Practical: Maps with R

Objectives

1. Gain experience using mapping packages for R;
2. Use R mapping packages to generate cartographic data graphics;
3. Critically examine your maps, improving them through an iterative design process.

You are encouraged to work in groups of three or four people in order to troubleshoot problems which might arise. However, you should be able to complete this practical while working alone.

The Required Packages

Before commencing, you may need to download the latest version of Rtools and install it. You can download Rtools here: Rtools for Windows. Rtools provides you will a set of tools to handle packages. Be sure to select BOTH check boxes on the last page of the installation wizard BEFORE you exit (this ensures your library paths are correctly linked).

There are specific libraries needed for this practical. Two through five are specific to spatial analysis and visualisation.

1. Rtools
2. maptools (for handeling spatial data)
3. rgdal (R geometry engine)
4. ggplot2 (for specialized plotting of maps), and
5. ggmap (for integrating with Google maps).
6. classInt (for later in the practical)
7. RColorBrewer (explanation is in Colors and Legends)

Downloading a Package

You must download packages using `install.packages` and then load it as a library using the `library` command:

```
> install.packages("maptools")
> library(maptools)
```

Repeat this for the other packages above. It is worth noting that there are many other mapping packages for R, such as GISTools.

If you have attempted this tutorial previously, check that you remove the package 'gpclib'.

```
> remove.packages("gpclib")
```

Acquiring Data

For this exercise, we will use European adminstrative boundaries and statistical data on expenditure on education from the Eurostat web site, http://epp.eurostat.ec.europa.eu/portal/page/portal/eurostat/home/. The data has been prepared and made available from the course website. Other sources of shapefile data include GADM (Global Administrative Areas http://gadm.org/) and CShapes (http://nils.weidmann.ws/projects/cshapes), which also includes temporal data about changes in administrative boundaries.

Reading in Data

To make the paths to the data sets easier to manage, you should use the `getwd()` (get working directory) and `setwd()` (set working directory) commands. Read in the data to R using the following commands:

```
> eurMap <- readOGR(dsn="your_file_path","CNTR_RG_60M_2010")
> eurEdu <- read.csv("your_file_path/educ_thexp_1_Data.csv", stringsAsFactors = F)
> # Note, your_file_path must be specified using forward slashes '/'
> # and, importantly, it must not end in a slash, e.g., dsn="C:/Data".

> plot(eurMap)
> # R can display the boundaries we have loaded, using the plot command.
```

ggplot2

We can do better with ggplot2. To use ggplot2, the eurMap object must first be converted to a data frame (rather than a SpatialPolygonsDataFrame, see `class(eurMap)`). ggplot2 uses '`fortify`'.

```
> eurMapDf <- fortify(eurMap,region="CNTR_ID")
> # We can now view the most basic data in ggplot2:
> ggplot(eurMapDf)+ aes(long,lat,group=group)+geom_polygon()
```

Note the slightly unusual use of '+'. Typical ggplot maps arise from this type of combination of components, each component representing a "layer" or a ggplot operation. Next, we can combine spatial and attribute data together using the merge command:

```
> eurEduMapDf <- merge(eurMapDf, eurEdu, by.x="id", by.y="GEO")
```

As our education data only pertains to certain countries in the world, we can restrict our data to our region of interest using the following:

```
> eurEduMapDf <- eurEduMapDf[order(eurEduMapDf$order),]
> europe.limits <- geocode(c("Cape Fligely, Franz Josef Land, Russia", "Gavdos, Greece", "Faja Grande, Azores", "Severny Island, Novaya Zemlya, Russi
> eurEduMapDf <- subset(eurEduMapDf, long > min(europe.limits$lon) & long < max(europe.limits$lon) & lat > min(europe.limits$lat) & lat < max(europe.
```

Now the data is ready to redisplay as a choropleth map:

```
> ggplot(eurEduMapDf)+aes(long,lat,group=group,fill=Value)+geom_polygon()
```

The data in the `Value` column is total expenditure on education as a percentage of GDP.

More advanced mapping

Our initial map design can be improved in several key areas. To get you started, here are examples of immediate stages in an iterative redesign.

Map projections

The mapped data appears stretched because of the default map projection used. Try for example:

```
> ggplot(eurEduMapDf,aes(long,lat,group=group,fill=Value)) + geom_polygon() + coord_equal()
```

ggplot2 understands a wide range of other projections. Experiment with replacing `coord_equal` with instead `coord_map`, e.g., `coord_map("orthographic")`, `coord_map("stereographic")`, `coord_map("conic", lat0 = 30)` (see `?coord_map` for more ideas and information).

Classification

One of our first tasks should be to classify systematically our data about education expenditure. This can be done with the `classInt` package.

- Ensure you have installed and loaded the "classInt" package.
- Compute the class breaks the education data using the equal interval method:

  ```
  > intervals <- classIntervals(eurEduMapDf$Value, n=5, style="equal")
  ```

- Classify the data using these class breaks:

  ```
  > rank <- findInterval(intervals$var,intervals$brks)
  ```

- Store the classified data as a new column in the data frame:

  ```
  > eurEduMapDf$rank <- rank
  ```

- Finally, replot the data with the changed classification.

  ```
  > ggplot(eurEduMapDf) + aes(long,lat,group=group,fill=rank)+geom_polygon()
  ```

Rebuild the map using different classification methods, e.g., quantiles, jenks, different numbers of classes. Compare the different versions. Which is better?

Colors and legend

The presentation of the mapped data, the expenditure on education as a percentage of GDP, is somewhat arbitrary. Note that the legend is continuous where as the classes are discrete. This can be fixed through the ggplot2 scale_fill_continuous() function. Try the following amendment to the plot:

```
> ggplot(eurEduMapDf) + aes(long,lat,group=group,fill=rank) + geom_polygon() + scale_fill_continuous(guide="legend")
```

As the ggplot commands become longer, they rapidly become more tedious to write. However, note that you can store in a variable and of the components of a ggplot summation. For example:

```
> m0 <- ggplot(eurEduMapDf)
> m1 <- aes(long,lat,group=group,fill=rank)
> m2 <- geom_polygon()
> m3 <- scale_fill_continuous(guide="legend")
> m0 + m1 + m2 + m3
```

The colors used are also somewhat arbitrary. To help with color selection for maps, Cynthia Brewer and colleagues have developed ColorBrewer, http://colorbrewer2.org/., with carefully chosen color schemes that are clearly visible with strong value message. In fact, R also "speaks ColorBrewer" through the "RColorBrewer" package. You can apply a ColorBrewer palette directly through ggplot, using the `scale_fill_brewer` function. However, first we must change the numerical data in the ranks to ordinal (factor) data:

```
> eurEduMapDf$rank <- as.factor(rank)
> m0 + m1 + m2 + scale_fill_brewer(type="seq",palette=7)
```

The map may appear somewhat washed out as the country borderlines are not yet represented:

```
> m0 + m1 + m2 + scale_fill_brewer(type="seq",palette=7) + geom_path(color='dark grey')
```

There are many other improvements to explore. ggplot2 alters many aspects of appearance through "themes". For example:

```
> m0 + m1 + m2 + scale_fill_brewer(type="seq",palette=7) + geom_path(color='dark grey') + theme(panel.background = element_rect(fill="white"))
```

This will remove the background grey color. Other improvements you should investigate include

- clearer labels for the legend;
- a title;
- thinner lines for country boundaries;
- removing the "lat" and "long" labels;
- adding Google map backgrounds; and many other options.

Coda

Core material based on an original exercise by Max Marchi, http://www.milanor.net/blog/?p=634.

Creative Commons Additional material and adaptations by Matt Duckham and Nathaniel Carpenter, licensed under a Creative Commons Attribution 3.0 Unported License.