

# Gapminder In-Class Lab

*Tiffany Tang*

*September 06, 2019*

## 1 Introduction

[Gapminder](#) is an excellent organization aimed at increasing the use and understanding of statistics on a number of global topics. They collect a variety of data from many sources and aim to produce fact-based statistics reflecting the current state of our world. In addition, Gapminder has developed easily-accessible [tools](#) for visualizing the data in creative and informative ways.

The data we will be exploring throughout this in-class guide consists of population, life expectancy, and GDP information for many countries over time. If you would like to download this data yourself, click [here](#). This data can also be pulled from the class GitHub repository.

In this in-class lab, our aims will be two-fold:

1. to gain some experience making visualizations with `ggplot()`, and
2. to illustrate little-known tips and tricks in R Markdown which can make life far easier.

**First Tip:** Outside of the code chunks, we can use markdown and latex like normal. We can cross-referencing to different sections (e.g., Section 1), add bullet points, write in-line math equations (e.g.,  $\sqrt{25} + \frac{1}{2}$ ), longer math equations, etc.

$$\begin{aligned}(\alpha + \beta)^2 &= \alpha^2 + \alpha\beta + \beta\alpha + \beta^2 \\ &= \alpha^2 + 2\alpha\beta + \beta^2\end{aligned}$$

We will begin this in-class lab by loading and cleaning the data in Section 2. In Section 3, we will proceed to visualize the data in various ways.

**Tip:** In the first code chunk, you can set global knitr options that will serve as the default settings for all subsequent code chunks.

```
# setting default knitr options (this will save you typing; this way, you no longer have to set echo = TRUE)
knitr::opts_chunk$set(
  echo = FALSE, # don't print the code chunk
  warning = FALSE, # don't print warnings
  message = FALSE, # don't print messages
  fig.width = 6, # set default width of figures
  fig.height = 4, # set default height of figures
  fig.align = "center", # always align figure in center
  fig.pos = "H", # always plot figure at the exact location of the code chunk
  cache = FALSE) # setting cache = TRUE will save the output of the evaluated code chunk locally so that it can be reused

# load useful libraries
library(tidyverse)
library(knitr)
library(R.utils)
library(kableExtra)
```

## 2 Data

Let's begin by loading and cleaning the data. To improve readability and modularity, I have written two external functions, `loadGapminderData()` and `cleanGapminderData()`, and source these functions from their respective files, `load.R` and `clean.R`. Please open these files to see what `loadGapminderData()` and `cleanGapminderData()` are doing and note the function documentation.

Fortunately, the data was already very clean, so we did not conduct any major modifications to the data. In future labs (especially lab 1), if you do need to perform data cleaning, think carefully about the choices you make in the data cleaning stage. Be sure to document how you cleaned the data and why you made those choices.

## 3 Visualizing the gapminder data (ggplot2)

Next, we put our visualization skills to the test and create different plots with `ggplot()`.

1. We are interested in exploring life expectancy as a function of GDP. Create a scatterplot of life expectancy versus GDP for the year 2007 using `ggplot()`, where the size of points are based on the population of the country and they are colored by the continent the country resides in.

**Tip:** The figures will be automatically numbered, and we can easily refer to figures (and tables) by the code chunk name, e.g., Figure ??.

2. Next, we explore change in life expectancy over time. For each continent excluding Oceania, use `ggplot()` to create a series of boxplots over time, where each data point corresponds to the life expectancy of a country for the given year in the given continent.

**Tip:** We can change the size of the figure by modifying `out.width`, `fig.width`, `fig.height`, and/or other knitr options in the header of the code chunk. You can read more about other knitr options [here](#).

### 3.1 Comparing GDP across continents (dplyr)

1. Compute the mean and variance of the GDP for each continent without using `dplyr()`.

```
## [1] 2193.755
## [1] 14469.48
## [1] 7136.11
## [1] 7902.15
```

Perform the same computation using `group_by()` and `summarise()`. Name the resulting tibble `gdp_stats`.

We can display `gdp_stats` in a publication-quality table using `kable()` and some related functions from the `kableExtra` library, which has been loaded. To evaluate the following code chunk and see the resulting table, change `eval = FALSE` to `eval = TRUE` in the following code chunk header.

**Tip:** Like with figures, we can reference tables by name (e.g., Table ??). Also, try setting `booktabs = FALSE` in `kable()`. I think the table with `booktabs = TRUE` looks far better than that with `booktabs = FALSE`, but this is only my opinion.

3. Next, we want to ask about raw GDP (i.e. overall GDP for each country, rather than standardized by per capita). Create a table using `kable()` that shows the average total GDP for each continent in 2007.

**Tip:** We can evaluate R code outside of the code chunks by placing the code inside single backquotes. For instance, the mean raw GDP for Asia is approximately .

## 3.2 Using `tidyr()` with the `gapminder` data

The `gapminder` data that we used for visualization was already in a clean usable format. Here we are given a dataset that requires some processing to get in a more useful form. Our goal is to transform the `gapminder_wide` dataset so that it is in the same form as the original `gapminder` dataset. Let us first load in the `gapminder_wide` dataset and quickly compare it to the original `gapminder` dataset.

```
## [1] 142 38
## [1] 1704 6
```

##	continent	country	gdpPercap_1952	gdpPercap_1957	pop_2002	pop_2007
## 1	Africa	Algeria	2449.0082	3013.9760	31287142	33333216
## 2	Africa	Angola	3520.6103	3827.9405	10866106	12420476
## 3	Africa	Benin	1062.7522	959.6011	7026113	8078314
## 4	Africa	Botswana	851.2411	918.2325	1630347	1639131
## 5	Africa	Burkina Faso	543.2552	617.1835	12251209	14326203
## 6	Africa	Burundi	339.2965	379.5646	7021078	8390505

  

##	country	year	population	continent	life_exp	gdp_per_cap
## 1	Afghanistan	1952	8425333	Asia	28.801	779.4453
## 2	Afghanistan	1957	9240934	Asia	30.332	820.8530
## 3	Afghanistan	1962	10267083	Asia	31.997	853.1007
## 4	Afghanistan	1967	11537966	Asia	34.020	836.1971
## 5	Afghanistan	1972	13079460	Asia	36.088	739.9811
## 6	Afghanistan	1977	14880372	Asia	38.438	786.1134

We can see that the wide version now has a separate column for each year of GDP, life expectancy, and population. This data becomes much easier to work with and understand if we can make year into a column.

1. Use the `gather()` and `separate()` functions to create a long version of the data where we only have five columns: continent, country, the value of an observation, the type of observation (i.e. GDP, life expectancy, or population size), and the year of the observation.

2. Finally, use `spread()` to convert the long version of the data to get the original intermediate version.

See <http://swcarpentry.github.io/r-novice-gapminder/14-tidyr/> for more ways to use `tidyr()` on this data.

**One Last Random Tip:** In future labs or research projects, it may be necessary to cite papers in your writeup. This can be easily done by creating a `.bib` file (using [JabRef](#) or your favorite bibliography manager) and setting `bibliography: name_of_bibliography.bib` in the header of the `.Rmd` file. Then, you can easily cite papers as you would in latex (e.g., Rosling and Zhang [2011]).

## Bibliography

Hans Rosling and Zhongxing Zhang. Health advocacy with gapminder animated statistics. *Journal of epidemiology and global health*, 1(1):11–14, 2011.