This post will make use of that data to create maps like the one above as well as some others.

**Step 1: Load the Libraries and the Data**

You can read in the data by following the steps below.

**1.1 Open the github site which holds the data.**

url <- "https://github.com/educatorsRlearners/datasets/blob/master/scores\_and\_geoms.RDS?raw=true"

**1.2 Create a temp file and then save the url in the temp file.**

temp <- tempfile()

download.file(url, temp)

**1.3 Read in the file and delete the temp.**

scores\_and\_geoms <- readRDS(temp)

#remove the unnecessary variables

rm(temp, url)

Now you may be asking:

***“Why did you name it scores\_and\_geoms?”***

The data frame contains the IELTS scores of interest and the geoms (aka the [Generic Earth Observation Metadata Standard](https://en.wikipedia.org/wiki/GEOMS_%E2%80%93_Generic_Earth_Observation_Metadata_Standard)) which allows us to draw our lines on a map.

Or you may be asking:

***“Why didn’t you just save the data frame as a .csv file?”***

The column geoms in scores\_and\_geoms is a NULL and, as this thread clearly highlights, saving a data frame which contains a list as a .csv is beyond my current level.

**Step 2: Make the First Map**

Now, all we have to do is the following.

**2.1 Convert scores\_and\_geoms to an sf object**

tmap only reads sf objects so we have to convert our data frame like so:

library(sf)

scores\_and\_geoms <- st\_as\_sf(scores\_and\_geoms)

Key point, sf::st\_as\_sf() automatically recognizes scores\_and\_geoms$geom as the column of interest. However, if the data frame consisted of say city data, and had one column each for latitude and longitude like this:

city <- c("Beijing", "Shanghai", "Guangzhou")

latitude <- c(39.928819, 31.222222, 23.116667)

longitude <- c(116.388869, 121.458056, 113.25)

city\_data <- data.frame(city, latitude, longitude)

| Table 1: Data Sourced from Simplemaps.com | | |
| --- | --- | --- |
| **city** | **latitude** | **longitude** |
| Beijing | 39.92882 | 116.3889 |
| Shanghai | 31.22222 | 121.4581 |
| Guangzhou | 23.11667 | 113.2500 |

then we’d pass those column names as an argument in coords like this:

city\_data\_sf <- st\_as\_sf(x = city\_data, coords = c("latitude", "longitude"))

resulting in the following:

| Table 2: Ready for Plotting | |
| --- | --- |
| **city** | **geometry** |
| Beijing | c(39.928819, 116.388869) |
| Shanghai | c(31.222222, 121.458056) |
| Guangzhou | c(23.116667, 113.25) |

But I digress.

**2.2 Make a World Map**

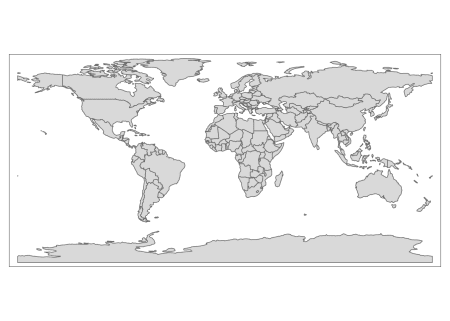
As long as the data is formatted properly, making maps in tmap is really easy. To do so:

1. Load the library
2. Add the shape file
3. Add the borders

library(tmap) #libary

tm\_shape(scores\_and\_geoms) + #shape

tm\_polygons() #borders



*Viola!* we’ve made our first map! Now lets start making it meaningful.

**3 Customize**

tmap is similar to ggplot in that

* everything is a layer and
* it works really well with tidyverse packages

Therefore, we can use dplyr to include the countries/regions which we want to display on map.

For instance, we don’t have IELTS data for Antarctica so lets remove it from the map.

library(tidyverse)

scores\_and\_geoms %>%

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

tm\_shape() +

tm\_polygons()



Going through the code above, we are telling R to:

1. take scores\_and\_geoms
2. keep all rows as long as the observation in scores\_and\_geoms$continent does not equal Antarctica
3. create the shape (aka, the blank map)
4. add country lines

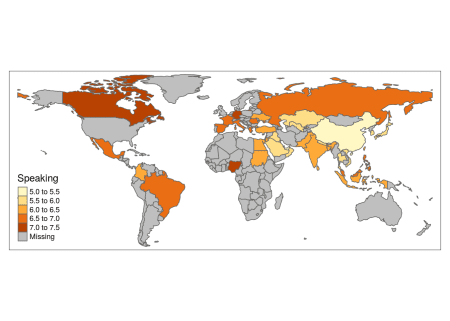
Now lets add some IELTS data. I’m interested to see speaking scores so lets have the color of the country reflect the average speaking score. To do so, pass the variable of interest, in our case scores\_and\_geoms$Speaking, to the color argument in the tm\_polygons() layer.

scores\_and\_geoms %>%

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

tm\_shape() +

tm\_polygons(col = "Speaking")



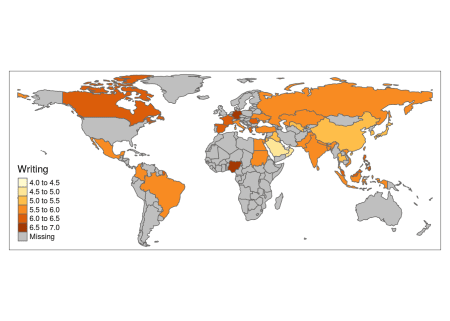
What if you wanted to look at writing? Just change color to “Writing”

scores\_and\_geoms %>%

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

tm\_shape() +

tm\_polygons(col = "Writing")



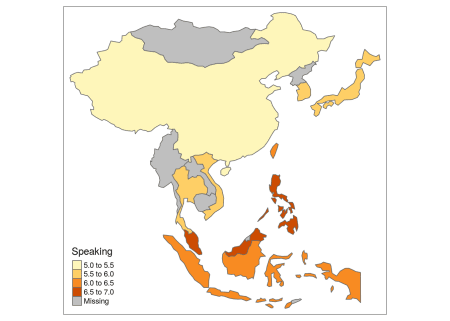
Since I’m based in China, I’m curious to see how East Asian countries compare so lets use dplyr to only plot countries from East and South East Asia.

scores\_and\_geoms %>%

filter(subregion %in% c("South-Eastern Asia", "Eastern Asia")) %>%

tm\_shape() +

tm\_polygons(col = "Speaking")



As for the title, add a tm\_layout() layer and pass it as an argument.

scores\_and\_geoms %>%

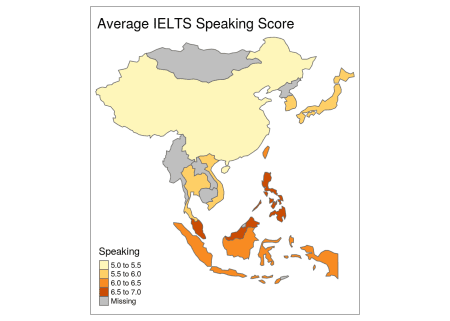
filter(subregion %in% c("South-Eastern Asia", "Eastern Asia")) %>%

tm\_shape() +

tm\_polygons(col = "Speaking") +

tm\_layout(title = "Average IELTS Speaking Score",

inner.margins = c(0.1, 0.02, 0.1, 0.02))



***“What is inner.margins?”***

Those are coordinates for how to position the title in the plot. Play around with them until you get them where you want them.

**Conclusion**

This post is already longer than I planned so we’ll explore some more features in the next post. Until then, happy coding!

Full Code

library(sf)

library(tmap)

library(tidyverse)

url <- "https://github.com/educatorsRlearners/datasets/blob/master/scores\_and\_geoms.RDS?raw=true"

temp <- tempfile()

download.file(url, temp)

scores\_and\_geoms <- readRDS(temp)

#remove the unnecessary variables

rm(temp, url)

#convert to shape file object

scores\_and\_geoms <- st\_as\_sf(scores\_and\_geoms)

#world map

tm\_shape(scores\_and\_geoms) + #shape

tm\_polygons() #borders

#world map omiting Antarctica

scores\_and\_geoms %>%

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

tm\_shape() +

tm\_polygons()

#Same as above but with country color reflecting average IELTS Speaking Score

scores\_and\_geoms %>%

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

tm\_shape() +

tm\_polygons(col = "Speaking")

#Same as above but reflecting average IELTS Writing Score

scores\_and\_geoms %>%

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

tm\_shape() +

tm\_polygons(col = "Writing")

#East Asia with country color reflecting average IELTS Speaking Score

scores\_and\_geoms %>%

filter(subregion %in% c("South-Eastern Asia", "Eastern Asia")) %>%

tm\_shape() +

tm\_polygons(col = "Speaking")

#Same as above but with a title

scores\_and\_geoms %>%

filter(subregion %in% c("South-Eastern Asia", "Eastern Asia")) %>%

tm\_shape() +

tm\_polygons(col = "Speaking") +

tm\_layout(title = "Average IELTS Speaking Score",

inner.margins = c(0.1, 0.02, 0.1, 0.02))