

## ASSIGNMENT 1

The goal of this assignment is to familiarize you with the fundamentals of R, data visualization, basic data wrangling, and exploration. The first part of this assignment is quite easy to complete because you will have a chance to check the solution at each step, interactively. However, I strongly recommend you do **not** check a solution before you attempt to answer a question. Otherwise, you will not learn, and you will struggle with the rest of the course. So, please!

If you don't cut corners by checking solutions even before you attempt, this assignment will take quite some time depending on how familiar you are with R. Please plan ahead and start early.

### Assignment Instructions (Part I)

Please complete the following RStudio Cloud Primers and take the screenshots showing both the completion of each Primer and your username together. You can take screenshots on a PC using the [Snipping Tool](#) (Mac users, [see this](#)). A sample submission for Part I is on the last page.

*Before you click on the following links and start working on the assignment, please visit [rstudio.cloud](#) and [log in](#) first. If you don't log in, your name will not come up in the screenshots.*

This assignment has five mandatory parts:

1. [Data Visualization Basics](#)
2. [Programming Basics](#)
3. [Working with Tibbles](#)
4. [Isolating Data with dplyr](#)
5. [Deriving Information with dplyr](#)

and two highly recommended sections<sup>1</sup>:

1. [Visualize Data](#)
2. [Tidy Your Data](#)

and two more recommended sections:

1. [Iterate](#)
2. [Write Functions](#)

---

<sup>1</sup> In the past, students who completed the recommended sections [increased the probability of getting an A by 39%](#), on average. Don't attempt to finish them all at once. You will **not** submit the optional sections in your assignment.

## Submission for Part I

Your name and UID: Mansi Kosamkar (116904988)

### 1. Data Visualization Basics

The screenshot shows the RStudio Cloud web interface. The browser address bar displays 'rstudio.cloud/learn/primers/1.1'. The page title is 'Data Visualization Basics'. The left sidebar contains navigation links: Spaces (Your Workspace, New Space), Learn (Guide, What's New, Primers, Cheat Sheets, Feedback and Questions), and Info (Terms and Conditions, System Status). The 'Primers' link is selected. The main content area is titled 'The Basics' and contains the following text:

What should you know about packages?

When you first install R, you get a small collection of core packages known as **base R**. The remaining packages—there are over 10,000 of them—are optional. You don't need to install them unless you want to use them.

ggplot2 is one of these optional packages, so are the other packages that we will look at in these tutorials. Some of the most popular and most modern parts of R come in the optional packages.

You don't need to worry about installing packages in these tutorials. Each tutorial comes with all of the packages that you need pre-installed; this is how we make the tutorials easy to use.

However, one day, you may want to use R outside of these tutorials. When that day comes, you'll want to remember which packages to download to acquire the functions you use here. Throughout the tutorials, I will try to make it as clear as possible where each function comes from!

If you'd like to learn more about installing R packages (or R or the RStudio IDE), the [Set Up](#) video tutorial walks you through the process of setting up R on your own computer.

Where to from here

Congratulations! You can use the ggplot2 code template to plot any dataset in many different ways. As you begin exploring data, you should incorporate these tools into your workflow.

There is much more to ggplot2 and Data Visualization than we have covered here. If you would like to learn more about visualizing data with ggplot2, check out RStudio's primer on [Data Visualization](#).

Your new data visualization skills will make it easier to learn other parts of R, because you can now visualize the results of any change that you make to data. You'll put these skills to immediate use in the next tutorial, which will show you how to extract values from datasets, as well as how to compute new variables and summary statistics from your data. See you there.

Previous Topic

### 2. Programming Basics

The screenshot shows the RStudio Cloud web interface. The browser address bar displays 'rstudio.cloud/learn/primers/2.1'. The page title is 'Programming basics'. The left sidebar contains navigation links: Spaces (Your Workspace, New Space), Learn (Guide, What's New, Primers, Cheat Sheets, Feedback and Questions), and Info (Terms and Conditions, System Status). The 'Primers' link is selected. The main content area is titled 'The Basics' and contains the following text:

Congratulations!

Congratulations. You now have a formal sense of how the basics of R work. Although you may think of your self as a Data Scientist, this brief Computer Science background will help you as you analyze data. Whenever R does something unexpected, you can apply your knowledge of how R works to figure out what went wrong.

Previous Topic

### 3. Working with Tibbles

Studio Cloud

Spaces

Your Workspace

New Space

Learn

Guide

What's New

Primers

Cheat Sheets

Feedback and Questions

Info

Terms and Conditions

System Status

Learn

Guide

What's New

Primers

Cheat Sheets

Mansi Kosamkar

Work with Data

Working with Tibbles

Welcome

babynames

tibbles

tidyverse

Start Over

Which package is not loaded by `library("tidyverse")`

☐ ggplot2 X

☐ dplyr X

☐ tibble X

☒ babynames ✓

Correct!

Now that you are familiar with the data set, and have loaded the necessary packages, let's explore the data.

Recap

Tibbles and the tidyverse package are two tools that make life with R easier. Ironically, you may not come to appreciate their value right away: these tutorials pre-load packages for you, and they wrap data frames into an interactive table for display (at least the tutorials in the primers that follow will). However, you will want to utilize tibbles and the tidyverse package when you move out of the tutorials and begin doing your own work with R inside of the RStudio IDE.

This tutorial also introduced the babynames dataset. In the next tutorial, you will use this data set to plot the popularity of *your* name over time. Along the way, you will learn how to filter and subset data sets in R.

Continue

### 4. Isolating Data with dplyr

Studio Cloud

Spaces

Your Workspace

New Space

Learn

Guide

What's New

Primers

Cheat Sheets

Feedback and Questions

Info

Terms and Conditions

System Status

Learn

Guide

What's New

Primers

Cheat Sheets

Mansi Kosamkar

Work with Data

Isolating Data with dplyr

Welcome

Your name

select()

filter()

arrange()

%>%

Start Over

Recap

Together, `select()`, `filter()`, and `arrange()` let you quickly find information displayed within your data.

The next tutorial will show you how to derive information that is implied by your data, but not displayed within your data set.

In that tutorial, you will continue to use the `%>%` operator, which is an essential part of programming with the

Pipes help make R expressive, like a spoken language. Spoken languages consist of simple words that you combine into sentences to create sophisticated thoughts.

In the tidyverse, functions are like words: each does one simple task well. You can combine these tasks into pipes with `%>%` to perform complex, customized procedures.

Previous Topic

## 5. Deriving Information with dplyr

The screenshot shows the RStudio Cloud interface. On the left is a sidebar with navigation links: Spaces (Your Workspace, New Space), Learn (Guide, What's New, **Primers**, Cheat Sheets), Feedback and Questions, Info (Terms and Conditions, System Status). The top navigation bar includes Learn, Guide, What's New, **Primers**, and Cheat Sheets. The main content area is titled 'Derive Information with dplyr' and is part of the 'Work with Data' section. It features a 'Welcome' message, a list of topics (The most popular names, summarise(), group\_by(), mutate(), Challenges), and a 'Where to from here' section. A green callout box contains a quote about name variety. The 'Where to from here' section explains that the user can use dplyr's grammar to access data, calculate summary statistics, and compute new variables. It also lists three basic skills for working with R: 1. How to visualize data, 2. How to work with data, and 3. How to program with R code. A 'Previous Topic' button is at the bottom.

## 6. Visualize data

The screenshot shows the RStudio Cloud interface. On the left is a sidebar with navigation links: Spaces (Your Workspace, New Space), Learn (Guide, What's New, **Primers**, Cheat Sheets), Feedback and Questions, Info (Terms and Conditions, System Status). The top navigation bar includes Learn, Guide, What's New, **Primers**, and Cheat Sheets. The main content area is titled 'Visualize Data' and is part of the 'Visualize Data' section. It features a 'Welcome' message, a list of topics (Exploratory Data Analysis, Variation, Covariation), and a 'Where to from here' section. A green callout box contains a quote about name variety. The 'Where to from here' section explains that the user can use dplyr's grammar to access data, calculate summary statistics, and compute new variables. It also lists three basic skills for working with R: 1. How to visualize data, 2. How to work with data, and 3. How to program with R code. A 'Previous Topic' button is at the bottom.

Studio Cloud

Spaces

Your Workspace
New Space

Learn

Guide
What's New
Primers
Cheat Sheets
Feedback and Questions

Info

Terms and Conditions
System Status

Learn
Guide
What's New
Primers
Cheat Sheets

Visualize Data

Mansi Kosamkar

HISTOGRAMS

Welcome
HISTOGRAMS
SIMILAR GEOMS

Start Over

Sometimes outliers cannot be seen in a plot, but can be inferred from the range of the x axis. For example, many of the plots in this tutorial seemed to extend well past the end of the data. Why? Because the range was stretched to include outliers. When your data set is large, like `diamonds`, the bar that describes an outlier may be invisible (i.e. less than one pixel high).

- Look for clusters.
- Look for shape. The shape of a histogram can often indicate whether or not a variable behaves according to a known probability distribution.

The most important thing to remember about histograms, frequency polygons, and dotplots is to explore different binwidths. The binwidth of a histogram determines what information the histogram displays. You cannot predict ahead of time which binwidth will reveal unexpected information.

Previous Topic

Studio Cloud

Spaces

Your Workspace
New Space

Learn

Guide
What's New
Primers
Cheat Sheets
Feedback and Questions

Info

Terms and Conditions
System Status

Learn
Guide
What's New
Primers
Cheat Sheets

Visualize Data

Mansi Kosamkar

SCATTERPLOTS

Welcome
SCATTERPLOTS
LAYERS
COORDINATE SYSTEMS

Start Over

Recap

Scatterplots are one of the most useful types of plots for data science. You will have many chances to use `geom_point()`, `geom_smooth()`, and `geom_label_repel()` in your day to day work.

However, this tutor introduced important two concepts that apply to more than just scatterplots:

- You can add **multiple layers** to any plot that you make with ggplot2
- You can add a different **coordinate system** to any plot that you make with ggplot2

Give Feedback

Previous Topic

Studio Cloud

Spaces

Your Workspace
New Space

Learn

Guide
What's New
Primers
Cheat Sheets
Feedback and Questions

Info

Terms and Conditions
System Status

Learn
Guide
What's New
Primers
Cheat Sheets

Visualize Data

Mansi Kosamkar

LINE PLOTS AND MAPS

Welcome
LINE GRAPHS
SIMILAR GEOMS
MAPS

Start Over

Recap

You can now make all of the plots recommended in the [Exploratory Data Analysis tutorial](#). The last tutorial in this primer will teach you several strategies for dealing with overplotting, a problem that can occur when you have large data or low resolution data.

Previous Topic

5/5

The image displays two screenshots of the RStudio Cloud interface, specifically the 'Visualize Data' section.

**Top Screenshot: Overplotting Primer**

- Header:** Studio Cloud, Learn, Guide, What's New, **Primers**, Cheat Sheets, Mansi Kosamkar
- Left Sidebar:** Spaces (Your Workspace, New Space), Learn (Guide, What's New, **Primers**, Cheat Sheets, Feedback and Questions), Info (Terms and Conditions, System Status).
- Main Content:**
  - Visualize Data** (Menu)
  - Overplotting** (Title)
  - Recap:** Overplotting is a common phenomenon in plots because the causes of overplotting are common in data sets. Data sets often:
    - round values to a common set of values, or
    - are too big to visualize easily without overplotting.
  - Text:** When overplotting results from rounding errors, you can work around it by manipulating the transparency or location of the points. For larger datasets you can use geoms that summarise the data to display relationships without overplotting. This is an effective tactic for truly big data as well, and it also works for the first case of overplotting due to rounding. One final tactic is to sample your data to create a sample data set that is small enough to visualize without overplotting. You've now learned a complete toolkit for exploring data visually. The final tutorial in this primer will show you how to polish the plots you make for publication. Instead of learning how to visualize data, you will learn how to add titles and captions, customize color schemes and more.
  - Buttons:** Start Over, Continue

**Bottom Screenshot: Customize plots Primer**

- Header:** Studio Cloud, Learn, Guide, What's New, **Primers**, Cheat Sheets, Mansi Kosamkar
- Left Sidebar:** Same as the top screenshot.
- Main Content:**
  - Visualize Data** (Menu)
  - Customize plots** (Title)
  - Table of Contents:** Welcome, Zooming, Labels, Themes, Scales, Legends, Quiz.
  - Plot:** A scatter plot titled 'Log Price' vs 'Log Carat Size'. The y-axis is labeled 'Log Price' with a tick at 1000. The x-axis is labeled 'Log Carat Size' with a tick at 1. The plot shows a positive correlation with many black points. Three green lines represent different diamond quality levels: 'Very Good', 'Premium', and 'Ideal'. The data is attributed to 'Data provided by Hadley Wickham'.
  - Buttons:** Start Over, Previous Topic

## Assignment Instructions (Part II)

In this part of the assignment, you will apply what you learned in the first part. You will work on the Gapminder dataset, which is available as a library in R (how convenient!). You can read more about the source of the dataset [here](#), and enjoy [this talk](#) from 2007 if you have not yet done so. See [How to submit labs/assignments](#) to learn how to submit the second part of the assignment.

Use the following R libraries in this assignment: tidyverse, tidymodels, plotly, skimr, gapminder

## Data Overview

1. country: Name of the country
2. continent: Name of the continent a country belongs in
3. year: Year for which the data is collected

4. lifeExp: Average life expectancy of the people in a country in a year
5. pop: Total population of a country in a year
6. gdpPercap: GDP per capita of a country in a year

## Questions

### 1. Create an R Notebook

- a. Start with [the assignment template](#) and set your working directory using `setwd`
- b. From here on, use a new chunk for each question to make your code readable

### 2. Load the data

- a. In a new chunk, load the `gapminder` library, and use this line to create `dfGap`:  
`dfGap <- gapminder`

### 3. Explore the data

- a. Use the `skim` function on the `dfGap` dataframe to get summary statistics in a nice format. I suggest you use the widest screen possible for the best reading.
- b. Filter `dfGap` for the year 2007 and sort it in descending order of life expectancy. Don't forget to use pipes!

- i. What are the names of the countries with a life expectancy over 81?

**Ans. Japan, Hongkong, Iceland, Switzerland, Australia.**

- c. Add a calculated column `totalGDP` to `dfGap` showing the total GDP per country, filter the dataframe for 2007, and sort in descending order for `totalGDP`. If you like, save the new dataframe as a new one for repeated use.

- i. What are some names of the countries with the top levels of total GDP?

**Ans. United States, China, Japan, India, Germany**

- ii. Which ones of these countries overlap with the countries from **3-b**?

**Ans. Japan**

- iii. What if you selected only the two columns `country` and `gdpPercap` and sorted the dataframe in descending order for `gdpPercap`? Do you observe more of an overlap now? What do you infer from this difference?

**Ans. Yes. Hong Kong, Switzerland and Iceland. It shows that the higher the GDP, the higher the life expectancy and there is a high positive correlation.**

- d. Filter `dfGap` for 2007, group it by continent, and then calculate the median life expectancy and *median* total GDP (so you need to have `totalGDP` already). Remember, you will pipe the filtered and grouped dataframe into `summarize()` to get the medians. Then, sort it in descending order for the median life expectancy. Before you sort it, don't forget to use `ungroup()` to ungroup.

- i. What continent has the highest median of life expectancy?

**Ans. Oceania**

- ii. Does it seem to be correlated with the median total GDP

**Ans. No, it doesn't seem to be correlated. This can be concluded from the similar values of Asia and America for median life expectancy but significantly different values for Median GDP.**

### 4. Visualize the data

- a. Now that you have explored the relationship between life expectancy and `totalGDP` in a table format, let's also visualize it to see a bigger picture.

- i. Create a scatter plot to understand the relationship between life expectancy (y-axis) and totalGDP (x-axis) in 2007. Does this plot help?

**Ans. No, it does not help explain any relationship between life expectancy and totalGDP**

- ii. Copy the same code, but this time also filter for countries with a totalGDP of over a billion (use the scientific notation  $1e+12$ ). What about now?

**Ans. It shows that countries with around the same GDP has similar life expectancy with a few anomalies.**

- iii. Copy the same code, and add labels this time. Do you see a cluster now? What are the names of the countries that are outside of the cluster?

**Ans. Yes, I see a cluster now. The names of the countries that are outside of the cluster are India, China, United States.**

- iv. Here is a pro tip. The labels you used in (iii) overlap and hide the points. This causes poor visibility. Install and load the `ggrepel` library. After that, copy the same code and use `geom_label_repel()` function instead of `geom_label()`. Does it look better now? Describe what has changed.

**Ans. The labels correlating to a point are pointed to by a straight line so as to clearly show the points.**

- v. Copy the same code. This time, add a color for the continent. What are the continents that are missing from your visual? Why do you think so?

**Ans. Australia, Africa and Antarctica are missing from the visual. This maybe because these countries are not big economic power houses.**

- b. You have an idea about the relationship between life expectancy and totalGDP even though you have not tested it statistically. Now, let's examine a more realistic relationship between life expectancy and gdpPercap (GDP per capita). Plot life expectancy (y-axis) against gdpPercap (x-axis) for 2007, add a smoothed line (no need to define any parameters, use the defaults). What do you observe about the overall relationship? Don't use any labels, just focus on the aggregate.

**Ans. The life expectancy keeps increasing with GDP per capita.**

- c. Now let's find out the variations in life expectancy across different continents. Create box plots for each continent (in the same plot) and add a title this time. What do you observe? Describe your observations and answer the questions:

- i. Which continent has the highest median life expectancy?

**Ans. Oceania has the highest median life expectancy.**

- ii. Which continent has the largest range of life expectancy?

**Ans. Asia has the largest range of life expectancy.**

- iii. Save your plot as `boxPlotsForAll` and put it into the `ggplotly()` function. More useful, right? Report the actual medians per continent by reading from the new interactive plot `ggplotly()` has created for you.

**Ans. Africa-47.79 , Americas- 67.05, Asia- 61.79, Europe- 72.24, Oceania- 73.66.**



- d. Finally, it is time to create a more advanced (and likely more helpful) plot. Create a line plot to show how **median GDP per capita** by continent changes **over time**. [Hint: For the continents, use the color parameter]. Describe what you observe.
- i. What continents have a clearer trend than others? Why do you think so?  
**Ans. Oceania, Europe and Africa have a clearer trend than others. This is because the median GDP per capita increases faster with the year than others.**
  - ii. Change the summary metric from median to mean. What has changed? Why do you think so?  
**Ans. Asia overtook the Americas when we changed the summary metric from median to mean. This is because taking the mean skews the result because of outliers or extreme values. Whereas, the median is a more centralized value.**
  - iii. Finally, don't you think these plots would be much more useful in plotly? Pick one and save it as `gdpOverTime` and call `ggplotly()` on it. You can now read the actual GDP values per year. What are some of the breakthrough years (steep changes) for GDP in different continents?  
**Ans. The 1990s and 2000s show significant increase in the GDP per capita of Oceania and Europe. Asia saw some growth in the period between late 1960s and early 1970s and the 2000s.**