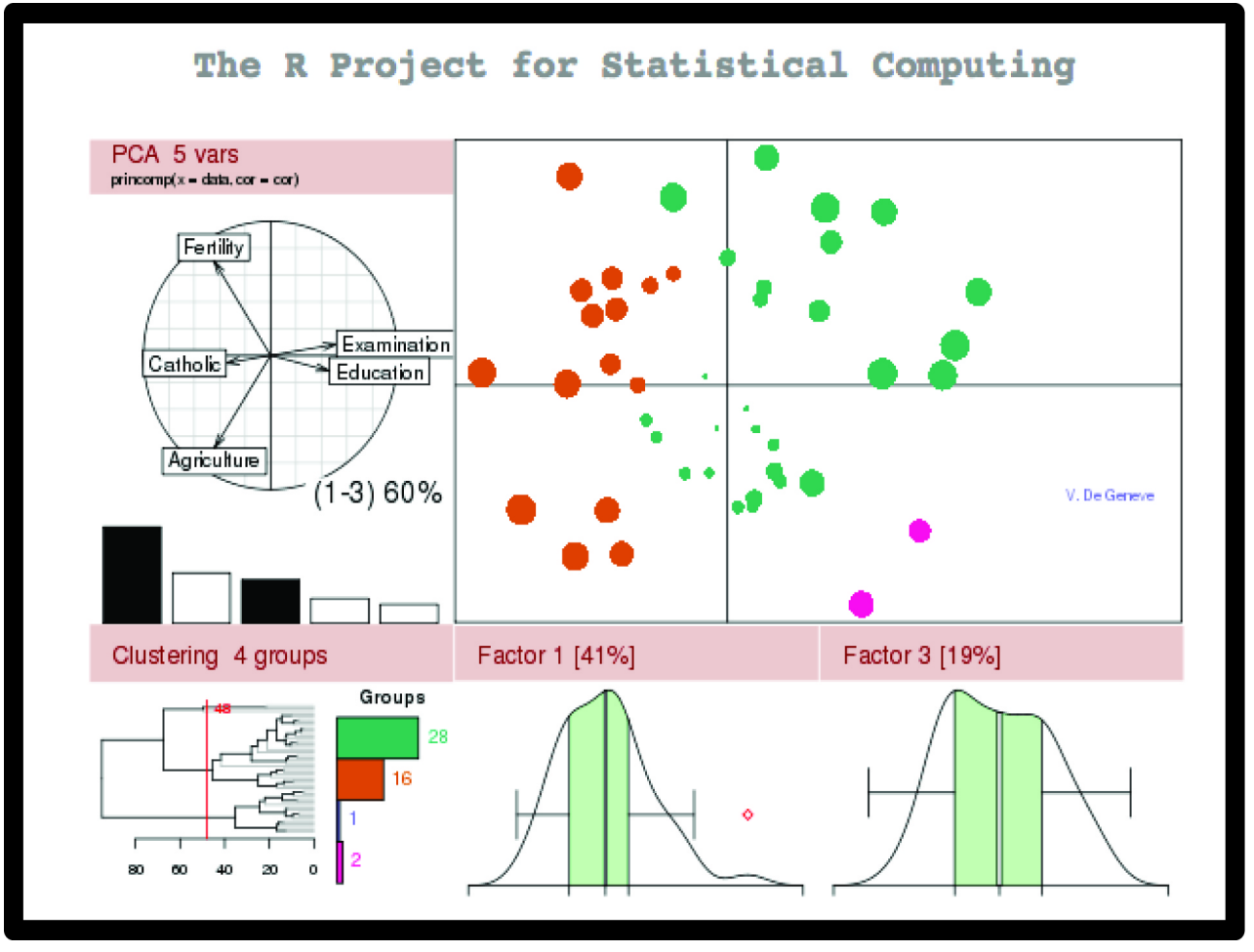
GIS with R

Francisco Rodriguez-Sanchez

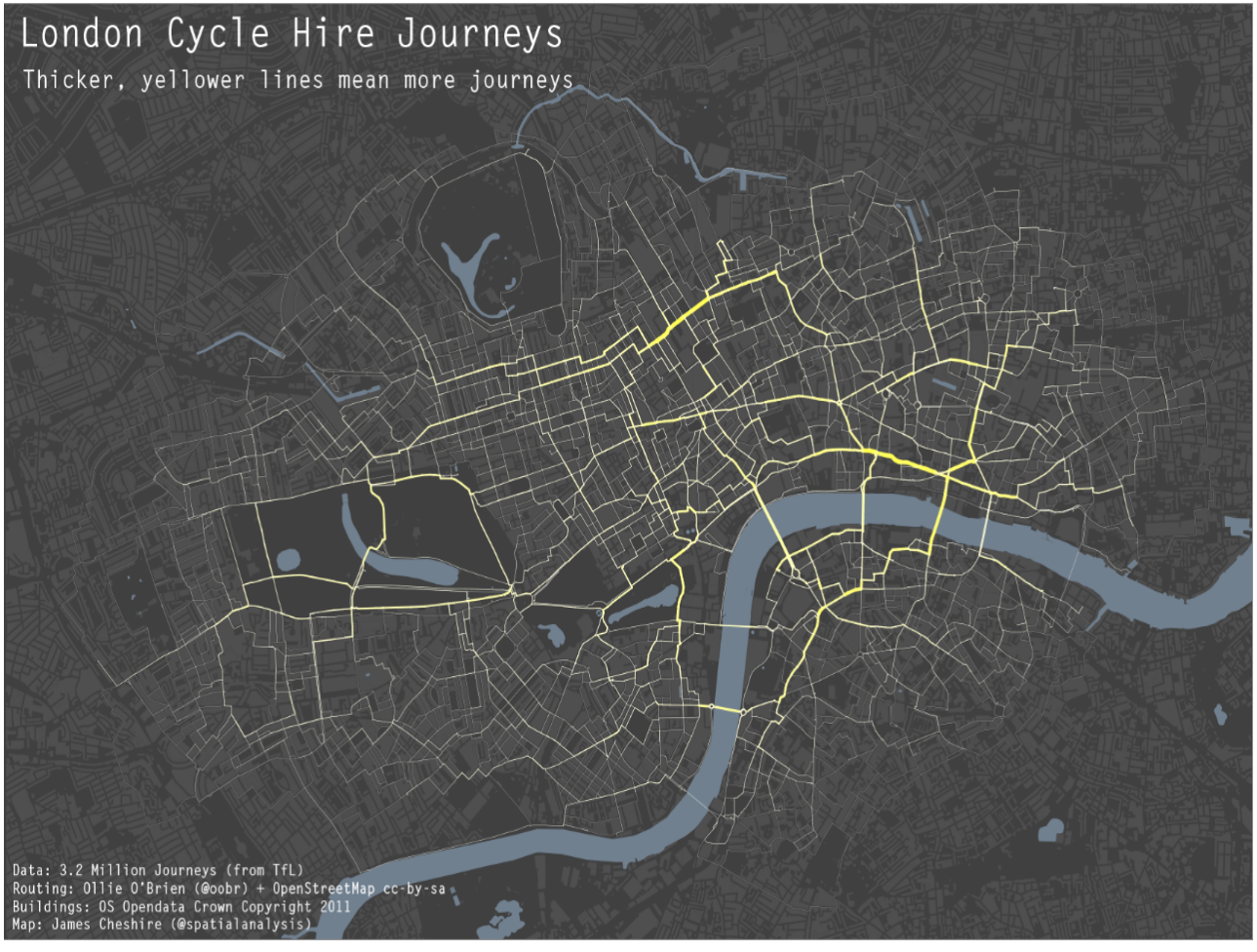
@frod_san

R: Not only for stats



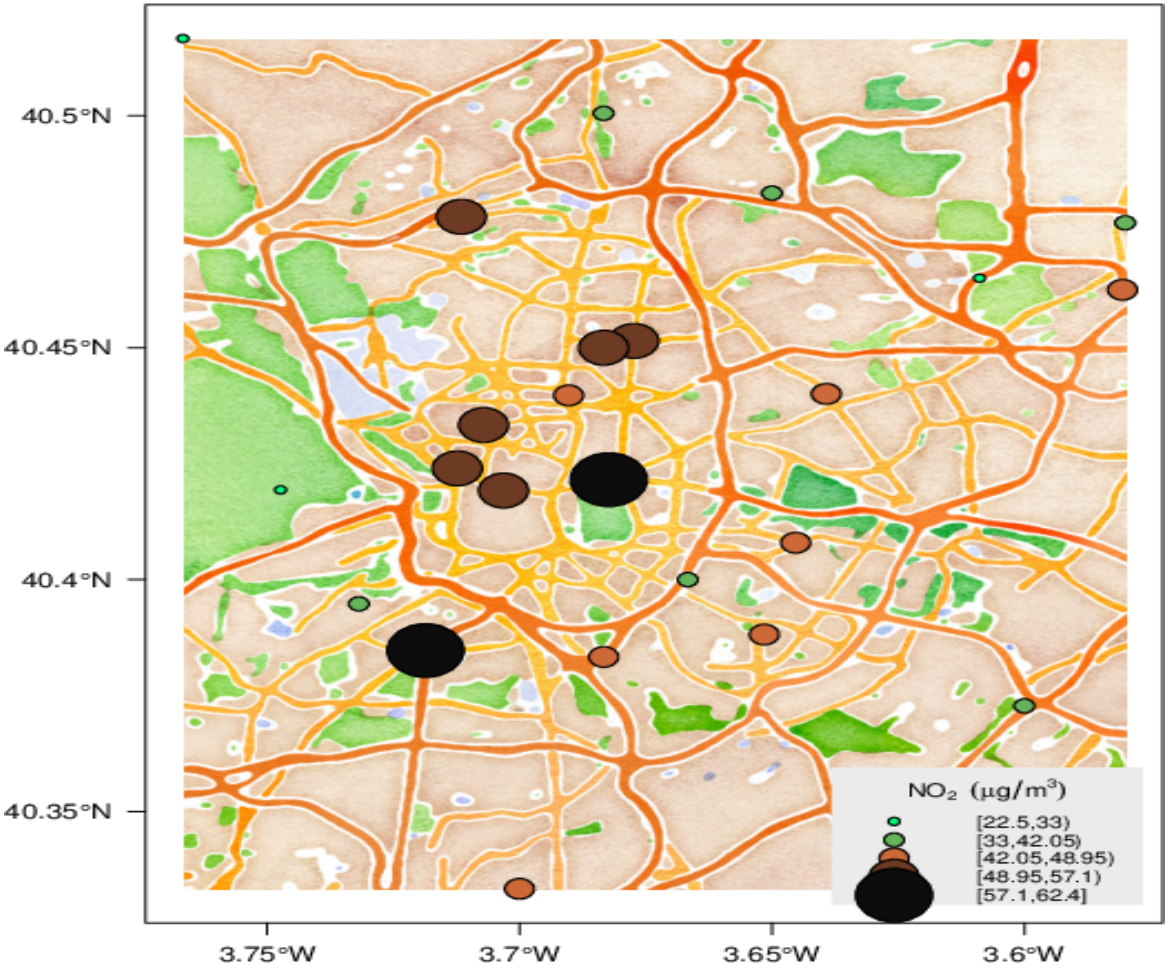
R can make beautiful maps

Made in R



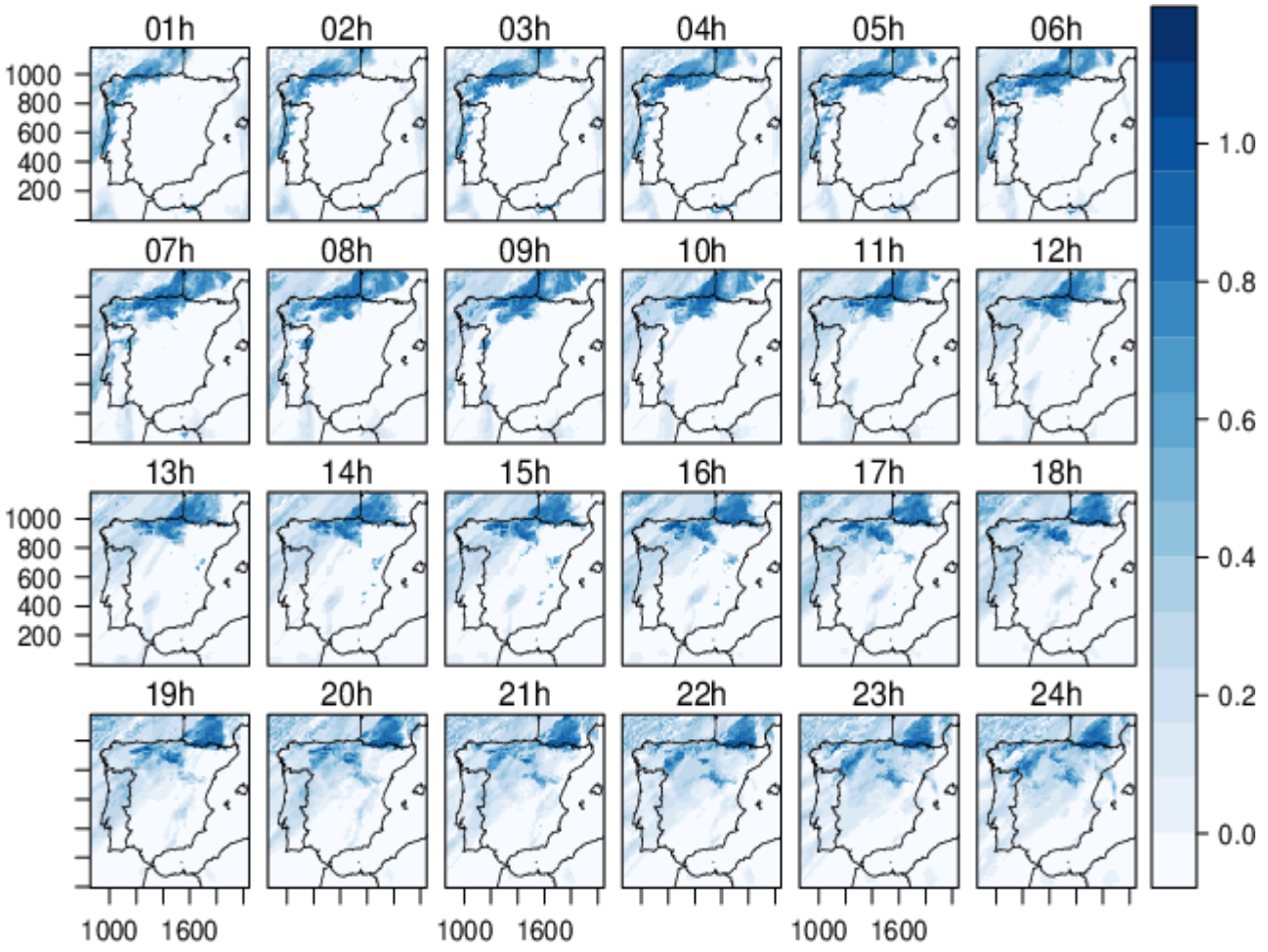
http://spatial.ly/2012/02/great-maps-ggplot2/

Made in R



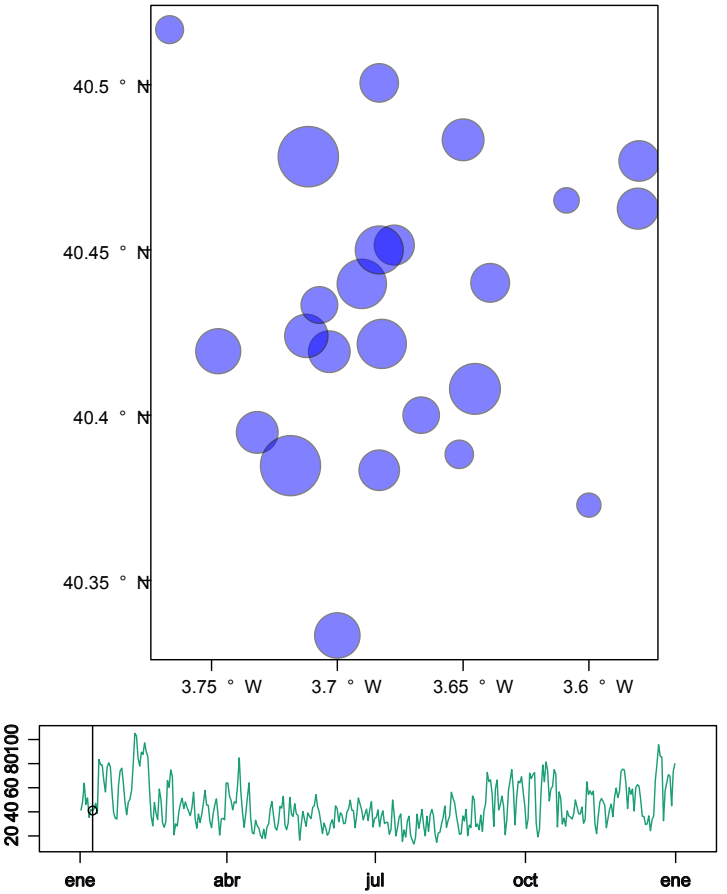
http://oscarperpinan.github.io/spacetime-vis/

Made in R



http://oscarperpinan.github.io/spacetime-vis/

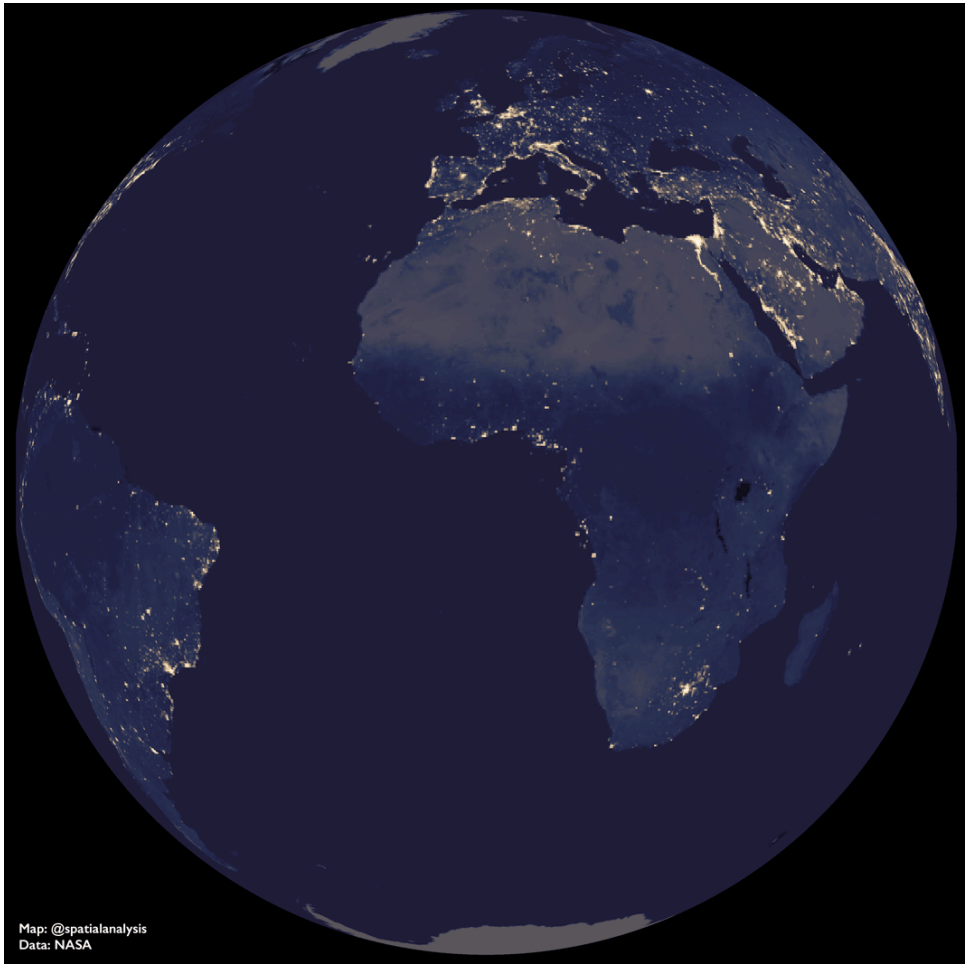
Made in R


40.5 ° N
40.45 ° N
40.4 ° N
40.35 ° N
3.75 ° W
3.7 ° W
3.65 ° W
3.6 ° W
20 40 60 80100
0
ene
abr
jul
oct
ene

Made in R

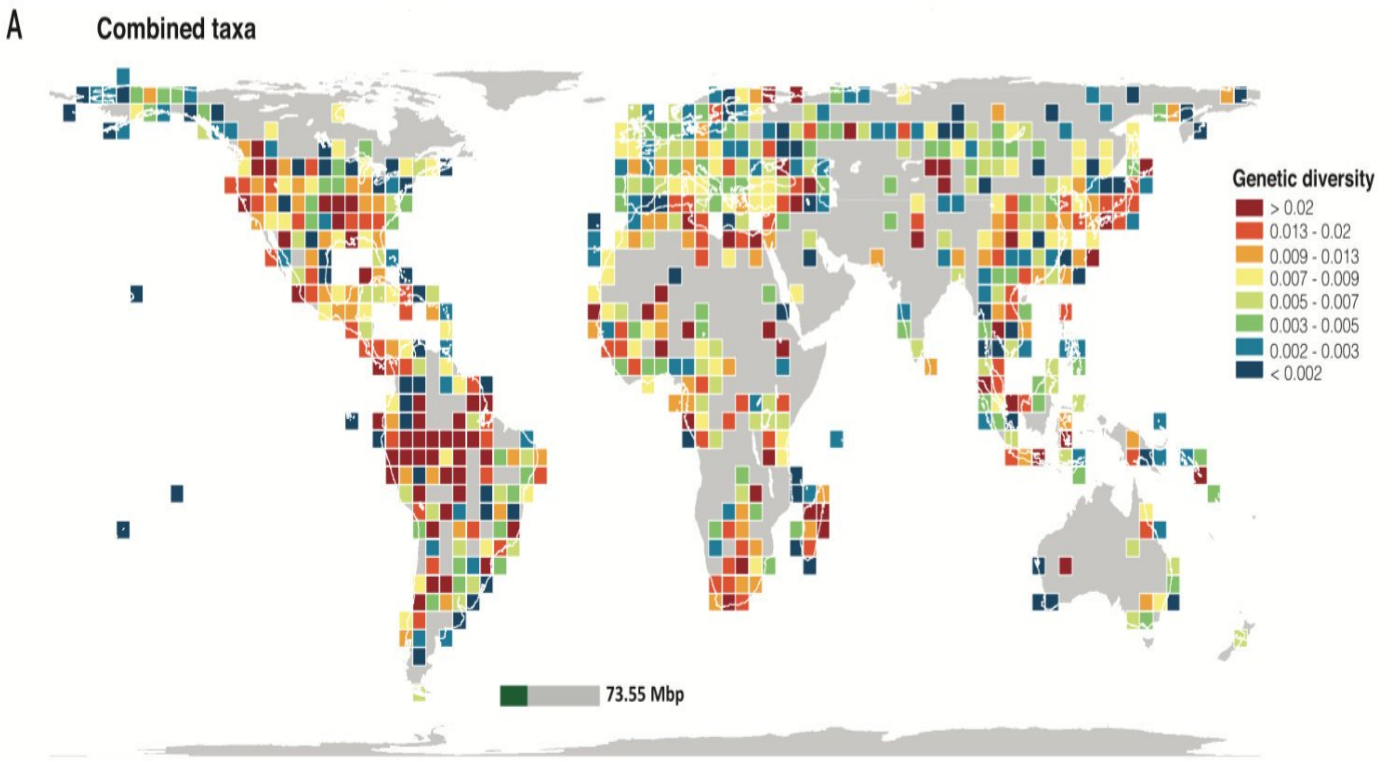
https://rstudio.github.io/leaflet/

Made in R



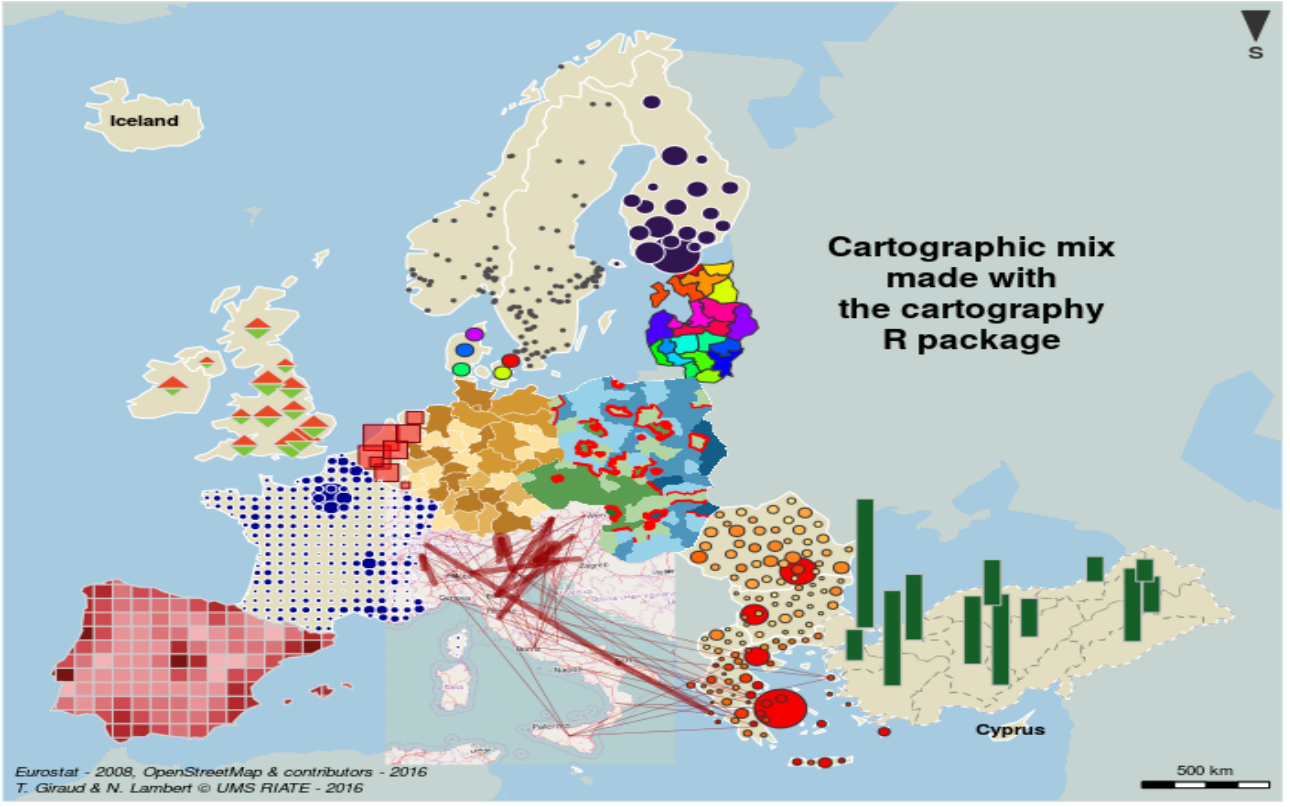
http://spatial.ly/2017/05/spinning-globes-with-r/

Made in R



http://science.sciencemag.org/content/suppl/2016/09/28/353.6307.1532.DC1

Made in R



https://cran.r-project.org/package=cartography

Made in R

https://cran.r-project.org/package=tmap

Made in R

https://github.com/tylermorganwall/rayshader

Made in R

https://github.com/tylermorganwall/rayshader

R can make beautiful maps

And beautiful stats too

Spatial data in R

Using R as a GIS

Basic packages for spatial data

- sf (sp)
- raster (stars)
- rgeos
- rgdal

And many more: see

- Spatial CRAN Task View
- Mapping Task View

Spatial data types in R

- **Vectorial** (sp/sf):
  - Points
  - Lines
  - Polygons
- **Raster**:
  - RasterLayer:
    - 1 grid
  - RasterStack:
    - multiple layers with same extent, resolution & projection
  - RasterBrick:
    - multiple layers (= RasterStack) but stored in a single file

Vector data

Importing vector data

```
library(sf)
countries <- st_read("data/eucountries.shp")  # or gpkg, etc
```

```
## Reading layer `eucountries' from data source `C:\Users\FRS\Dropbox\Rcode\m
## Simple feature collection with 38 features and 4 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: -24.32618 ymin: 34.91999 xmax: 40.08079 ymax: 80.657
## epsg (SRID):    4326
## proj4string:    +proj=longlat +datum=WGS84 +no_defs
```

sf objects are data.frames! (w/ geometry column)

```
head(countries)
```

```
## Simple feature collection with 6 features and 4 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: 2.513573 ymin: 39.625 xmax: 32.69364 ymax: 56.16913
## epsg (SRID):    4326
## proj4string:    +proj=longlat +datum=WGS84 +no_defs
##                name  pop_est gdp_md_est       subregion
## 1           Albania  3639453      21810 Southern Europe
## 2           Austria  8210281     329500  Western Europe
## 3           Belgium 10414336     389300  Western Europe
## 4          Bulgaria  7204687      93750  Eastern Europe
## 5 Bosnia and Herz.  4613414      29700 Southern Europe
## 6           Belarus  9648533     114100  Eastern Europe
##                           geometry
## 1 MULTIPOLYGON (((20.59025 41...
## 2 MULTIPOLYGON (((16.97967 48...
## 3 MULTIPOLYGON (((3.314971 51...
## 4 MULTIPOLYGON (((22.65715 44...
## 5 MULTIPOLYGON (((19.00549 44...
## 6 MULTIPOLYGON (((23.48413 53...
```

So we can easily manipulate them (e.g. dplyr)

Remove column:

```
library(dplyr)
countries <- dplyr::select(countries, -gdp_md_est)
```

So we can easily manipulate them (e.g. dplyr)

Filter cases:

```
west.eu <- filter(countries, subregion == "Western Europe")
west.eu
```

```
## Simple feature collection with 7 features and 3 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: -4.59235 ymin: 41.38001 xmax: 16.97967 ymax: 54.9831
## epsg (SRID):    4326
## proj4string:    +proj=longlat +datum=WGS84 +no_defs
##          name  pop_est      subregion                       geometry
## 1     Austria  8210281 Western Europe MULTIPOLYGON (((16.97967 48...
## 2     Belgium 10414336 Western Europe MULTIPOLYGON (((3.314971 51...
## 3 Switzerland  7604467 Western Europe MULTIPOLYGON (((9.594226 47...
## 4     Germany 82329758 Western Europe MULTIPOLYGON (((9.921906 54...
## 5      France 64057792 Western Europe MULTIPOLYGON (((9.560016 42...
## 6  Luxembourg   491775 Western Europe MULTIPOLYGON (((6.043073 50...
## 7 Netherlands 16715999 Western Europe MULTIPOLYGON (((6.074183 53...
```

So we can easily manipulate them (e.g. dplyr)

Summarise data:

```
countries %>%
  group_by(subregion) %>%
  summarise(mean(pop_est))
```

```
## Simple feature collection with 4 features and 2 fields
## geometry type:  GEOMETRY
## dimension:      XY
## bbox:           xmin: -24.32618 ymin: 34.91999 xmax: 40.08079 ymax: 80.657
## epsg (SRID):    4326
## proj4string:    +proj=longlat +datum=WGS84 +no_defs
## # A tibble: 4 x 3
##   subregion    `mean(pop_est)`                                      geometr
##   <fct>                  <dbl>                                <GEOMETRY [Â°]
## 1 Eastern Eur~       17017028. POLYGON ((28.55808 43.70746, 28.0391 43.293
## 2 Northern Eu~        9834469. MULTIPOLYGON (((-6.197885 53.86757, -6.0329
## 3 Southern Eu~       12230634. MULTIPOLYGON (((23.69998 35.705, 24.24667 3
## 4 Western Eur~       27117773. MULTIPOLYGON (((9.560016 42.15249, 9.229752
```

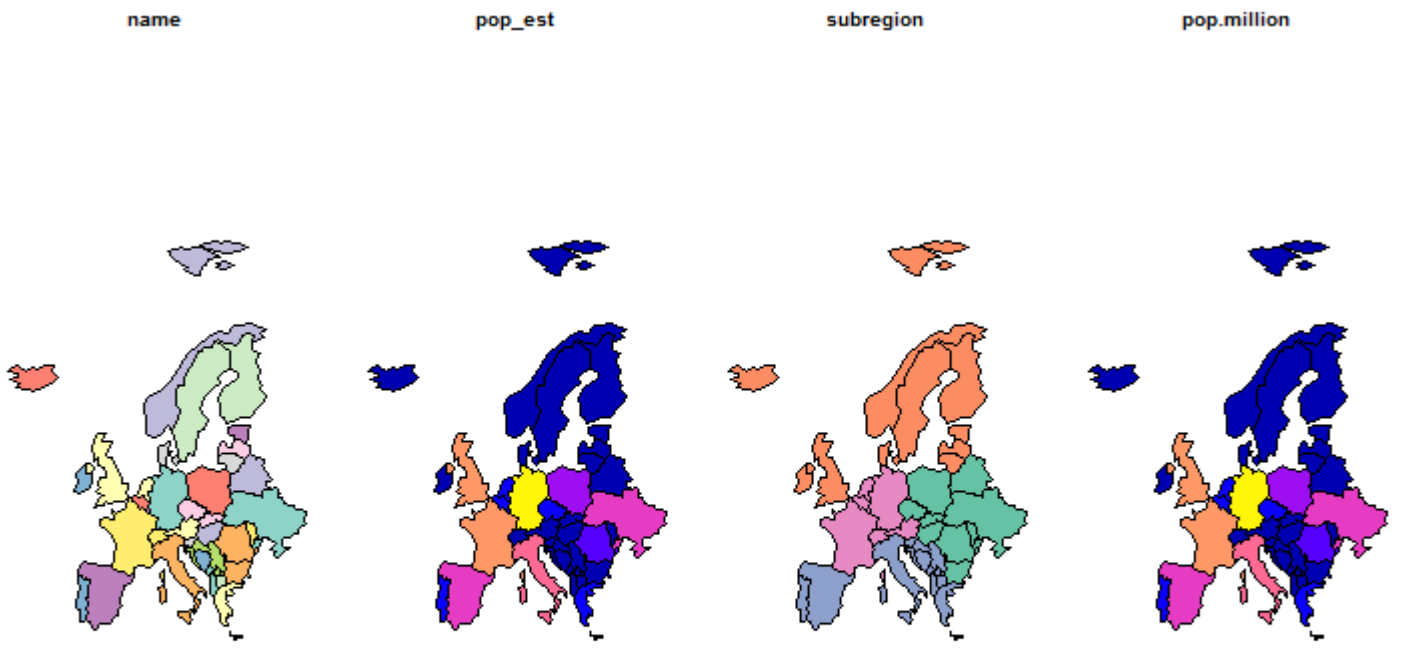
So we can easily manipulate them (e.g. dplyr)

Create new columns:

```
countries <- mutate(countries, pop.million = pop_est/1000000)
```

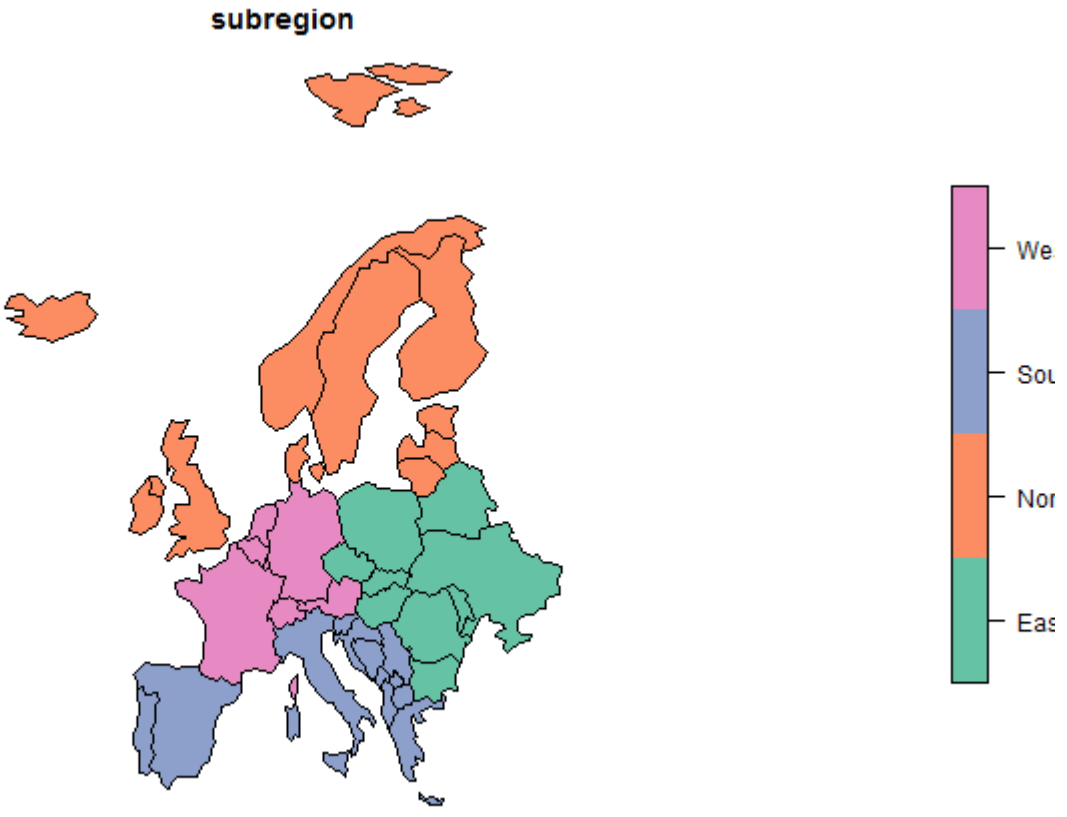
Basic plotting

```
plot(countries)
```



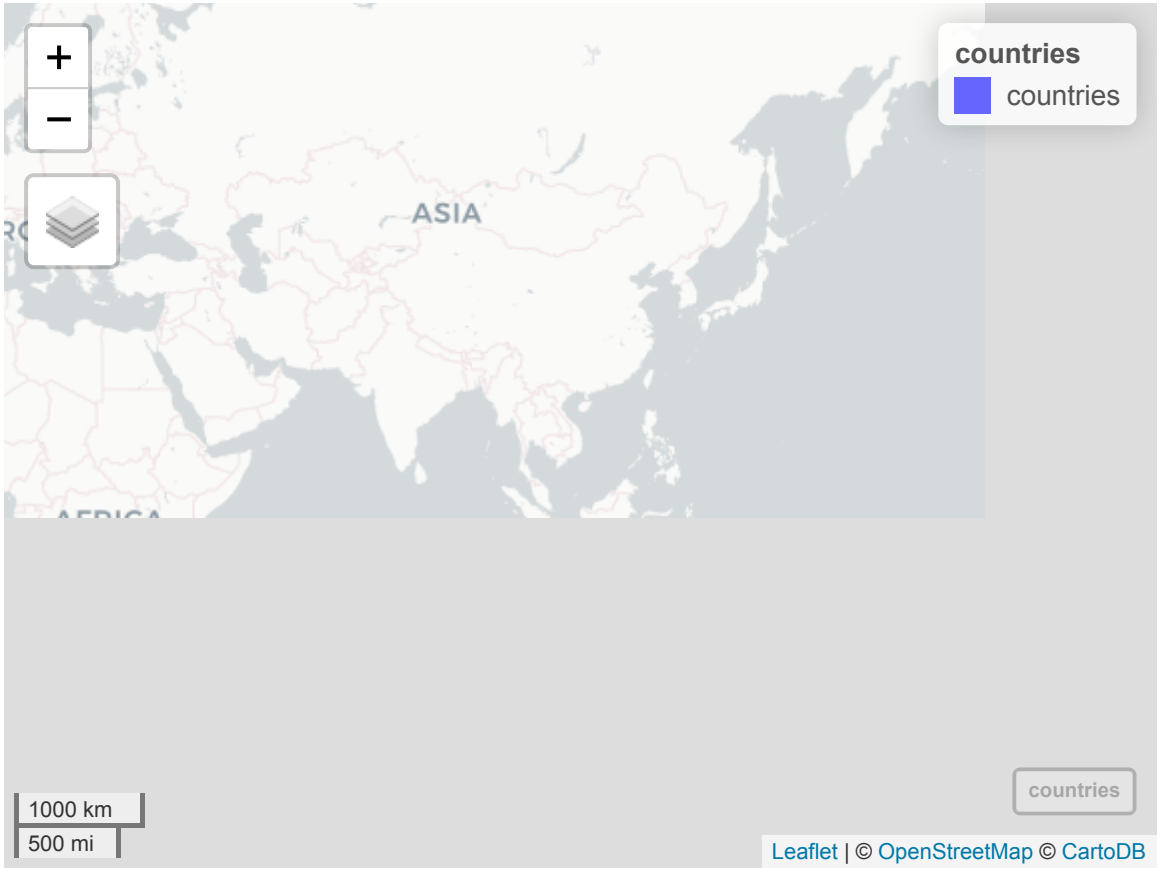
Basic plotting

```
plot(countries["subregion"])
```



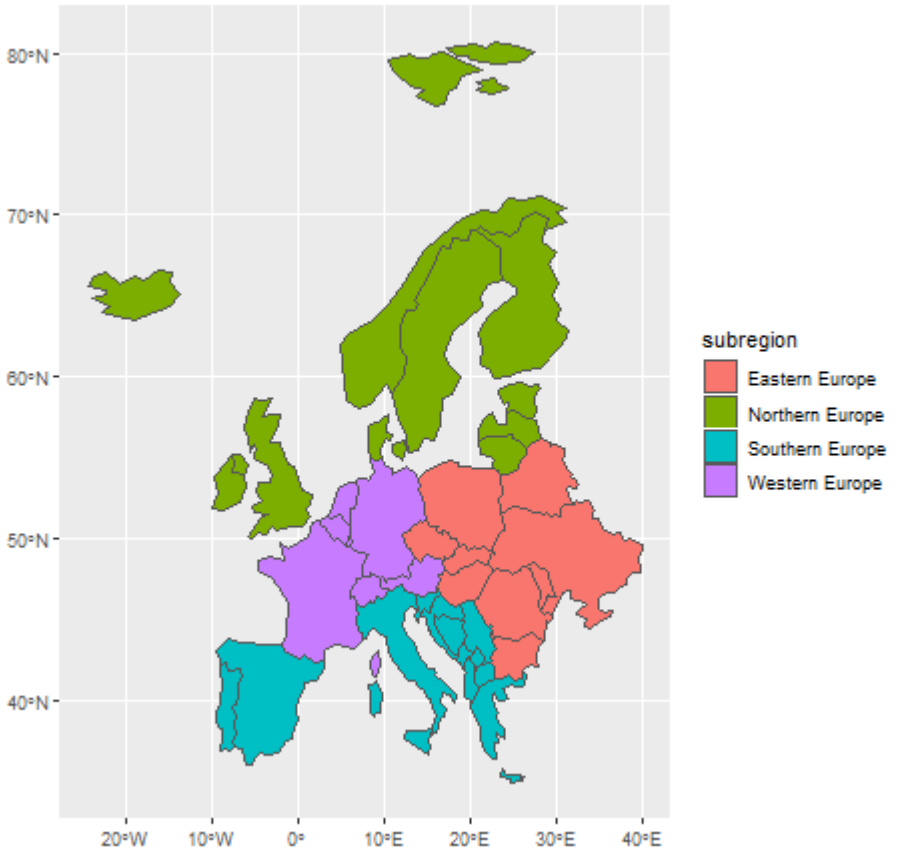
Interactive plot (leaflet)

```
library(mapview)
mapview(countries)
```



Plotting sf objects with ggplot2

```
ggplot() +
  geom_sf(data = countries, aes(fill = subregion))
```

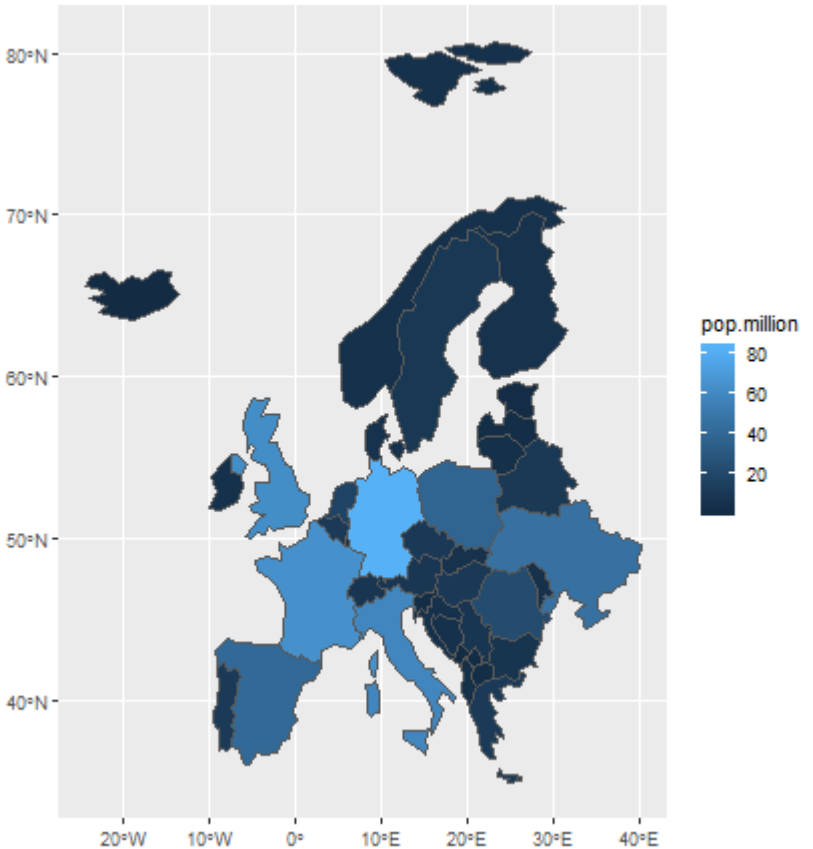


Plotting sf objects with ggplot2

```
ggplot() +
  geom_sf(data = countries, aes(fill = name)) +
  theme(legend.position = "none")
```

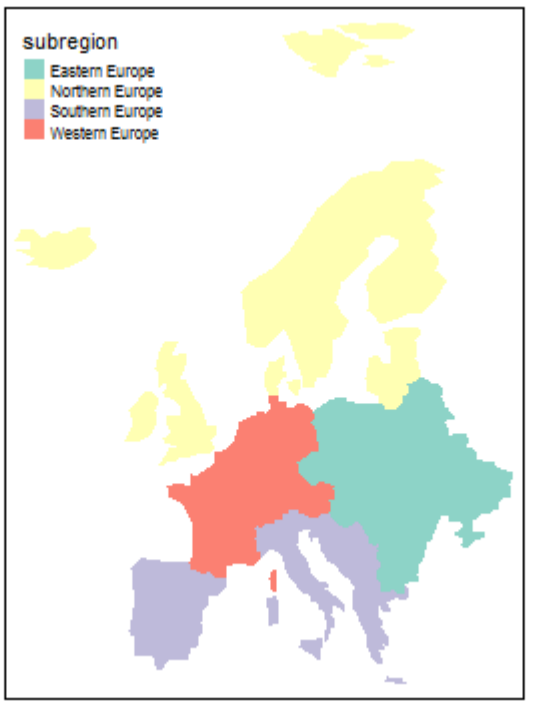
Plotting sf objects with ggplot2

```
ggplot() +
  geom_sf(data = countries, aes(fill = pop.million))
```



Plotting with tmap

```
library(tmap)
tm_shape(countries) +
  tm_fill(col = "subregion") +
  tm_layout(legend.position = c("left", "top"))
```



How to create sf from an R object?

Making a data frame *spatial*

```
mydf <- read.csv("data/occs.csv")
head(mydf)
```

```
##                  species      x      y
## 1 Solanum acaule Bitter -66.10 -21.90
## 2 Solanum acaule Bitter -71.00 -13.50
## 3 Solanum acaule Bitter -66.43 -24.22
## 4 Solanum acaule Bitter -72.07 -13.35
## 5 Solanum acaule Bitter -68.97 -15.23
## 6 Solanum acaule Bitter -64.95 -17.75
```

Making a data frame *spatial*

```
occs <- st_as_sf(mydf, coords = c("x", "y"))
head(occs)
```

```
## Simple feature collection with 6 features and 1 field
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: -72.07 ymin: -24.22 xmax: -64.95 ymax: -13.35
## epsg (SRID):    NA
## proj4string:    NA
##                 species              geometry
## 1 Solanum acaule Bitter   POINT (-66.1 -21.9)
## 2 Solanum acaule Bitter     POINT (-71 -13.5)
## 3 Solanum acaule Bitter POINT (-66.43 -24.22)
## 4 Solanum acaule Bitter POINT (-72.07 -13.35)
## 5 Solanum acaule Bitter POINT (-68.97 -15.23)
## 6 Solanum acaule Bitter POINT (-64.95 -17.75)
```

Setting the projection (Coordinate Reference System)

```
st_crs(occs) <- "+proj=longlat +ellps=WGS84 +datum=WGS84"
```

See http://spatialreference.org

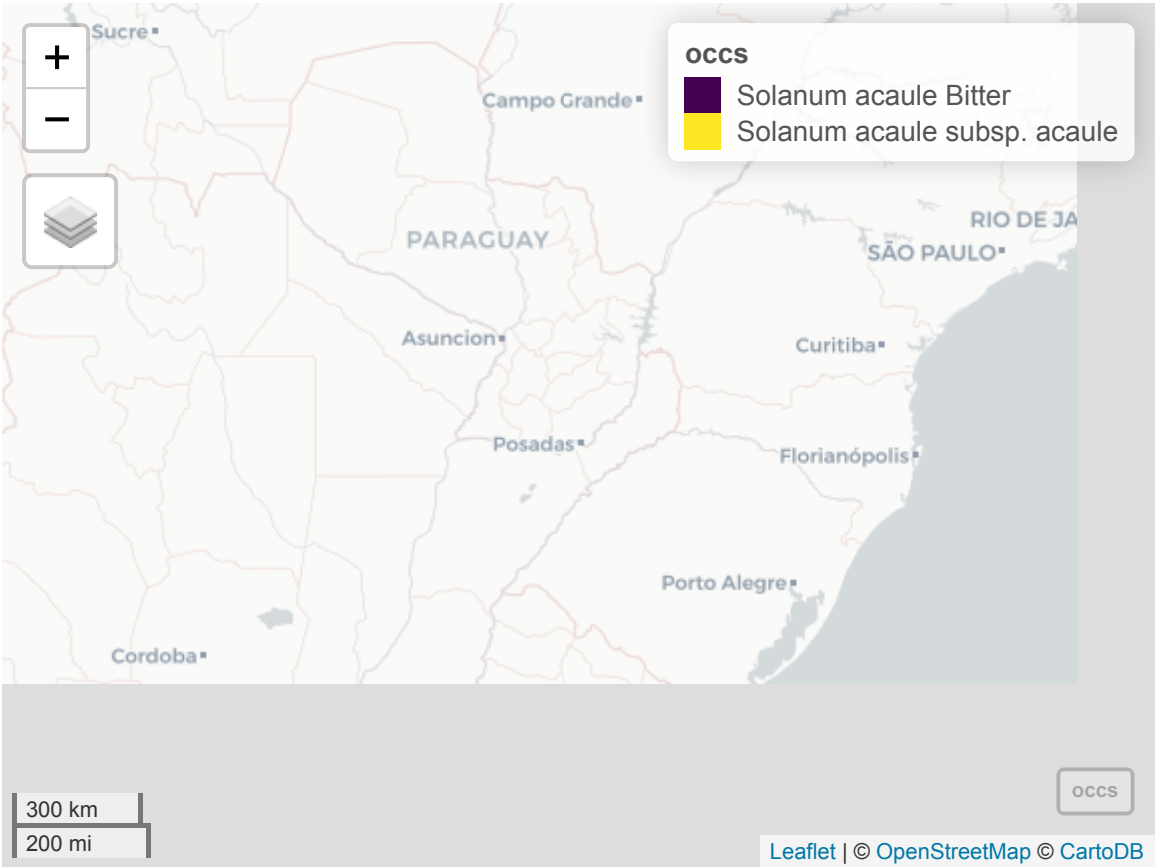
Changing projection

```
occs.laea <- st_transform(occs, crs = 3035)
occs.laea
```

```
## Simple feature collection with 49 features and 1 field
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: -4831320 ymin: -1011814 xmax: -4354226 ymax: 1006172
## epsg (SRID):    3035
## proj4string:    +proj=laea +lat_0=52 +lon_0=10 +x_0=4321000 +y_0=3210000 +
## First 10 features:
##                  species                   geometry
## 1  Solanum acaule Bitter POINT (-4514047 -755397.4)
## 2  Solanum acaule Bitter  POINT (-4750733 726537.4)
## 3  Solanum acaule Bitter   POINT (-4560712 -998976)
## 4  Solanum acaule Bitter  POINT (-4827141 862365.9)
## 5  Solanum acaule Bitter  POINT (-4639318 310914.9)
## 6  Solanum acaule Bitter POINT (-4354641 -389505.2)
## 7  Solanum acaule Bitter POINT (-4425100 -548771.6)
## 8  Solanum acaule Bitter POINT (-4354226 -588275.4)
## 9  Solanum acaule Bitter  POINT (-4667605 73818.71)
## 10 Solanum acaule Bitter  POINT (-4485788 -1011814)
```

Leaflet map (mapview)

```
mapview(occs)
```

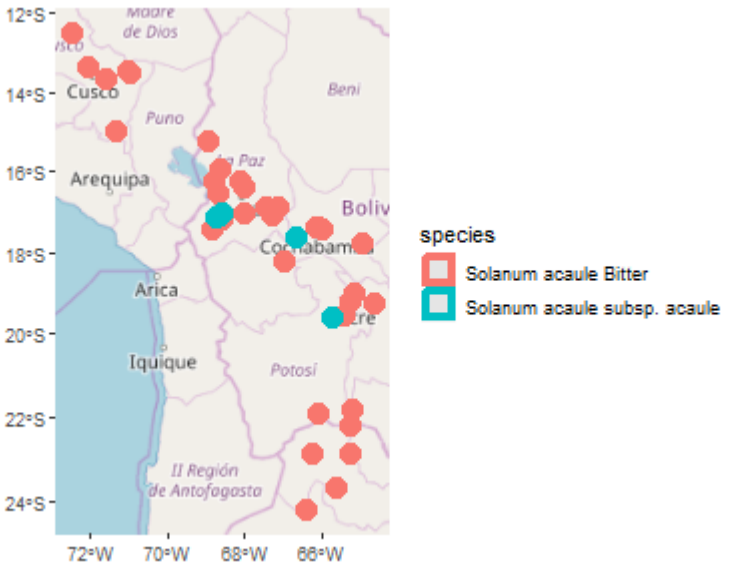


Leaflet map (leaflet)

```
leaflet(occs) %>%
  addTiles() %>%
  addMarkers(popup = ~species)
```

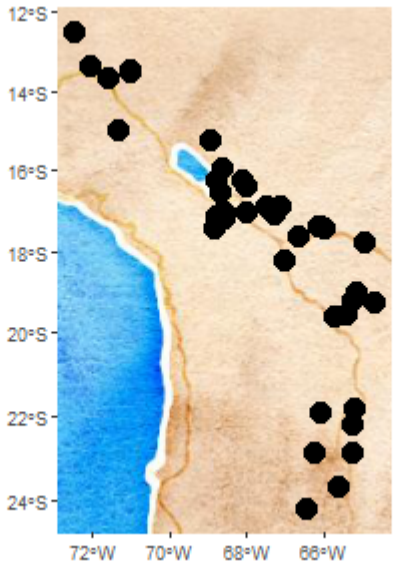
Plotting with ggspatial

```
library(ggspatial)
ggplot() +
  annotation_map_tile() +
  layer_spatial(occs, size = 5, aes(colour = species))
```



Plotting with ggspatial

```
ggplot() +
  annotation_map_tile(type = "stamenwatercolor") +
  layer_spatial(occs, size = 5)
```



Basemap Copyright OpenStreetMap Contributors CC-BY-SA

Convert sf to Spatial* object (sp)

```
occs.sp <- as(occs, "Spatial")
occs.sp
```

```
## class       : SpatialPointsDataFrame
## features    : 49
## extent      : -72.5, -64.67, -24.22, -12.5  (xmin, xmax, ymin, ymax)
## coord. ref. : +proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,
## variables   : 1
## names       :                       species
## min values  :        Solanum acaule Bitter
## max values  : Solanum acaule subsp. acaule
```

How to save/export vector data?

Saving vector data

```
st_write(countries, "data/countries.gpkg", delete_dsn = TRUE)
```

```
## Deleting source `data/countries.gpkg' using driver `GPKG'
## Writing layer `countries' to data source `data/countries.gpkg' using drive
## features:       38
## fields:         4
## geometry type:  Multi Polygon
```

Geocoding

Geocoding

```
myplace <- tmaptools::geocode_OSM("Granada, Spain")
myplace
```

```
## $query
## [1] "Granada, Spain"
##
## $coords
##         x          y
## -3.602193 37.183054
##
## $bbox
##      xmin      ymin      xmax      ymax
## -3.700362 37.136703 -3.496324 37.225017
```

Other packages: opencage, dismo, ggmap, geocodeHERE, rmapzen...

Raster data

Download raster (and vector) data

```
library(raster)
bioclim <- getData('worldclim', var = "bio", res = 10)
bioclim
```

```
## class       : RasterStack
## dimensions  : 900, 2160, 1944000, 19  (nrow, ncol, ncell, nlayers)
## resolution  : 0.1666667, 0.1666667  (x, y)
## extent      : -180, 180, -60, 90  (xmin, xmax, ymin, ymax)
## coord. ref. : +proj=longlat +datum=WGS84 +ellps=WGS84 +towgs84=0,0,0
## names       :  bio1,  bio2,  bio3,  bio4,  bio5,  bio6,  bio7,  bio8,  bio
## min values  :  -269,     9,     8,    72,   -59,  -547,    53,  -251,  -45
## max values  :   314,   211,    95, 22673,   489,   258,   725,   375,   36
```

Importing raster data from disk

One grid only (1 layer):

```
ras <- raster("wc10/bio1.bil")
ras
```

```
## class       : RasterLayer
## dimensions  : 900, 2160, 1944000  (nrow, ncol, ncell)
## resolution  : 0.1666667, 0.1666667  (x, y)
## extent      : -180, 180, -60, 90  (xmin, xmax, ymin, ymax)
## coord. ref. : +proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,
## data source : C:/Users/FRS/Dropbox/Rcode/myRcode/courses_talks/GISwithR/wc
## names       : bio1
## values      : -269, 314  (min, max)
```

Importing raster data from disk

Multiple grids:

```
files <- list.files("wc10", pattern = "bio\\d+.bil", full.names = TRU
manylayers <- stack(files)
manylayers
```

```
## class       : RasterStack
## dimensions  : 900, 2160, 1944000, 19  (nrow, ncol, ncell, nlayers)
## resolution  : 0.1666667, 0.1666667  (x, y)
## extent      : -180, 180, -60, 90  (xmin, xmax, ymin, ymax)
## coord. ref. : +proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,
## names       :  bio1, bio10, bio11, bio12, bio13, bio14, bio15, bio16, bio1
## min values  :  -269,   -97,  -488,     0,     0,     0,     0,     0,
## max values  :   314,   380,   289,  9916,  2088,   652,   261,  5043,  215
```

Setting the projection (Coordinate Reference System)

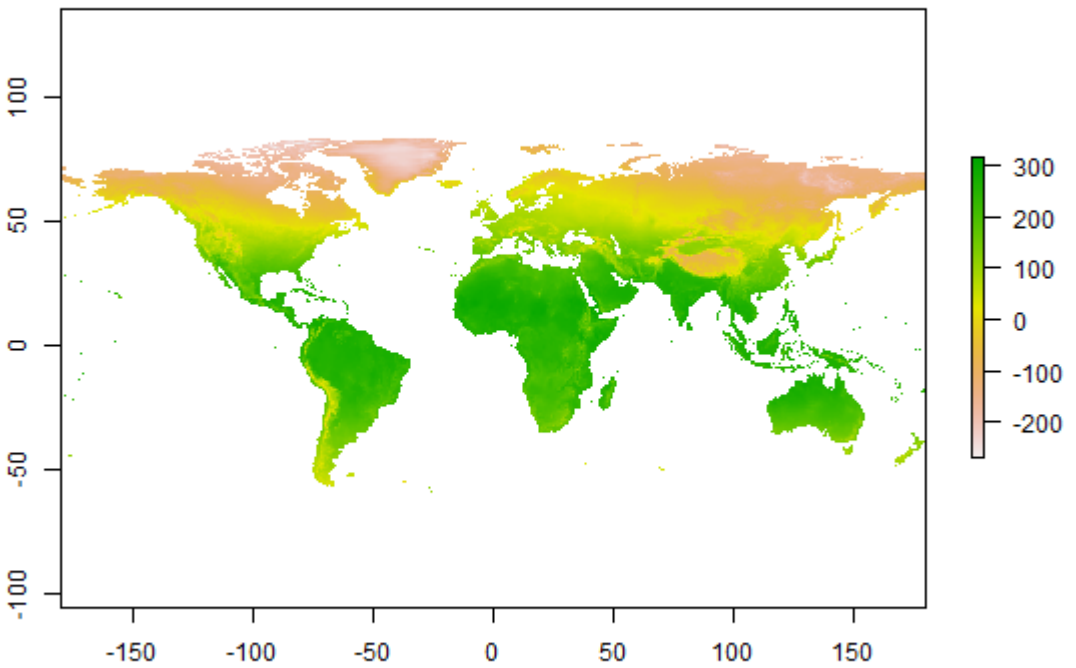
```
crs(ras) <- "+proj=longlat +ellps=WGS84 +datum=WGS84"
```

See http://spatialreference.org

To change projection: `projectRaster`

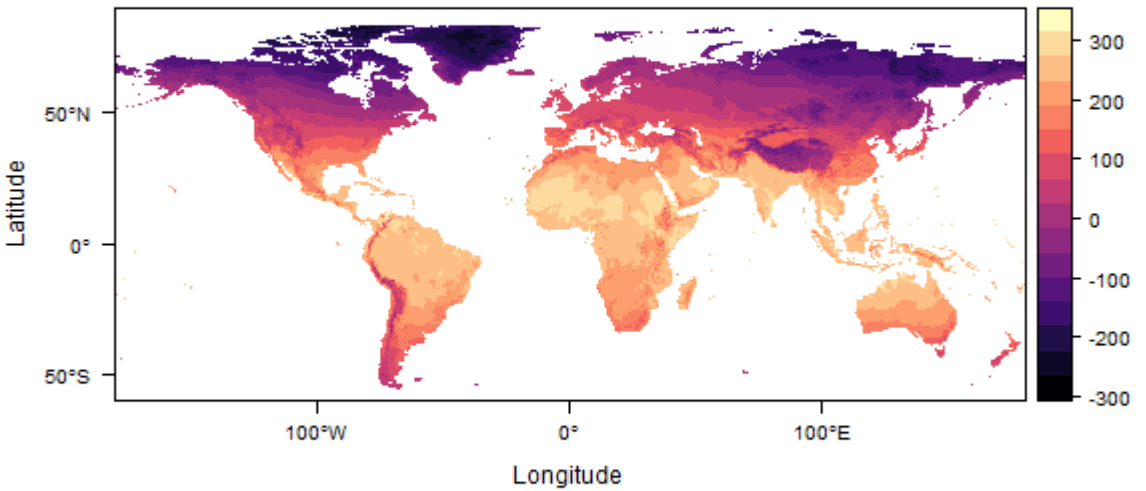
Basic raster plotting

```
plot(ras)
```



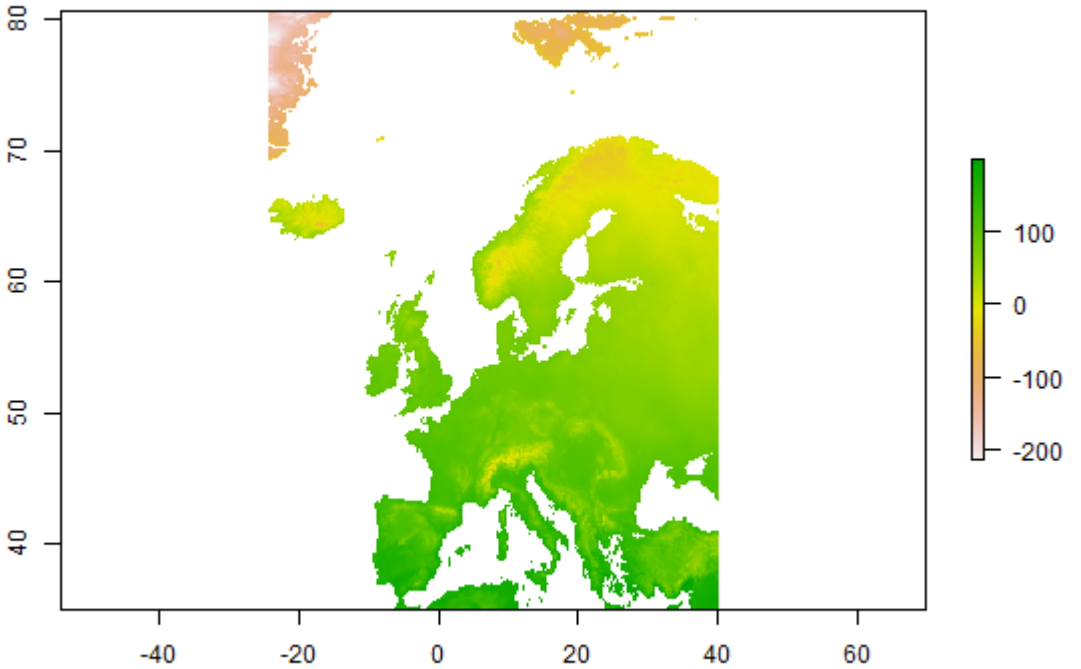
rasterVis

```
library(rasterVis)
levelplot(ras, margin = FALSE)
```



Crop (change extent)

```
ras.crop <- crop(ras, countries)
plot(ras.crop)
```



Plot raster with ggplot2

```
ras.df <- as.data.frame(ras.crop, xy = TRUE)
ggplot(ras.df) +
  geom_raster(aes(x = x, y = y, fill = bio1))
```

Change resolution

```
ras.coarse <- aggregate(ras.crop, fact = 4, fun = mean)
ras.coarse
```

```
## class       : RasterLayer
## dimensions  : 69, 97, 6693  (nrow, ncol, ncell)
## resolution  : 0.6666667, 0.6666667  (x, y)
## extent      : -24.33333, 40.33333, 34.66667, 80.66667  (xmin, xmax, ymin,
## coord. ref. : +proj=longlat +ellps=WGS84 +datum=WGS84 +towgs84=0,0,0
## data source : in memory
## names       : bio1
## values      : -206.375, 194  (min, max)
```

Extract values from rasters

```
vals <- extract(ras, occs)
head(vals)
```

```
## [1]  88 100  52  98  86 154
```

Extract values from rasters

```
vals <- extract(ras, countries, fun = mean)
head(vals)
```

```
##           [,1]
## [1,] 118.00000
## [2,]  57.86376
## [3,]  95.43704
## [4,] 102.15935
## [5,]  90.21891
## [6,]  62.47996
```

Save raster data

```
writeRaster(ras, filename = "myraster.grd")
```

KML (Google Earth):

```
KML(ras, filename = "myraster.kmz", overwrite = TRUE)
```

Remote sensing too

http://bleutner.github.io/RStoolbox/

And many more packages! (MODIS, Landsat, LiDAR...)

Why doing GIS in R

- Harness all **stats & data management power** from R
  - Data wrangling
  - Modelling
  - Dataviz
- Fully-**reproducible** scripts

Running GIS geoprocessing algorithms from R

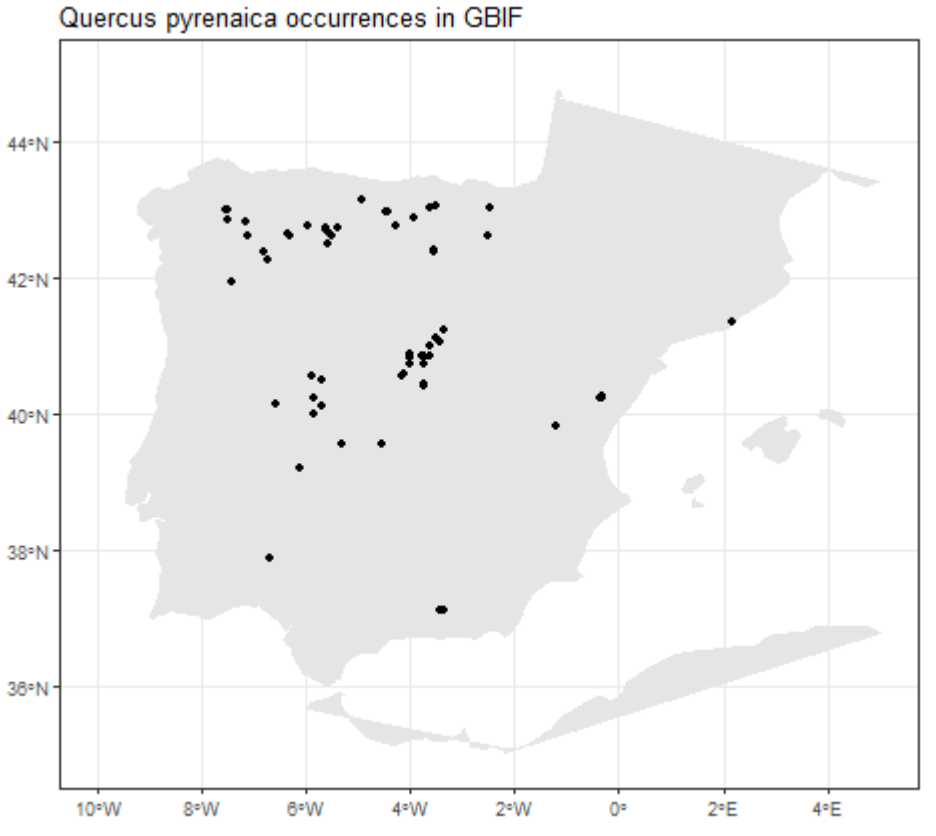
- RQGIS
- rgrass7
- RSAGA
- ArcGIS

Some tutorials

- https://geocompr.robinlovelace.net/
- https://bhaskarvk.github.io/user2017.geodataviz/
- https://github.com/Nowosad/gis_with_r_how_to_start
- http://www.rspatial.org/
- http://r-spatial.github.io/sf/
- http://book.ecosens.org/
- http://pakillo.github.io/R-GIS-tutorial
- https://datacarpentry.org/geospatial-workshop/
- http://jafflerbach.github.io/spatial-analysis-R/intro_spatial_data_R.html
- https://github.com/USEPA/intro_gis_with_r
- http://www.nickeubank.com/gis-in-r/
- Spatial R cheatsheet

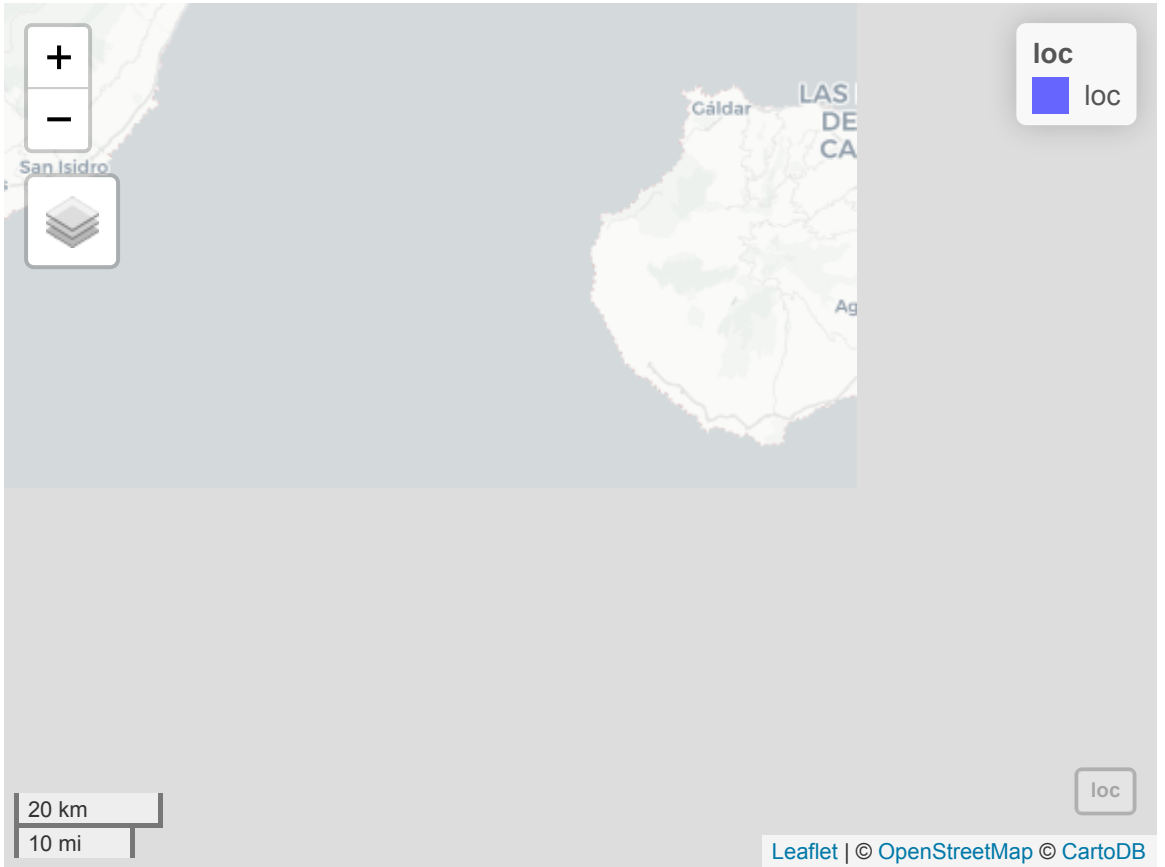
Exercises

Map distribution of species occurrences (rgbif)

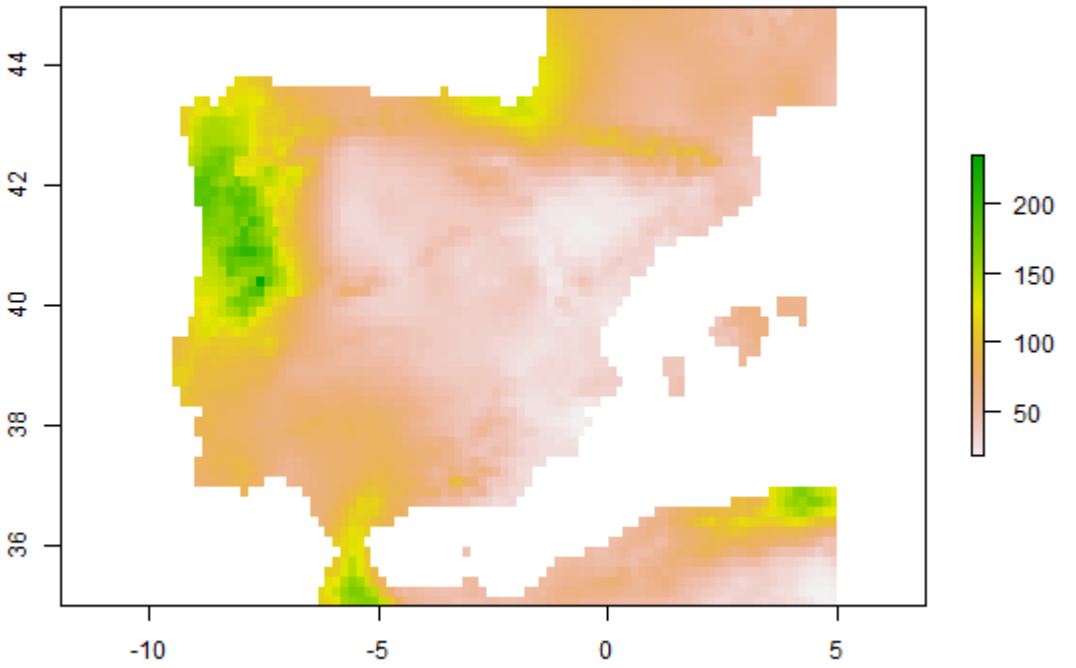


Geocode and map address

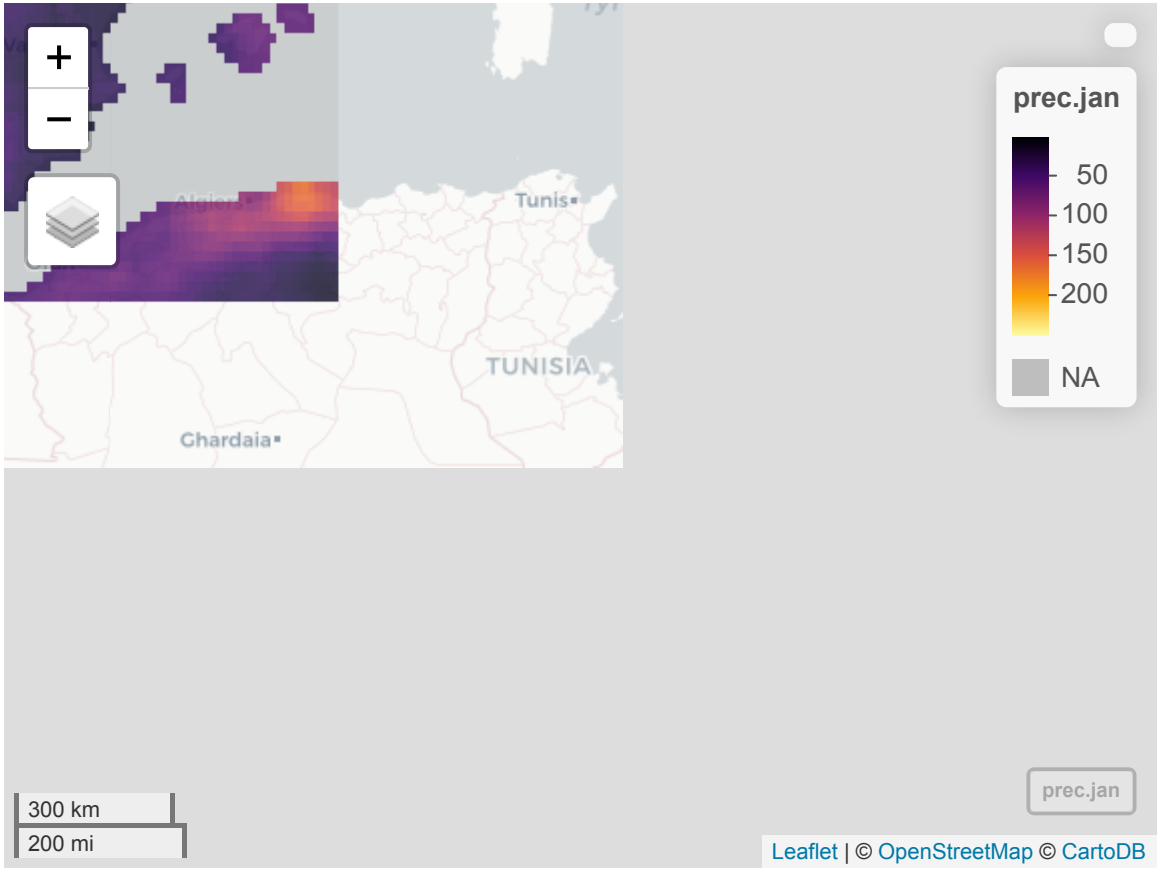
Interactive map



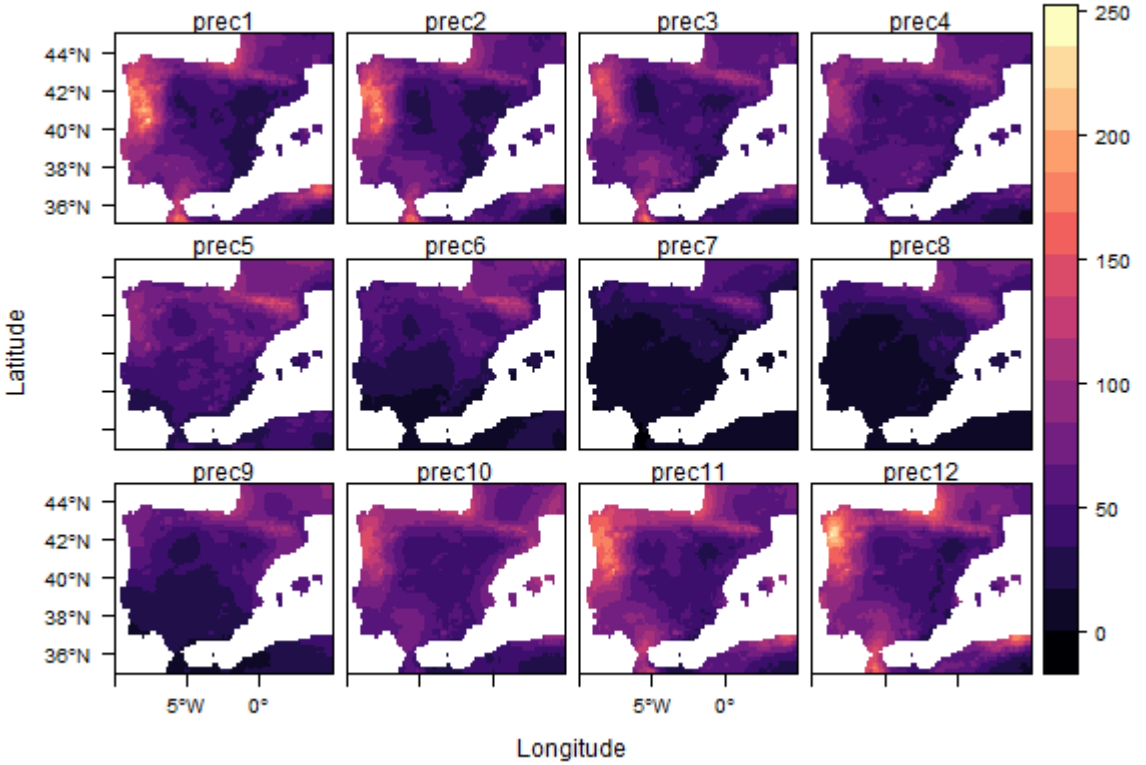
January Precipitation in Spain (raster)



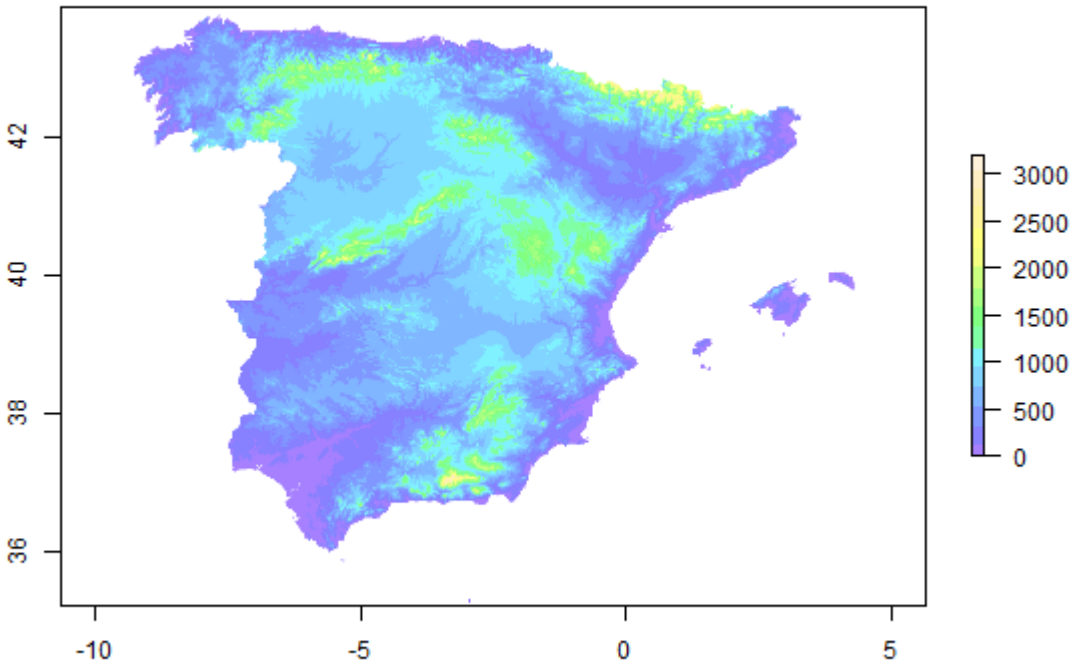
January Precipitation in Spain (leaflet)



Monthly Precipitation in Spain (rasterVis)



Elevation map of Spain



END

Slides and source code available at https://github.com/Pakillo/GISwithR