Thematic Maps with cartography

Timothée Giraud

useR! 2019 - Toulouse - 2019/07/09
Produce publication ready thematic maps
Standing on the Shoulders of Giants

Initially based on \texttt{sp} & \texttt{rgeos}…

… and updated to \texttt{sf} with version 2.0.0 (Sep, 2017)
Main Features: Symbologies

**Choropleth**
```r
choroLayer(x = mtq, var = "myvar", method = "quantile", nclass = 8)
```

**Typology**
```r
typolayer(x = mtq, var = "myvar")
```

**Proportional Symbols**
```r
propSymbolsLayer(x = mtq, var = "myvar", inches = 0.1, symbols = "circle")
```

**Colorized Proportional Symbols (relative data)**
```r
propSymbolsChorolayer(x = mtq, var = "myvar", var2 = "myvar2")
```

**Double Proportional Symbols**
```r
propTrianglesLayer(x = mtq, var1 = "myvar", var2 = "myvar2")
```

**OpenStreetMap Basemap**
```r
tiles <- getTiles(x = mtq, type = "osm")
tilesLayer(tiles)
```

**Isopleth**
```r
smoothLayer(x = mtq, var = "myvar", typefc = "exponential", span = 500, beta = 2)
```

**Discontinuities**
```r
discLayer(x = mtq.borders, df = mtq, var = "myvar", threshold = 0.5)
```

**Flows**
```r
propLinkLayer(x = mtq_link, df = mtq_df, var = "fij")
```

**Dot Density**
```r
dotDensityLayer(x = mtq, var = "myvar")
```

**Labels**
```r
labelLayer(x = mtq, txt = "myvar", halo = TRUE, overlap = FALSE)
```
Main Features: Transformations

Points to Links
mtq_link <- getLinkLayer(x = mtq, df = link)

Polygons to Borders
mtq_border <- getBorders(x = mtq)

Polygons to Pencil Lines
mtq_pen <- getPencilLayer(x = mtq)

Polygons to Grid
mtq_grid <- getGridLayer(x = mtq, cellsize = 3.6e+07, type = "hexagonal", var = "myvar")
Main Features: Map Layout

```r
legendChoro()

legendChoro(pos = "topleft",
title.txt = "legendChoro()
breaks = c(0,20,40,60,80,100),
col = carto.pal("green", 5),
nodata = TRUE, nodata.txt = "No Data")

legendTypo()

legendTypo(title.txt = "legendTypo()
col = c("peru", "skyblue", "gray77"),
categ = c("type 1", "type 2", "type 3"),
nodata = FALSE)

legendCirclesSymbols()

legendCirclesSymbols(var = c(10,100),
title.txt = "legendCirclesSymbols()
col = "#a7db4d",
inches = 0.3)

North Arrow:
north(pos = "topright")

Scale Bar:
barscale(size = 5)

Full Layout:
layoutLayer(
title = "Martinique",
tabtitle = TRUE,
frame = TRUE,
author = "Author",
sources = "Sources",
north = TRUE,
scale = 5)
```
Cheat Sheet

Thematic maps with cartography

**Symbology**
- In most functions the `x` argument should be an object; `x` objects are handled through `sp` and `s2r` arguments.
- `Choropleth` - `x`, `fig`, `name = "Choropleth"`, `col = "red"`.
- `Typology` - `x`, `fig`, `name = "Typology"`, `col = "black"`.
- `Popartic Symbology` - `x`, `fig`, `name = "Popartic Symbology"`, `col = "blue"`.
- `Colored Proportional Symbology` - `x`, `fig`, `name = "Colored Proportional Symbology"`, `col = "green"`.
- `Double Proportional Symbology` - `x`, `fig`, `name = "Double Proportional Symbology"`, `col = "yellow"`.
- `OpenStreetMap BaseMap` - `x`, `fig`, `name = "OpenStreetMap BaseMap"`.
- `RGB` - `x`, `fig`, `name = "RGB"`.
- `Discoloration` - `x`, `fig`, `name = "Discoloration"`.
- `Flow` - `x`, `fig`, `name = "Flow"`.
- `Cell Density` - `x`, `fig`, `name = "Cell Density"`.

**Transformations**
- `Polygons to Grid` - `x`, `fig`, `name = "Polygons to Grid"`, `col = "red"`.
- `Points to Link` - `x`, `fig`, `name = "Points to Link"`, `col = "blue"`.
- `Polygons to Borders` - `x`, `fig`, `name = "Polygons to Borders"`, `col = "green"`.
- `Polygons to Point Cluster` - `x`, `fig`, `name = "Polygons to Point Cluster"`, `col = "yellow"`.

**Map Layout**
- `North Arrow` - `x`, `fig`, `name = "North Arrow"`, `col = "purple"`.
- `Scale Bar` - `x`, `fig`, `name = "Scale Bar"`, `col = "blue"`.

**Legend**
- `Legend tj` - `x`, `fig`, `name = "Legend tj"`, `col = "red"`.
- `Legend tj` - `x`, `fig`, `name = "Legend tj"`, `col = "blue"`.

**Color Palettes**
- `cartoplasmatel" + "pat"`, `fig`, `name = "Color Palettes"`, `col = "red"`.

7/22
cartography

Create and integrate maps in your R workflow!

This package helps to design cartographic representations such as proportional symbols, choropleths, typology, flow or discontinuities maps. It also offers several features that improve the graphic presentation of maps, for instance, map palettes, layout elements (scale, north arrow, title...), labels or legends.

Cheat Sheet

The cheat sheet displays a quick overview of cartography’s main features.

Vignette

The vignette contains commented scripts on how to build various types of maps with cartography.
Alternative Solutions

**ggplot2** (Wickham, 2016) + **ggspatial** (Dunnington, 2018)
- A general purpose graphic library

**tmap** (Tennekes, 2018)
- A mapping library with similar functionalities
- Uses a different grammar (à la **ggplot2**)
- Allows interactive maps
library(sf)
library(cartography)

# path to the geo file embedded in cartography
path <- system.file("gpkg/mtq.gpkg",
    package = "cartography")

# import to an sf object
mtq <- st_read(dsn = path, quiet = TRUE)

# Set figure margins
par(mar = c(0,0,1.2,0), bg = "#b5d0d0")

# Plot the choropleth map
choroLayer(
    x = mtq, var = "MED",
    method = "equal", nclass = 4,
    col = carto.pal(pal1 = "sand.pal", n1 = 4),
    legend.values.rnd = -2, legend.pos = "left",
    legend.title.txt = "Median Income\n(in euros)"
)

# Plot a Layout
layoutLayer(
    title="Wealth in Martinique, 2015",
    author = "T. Giraud, 2019",
    sources = "Sources: Insee and IGN - 2018",
    scale = 5, north = TRUE,
    tabtitle = TRUE, frame = FALSE
)
- **osrm**: Gets road distances (time) between each municipalities.

- **SpatialPosition**: Computes gravitational accessibility based on road distances.

- **cartography::getBreaks()**: Classifies data

- **cartography::carto.pal()**: Uses color palettes from the package

- **cartography::choroLayer()**: Plots the choropleth map
- **sf**: Provides geodata transformations
- **cartography::propSymbolsLayer()**: Plots the proportional symbols
- **cartography::layoutLayer()**: Plots the map layout
- `cartography::getPencilLayer()`: Transforms polygons to hand-drawn polylines

- `cartography::choroLayer()`: Plots the choropleth map
- `spatstat`: Computes Kernel Density Estimation
- `tanaka`: Plots shaded contour lines
- `cartography::LegendChoro()`: Plots legend
- `cartography::layoutLayer()`: Plots map layout
- `cartography::choroLayer()`: Plots the choropleth map
- `cartography::getBorders()`: Extract borders between countries
- `cartography::discLayer()`: Plots discontinuities on borders
- `cartography::layoutLayer()`: Plots the map layout
Thank You

frama.link/cartography

github.com/riatelab/cartography

@rgeomatic

rgeomatic.hypotheses.org