Creating thematic maps in R

OpenGeoHUB Summer School 2019 Münster

Martijn Tennekes
tmap: R package for thematic maps

- A thematic map is a visualization where statistical information with a spatial component is shown.

- Thematic maps can be also made with other R packages:
  - plot (from the spatial method packages, e.g. sf and raster) Fast to plot spatial objects, but requires manual work to create thematic maps. Only static maps.
  - ggplot2, Popular general data visualization package. Thematic maps can be made easily, but the layout requires some attention. Only static maps.
  - leaflet, R interface to the popular Javascript library. Easy to produce maps of spatial objects, but requires manual work to create thematic maps. Only interactive maps.
  - mapview, Excellent package to explore spatial objects quickly. Only interactive maps.

- The syntax of tmap is based on ggplot2 and the Grammar of Graphics, but works fluently with spatial objects from the sf, sp and raster packages.

- It supports two modes: plot (static maps) and view (interactive maps)


- Development site http://github.com/mtennekes/tmap
## The history of tmap

<table>
<thead>
<tr>
<th>Package / version</th>
<th>Date</th>
<th>Description / new features</th>
</tr>
</thead>
<tbody>
<tr>
<td>geoNL</td>
<td>2014</td>
<td>General functions to create thematic maps for the Netherlands</td>
</tr>
<tr>
<td>geo</td>
<td>2014-07</td>
<td><strong>ggplot2</strong> style approach to create thematic maps</td>
</tr>
<tr>
<td>tmap 0.6</td>
<td>2014-07</td>
<td>geo was accepted on CRAN, but had to be renamed... “geo” was too general 😊</td>
</tr>
<tr>
<td>tmap 1.0</td>
<td>2015-05</td>
<td>First stable release of tmap</td>
</tr>
<tr>
<td>tmap 1.4</td>
<td>2016-03</td>
<td>View mode added (i.e. interactive maps)</td>
</tr>
<tr>
<td>tmap 1.11-2</td>
<td>2018-04</td>
<td>Version described in the JSS paper</td>
</tr>
<tr>
<td>tmap 2.0</td>
<td>2018-07</td>
<td>Migration from sp to sf</td>
</tr>
<tr>
<td>tmap 2.3</td>
<td>2019-07</td>
<td>shiny integration (with tmapOutput and renderTmap)</td>
</tr>
</tbody>
</table>
The Grammar of Graphics

**ggplot2**
Layered Grammar of Graphics

- Defaults
  - Data
  - Aesthetics
- Layers
  - Data
  - Aesthetics
  - Geometry
  - Statistics
  - Position
- Scales
- Coordinates
- Facets

**tmap**
Layered Grammar of Thematic Maps

- Shape
  - Coordinates and topology. Spatial types:
    - Polygons
    - Points
    - Lines
    - Raster
  - Data
  - Map projection
  - Bounding box
- Layers
  - Aesthetics
  - Statistics
  - Scale
- Facets

Group

1

1 or more
Example: choropleth

```r
# load example datasets
data("World")

# draw polygons
tm_shape(World) + tm_polygons()

# draw polygons with a specific color
tm_shape(World) + tm_polygons("blue")

# draw polygons colored by a data variable
# the result is called a choropleth
tm_shape(World) + tm_polygons("income_grp")
```
# load example dataset
data("metro")

# draw dotstm_shape(metro) +
  tm_dots()

# draw a bubble map
tm_shape(metro) +
  tm_bubbles("pop2020")

# draw a colored bubble map
tm_shape(metro) +
  tm_bubbles("pop2020", col = "growth")

# combine choropleth with bubble map
tm_shape(world) +
  tm_polygons("income_grp") +
  tm_shape(metro) +
  tm_bubbles("pop2020", col = "growth")
Example: choropleth with bubble map
Two modes: plot and view

tmap contains two modes:

**plot**: static maps, shown in graphics device window; can be exported to png, jpg, pdf, etc.

**view**: interactive maps, shown in the viewing window or in the browser; can be exported to standalone HTML files

```r
# switch to plot mode:
tmap_mode("plot")

# switch to view mode:
tmap_mode("view")

# toggle between modes:
ttm()
```
The last plot in view mode

# switch to view mode:
tmap_mode("view")

# repeat the last plot (but now in view mode)
tmap_last()
Change style

... + tm_style("classic")
Facets

```r
# specify multiple variables
tm_shape(world) +
tm_polygons(c("HPI", "gdp_cap_est"))
```
Facets

```
facets <- tm_shape(crimes_city) +
  tm_dots() +
  tm_facets(by = "Crime.type", free.coords = FALSE)
```
Output functions

Save to image:

```r
tm_twitter <- tm_shape(NLD_muni) + tm_polygons() +
  tm_shape(NLD_twitter) + tm_dots()

tmap_save (tm_twitter, filename = "twitter.png", width = 600, height = 800)
```

Save to interactive website:

```r
tmap_save (tm_twitter, filename = "twitter.html")
```

Create animation:

```r
tmap_animation(...)```
tmap in shiny apps

ui <- fluidPage(
  tmapOutput("map"),
  selectInput("var", "Variable", world_vars)
)

server <- function(input, output, session) {
  # initial map
  output$map <- renderTmap({
    tm_shape(World) +
    tm_polygons(world_vars[1], zindex = 401)
  })

  # update map
  observe({
    var <- input$var
    tmapProxy("map", session, {
      tm_remove_layer(401) +
      tm_shape(World) +
      tm_polygons(var, zindex = 401)
    })
  })
}

Required tmap >= 2.3
Which mapping package to use?

• If you familiar with ggplot2, and do not care about interactive maps, and do not prefer to learn yet another package? **ggplot2**

• Else if you want interactive maps as flexible as possible (albeit with more code)? **leaflet**

• Else if you just want to explore spatial objects of any sort interactively? **mapview**

• Else **tmap**
Summary

- **tmap** is a powerful package for spatial data visualization
- It is based on the syntax of **ggplot2**, but tailored for maps
- Other awesome mapping packages are **ggplot2**, **leaflet**, **mapview**
- The key qualities of **tmap** are:
  - Intuitive and easy to understand syntax (therefore very suitable for educational purposes)
  - Many options to configure the map
  - Two modes: static and interactive maps
- The users are key in the development of software. Therefore, please do not hesitate to post questions, bug reports, or suggestions.
- Please use
  - **StackOverflow** for general questions, and
  - **github** for bug reports and suggestions.