topright",
```

```
  flow_opacity = 0.55,
  provider_tiles = "CartoDB.Positron",
  popup = TRUE,
  label = TRUE,
  prefer_canvas = FALSE,
  map_fill = "#f3f4f6",
  map_colour = "#ffffff",
  map_weight = 1,
  map_opacity = 0.9,
  donut_colour = "#ffffff",
  donut_weight = 1,
  donut_opacity = 0.9,
  donut_smooth_factor = 0
)
```

Arguments

`data`	A data frame or `sf` object. Each row is one category for one donut location.
`id`	Unquoted column identifying each donut location.
`category`	Unquoted column identifying donut categories.
`value`	Unquoted numeric column giving non-negative category values.
`map`	Optional `sf` object used as a background layer.
`lon, lat`	Unquoted longitude and latitude columns. Required when `data` is not an `sf` object.
`input_crs`	Coordinate reference system for `lon` and `lat`, or for an `sf` object with missing CRS. Defaults to EPSG:4326.
`crs`	Target projected CRS used to build interactive donut and trajectory geometries. Defaults to EPSG:3857, Leaflet's default display projection, so donut circles and sector separators remain visually regular on screen.
`radius_range`	Numeric vector of length 2 giving minimum and maximum donut radii in map units. If `NULL`, a range is derived from the map extent.
`inner_radius`	Numeric value in `(0, 1)` giving the inner radius as a proportion of the outer radius.
`n`	Number of points used to approximate a complete outer circle.
`colours`	Optional category colours. Use a named vector for stable category-colour matching.
`flows`	Optional data frame of origin-destination flows.
`from, to`	Unquoted columns in `flows` identifying origin and destination ids. Required when `flows` is supplied.
`flow_value`	Optional unquoted numeric column in `flows` used to scale line widths. If omitted, each flow receives value 1.
`flow_group`	Optional unquoted column in `flows` used to colour flow lines and arrowheads by group.
`flow_min`	Optional minimum flow value to draw.

`flow_weight_range`
: Numeric vector of length 2 controlling interactive flow line weights.

`flow_curvature`
: Numeric curvature for trajectory lines. Use `0` for straight lines, positive values for one bend direction, and negative values for the opposite direction.

`flow_n`
: Number of points used to approximate each curved trajectory.

`flow_arrow`
: Should interactive flow trajectories include arrowheads?

`flow_arrow_size`
: Arrowhead length in projected map units. If `NULL`, a size is derived from the donut radii.

`flow_colour`
: Flow line colour used when `flow_group` is not supplied.

`flow_colours`
: Optional colours for `flow_group`. Use a named vector for stable group-colour matching. If omitted and flow groups match donut categories, `colours` is reused.

`flow_legend`
: Should a separate flow colour legend be shown when `flow_group` is supplied?

`flow_legend_position`
: Position of the flow colour legend.

`flow_opacity`
: Flow line opacity.

`provider_tiles`
: Leaflet provider tiles. Use `NULL` to skip tile layers.

`popup`
: Should popups be attached to donut segments and flow lines?

`label`
: Should hover labels be attached to donut segments and flow lines?

`prefer_canvas`
: Should Leaflet prefer Canvas over SVG for vector rendering? The default `FALSE` gives crisper small donut separators.

`map_fill`, `map_colour`
: Background map fill and outline colours.

`map_weight`
: Background map border weight.

`map_opacity`
: Background map opacity.

`donut_colour`
: Donut segment border colour.

`donut_weight`
: Donut segment border weight.

`donut_opacity`
: Donut segment fill opacity.

`donut_smooth_factor`
: Leaflet path simplification factor for donut polygons. The default `0` preserves the donut vertices so arcs and sector separators are not simplified by Leaflet.

Value

A `leaflet` htmlwidget.

Examples

```
demo <- data.frame(
  place = rep(c("A", "B", "C"), each = 3),
  lon = rep(c(-71.35, -71.2, -71.05), each = 3),
  lat = rep(c(46.75, 46.82, 46.73), each = 3),
  category = rep(c("x", "y", "z"), times = 3),
  value = c(10, 20, 5, 5, 15, 10, 12, 4, 9)
```

```
)

donut_leaflet(demo, place, category, value, lon = lon, lat = lat)
```

---

donut_map *Draw a donut map*

Description

donut_map() returns a ggplot2 map with donut charts located on top of an optional sf background map. It can also add origin-destination links or curved trajectories between donut locations.

Usage

```
donut_map(
  data,
  id,
  category,
  value,
  map = NULL,
  lon = NULL,
  lat = NULL,
  input_crs = 4326,
  crs = NULL,
  radius_range = NULL,
  inner_radius = 0.55,
  n = 96,
  colours = NULL,
  flows = NULL,
  from = NULL,
  to = NULL,
  flow_value = NULL,
  flow_group = NULL,
  flow_colours = NULL,
  flow_min = NULL,
  flow_linewidth_range = c(0.2, 2.5),
  flow_curvature = 0.18,
  flow_n = 30,
  flow_arrow = TRUE,
  flow_arrow_length = 0.12,
  flow_colour = "grey35",
  flow_alpha = 0.45,
  map_fill = "grey96",
  map_colour = "white",
  donut_colour = "white",
  donut_linewidth = 0.15
)
```

Arguments

data	A data frame or sf object. Each row is one category for one donut location.
id	Unquoted column identifying each donut location.
category	Unquoted column identifying donut categories.
value	Unquoted numeric column giving non-negative category values.
map	Optional sf object used as a background layer.
lon, lat	Unquoted longitude and latitude columns. Required when data is not an sf object.
input_crs	Coordinate reference system for lon and lat, or for an sf object with missing CRS. Defaults to EPSG:4326.
crs	Target projected CRS used to build the map.
radius_range	Numeric vector of length 2 giving minimum and maximum donut radii in map units. If NULL, a range is derived from the map extent.
inner_radius	Numeric value in (0, 1) giving the inner radius as a proportion of the outer radius.
n	Number of points used to approximate a complete outer circle.
colours	Optional category colours. Use a named vector for stable category-colour matching.
flows	Optional data frame of origin-destination flows.
from, to	Unquoted columns in flows identifying origin and destination ids. Required when flows is supplied.
flow_value	Optional unquoted numeric column in flows used to scale line widths. If omitted, each flow receives value 1.
flow_group	Optional unquoted column in flows used to colour flow lines and arrowheads by group.
flow_colours	Optional colours for flow_group. Use a named vector for stable group-colour matching. If omitted and flow groups match donut categories, colours is reused.
flow_min	Optional minimum flow value to draw.
flow_linewidth_range	Numeric vector of length 2 controlling flow line widths.
flow_curvature	Numeric curvature for trajectory lines. Use 0 for straight lines, positive values for one bend direction, and negative values for the opposite direction.
flow_n	Number of points used to approximate each curved trajectory.
flow_arrow	Should static flow trajectories include arrows?
flow_arrow_length	Arrow length in inches when flow_arrow = TRUE.
flow_colour, flow_alpha	Flow line colour and alpha. flow_colour is used when flow_group is not supplied.
map_fill, map_colour	Background map fill and outline colours.
donut_colour, donut_linewidth	Donut segment border colour and linewidth.

Value

A ggplot object.

Examples

```
demo <- data.frame(
  place = rep(c("A", "B", "C"), each = 3),
  lon = rep(c(-71.35, -71.2, -71.05), each = 3),
  lat = rep(c(46.75, 46.82, 46.73), each = 3),
  category = rep(c("x", "y", "z"), times = 3),
  value = c(10, 20, 5, 5, 15, 10, 12, 4, 9)
)

flows <- data.frame(
  from = c("A", "B"),
  to = c("B", "C"),
  trips = c(30, 10)
)

donut_map(
  demo,
  place,
  category,
  value,
  lon = lon,
  lat = lat,
  flows = flows,
  from = from,
  to = to,
  flow_value = trips
)
```

---

donut_polygons *Compute donut polygons*

Description

donut_polygons() turns tidy category values at spatial locations into an sf polygon layer. Each row of data represents one category for one location.

Usage

```
donut_polygons(
  data,
  id,
  category,
  value,
  lon = NULL,
  lat = NULL,
```

```
  input_crs = 4326,
  crs = NULL,
  map = NULL,
  radius_range = NULL,
  inner_radius = 0.55,
  n = 96,
  start_angle = pi/2
)
```

Arguments

data	A data frame or sf object. If data is not an sf object, lon and lat must be supplied.
id	Unquoted column identifying each donut location.
category	Unquoted column identifying donut categories.
value	Unquoted numeric column giving non-negative category values.
lon, lat	Unquoted longitude and latitude columns. Required when data is not an sf object.
input_crs	Coordinate reference system for lon and lat, or for an sf object with missing CRS. Defaults to EPSG:4326.
crs	Target projected CRS used to build the donut polygons. If NULL, a projected CRS is chosen from map, data, or an estimated UTM zone.
map	Optional sf object used only to choose the working CRS and default donut radius range.
radius_range	Numeric vector of length 2 giving minimum and maximum donut radii in map units. If NULL, a range is derived from the map extent.
inner_radius	Numeric value in (0, 1) giving the inner radius as a proportion of the outer radius.
n	Number of points used to approximate a complete outer circle.
start_angle	Start angle in radians. The default starts at 12 o'clock.

Value

An sf object with one polygon per non-zero location-category pair.

Examples

```
demo <- data.frame(
  place = rep(c("A", "B"), each = 3),
  lon = rep(c(-71.3, -71.1), each = 3),
  lat = rep(c(46.75, 46.85), each = 3),
  category = rep(c("x", "y", "z"), times = 2),
  value = c(10, 20, 5, 5, 15, 10)
)

donuts <- donut_polygons(demo, place, category, value, lon = lon, lat = lat)
plot(donuts["category"])
```

flow_lines *Compute origin-destination flow lines*

Description

`flow_lines()` creates `sf` line geometries joining origin and destination locations. Lines can be straight or curved trajectories.

Usage

```
flow_lines(
  flows,
  locations,
  from,
  to,
  value = NULL,
  id,
  group = NULL,
  lon = NULL,
  lat = NULL,
  input_crs = 4326,
  crs = NULL,
  drop_self = TRUE,
  flow_curvature = 0,
  flow_n = 30
)
```

Arguments

`flows`	A data frame containing origin-destination pairs.
`locations`	A data frame or `sf` object containing one or more rows per location. Repeated locations are reduced to the first geometry per `id`.
`from, to`	Unquoted columns in `flows` identifying origin and destination ids.
`value`	Optional unquoted numeric column in `flows` used as flow value. If omitted, each flow receives value 1.
`id`	Unquoted location id column in `locations`.
`group`	Optional unquoted column in `flows` used to group or colour flow lines.
`lon, lat`	Unquoted longitude and latitude columns in `locations`. Required when `locations` is not an `sf` object.
`input_crs`	Coordinate reference system for `lon` and `lat`, or for an `sf` object with missing CRS. Defaults to EPSG:4326.
`crs`	Target projected CRS. If `NULL`, a projected CRS is selected from `locations` or an estimated UTM zone.
`drop_self`	Should self-flows be removed?

flow_curvature	Numeric curvature for trajectory lines. Use 0 for straight lines, positive values for one bend direction, and negative values for the opposite direction.
flow_n	Number of points used to approximate each curved trajectory.

Value

An sf object with one line per retained flow.

Examples

```
locations <- data.frame(
  place = c("A", "B"),
  lon = c(-71.3, -71.1),
  lat = c(46.75, 46.85)
)
flows <- data.frame(from = "A", to = "B", trips = 15)

lines <- flow_lines(
  flows,
  locations,
  from,
  to,
  trips,
  place,
  lon = lon,
  lat = lat
)
plot(lines["value"])
```

Index