Technical Guidelines Series

Part 1 - Conversion to the Robinson Projection

Prepared by Joy Kumagai - Technical Support Unit (TSU) of Knowledge and Data
Reviewed by Aidin Niamir - Head of the Technical Support Unit of Knowledge and Data
For any inquires please contact tsu.data@ipbes.net

Version: 2.1
Last Updated: 15 July 2022

DOI: 10.5281/zenodo.6840235

The guide will show how to convert raster and vector data from a projection to the Robinson projection using R and is intended for IPBES experts creating maps. IPBES has adopted the Robinson projection for visualizing global scale maps as it balances distortions in area, direction, distance, and distortions near the poles. Please note that the projection used for analysis and whether one would like to display a Pacific centered or Greenwich centered map depends on the application. For example, If calculating area, an equal-area projection is needed.

Begin by loading the following packages.

```
library(terra)
library(raster)
library(graticule)
library(rgdal)
library(rworldmap)
```

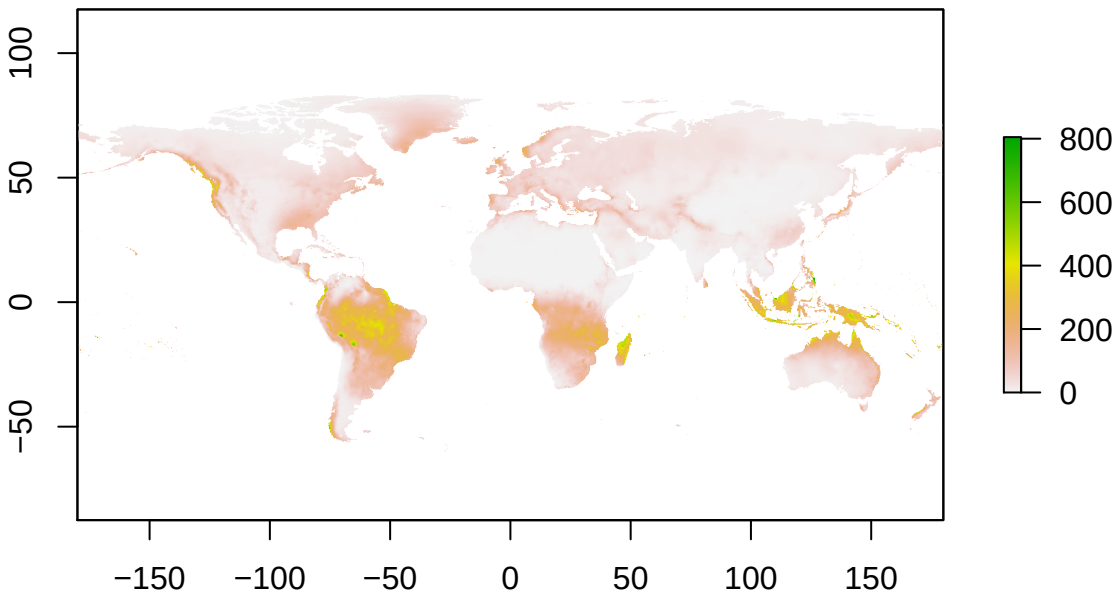
Next, we define the Robinson projection using PROJ.4 notation. This information can be found at the this link.

```
crs <- "+proj=robin +lon_0=0 +x_0=0 +y_0=0 +datum=WGS84 +units=m"
```

I. Raster Data

To produce a Greenwich centered map in the Robinson projection with raster data, we will load precipitation data from world clim and plot the result, but please load and use the raster data that you will be working with. The crs() function will output the projection that your data is in.

```
prec <- raster::getData(name = "worldclim", var = "prec", res = 10)[[1]] # load data
plot(prec)
```




```
raster::crs(prec)
```

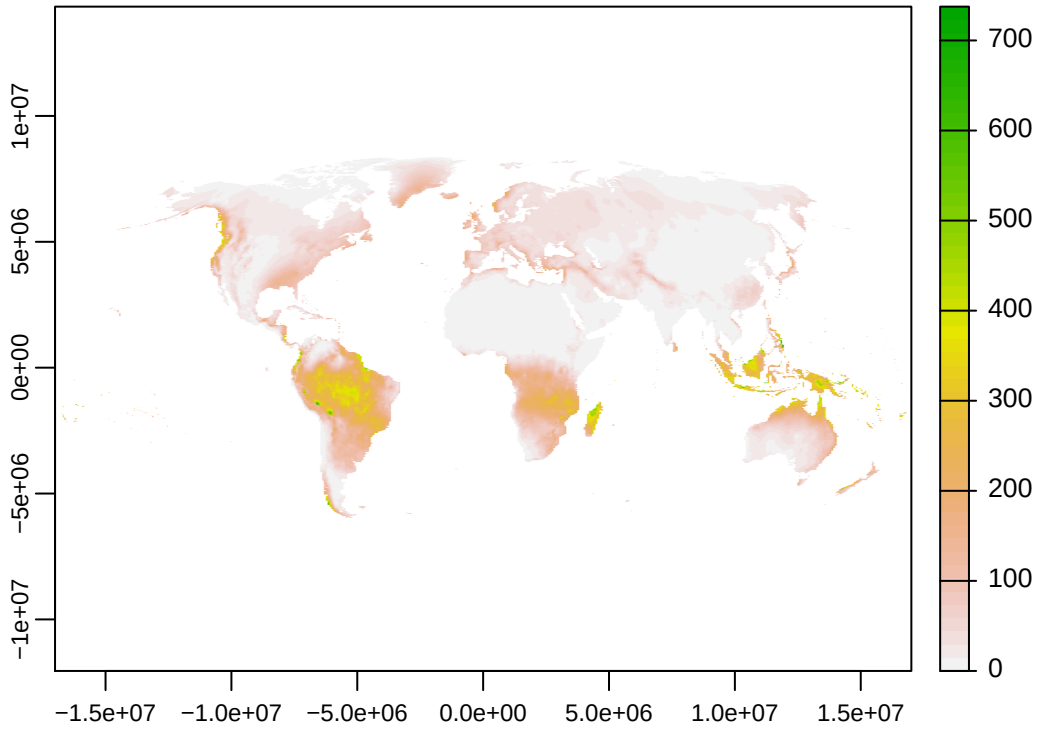
```
## CRS arguments: +proj=longlat +datum=WGS84 +no_defs
```

Next, we will create a mask, and then project the data with this mask so repeated areas of the world map are removed.

```
jp <- terra::rast(prec$prec1)
jp <- jp * 1 # to deal with NAs in this datasaet
rob <- terra::project(jp, crs, mask=TRUE)
```

Now we can plot our results to ensure it is correct.

```
plot(rob) #Plot the raster
```



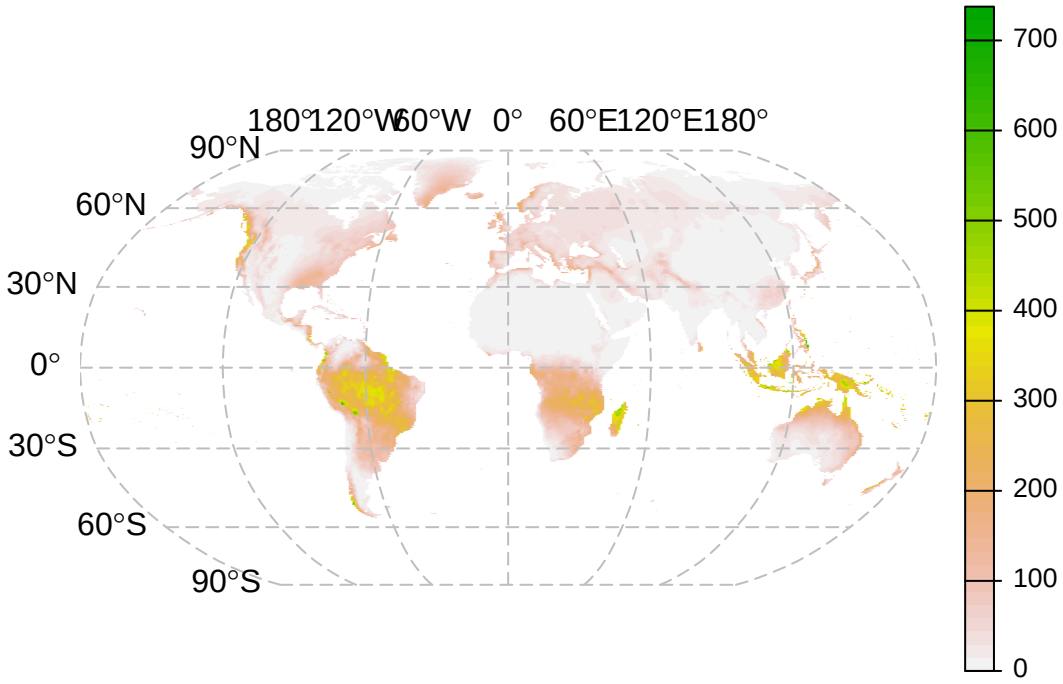
The next step is to create graitcules and labels to add to all of the next plots.

```
# Creates latitude and longitude labels and graticules
lat <- c(-90, -60, -30, 0, 30, 60, 90)
long <- c(-180, -120, -60, 0, 60, 120, 180)
labs <- graticule::graticule_labels(lons = long, lats = lat, xline = -180, yline = 90, proj = crs) # la
lines <- graticule::graticule(lons = long, lats = lat, proj = crs) # graticules
```

The warnings of discarding the datum can be safely ignored in this case*.

Finally, we will plot the raster with the graticules we just created.

```
plot(rob,axes = F) #Plot the raster
plot(lines, lty = 5, col = "grey", add = TRUE) # plots graticules
text(subset(labs, labs$islon), lab = parse(text = labs$lab[labs$islon]), pos = 3, xpd = NA) # plots lon
text(subset(labs, !labs$islon), lab = parse(text = labs$lab[!labs$islon]), pos = 2, xpd = NA) # plots l
```



II. Vector Data

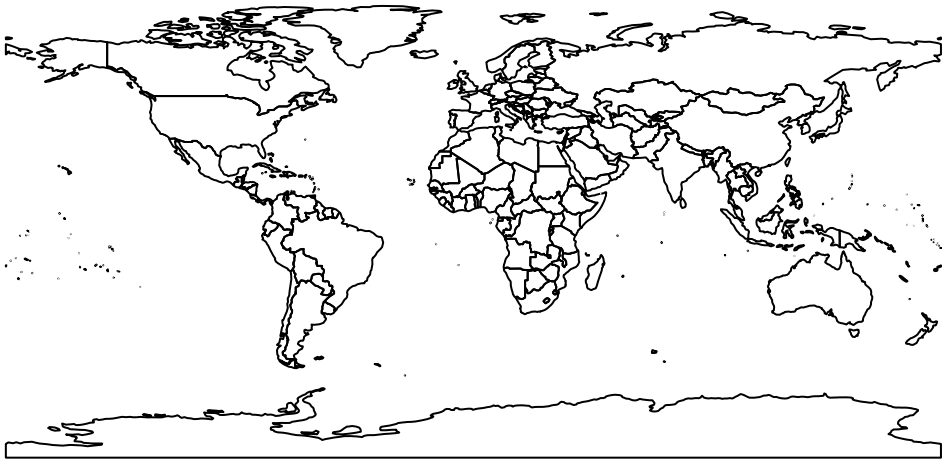
In this section, we will produce a world map of countries with vector data (points, lines, polygons). First, we will load, check the projection, and plot a world map of countries.

```
worldmap <- rworldmap::getMap(resolution = "coarse") # Load countries
raster::crs(worldmap)
```

```
## CRS arguments:
##  +proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs
```

```
plot(worldmap) # Plot world map (vector)
```

```
## Warning in wkt(obj): CRS object has no comment
```



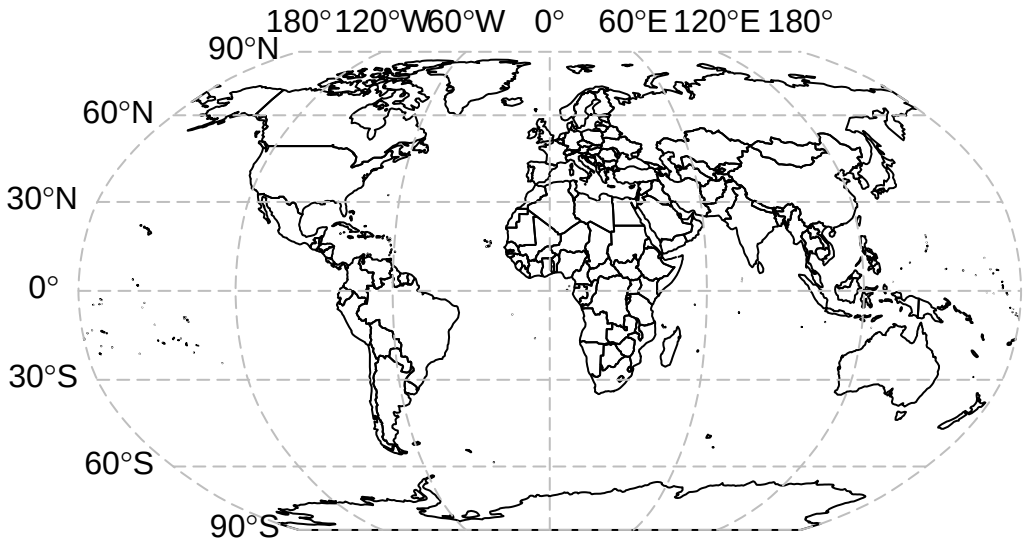
Now, we project the vector data into the Robinson projection and plot the map with graticules.

```
worldmap <- sp::spTransform(worldmap,crs) # Project to Robinson
```

```
## Warning in spTransform(xSP, CRSobj, ...): NULL source CRS comment, falling back
## to PROJ string
```

```
## Warning in wkt(obj): CRS object has no comment
```

```
plot(worldmap)
plot(lines, lty = 5, col = "grey", add = TRUE) # plots graticules
text(subset(labs, labs$islon), lab = parse(text = labs$lab[labs$islon]), pos = 3, xpd = NA) # plots lon
text(subset(labs, !labs$islon), lab = parse(text = labs$lab[!labs$islon]), pos = 2, xpd = NA) # plots l
```

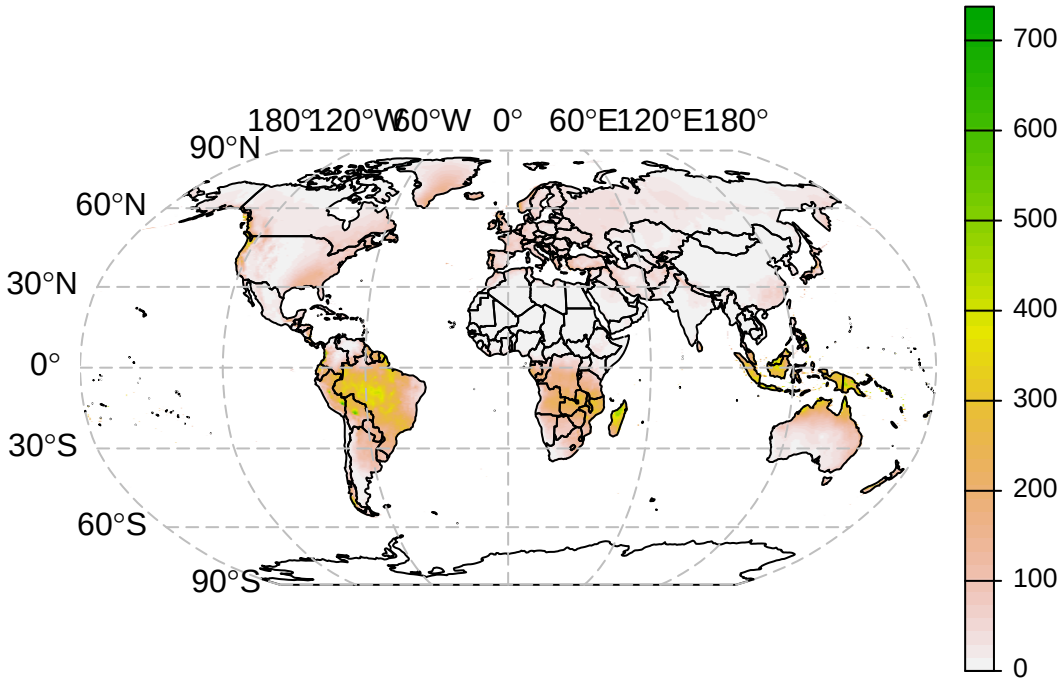


If there are polygons that cross the date line, it is possible that some erroneous polygons may appear. If this occurs, the solution is to run the function st_wrap_dateline() before projecting.

III. Combine Vector and Raster

Finally, we will visualize both the raster and vector data together in one map.

```
## Vector & Raster
plot(rob,axes = F) # Plot the raster
plot(worldmap,add=T) # Plot the vector data
plot(lines, lty = 5, col = "grey", add = TRUE) # plots graticules
text(subset(labs, labs$islon), lab = parse(text = labs$lab[labs$islon]), pos = 3, xpd = NA) # plots lon
text(subset(labs, !labs$islon), lab = parse(text = labs$lab[!labs$islon]), pos = 2, xpd = NA) # plots l
```



Your feedback on this content is welcome. Let us know what other useful material would you like to see here by emailing tsu.data@ipbes.net

*The warnings of discarding the datum but preserving the `+towgs1984 = values` stem from an update from PROJ4 to PRROJ6 but is not worriesome in this case. The `+datum=` part is depreciated from GDAL >3 and sf, rgdal, and raster packages use GDAL to read files. There is a stackoverflow thread with more information here