Making Maps with R

Shaun Truelove

Friday, June 19, 2015

Making Maps with R

- rworldmap
- ggmaps

Setup your packages

```
## If necessary
install.packages(c("rworldmap", "rworldxtra", "RColorBrewer",
    "maptools", "classInt"))

## Load packages
library('rworldmap')
library('rworldxtra')
library('RColorBrewer')
library('maptools')
library('classInt')
```

Load the map data

First we need to load the data. For this we will be using data provided by the package `rworldmap` for our maps. You can do the same thing with your own map files or using packages that use freely available maps, like `Google Maps` or `OpenStreetMap` (https://www.openstreetmap.org).

```
worldmap <- getMap(resolution = "high")
dim(worldmap)
```

```
## [1] 253  51
```

What's in the data object

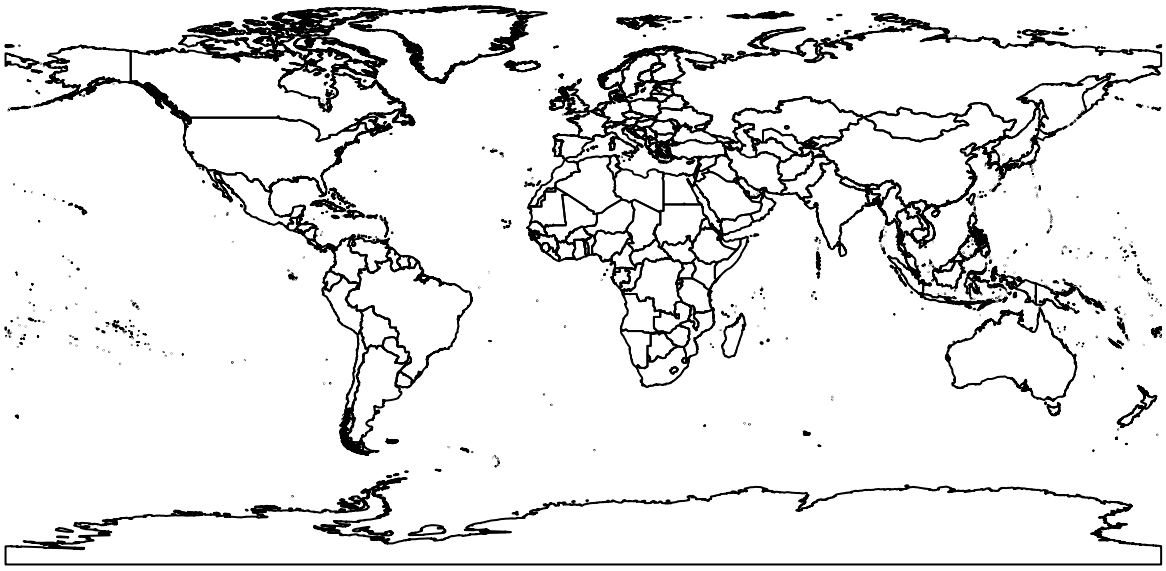
```
names(worldmap)
```

```
##  [1] "ne_10m_adm"   "ScaleRank"    "LabelRank"    "FeatureCla"
##  [5] "OID_"         "SOVEREIGNT"   "SOV_A3"       "ADM0_DIF"
##  [9] "LEVEL"        "TYPE"         "ADMIN"        "ADM0_A3"
## [13] "GEOU_DIF"     "GEOUNIT"      "GU_A3"        "SU_DIF"
## [17] "SUBUNIT"      "SU_A3"        "NAME"         "ABBREV"
## [21] "POSTAL"       "NAME_FORMA"   "TERR_"        "NAME_SORT"
## [25] "MAP_COLOR"    "POP_EST"      "GDP_MD_EST"   "FIPS_10_"
## [29] "ISO_A2"       "ISO_A3"       "ISO_N3"       "ISO3"
```

```
## [33] "LON"          "LAT"          "ISO3.1"       "ADMIN.1"
## [37] "REGION"       "continent"    "GEO3major"    "GEO3"
## [41] "IMAGE24"      "GLOCAF"       "Stern"        "SRESmajor"
## [45] "SRES"         "GBD"          "AVOIDnumeric" "AVOIDname"
## [49] "LDC"          "SID"          "LLDC"
```

Plot the world

```
par(mar=c(0,0,0,0))     # Set 0 margins
plot(worldmap)          # Plot
```



Plot a smaller area

Setting the `xlim` and `ylim` sets our plotted area to a specific limit of lattitude and longitude coordinates.

- ylim = Lattitude
- xlim = Longitude

```
par(mar=c(0,0,0,0))     # Set 0 margins
plot(worldmap, xlim = c(-20, 59), ylim = c(35, 71), asp = 1)
```

Plot a Region, Country, or other Area

We can also select only certain regions or countries if we want.

```
t(t(table(worldmap$REGION)))
```

```
##
##                                  [,1]
##   Africa                           57
##   Antarctica                        1
##   Asia                             46
##   Australia                        27
##   Europe                           70
##   North America                     5
##   South America and the Caribbean  44
```

More

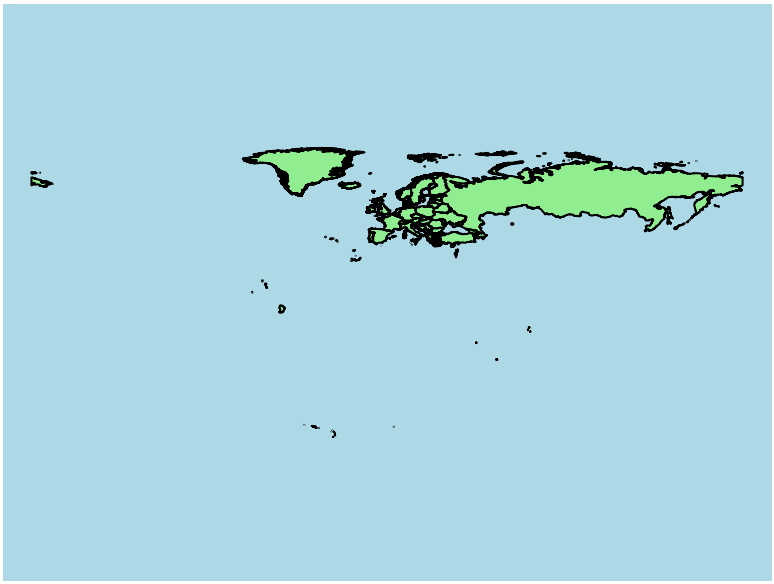
```
table(worldmap$GEO3)
```

```
##
##                 Antarctic         Arabian Peninsula
```

```
##                          1                          6
## Australia and New Zealand                     Canada
##                          5                          2
##                  Caribbean             Central Africa
##                         23                          9
##               Central Asia             Central Europe
##                          5                         21
##             Eastern Africa             Eastern Europe
##                          9                          8
##                    Mashriq               Meso-America
##                          5                          8
##                North Africa  NW Pacific and East Asia
##                          8                          9
##                      Polar              South America
##                          1                         13
##                 South Asia              South Pacific
##                         10                         22
##             Southeast Asia            Southern Africa
##                         11                         12
##                         US             Western Africa
##                          3                         15
##             Western Europe       Western Indian Ocean
##                         40                          4
```

Plot Europe

```
par(mar=c(0,0,0,0))      # Set 0 margins
europe <- worldmap[which(worldmap$REGION=="Europe"),]
plot(europe, col="lightgreen", bg="lightblue")
```



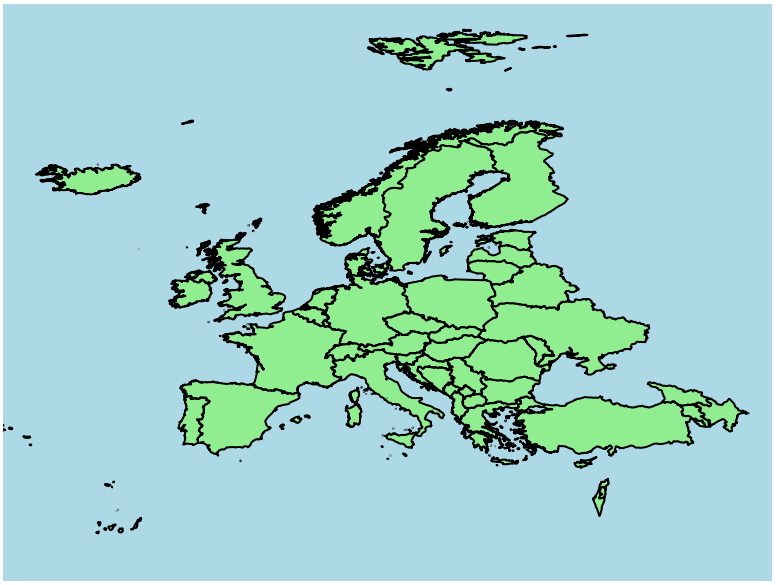
This looks a little weird because of Russia and the inclusion of the island territories.

Europe excluding Russia and Territories

```
par(mar=c(0,0,0,0))
europe <- worldmap[which(grepl("Europe", worldmap$GEO3) &
                           as.character(worldmap$NAME) != "Russia"),]
plot(europe, col="lightgreen", bg="lightblue")
```

Zoom to an area

```
par(mar=c(0,0,0,0))
plot(europe, col="lightgreen", bg="lightblue",
     xlim = c(-25, 50), ylim = c(35, 71), asp = 1)
```



Mapping Data

Adding Population data

First we need to load our population data:

```
world.pop <- read.csv("../data/world.population.csv",
                      header=TRUE)
row.names(world.pop) <- world.pop[,1]
```

Check Matching

Check which countries we do not have population data for:

```
country.codes <- as.character(worldmap$ADM0_A3)
worldmap$ADMIN[which(!(country.codes %in%
    world.pop$CountryCode))]
```

```
##  [1] Anguilla
##  [2] Aland
##  [3] Antarctica
##  [4] Ashmore and Cartier Islands
##  [5] French Southern and Antarctic Lands
##  [6] Saint Barthelemy
##  [7] Clipperton Island
##  [8] Cyprus No Mans Area
##  [9] Cook Islands
## [10] Coral Sea Islands
## [11] Northern Cyprus
## [12] Dhekelia Sovereign Base Area
## [13] Falkland Islands
## [14] Gaza
## [15] Guernsey
## [16] Gibraltar
## [17] Heard Island and McDonald Islands
## [18] Indian Ocean Territories
## [19] British Indian Ocean Territory
## [20] Jersey
## [21] Baykonur Cosmodrome
## [22] Siachen Glacier
## [23] Korea No Mans Area
## [24] Montserrat
## [25] Norfolk Island
## [26] Niue
## [27] Nauru
## [28] Pitcairn Islands
## [29] Western Sahara
## [30] South Sudan
## [31] South Georgia and South Sandwich Islands
## [32] Saint Helena
## [33] Somaliland
## [34] Saint Pierre and Miquelon
```

```
## [35] Taiwan
## [36] United States Minor Outlying Islands
## [37] US Naval Base Guantanamo Bay
## [38] Vatican
## [39] British Virgin Islands
## [40] West Bank
## [41] Wallis and Futuna
## [42] Akrotiri Sovereign Base Area
## 253 Levels: Afghanistan Akrotiri Sovereign Base Area Aland ... Zimbabwe
```

Check Matching

Look for one that didn't match

```
grep("west bank", world.pop$CountryName, ignore.case=TRUE, value=TRUE)
```

```
## [1] "West Bank and Gaza"
```

Check Matching

Check what from the population data is not in the map data

```
as.character(world.pop$CountryName)[
  which(!(world.pop$CountryCode %in% country.codes))]
```

```
##  [1] "Arab World"
##  [2] "Central Europe and the Baltics"
##  [3] "Channel Islands"
##  [4] "Caribbean small states"
##  [5] "East Asia & Pacific (developing only)"
##  [6] "East Asia & Pacific (all income levels)"
##  [7] "Europe & Central Asia (developing only)"
##  [8] "Europe & Central Asia (all income levels)"
##  [9] "Euro area"
## [10] "European Union"
## [11] "Fragile and conflict affected situations"
## [12] "High income"
## [13] "Heavily indebted poor countries (HIPC)"
## [14] "Not classified"
## [15] "Latin America & Caribbean (developing only)"
## [16] "Latin America & Caribbean (all income levels)"
## [17] "Least developed countries: UN classification"
## [18] "Low income"
## [19] "Lower middle income"
## [20] "Low & middle income"
## [21] "Middle East & North Africa (all income levels)"
## [22] "Middle income"
## [23] "Middle East & North Africa (developing only)"
## [24] "North America"
## [25] "High income: nonOECD"
## [26] "High income: OECD"
```

```
## [27] "OECD members"
## [28] "Other small states"
## [29] "Pacific island small states"
## [30] "South Asia"
## [31] "Sub-Saharan Africa (developing only)"
## [32] "South Sudan"
## [33] "Sub-Saharan Africa (all income levels)"
## [34] "Small states"
## [35] "Upper middle income"
## [36] "West Bank and Gaza"
## [37] "World"
```

Looks like all of the unmatched are aggregates, with a couple exceptions.

Add the population data to the map data

```
Pop2013 <- world.pop[,c("CountryCode", "X2013")]
colnames(Pop2013)
```

```
## [1] "CountryCode" "X2013"
```

```
colnames(Pop2013)[2] <- "Pop2013"

worldmap$ADM0_A3 <- as.character(worldmap$ADM0_A3)
worldmap <- merge(worldmap, Pop2013,
  by.x="ADM0_A3", by.y="CountryCode", all.x=TRUE)
```

```
## Warning in .local(x, y, ...): 37 records in y cannot be matched to x
```

Other options for adding data to map data

```
joinCountryData2Map()
```

Part of the `rworldmap` package

Joins user data referenced by country codes or names to an internal map, ready for plotting using mapCountryData. Reports join successes and failures.

Setting up to Plot

To plot population, we will give the country a color based on the population. To do this, we need to create population categories/intervals.

Option 1

```
quantile(worldmap$Pop2013, na.rm=TRUE)
```

```
##           0%          25%          50%          75%         100%
##       9876.0     836272.5    6333135.0   22549754.5 1357380000.0
```

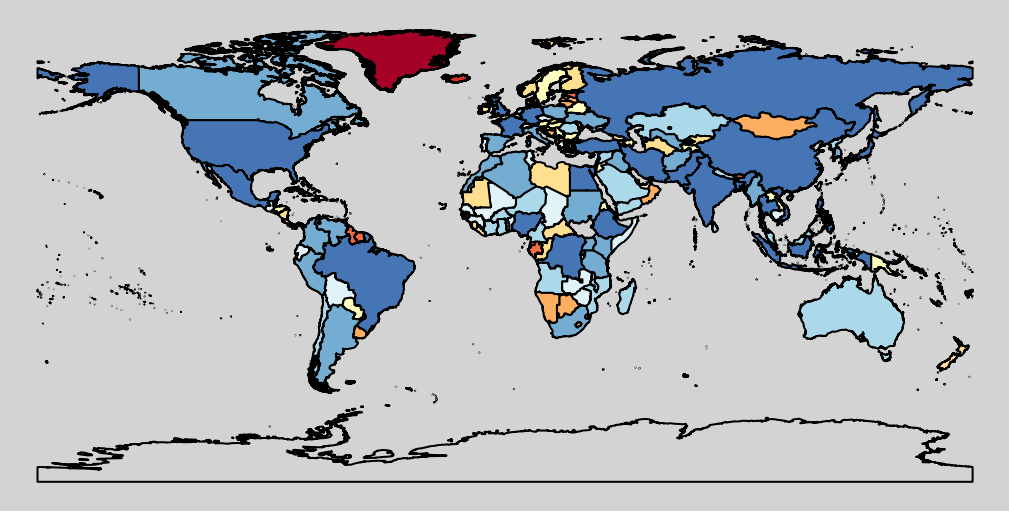
```
library(classInt)
brks <- classIntervals(worldmap$Pop2013[
    which(!is.na(worldmap$Pop2013))],
       n=10, style="quantile")
brks <- brks$brks
colors <- brewer.pal(length(brks), "RdYlBu")
```

Option 2

```
pop_cuts <- c(100000, 500000, 1000000, 5000000, 25000000,
              100000000, 500000000, 1000000000, 1500000000)
colors2 <- brewer.pal(length(pop_cuts) + 1, "RdYlBu")
```

Plot the world map

```
plot(worldmap, col=colors[findInterval(worldmap$Pop2013,
      brks, all.inside=TRUE)], axes=FALSE, bg="lightgray")
```



This does not let us distinguish very well. We know the U.S. has a population of ~330 million, yet it is the same color as India and China, each with over 1 billion. We can define our population cuts better for this.

Plot World Population - 2

```
plot(worldmap, col=colors2[findInterval(worldmap$Pop2013,
      pop_cuts, all.inside=TRUE)], axes=FALSE, bg="lightgray")
title("Population by Country, World 2013") #add a title
legend("bottomleft", legend=leglabs(round(pop_cuts)), #add a legend
       fill=colors2, bty="n", cex=.6)
```