Package ‘cartogram’

August 26, 2020

Title Create Cartograms with R
Version 0.2.2
Description Construct continuous and non-contiguous area cartograms.
URL https://github.com/sjewo/cartogram
BugReports https://github.com/sjewo/cartogram/issues
Imports methods, sf, packcircles
Suggests rgeos, sp, rgdal, maptools
License GPL-3
Encoding UTF-8
LazyData true
RoxygenNote 7.0.2
NeedsCompilation no
Author Sebastian Jeworutzki [aut, cre]
  (<https://orcid.org/0000-0002-2671-5253>),
  Timothee Giraud [ctb],
  Nicolas Lambert [ctb],
  Roger Bivand [cph],
  Edzer Pebesma [cph],
  Jakub Nowosad [ctb] (<https://orcid.org/0000-0002-1057-3721>)
Maintainer Sebastian Jeworutzki <sebastian.jeworutzki@ruhr-uni-bochum.de>
Repository CRAN
Date/Publication 2020-08-26 20:20:11 UTC

R topics documented:

  cartogram_cont ......................................................... 2
  cartogram_dorling ..................................................... 4
  cartogram_ncont ....................................................... 5

Index 7
Description

Construct a continuous area cartogram by a rubber sheet distortion algorithm (Dougenik et al. 1985)

Usage

```
cartogram_cont(
  x,                 # SpatialPolygonDataFrame or an sf object
  weight,            # Name of the weighting variable in x
  itermax = 15,      # Maximum iterations for the cartogram transformation, if maxSizeError ist not reached
  maxSizeError = 1.0001, # Stop if meanSizeError is smaller than maxSizeError
  prepare = "adjust",  # adjust
  threshold = 0.05    # adjust
)
```

## S3 method for class 'SpatialPolygonsDataFrame'
cartogram_cont(
  x,                 # SpatialPolygonDataFrame or an sf object
  weight,            # Name of the weighting variable in x
  itermax = 15,      # Maximum iterations for the cartogram transformation, if maxSizeError ist not reached
  maxSizeError = 1.0001, # Stop if meanSizeError is smaller than maxSizeError
  prepare = "adjust",  # adjust
  threshold = 0.05    # adjust
)

## S3 method for class 'sf'
cartogram_cont(
  x,                 # SpatialPolygonDataFrame or an sf object
  weight,            # Name of the weighting variable in x
  itermax = 15,      # Maximum iterations for the cartogram transformation, if maxSizeError ist not reached
  maxSizeError = 1.0001, # Stop if meanSizeError is smaller than maxSizeError
  prepare = "adjust",  # adjust
  threshold = 0.05    # adjust
)

Arguments

- `x`: SpatialPolygonDataFrame or an sf object
- `weight`: Name of the weighting variable in `x`
- `itermax`: Maximum iterations for the cartogram transformation, if `maxSizeError` is not reached
- `maxSizeError`: Stop if `meanSizeError` is smaller than `maxSizeError`
Weighting values are adjusted to reach convergence much earlier. Possible methods are "adjust", adjust values to restrict the mass vector to the quantiles defined by threshold and 1-threshold (default), "remove", remove features with values lower than quantile at threshold, "none", don’t adjust weighting values

`threshold` Define threshold for data preparation

**Value**

An object of the same class as `x`

**References**


**Examples**

```r
library(maptools)
library(cartogram)
library(rgdal)
data(wrld_simpl)

# Remove uninhabited regions
afr <- spTransform(wrld_simpl[wrld_simpl$REGION==2 & wrld_simpl$POP2005 > 0,], CRS("+init=epsg:3395"))

# Create cartogram
afr_carto <- cartogram_cont(afr, "POP2005", 3)

# Plot
par(mfcol=c(1,2))
plot(afr, main="original")
plot(afr_carto, main="distorted (sp)")

# Same with sf objects
library(sf)
afr_sf = st_as_sf(afr)
afr_sf_carto <- cartogram_cont(afr_sf, "POP2005", 3)

# Plot
par(mfcol=c(1,3))
plot(afr, main="original")
plot(afr_carto, main="distorted (sp)")
plot(st_geometry(afr_sf_carto), main="distorted (sf)")
```
cartogram_dorling  

Calculate Non-Overlapping Circles Cartogram

Description

Construct a cartogram which represents each geographic region as non-overlapping circles (Dorling 1996).

Usage

cartogram_dorling(x, weight, k = 5, m_weight = 1, itermax = 1000)

## S3 method for class 'sf'
cartogram_dorling(x, weight, k = 5, m_weight = 1, itermax = 1000)

## S3 method for class 'SpatialPolygonsDataFrame'
cartogram_dorling(x, weight, k = 5, m_weight = 1, itermax = 1000)

Arguments

x  SpatialPolygonsDataFrame, SpatialPointsDataFrame or an sf object
weight  Name of the weighting variable in x
k  Share of the bounding box of x filled by the larger circle
m_weight  Circles’ movements weights. An optional vector of numeric weights (0 to 1 inclusive) to apply to the distance each circle moves during pair-repulsion. A weight of 0 prevents any movement. A weight of 1 gives the default movement distance. A single value can be supplied for uniform weights. A vector with length less than the number of circles will be silently extended by repeating the final value. Any values outside the range [0, 1] will be clamped to 0 or 1.
itermax  Maximum iterations for the cartogram transformation.

Value

Non overlapping proportional circles of the same class as x.

References


Examples

library(maptools)
library(cartogram)
library(rgdal)
data(wrld_simpl)
# Remove uninhabited regions
afr <- spTransform(wrld_simpl[wrld_simpl$REGION==2 & wrld_simpl$POP2005 > 0,],
                   CRS("+init=epsg:3395"))

# Create cartogram
afr_carto <- cartogram_dorling(afr, "POP2005")

# Plot
par(mfcol=c(1,2))
plot(afr, main="original")
plot(afr, main="distorted (sp)")
plot(afr_carto, col = "red", add=TRUE)

# Same with sf objects
library(sf)
afr_sf = st_as_sf(afr)
afr_sf_carto <- cartogram_dorling(afr_sf, "POP2005")

# Plot
par(mfcol=c(1,3))
plot(afr, main="original")
plot(afr_carto, main="distorted (sp)")
plot(st_geometry(afr_sf_carto), main="distorted (sf)")

---

**cartogram_ncont**

_Software for Cartogram Non-contiguity_ (Olson 1976)

**Cartogram Non-contiguity**

*Description*

Construct a non-contiguous area cartogram (Olson 1976).

**Usage**

```r
cartogram_ncont(x, weight, k = 1, inplace = TRUE)
```

## S3 method for class 'SpatialPolygonsDataFrame'
```r
cartogram_ncont(x, weight, k = 1, inplace = TRUE)
```

## S3 method for class 'sf'
```r
cartogram_ncont(x, weight, k = 1, inplace = TRUE)
```

**Arguments**

- **x**: SpatialPolygonDataFrame or an sf object
- **weight**: Name of the weighting variable in x
- **k**: Factor expansion for the unit with the greater value
inplace  If TRUE, each polygon is modified in its original place, if FALSE multi-polygons are centered on their initial centroid

Value

An object of the same class as x with resized polygon boundaries

References


Examples

library(maptools)
library(cartogram)
library(rgdal)
data(wrld_simpl)

# Remove uninhabited regions
afr <- spTransform(wrld_simpl[wrld_simpl$REGION==2 & wrld_simpl$POP2005 > 0,],
        CRS("+init=epsg:3395"))

# Create cartogram
afr_nc <- cartogram_ncont(afr, "POP2005")

# Plot
plot(afr)
plot(afr_nc, add = TRUE, col = 'red')

# Same with sf objects
library(sf)
afr_sf = st_as_sf(afr)
afr_sf_nc <- cartogram_ncont(afr_sf, "POP2005")

plot(st_geometry(afr_sf))
plot(st_geometry(afr_sf_nc), add = TRUE, col = 'red')
Index

cartogram_cont, 2

cartogram_dorling, 4

cartogram_ncont, 5