Penn_PREP_Workshop_2019

Binglan Li

3/19/2019

Multiple Testing Adjustment

"Simulated_p_values.txt" is comprised of randomly generated p-values. None of them suggests any statistical significant test results, despite the fact that some p-values are less than 0.05 out of chance.

1. Read simulated p-values to a vector named **p_values** using **read.table("simulated_p_values.txt")**
2. Adjust p-values using **p.adjust(vector, method = "fdr")** and store the output to a new vector named **fdr**
3. Adjust p-values using **p.adjust(vector, method = "bonferroni")** and store the output to a new vector named **bonferroni_p**
4. Use **length(which(vector < 0.05))** to check whether there are still any significant p-values.

```
# read in simulated p-values
p_values <- read.table("simulated_p_values.txt")
head(sort(p_values[,1]))
```

```
## [1] 0.0001345621 0.0006725208 0.0009421944 0.0026160872 0.0068040853
## [6] 0.0122261597
```

```
# FDR: most common in biological and medical research
fdr <- p.adjust(p_values[,1], method = "fdr")
head(sort(fdr))
```

```
## [1] 0.1345621 0.3140648 0.3140648 0.6540218 0.9466819 0.9466819
```

```
# Bonferroni: more conserved compared with FDR
bonferroni_p <- p.adjust(p_values[,1], method = "bonferroni")
head(sort(bonferroni_p))
```

```
## [1] 0.1345621 0.6725208 0.9421944 1.0000000 1.0000000 1.0000000
```

```
# check significant p-values
length(which(fdr < 0.05))
```

```
## [1] 0
```

```
length(which(bonferroni_p < 0.05))
```

```
## [1] 0
```

Preliminaries

Load the ggplot2 library and necessary datasets. Here we are using the Galton data, referred to as Pearson's height data as well, on the heights of parents and their children collected by sir Francis Galton (1822-1911), an English statistician. He founded many concepts in statistics, such as correlation, quartile, percentile and regression, that are still being used today. The famous Pearson's correlation coefficient was established by Karl Pearson based on a related idea introduced by sir Francis Galton.

The data were collected in the **late 19th century in England**. He coined the term regression towards mediocrity to describe the result of his linear model, which was too philosophical for me to understand. (Note that the paper was written in 1886.)

Exercise 1.0

1. Install **ggplot2** package using **install.packages()**.
2. Load **ggplot2** package using **library()**.
3. Create a new data frame called **heights** that reads data from **"Galton_Height.txt"** using **read.table()**. Notice that this is a tab delimited file with a header line. Let **read.table()** know you want to keep the header line of the data by adding the argument **header = TRUE** in the read.table() function, e.g. **read.table(file, header = TRUE)**.
4. (Optional) while reading the height data by read.table(), add the argument **stringsAsFactors = F**. Use **str()** to check the internal structure of the data frame of **heights** which tells you the dimension and types of data in each vector/column. Do you notice any difference after adding the argument?

```
# install ggplot2 package
#install.packages("ggpplot2")
# if the above doesn't work, try
#devtools::install_github("tidyverse/ggplot2")

# load libraries
library(ggplot2)

# load datasets
# An alternative way to load data is through "import dataset" bottom located
# at the top right console under the "Environment" tab
heights <- read.table("Galton_Height.txt", header = TRUE, stringsAsFactors = F)

# load the iris dataset
# this is another famous statistical dataset that you can play with
data("iris")
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

Exercise 1.1. Data cleaning

1. Check the height data using **head()**. What data do you think are in **heights**?

2. Check dimension of **heights** using either **dim()** or **str()**.
3. Extract "Father" and "Height" columns from heights and keep only rows whose "Gender" are males ("M"). Store the extracted data in a new data frame named **fs** (father-and-son pairs).
4. (Optional) Rstudio also provides a cool function that allows you to skim through the whole data frame and even do basic search, filtering, and sorting without making any actual changes to the data. Find the table of data under the tab of **Environment** at the top right. Find **heights**. Click on the small table icon to the right of **heights**.
5. (Optional) use **View()** to open a new tab in the coding console to view data in **fs**.

```
# check the content in the heights data frame
head(heights)
```

```
##   Family Father Mother Gender Height Kids
## 1      1   78.5   67.0      M   73.2    4
## 2      1   78.5   67.0      F   69.2    4
## 3      1   78.5   67.0      F   69.0    4
## 4      1   78.5   67.0      F   69.0    4
## 5      2   75.5   66.5      M   73.5    4
## 6      2   75.5   66.5      M   72.5    4
```

```
dim(heights)
```

```
## [1] 898   6
```

```
str(heights)
```

```
## 'data.frame':    898 obs. of  6 variables:
##  $ Family: chr  "1" "1" "1" "1" ...
##  $ Father: num  78.5 78.5 78.5 78.5 75.5 75.5 75.5 75.5 75 75 ...
##  $ Mother: num  67 67 67 67 66.5 66.5 66.5 66.5 64 64 ...
##  $ Gender: chr  "M" "F" "F" "F" ...
##  $ Height: num  73.2 69.2 69 69 73.5 72.5 65.5 65.5 71 68 ...
##  $ Kids  : int  4 4 4 4 4 4 4 4 2 2 ...
```

```
# extract only the father and son pairs
fs <- heights[which(heights$Gender == "M"), c("Father", "Height")]

# check content the newly generated data frame
head(fs)
```

```
##    Father Height
## 1    78.5   73.2
## 5    75.5   73.5
## 6    75.5   72.5
## 9    75.0   71.0
## 11   75.0   70.5
## 12   75.0   68.5
```

```
View(fs)
```

```
## Warning: running command ''/usr/bin/otool' -L '/Library/Frameworks/
## R.framework/Resources/modules/R_de.so'' had status 1
```

Exercise 1.2. Correlation

Correlation is a powerful approach to learn whether there is the linear relationship between two random variables. However, correlation does not indicate any causal-effect relation between two variables. For example, Even if the number of deaths per year are correlated with iphone sales, the former is not necessarily attributable to the latter. The estimation of correlation simply indicates 1) whether there is a linear relationsihp, 2) the strength of the relationship, and 3) the direction of the relationship.

1. Calculate the correlation between the father's and son's height using **cor()**.

```
# calculate the Pearson's correlation coefficient between father's and
# son's height
fs_cor <- cor(fs$Father, fs$Height)

# print the correlation
print(fs_cor)
```

```
## [1] 0.3913174
```

Exercise 1.3. Generate a Simple Linear Regression (SLR) Model

Regression is an approach for modelling the relationship between a quantitative response Y and a single predictor variable X. Regression can sometimes look a lot like correlation as it provides a statistical model explaining the relationship between X and Y (exploration) and predict the mean value of an unknown Y given a new X (prediction).

1. Generate a SLR between the son's height (**Height** in fs, Y) and the father's height (**Father** in fs, X) using **lm()**.
2. Store the SLR model in a variable named **fs_lm**.
3. Print the SLR model of father-and-son height. What are intercept and slope?
4. What is the unit increase of son's height when there is every one unit increase on father's height?

```
# generate a SLR model
lm(Height ~ Father, data = fs)
```

```
##
## Call:
## lm(formula = Height ~ Father, data = fs)
##
## Coefficients:
## (Intercept)       Father
##     38.2589       0.4477
```

```
# store the SLR model in a variable
fs_lm <- lm(Height ~ Father, data = fs)

# view the SLR model
print(fs_lm)
```

```
##
## Call:
```

```
## lm(formula = Height ~ Father, data = fs)
##
## Coefficients:
## (Intercept)       Father
##     38.2589       0.4477
```

Exercise 1.4. Reading Additional Output of SLR from summary()

1. Obtain the summary statistics of the father-and-son SLR model using **summary()**.
2. Is there a relationship between father's and son's height?
3. How accurate is the SLR model of father-and-son height? Find the R-square of the SLR model.

```
## obtain summary stat of SLR
summary(fs_lm)
```

```
##
## Call:
## lm(formula = Height ~ Father, data = fs)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.3774 -1.4968  0.0181  1.6375  9.3987
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 38.25891    3.38663   11.30   <2e-16 ***
## Father       0.44775    0.04894    9.15   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.424 on 463 degrees of freedom
## Multiple R-squared:  0.1531, Adjusted R-squared:  0.1513
## F-statistic: 83.72 on 1 and 463 DF,  p-value: < 2.2e-16
```

Exercise 1.5. Predict Using a SLR Model

1. Generate a new data frame named **new_f** that store a vector/column named **Father** with a single value - 75.
2. Predict the son's height when the father is 75 inches using **predict(regression model, new data)**.
3. Calculate the 95% confidence interval of son's height when the father is 75 inches tall using **predict(regression model, new data, interval = "confidence")**. What is the lower and upper bound of the confidence interval of predicted son's height on average?
4. Calculate the 95% prediction interval of son's height when the father is 75 inches tall using **predict(regression model, new data, interval = "prediction")**. What is the lower and upper bound of the prediction interval of predicted son's height?

```
# generate a new data frame storing new data on father's height
new_f <- data.frame(Father = 75)

# predict point values
predict(fs_lm, newdata = new_f)
```

```
##        1
## 71.84001
```

```
# calculate 95% confidence interval of predicted son's height when father is
# 75 inches tall
predict(fs_lm, newdata = new_f, interval = "confidence")
```

```
##        fit      lwr      upr
## 1 71.84001 71.23725 72.44276
```

```
# calculate 95% confidence interval of predicted son's height when father is
# 75 inches tall
predict(fs_lm, newdata = new_f, interval = "prediction")
```

```
##        fit      lwr      upr
## 1 71.84001 67.03793 76.64208
```

Exercise 1.6. (Optional) Checking Assumptions

1. Draw a scatter plot between father's (X) and son's height (Y) using **fs** and **geom_point()** in ggplot2. Do you observe a linear relationship between father's and son's height?
2. Create a new column/vector in **fs** named **residuals** which stores residuals (**fs_lm$residuals**) from the fitted SLR model.
3. Draw a residual plot for your SLR model on father-and-son data using **fs** data frame and **geom_point()** in ggplot2. Use father's height in **fs** as X-axis and use residuals in **fs** as Y-axis. Do you observe constant variance of errors?
4. Create a data frame named **fs_lagged_residuals** to store two columns of residuals with a lag of one. Let column 1 store all residuals but the last one. Let column 2 store all residuals but the first one.
5. Generate a linear regression on lagged residuals using **ggplot2** and **fs_lagged_residuals**. Print summary statistics of the SLR model of lagged residuals using **summary()**. Is there a significant linear relationship between lagged residuals?

You will find that father-and-son height data do not violate the assumptions of linearity or constant variance of errors, but residual i is correlated to residual i+1, suggesting **violation of independence of errors**. A closer look into the father-and-son height data will show that there are likely more than one sons in a family. As siblings from the same household might share the same factors, besides father's height, that contribute to their height, like diet and environmental exposures. So, the residuals of these siblings might be correlated. The easiest way to resolve the issue might be keep only one father-and-son pair from each family and redo regression analysis. Now you might understand why 80% of the time in a bioinformatic project is used in data cleaning.

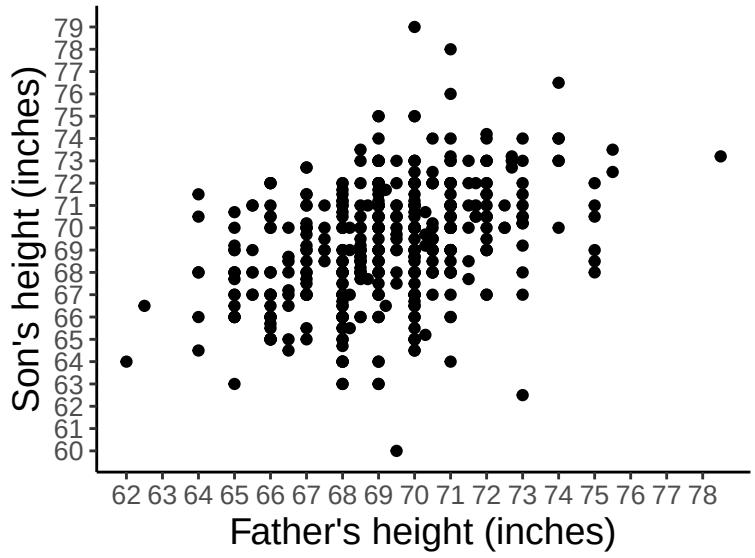
6. Use the following code to extract only one father-and-son pair from each family.

```
fs <- heights[which(heights$Gender == "M"), c("Family", "Father", "Height")]
temp <- fs[!duplicated(fs$Family), c("Father", "Height")]
head(temp)
fs <- temp
rm(temp)
```

7. Redo linear regression analysis on cleaned **fs** data set.

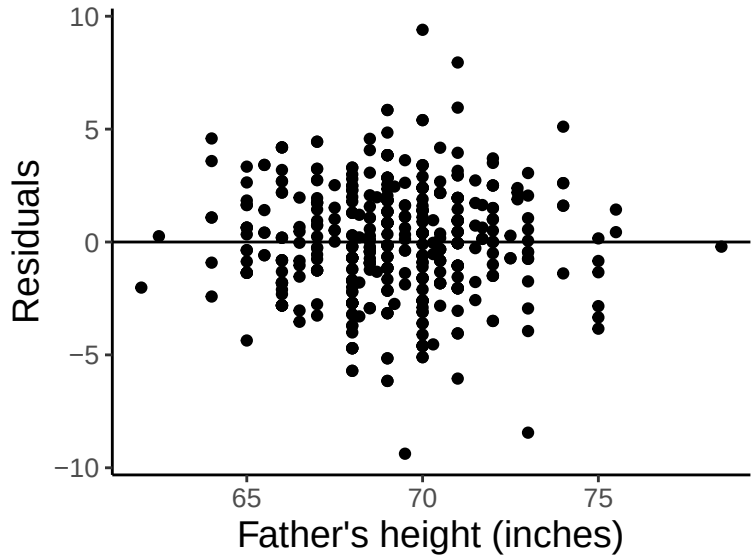
```r
# define general ggplot theme for all figures
th <- theme_classic() +
  theme(plot.title = element_text(hjust = 0.5, size = 14),
        axis.text = element_text(size = 10),
        axis.title = element_text(size = 14),
        strip.text = element_text(size = 10))

# scatter plot to check linearity
ggplot(data = fs, aes(x = Father, y = Height)) +
  geom_point() +
  labs(x = "Father's height (inches)", y = "Son's height (inches)") +
  scale_x_continuous(breaks = seq(min(fs$Father), max(fs$Father), 1)) +
  scale_y_continuous(breaks = seq(min(fs$Height), max(fs$Height), 1)) +
  th +
  ggsave("scatter_plot_father_and_son_heights_all.png")
```

```
## Saving 4 x 3 in image
```




```r
# residual plot to check linearity and constant variance of errors
fs$residuals <- fs_lm$residuals
ggplot(data = fs, aes(x = Father, y = residuals)) +
  geom_point() +
  geom_abline(intercept = 0, slope = 0) +
  labs(x = "Father's height (inches)", y = "Residuals") +
  th +
  ggsave("residual_plot_father_and_son_heights_all.png")
```

```
## Saving 4 x 3 in image
```




```
# regression or correlation between residuals to check independence of residuals
fs_lagged_residuals <- data.frame(x = fs$residuals[-length(fs$residuals)],
                                  y = fs$residuals[-1])
cor(fs_lagged_residuals$x, fs_lagged_residuals$y)
```

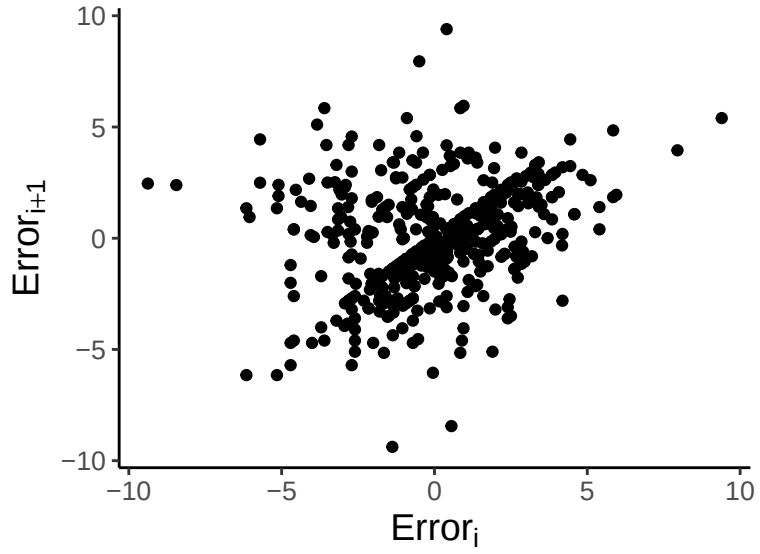
```
## [1] 0.2594269
```

```
fs_lagged_residuals_lm <- lm(y ~ x, data = fs_lagged_residuals)
summary(fs_lagged_residuals_lm)
```

```
##
## Call:
## lm(formula = y ~ x, data = fs_lagged_residuals)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.0197 -1.4034 -0.1341  1.4644  9.2955
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.0002413  0.1088112  -0.002    0.998
## x            0.2594971  0.0449435   5.774 1.42e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.344 on 462 degrees of freedom
## Multiple R-squared:  0.0673, Adjusted R-squared:  0.06528
## F-statistic: 33.34 on 1 and 462 DF,  p-value: 1.424e-08
```

```
ggplot(data = fs_lagged_residuals, aes(x = x, y = y)) +
  geom_point() +
  labs(x = bquote(Error[i]), y = bquote(Error[i+1])) +
  th +
  ggsave("scatter_plot_of_residuals.png")
```

```
## Saving 4 x 3 in image
```




```
# redo analysis with cleaned data
fs <- heights[which(heights$Gender == "M"), c("Family", "Father", "Height")]
temp <- fs[!duplicated(fs$Family), c("Father", "Height")]
head(temp)
```

```
##    Father Height
## 1    78.5   73.2
## 5    75.5   73.5
## 9    75.0   71.0
## 11   75.0   70.5
## 16   75.0   72.0
## 23   74.0   76.5
```

```
fs <- temp
rm(temp)
```

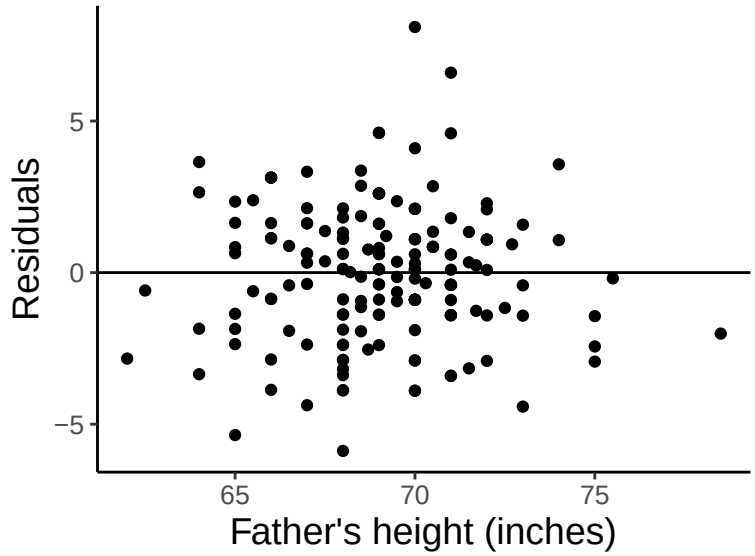
Exercise 1.7. (Optional) Identify outliers

1. Fit a SLR into the new **fs** data set that has only one father-and-son pair from each family using **lm()**. Store the regression result in a variable named **fs_lm**.
2. Draw a residual plot using the new **fs_lm** model.
3. Do you see outliers, high leverage points, or influential points?

```
# fit a SLR model
fs_lm <- lm(Height ~ Father, data = fs)

# draw a residual plot
fs$residuals <- fs_lm$residuals
ggplot(data = fs, aes(x = Father, y = residuals)) +
  geom_point() +
```

```
  geom_abline(intercept = 0, slope = 0) +
  labs(x = "Father's height (inches)", y = "Residuals") +
  th +
  ggsave("residual_plot_father_and_son_heights_cleaned.png")
```

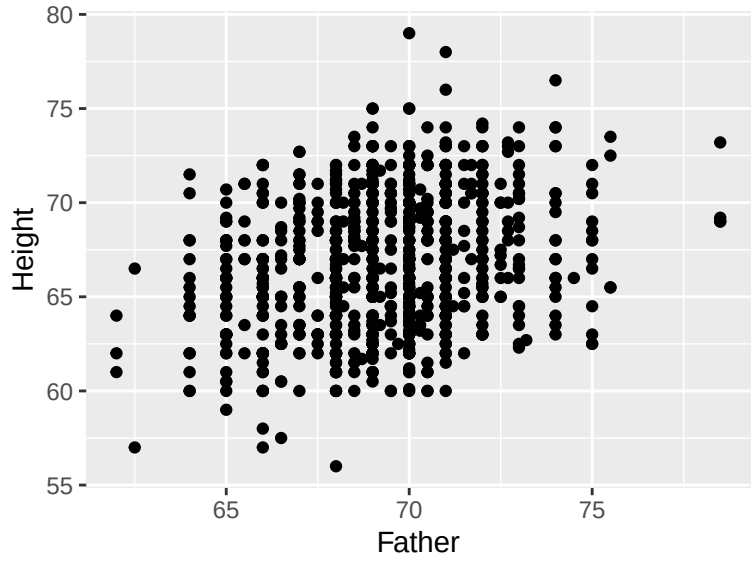
```
## Saving 4 x 3 in image
```



Exercise 2.1.

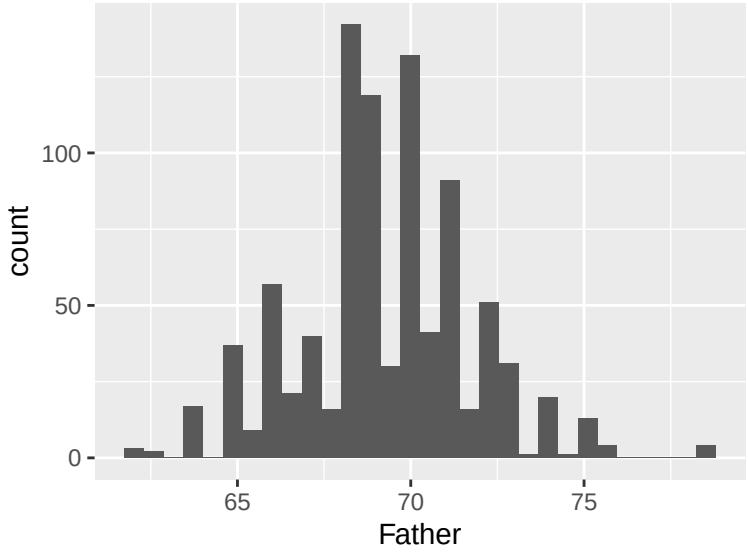
1. Draw a scatter plot between father's and children's height using **geom_point()** in ggplot2.
2. Draw a histogram of the distribution of father's height using **geom_histogram()** in ggplot2.

```
# scatter plot
ggplot(data = heights) +
  geom_point(mapping = aes(x = Father, y = Height))
```



```r
# histogram
ggplot(data = heights) +
  geom_histogram(mapping = aes(x = Father))
```

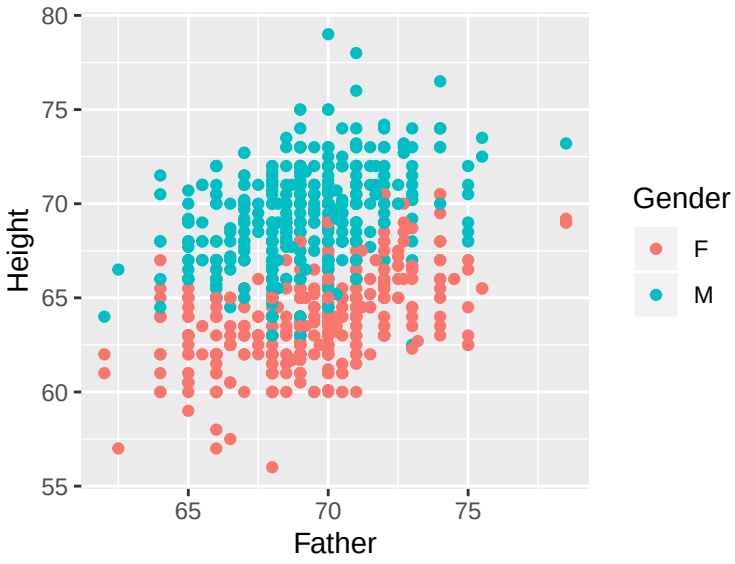
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



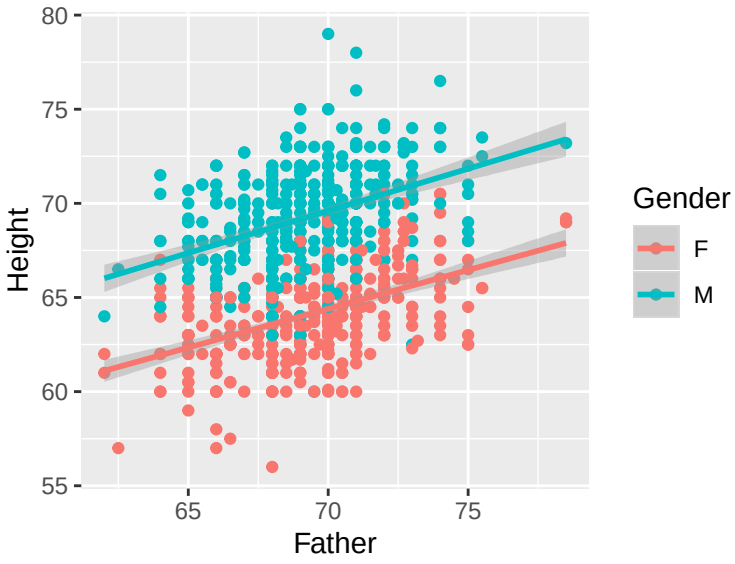
Exercise 2.2.

1. Add colors to scatter plot to differentiate sons and daughters using **geom_point(mapping = aes(color = Gender))** in ggplot2.
2. Following step 1, add a SLR regression line to the scatter plot using **geom_smooth(method = "lm")**.
3. Move aes from **geom_point()** and **geom_smooth()** to the **ggplot()**.
4. Draw a overall regression line on height data irrespect of gender using **geom_point()** and **geom_smooth()**. In order to do so, you will have to specify data in **geom_point()** and **geom_smooth()**, separately.

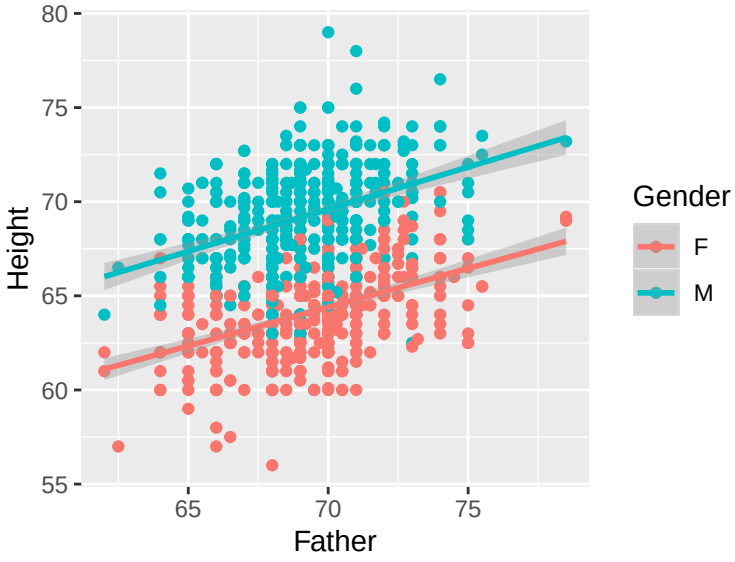
```r
# add colors
ggplot(data = heights) +
  geom_point(mapping = aes(x = Father, y = Height, color = Gender))
```



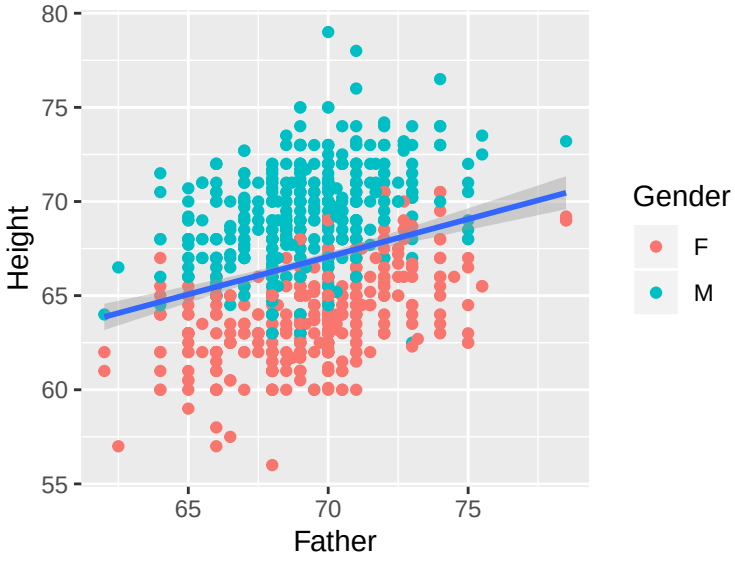

```
# add another layer
ggplot(data = heights) +
  geom_point(mapping = aes(x = Father, y = Height, color = Gender)) +
  geom_smooth(mapping = aes(x = Father, y = Height, color = Gender),
              method = "lm")
```




```
# move aes up
ggplot(data = heights, mapping = aes(x = Father, y = Height, color = Gender)) +
  geom_point() +
  geom_smooth(method = "lm")
```




```
# or don't move aes up
ggplot(data = heights) +
  geom_point(mapping = aes(x = Father, y = Height, color = Gender)) +
  geom_smooth(mapping = aes(x = Father, y = Height), method = "lm")
```

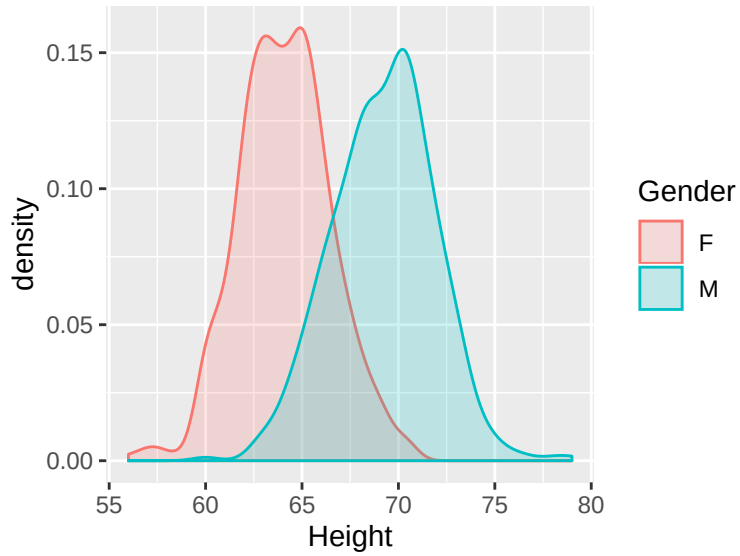


Exercise 2.3.

1. Draw a dentisy plot of the distributions of sons' and daughters' heights using **geom_density()**.
2. Create a new variable named **tg_summary** and read data from **"tooth_growth_summary.txt"** using **read.table(file, header = T, stringsAsFactors = F)**. Use **tg_summary***dose* $<-$ $factor(tg_summary$**dose)** to convert the column of dose from integer to factor. It describes the effect of Vitamin C on tooth growth in Guinea pigs. Three dose levels of Vitamin C (0.5, 1, and 2 mg) with each of two delivery methods [orange juice (OJ) or ascorbic acid (VC)] are used.
3. Draw a bar plot of the distributions of tooth growth of guinea pig under different VC doses and delivery methods using **geom_bar(stat="identity", position=position_dodge())** and

geom_errorbar(mapping = aes(ymin, ymax), width=.2, position=position_dodge(.9))

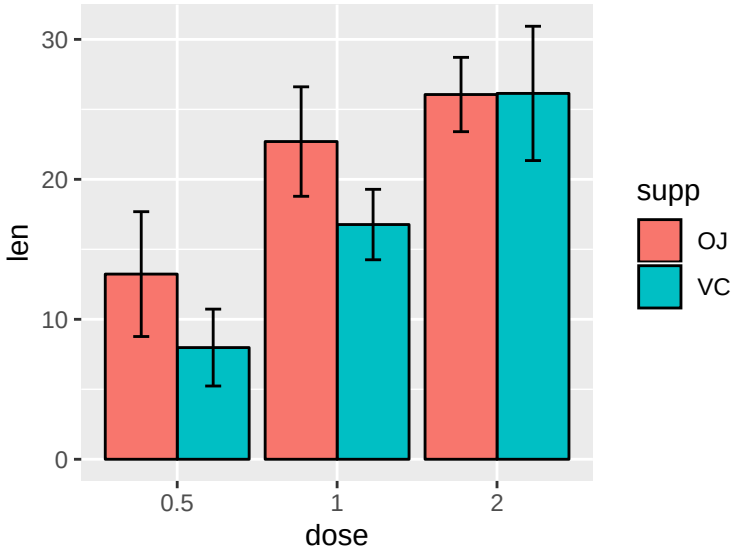
```
# draw density plot
ggplot(data = heights, aes(Height, fill = Gender, color = Gender)) +
  geom_density(alpha = 0.2)
```




```
# read tooth growth data form file
tg_summary <- read.table("tooth_growth_summary.txt", header = T)
tg_summary$dose <- factor(tg_summary$dose)
tg_summary
```

```
##   supp dose   len       sd
## 1   OJ  0.5 13.23 4.459709
## 2   OJ    1 22.70 3.910953
## 3   OJ    2 26.06 2.655058
## 4   VC  0.5  7.98 2.746634
## 5   VC    1 16.77 2.515309
## 6   VC    2 26.14 4.797731
```

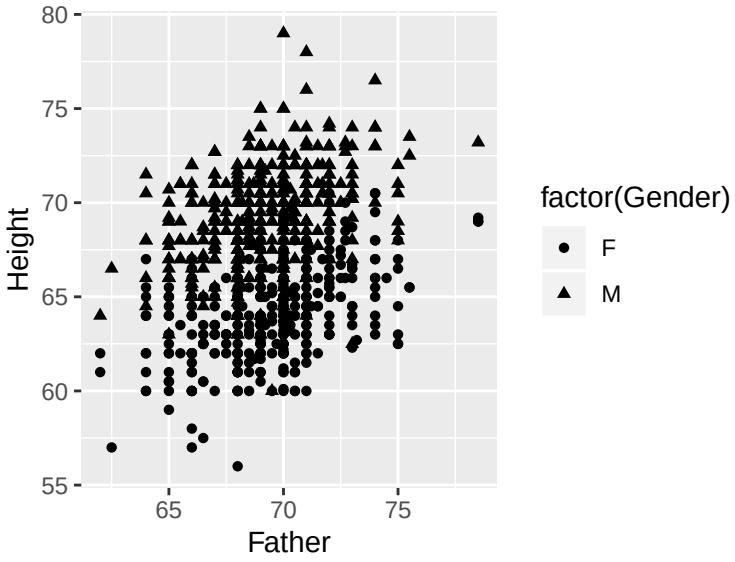
```
# draw a bar plot with one standar error bar
ggplot(tg_summary, aes(x=dose, y=len, fill=supp)) +
  geom_bar(stat="identity", color="black", position=position_dodge()) +
  geom_errorbar(aes(ymin=len-sd, ymax=len+sd), width=.2, position=position_dodge(.9))
```



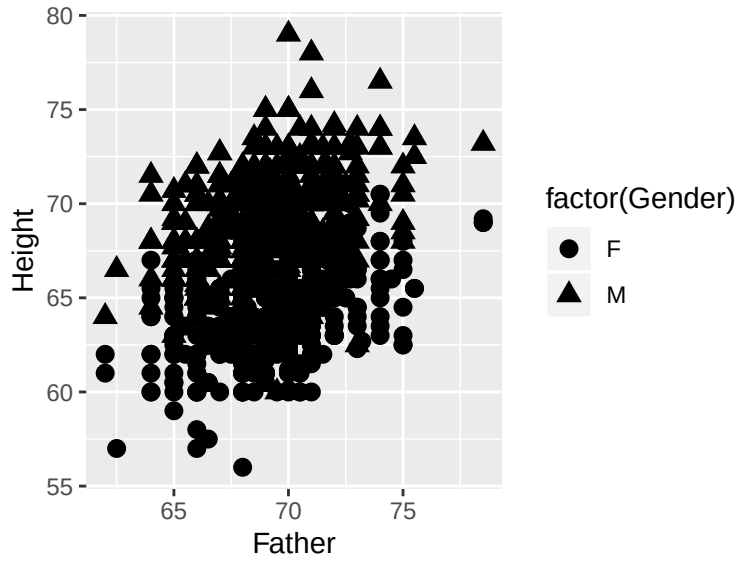
Exercise 2.4.

1. Use the shape of the points to represent an individual's gender using **geom_point(mapping = aes(shape = factor(Gender)))**
2. Change the size of the points using **geom_point(size = 3)**
3. Modify the title and axis labels using **labs(title, x, y)**
4. Change the overall theme of the plot using **theme_classic()** or other default theme you like.

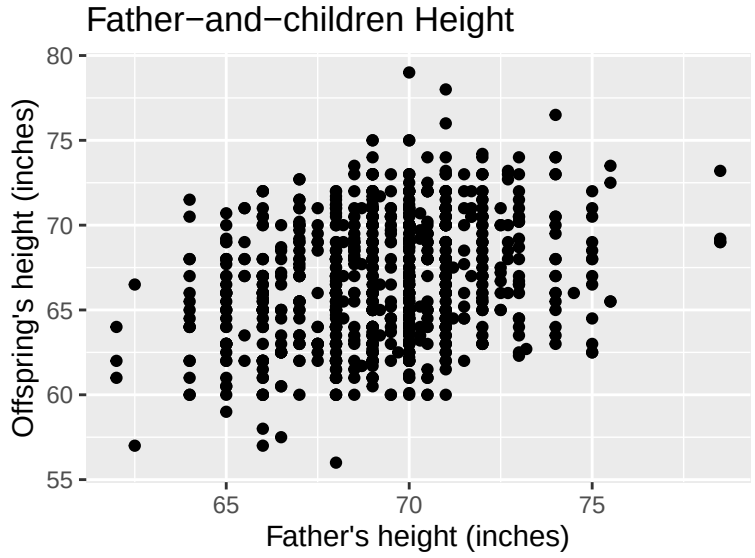
```
# use the shape of the points to represent the gender of observations
ggplot(data = heights) +
  geom_point(mapping = aes(x = Father, y = Height, shape = factor(Gender)))
```



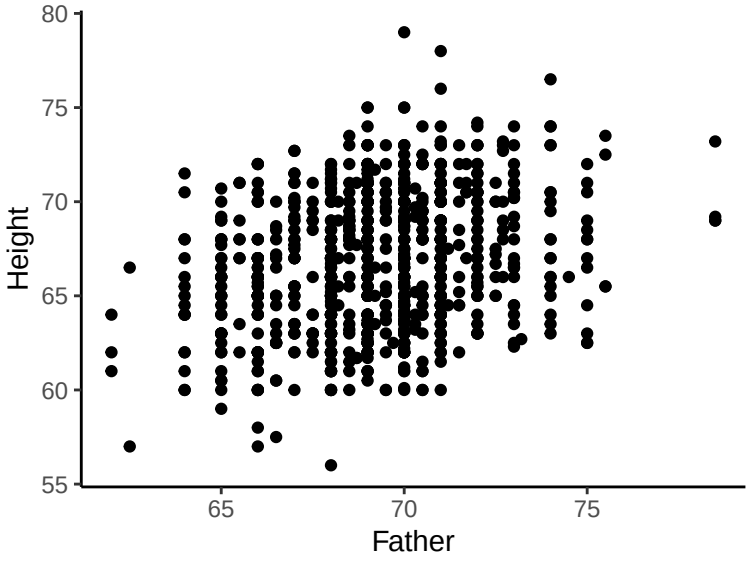

```
# change the szie of the points
ggplot(data = heights) +
  geom_point(mapping = aes(x = Father, y = Height, shape = factor(Gender)), size = 3)
```



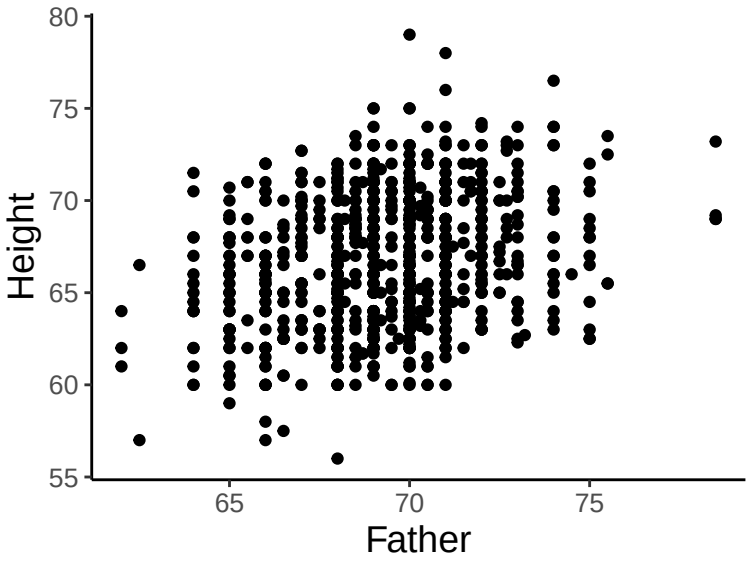

```
# modify label of axis
ggplot(data = heights) +
  geom_point(mapping = aes(x = Father, y = Height)) +
  labs(title = "Father-and-children Height",
       x = "Father's height (inches)",
       y = "Offspring's height (inches)")
```




```
# change overall theme of the plot
ggplot(data = heights) +
  geom_point(mapping = aes(x = Father, y = Height)) +
  theme_classic()
```




```
# my choice
# it's important to increase font sizes of axis and axis text for visual pleasure, especially in a pape
th <- theme_classic() +
  theme(plot.title = element_text(hjust = 0.5, size = 14),
        axis.text = element_text(size = 10),
        axis.title = element_text(size = 14),
        strip.text = element_text(size = 10))
ggplot(data = heights) +
  geom_point(mapping = aes(x = Father, y = Height)) +
  th
```

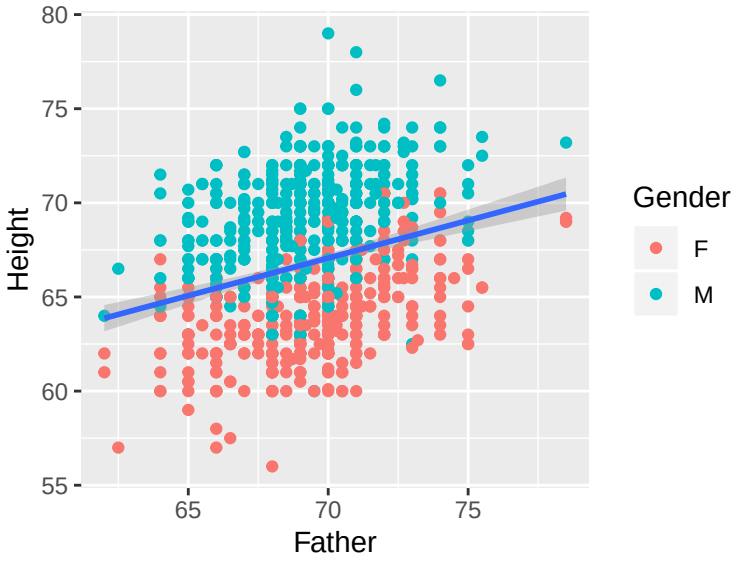


Exercise 2.5.

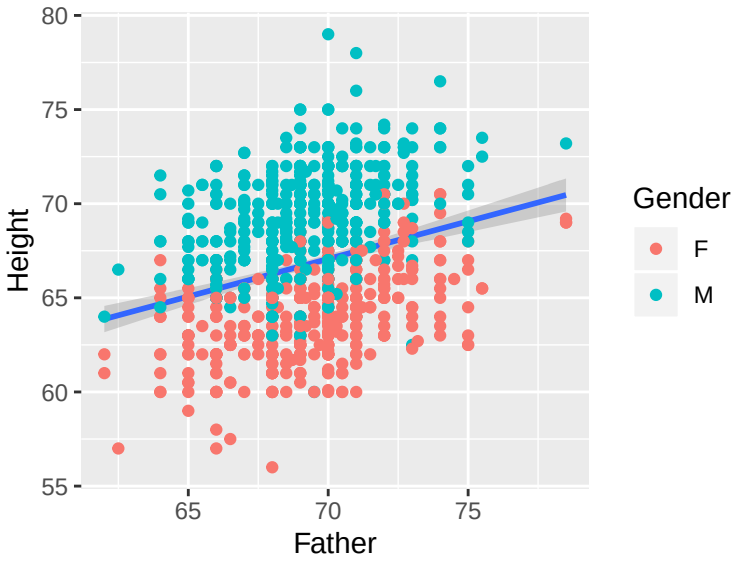
1. Change the sequence of **geom_points()** and **geom_smooth()**. Do you notice the difference?

2. Create a variable **cbPalette** to store the colorblind friendly color palette you like. Use a colorblind friendly color palette in your plot using **scale_colour_manual(values = cbPalette)**. ~~~ # This is my go-to color palette # This palette is from http://www.cookbook-r.com/Graphs/Colors_(ggplot2)/ # More info on colors can be found here: http://jfly.iam.u-tokyo.ac.jp/color/ cbbPalette <- c("#000000", "#E69F00", "#56B4E9", "#009E73", "#F0E442", "#0072B2", "#D55E00", "#CC79A7") ~~~
3. Divide your dataset into two facets according to gender using **facet_grid(. ~ Gender)**.
4. Create a variable named **p** that store a scatter plot with father's height as x-axis and children's height as y-axis.
5. Add a regression line to **p** using **geom_smooth(method = "lm")**.
6. Save a plot using **ggsave()**.

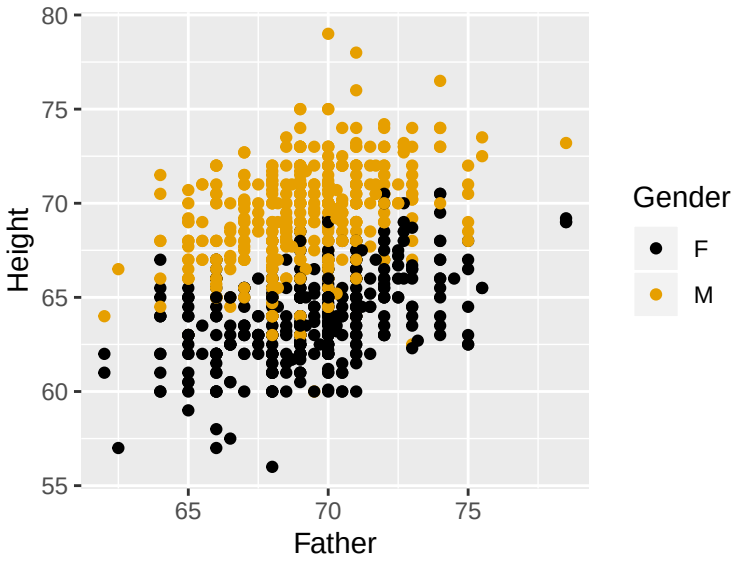
```
# change the sequence of plotting
ggplot(data = heights) +
  geom_point(mapping = aes(x = Father, y = Height, color = Gender)) +
  geom_smooth(mapping = aes(x = Father, y = Height), method = "lm")
```



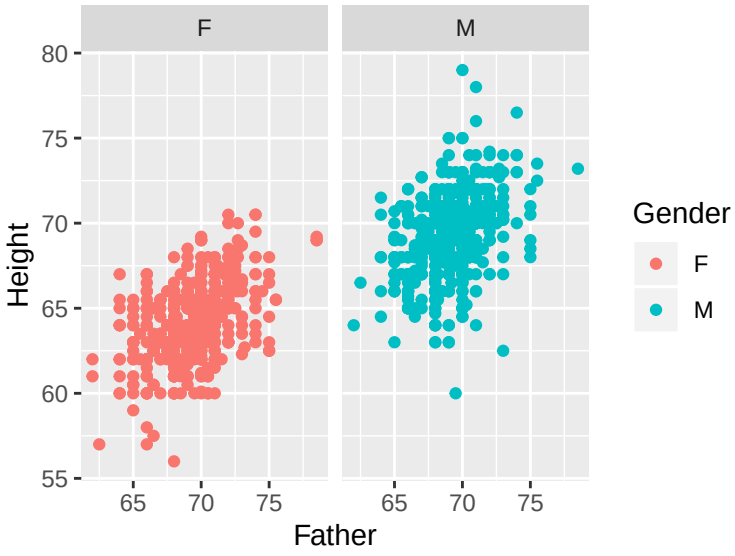

```
ggplot(data = heights) +
  geom_smooth(mapping = aes(x = Father, y = Height), method = "lm") +
  geom_point(mapping = aes(x = Father, y = Height, color = Gender))
```



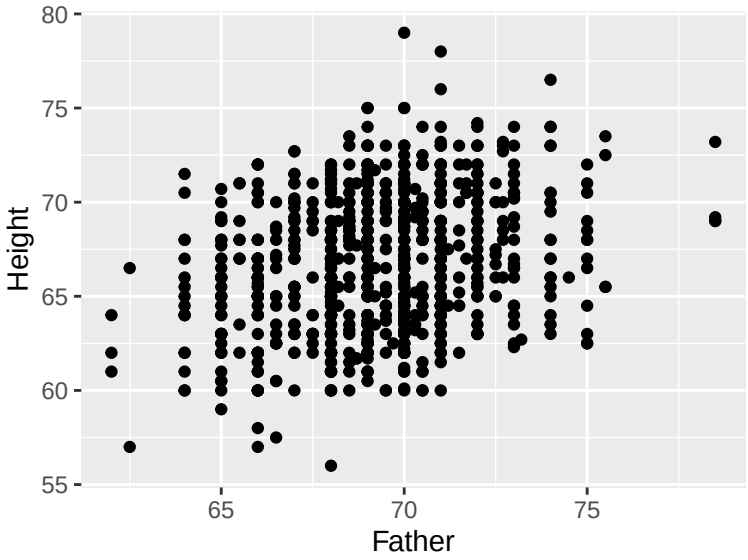

```
# change colors
cbPalette <- c("#000000", "#E69F00", "#56B4E9", "#009E73", "#F0E442",
               "#0072B2", "#D55E00", "#CC79A7")
ggplot(data = heights) +
  geom_point(mapping = aes(x = Father, y = Height, color = Gender)) +
  scale_colour_manual(values = cbPalette)
```



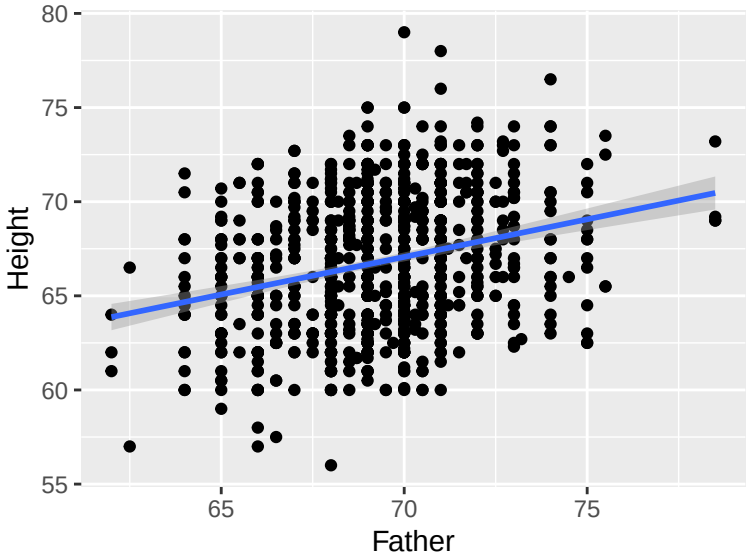

```
# create facet
ggplot(data = heights) +
  geom_point(mapping = aes(x = Father, y = Height, color = Gender)) +
  facet_grid(. ~ Gender)
```




```
# save the plot to a variable
p <- ggplot(data = heights, mapping = aes(x = Father, y = Height)) +
  geom_point()
print(p)
```

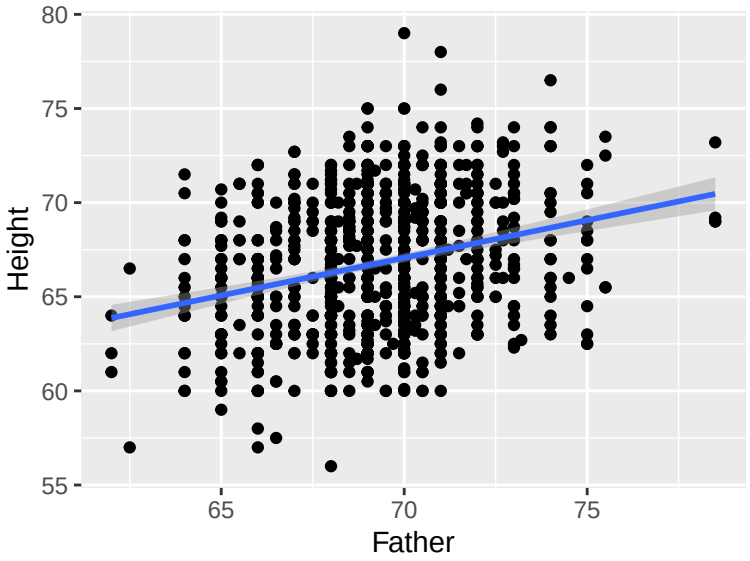



```
# add a regression line to p
p <- p + geom_smooth(method = "lm")
print(p)
```




```
# save a plot using ggsave()
p + ggsave("scatter_plot_height.png")
```

```
## Saving 4 x 3 in image
```



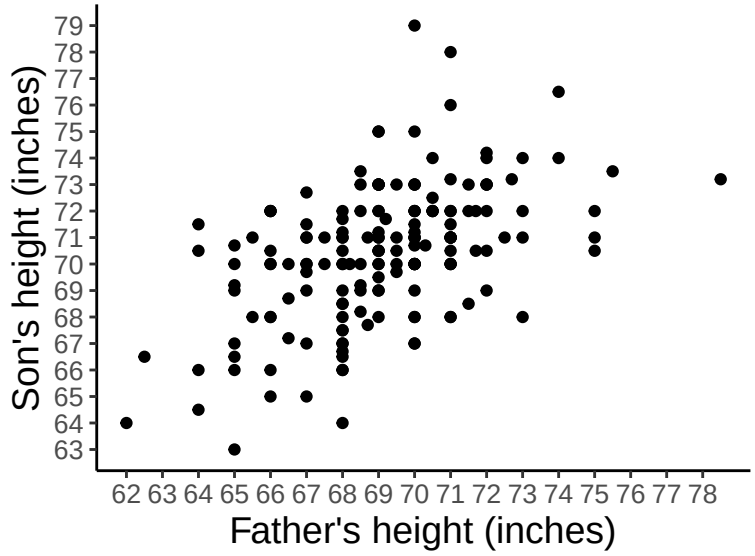
Additional R Codes

The following R codes were used in generation of the slides for SLR and R visualization.

```
####################################
## generate a scatter plot of father-and-son height
####################################

# define color palette and general ggplot theme for all figures
```

```
cbPalette <- c("#000000", "#E69F00", "#56B4E9", "#009E73", "#F0E442",
               "#0072B2", "#D55E00", "#CC79A7")
th <- theme_classic() +
  theme(plot.title = element_text(hjust = 0.5, size = 14),
        axis.text = element_text(size = 10),
        axis.title = element_text(size = 14),
        strip.text = element_text(size = 10))

# figure 1. Scatter plot of heights of all father-and-son pairs
ggplot(data = fs, aes(x = Father, y = Height)) +
  geom_point() +
  labs(x = "Father's height (inches)", y = "Son's height (inches)") +
  scale_x_continuous(breaks = seq(min(fs$Father), max(fs$Father), 1)) +
  scale_y_continuous(breaks = seq(min(fs$Height), max(fs$Height), 1)) +
  th +
  ggsave("scatter_plot_father_and_son_heights_all.png")
```

```
## Saving 4 x 3 in image
```




```
###################################
## example of car's stopping distance
###################################

# example of stopping distance of cars on the basis of the speed
data("cars", package = "datasets")
head(cars)
```

```
##   speed dist
## 1     4    2
## 2     4   10
## 3     7    4
## 4     7   22
## 5     8   16
## 6     9   10
```

```r
# generate a SLR of stopping distance of cars on the basis of the speed
cars_lm <- lm(dist ~ speed, data = cars)
print(cars_lm)
```

```
##
## Call:
## lm(formula = dist ~ speed, data = cars)
##
## Coefficients:
## (Intercept)        speed
##     -17.579        3.932
```

```r
# summary stat of car's stopping distance model
summary(cars_lm)
```

```
##
## Call:
## lm(formula = dist ~ speed, data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -29.069  -9.525  -2.272   9.215  43.201
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -17.5791     6.7584  -2.601   0.0123 *
## speed         3.9324     0.4155   9.464 1.49e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.38 on 48 degrees of freedom
## Multiple R-squared:  0.6511, Adjusted R-squared:  0.6438
## F-statistic: 89.57 on 1 and 48 DF,  p-value: 1.49e-12
```

```r
# predict based on new speeds
new_speed <- data.frame(speed = 19)
predict(cars_lm, newdata = new_speed)
```

```
##        1
## 57.13667
```

```r
# obtain 95% confidence interval for prediction
predict(cars_lm, newdata = new_speed, interval = "confidence")
```

```
##        fit      lwr      upr
## 1 57.13667 51.82913 62.44421
```

```r
# obtain 95% prediction tern
predict(cars_lm, newdata = new_speed, interval = "prediction")
```

```
##        fit      lwr      upr
## 1 57.13667 25.76176 88.51159
```

```
###################################
## generate a SLR model on father-and-daughter height
###################################

fd <- heights[which(heights$Gender == "F"), c("Father", "Height")]
fd_lm <- lm(Height ~ Father, data = fd)
```

Reference and Helpful Materials

Statistical Analysis in R

- An Introduction to Statistical Learning: With Applications in R
- Applied linear statistical models

Visualization in R

- R cheat sheet
- Google ggplot2 reference to to go the following website, https://ggplot2.tidyverse.org/reference/index.html
- Stack Overflow with keywords, "R" or "ggplot2"
- R for Data Science available at https://r4ds.had.co.nz/
- The R Graphics Cookbook by Winston Chang