A few months ago I have made an attempt to visualize the world population changes from 1800 to 2100:

Inspired by [@MaxCRoser](https://twitter.com/MaxCRoser?ref_src=twsrc%5Etfw) and [@jkottke](https://twitter.com/jkottke?ref_src=twsrc%5Etfw), I've tried to visualize the world population changes from 1800 to 2100. My new blog post at <https://t.co/XpBpkZLO9s> describes how this animation was made using [#rstats](https://twitter.com/hashtag/rstats?src=hash&ref_src=twsrc%5Etfw) and [#OpenData](https://twitter.com/hashtag/OpenData?src=hash&ref_src=twsrc%5Etfw). [pic.twitter.com/WI3gj0xUwU](https://t.co/WI3gj0xUwU)

— Jakub Nowosad (@jakub\_nowosad) [October 9, 2018](https://twitter.com/jakub_nowosad/status/1049685831475187712?ref_src=twsrc%5Etfw)

This way of visualization is good to show the ever-changing distribution of the population on a global scale.  
It allows seeing that, China and India dominated the world population, but also a large share of the world population had lived in Europe in 1800.  
At the same time, Australia, the Americas, or Africa were not densely populated.  
By 1900, there is a noticeable growth in relative population in North America (especially the USA).  
From 1950 onwards, a decrease in population in Europe and a tremendous increase in population in Africa is visible.  
Based on this prediction, it is also possible to see the relative population of China and India will decrease in the last decades of this century.

However, while we can see changes in proportion, we cannot see one important dimension – population growth on a global scale.  
Recently, Tomasz Stepinski, Pawel Netzel and I published a paper describing the landscapes change on a global scale between 1992 and 2015[1](https://nowosad.github.io/post/world-pop-change2/#fn1) showing how humans impacted our planet just in 24 years time.  
It is clear the population and its growth has an enormous impact on our planet as it results in more demand for land and natural resources.  
Therefore, I’ve updated my code to add this dimension…

Scaled cartogram of the world population changes between 1800 and 2100.

Now, we can see not only the relative distribution of the population for each year but also population growth on a global scale.

Similarly to [the previous post](https://nowosad.github.io/post/world-pop-change/), the rest of this blog post will focus on explaining the steps and code to create the above animation.  
It involves downloading the data from two sources, cleaning, merging, and preprocessing it.  
Prepared data can be transformed to cartogram to indicate spatial relations between countries at any given year, and scaled to indicate the global world population at any given year compared to the predicted population for 2100.

**Starting**

Let’s start with the packages.  
If you are new to R, you may want to take a read of [this](https://geocompr.robinlovelace.net/spatial-class.html) first, which points to various resources for setting-up R for geographic data.  
When you have a recent R version and the appropriate packages installed (e.g. by executing devtools::install\_github("geocompr/geocompkg")) the packages can be attached as follows:

library(sf) # spatial data classes

library(rnaturalearth) # world map data

library(readxl) # reading excel files

library(dplyr) # data manipulation

library(stringr) # data manipulation

library(tidyr) # data manipulation

library(purrr) # data manipulation

library(cartogram) # cartograms creation

library(gganimate) # animation creation

**Getting data**

To create the maps of the world population for each year we will need two datasets – one containing spatial data of the world’s countries and one non-spatial with information about the annual population in the world’s countries.  
The first one can be easily downloaded from the [Natural Earth](http://www.naturalearthdata.com/) website, for example using the **rnaturalearth** package:

world\_map = ne\_countries(returnclass = "sf")

The second one is available from the [Gapminder](https://www.gapminder.org/) foundation.  
Gapminder provides [a dataset with population data](https://www.gapminder.org/data/documentation/gd003/) for all countries and world regions from 1800 to 2100.  
We can download and read the dataset using the code below:

if(!dir.exists("data")) dir.create("data")

download.file("http://gapm.io/dl\_pop", destfile = "data/pop1800\_2100.xlsx")

world\_pop = read\_xlsx("data/pop1800\_2100.xlsx", sheet = 7)

**Cleaning**

As always when working with multiple datasets – some data cleaning is necessary.  
Our world\_map dataset has many columns irrelevant for cartograms creation and we do not need spatial data of Antarctica.  
We can also transform our data into a more appropriate projection[2](https://nowosad.github.io/post/world-pop-change2/#fn2).

world\_map = world\_map %>%

filter(str\_detect(type, "country|Country")) %>%

filter(sovereignt != "Antarctica") %>%

select(sovereignt) %>%

st\_transform(world\_map, crs = "+proj=moll")

We need to have a common identifier to combine our spatial and non-spatial datasets, for example, names of the countries.  
However, there are inconsistencies between some of the names.  
We can fix it manually:

world\_pop = world\_pop %>%

mutate(sovereignt = name) %>%

mutate(sovereignt = replace(sovereignt, sovereignt == "Tanzania", "United Republic of Tanzania")) %>%

mutate(sovereignt = replace(sovereignt, sovereignt == "United States", "United States of America")) %>%

mutate(sovereignt = replace(sovereignt, sovereignt == "Congo, Dem. Rep.", "Democratic Republic of the Congo")) %>%

mutate(sovereignt = replace(sovereignt, sovereignt == "Bahamas", "The Bahamas")) %>%

mutate(sovereignt = replace(sovereignt, sovereignt == "Serbia", "Republic of Serbia")) %>%

mutate(sovereignt = replace(sovereignt, sovereignt == "Macedonia, FYR", "Macedonia")) %>%

mutate(sovereignt = replace(sovereignt, sovereignt == "Slovak Republic", "Slovakia")) %>%

mutate(sovereignt = replace(sovereignt, sovereignt == "Congo, Rep.", "Republic of Congo")) %>%

mutate(sovereignt = replace(sovereignt, sovereignt == "Kyrgyz Republic", "Kyrgyzstan")) %>%

mutate(sovereignt = replace(sovereignt, sovereignt == "Lao", "Laos")) %>%

mutate(sovereignt = replace(sovereignt, sovereignt == "Cote d'Ivoire", "Ivory Coast")) %>%

mutate(sovereignt = replace(sovereignt, sovereignt == "Timor-Leste", "East Timor")) %>%

mutate(sovereignt = replace(sovereignt, sovereignt == "Guinea-Bissau", "Guinea Bissau"))

**Preparing**

Now we can join our two datasets, remove missing values and unimportant columns.  
We also should transform the data into a long format[3](https://nowosad.github.io/post/world-pop-change2/#fn3), which is accepted by the plotting packages (such as **tmap** or **ggplot2**).

world\_data = left\_join(world\_map, world\_pop, by = "sovereignt") %>%

na.omit() %>%

select(-geo, -name, -indicator) %>%

gather(key = "year", value = "population", `1800.0`:`2100.0`,

convert = TRUE)

Additionally, we can calculate total global populations in each year:

world\_data = world\_data %>%

group\_by(year) %>%

mutate(total\_pop = sum(as.numeric(population), na.rm = TRUE)) %>%

mutate(title = paste0("Year: ", year, "\nTotal population (billions): ",

round(total\_pop / 1e9, 2))) %>%

ungroup()

**Subsetting**

Now, our data contains information about the world population for each year between 1800 and 2100.  
It is possible to use it to create cartograms, however, to reduce calculation time and simplification of the results we will only use data for every 25 years:

world\_data = world\_data %>%

filter(year %in% seq(1800, 2100, by = 25))

**Transforming**

With newly created data, we are able to create our cartograms.  
We need to split our data into independent annual datasets, create cartograms based on the population variable, and combine all of the cartograms back into one object.

world\_data\_carto = world\_data %>%

split(.$year) %>%

map(cartogram\_cont, "population", maxSizeError = 2, threshold = 0.1) %>%

do.call(rbind, .)

**Scaling**

Predicted global population is the largest for the year 2100 and consists of more than 11 billion human beings.  
We can resize a map for each year using this value as our reference point, where the rest of the maps would be proportionally smaller than the one for 2100.  
To scale the areas of the other maps we need to divide a total population in each year by the maximum value and compute the square root of the result.  
The last step of calculations is to resize the geometries (coordinates) by multiplying it with the scales values[4](https://nowosad.github.io/post/world-pop-change2/#fn4):

world\_data\_scaled = world\_data\_carto %>%

mutate(scales = sqrt(total\_pop / max(total\_pop))) %>%

group\_by(year) %>%

mutate(geometry = geometry \* scales)

For more on the topic, we recommend checking out the [Affine transformations](https://geocompr.robinlovelace.net/geometric-operations.html#affine-transformations) section of Geocomputation with R.

**Visualizing**

Now when we have all of the pieces needed, the animated map can be created using either **tmap**[5](https://nowosad.github.io/post/world-pop-change2/#fn5) or **gganimate**.  
It consists of three steps:

1. Creating maps for each year using a combination of the ggplot(), geom\_sf(), and transition\_states() functions.
2. Rendering an animation using animate().  
   It allows for changing the size of a resulting image, and the speed of animation – you can control it with any two combinations of nframes, fps, and duration.
3. Saving an animation with anim\_save().

worlds\_anim = ggplot() +

geom\_sf(data = world\_data\_scaled, aes(fill = population / 1000000), lwd = 0.1) +

coord\_sf(datum = NA) +

scale\_fill\_viridis\_c(name = "Population (mln)") +

theme\_minimal() +

theme(

plot.title = element\_text(size = 22),

plot.background = element\_rect(fill = "#f5f5f4", color = NA),

panel.background = element\_rect(fill = "#f5f5f4", color = NA),

legend.background = element\_rect(fill = "#f5f5f4", color = NA),

legend.position = c(0.1, 0.2),

legend.title = element\_text(size = 18)

) +

labs(title = "{closest\_state}") +

transition\_states(title, transition\_length = 2, state\_length = 6) +

ease\_aes("cubic-in-out")

worlds\_animate = animate(worlds\_anim, width = 1200, height = 550,

duration = 30, fps = 4)

anim\_save("worlds\_animate.gif", animation = worlds\_animate)

1. You can learn more about it at <https://doi.org/10.1016/j.jag.2018.09.013> or by reading its preprint at <https://eartharxiv.org/k3rmn/>.[](https://nowosad.github.io/post/world-pop-change2/#fnref1)
2. You can read more about projections in the [Reprojecting geographic data](https://geocompr.robinlovelace.net/reproj-geo-data.html) chapter of Geocomputation with R.[](https://nowosad.github.io/post/world-pop-change2/#fnref2)
3. This process is called [gathering](http://r4ds.had.co.nz/tidy-data.html#gathering).[](https://nowosad.github.io/post/world-pop-change2/#fnref3)
4. Mutating of geometry columns is possible with [the recent improvements in the **sf** package](https://github.com/r-spatial/sf/issues/903).[](https://nowosad.github.io/post/world-pop-change2/#fnref4)
5. Visit [the animated maps chapter of Geocomputation with R](https://geocompr.robinlovelace.net/adv-map.html#animated-maps) or [my previous blog post](https://nowosad.github.io/post/world-pop-change/) to learn how to make animations with **tmap**.[](https://nowosad.github.io/post/world-pop-change2/#fnref5)