# HW1_jaeyounglee

Jaeyoung Lee

September 1, 2020

## Problem 1

I have practiced the visualization and programming basics in Primers

## Problem 2

### Part A

What I hope to get out of this class as follows,

1. How to be reproducible to collaborate with others and even myself
2. How to write effective codes
3. How to make nice reports using R Markdown (Be a nice statistician)
4. Getting used to R Markdown and GitHub

I feel lucky to take this class.

### Part B

**Binomial, Gamma, Beta pdfs from Casella & Berger**

$$P(X = x|n, p) = \binom{n}{x} p^x (1-p)^{1-x}. \tag{1}$$

$$P(X|\alpha, \beta) = \frac{1}{\gamma(\alpha)\beta^\alpha} x^{\alpha-1} e^{-\frac{x}{\beta}}. \tag{2}$$

$$P(X|\alpha, \beta) = \frac{1}{B(\alpha, \beta)} x^{\alpha-1}(1-x)^{\beta-1}. \tag{3}$$

## Problem 3

1. Rule 1. For Every Result, Keep Track of How It Was Produced
    - Recording every detail is important for reproducibility.
    - Details such as the name and version of the program, parameters and inputs are critical.
    - What can be a challenge? How to write details easy to read is a challenge when we record every detail.
2. Rule 2. Avoid Manual Data Manipulation Steps
    - Rely on codes, not on manual procedures such as Excel to reproduce.
    - What can be a challenge? From the beginning, we should get used to how to manipulate data only with codes not manually.
3. Rule 3. Archive the Exact Versions of All External Programs Used
    - A newer version of a program may not run even without any change of inputs.
    - What can be a challenge? It is easy to think that all the newer versions work well. This step also can be included to Rule 1.
4. Rule 4. Version Control All Custom Scripts
    - Only one exact script may be able to produce that exact output.
    - Sometimes, backtracking to a code state is needed.
    - Use GitHub.
    - What can be a challenge? One should get used to GitHub.
5. Rule 5. Record All Intermediate Results, When Possible in Standardized Formats
    - Looking through intermediate results can uncover discrepancies toward what is assumed, and can find bugs or faulty interpretations.
    - It reveals consequences of alternative programs and parameter choices at each steps.
    - It allows parts of the process to be rerun.
    - One can track the steps where the problems appears.
    - Without full operation, one can examine the full process.
    - What can be a challenge? If one write a long function, a challenge might occur regarding rule 5.
6. Rule 6. For Analyses That Include Randomness, Note Underlying Random Seeds
    - Providing the same random seed allows results to be reproduced exactly in future runs.
    - What can be a challenge? The codes might work only few random seeds. If one runs the code from that random seeds, the one can mislead the results from the random seeds.
7. Rule 7. Always Store Raw Data behind Plots
    - It allows raw data for a given figure to be easily retrieved.
    - one can easily modify the plotting procedure, without redoing the whole analysis.
    - What can be a challenge? It is not challenging but One should know which data is used for the plot.
8. Rule 8. Generate Hierarchical Analysis Output, Allowing Layers of Increasing Detail to Be Inspected
    - Hypertext is the best example.

- By simple clicks, we can easily view the full data underlying the summary of results with links.
- What can be a challenge? Organizing the full data under hypertext might be a challenge.

9. Rule 9. Connect Textual Statements to Underlying Results
    - Connect results to the statements that are initially formulated such as notes or emails.
    - It is important to provide details along with your textual interpretations to the results to be tracked down in the future.
    - What can be a challenge? One should write proper ReadMe files.
10. Rule 10. Provide Public Access to Scripts, Runs, and Results
    - All inputs, scripts, versions, parameters, and intermediate results should be provided publicly and easily accessible.
    - GitHub is a good tool.
    - What can be a challenge? Getting used to GitHub can be hard to the beginners.

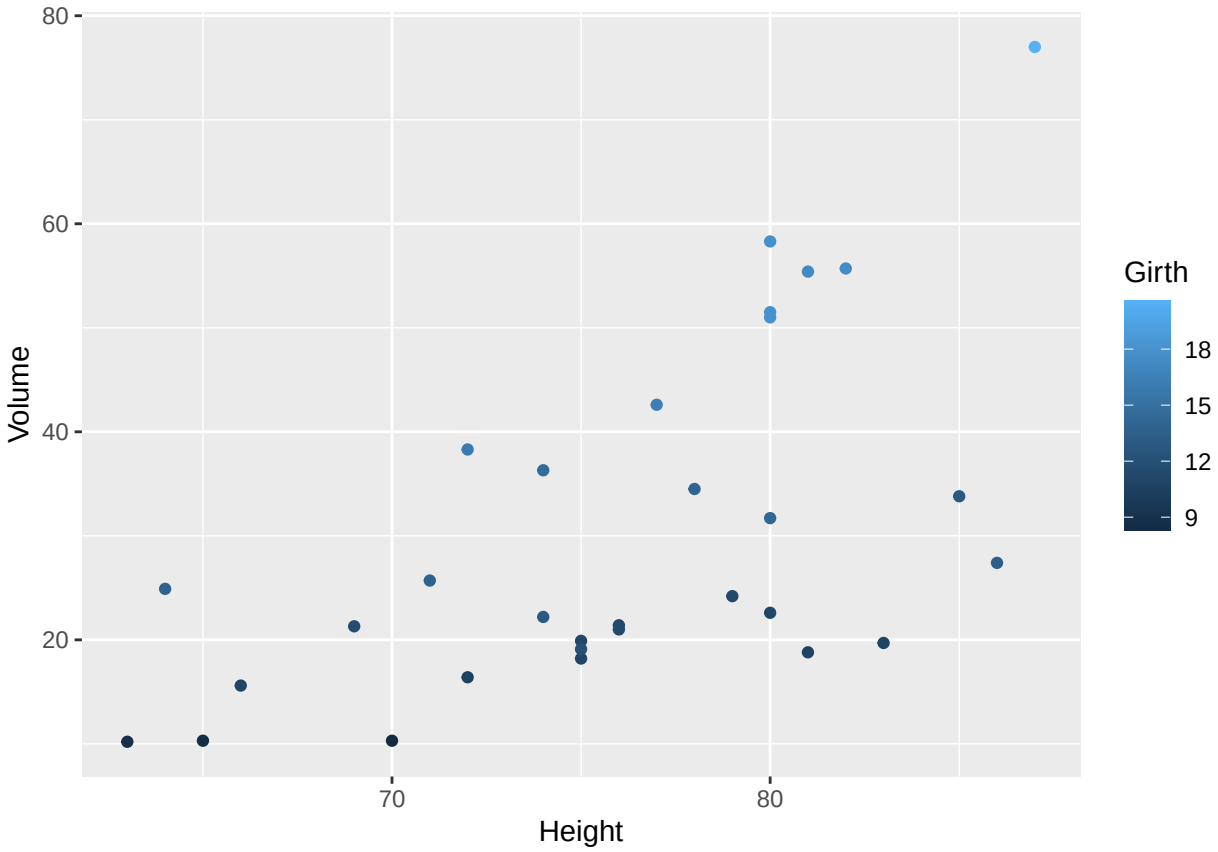
# Problem 4

```
### Problem 4 : A Scatter Plot and A Histogram ###

# R version 4.0.2
library(help = 'datasets')  # To get a list of the datasets
summary(trees)              # Summary of trees data
```

```
##      Girth           Height       Volume
##  Min.   : 8.30   Min.   :63   Min.   :10.20
##  1st Qu.:11.05   1st Qu.:72   1st Qu.:19.40
##  Median :12.90   Median :76   Median :24.20
##  Mean   :13.25   Mean   :76   Mean   :30.17
##  3rd Qu.:15.25   3rd Qu.:80   3rd Qu.:37.30
##  Max.   :20.60   Max.   :87   Max.   :77.00
```

```
### A basic scatter plot from 'trees' data using ggplot2
library(ggplot2)            # To use ggplot function
ggplot(data = trees) +
  geom_point(mapping = aes(x = Height, y = Volume, color = Girth))
```




```
### Histogram for 'trees' data
hist(trees$Volume, main ='Histogram of Volume of Trees', xlab = 'Volume')
```


## Problem 5

Push to GitHub

## Appendix : R codes

```
knitr::opts_chunk$set(echo = TRUE)

### Problem 4 : A Scatter Plot and A Histogram ###

# R version 4.0.2
library(help = 'datasets')  # To get a list of the datasets
summary(trees)              # Summary of trees data

### A basic scatter plot from 'trees' data using ggplot2
library(ggplot2)            # To use ggplot function
ggplot(data = trees) +
  geom_point(mapping = aes(x = Height, y = Volume, color = Girth))

### Histogram for 'trees' data
hist(trees$Volume, main ='Histogram of Volume of Trees', xlab = 'Volume')
```