Problem Set 2
MFE 402: Econometrics
Professor Rossi

This problem set is designed to review material on the sampling distribution of least squares.

Question 1

(a.) The least square intercept can be expressed as:

$$b_0 = \bar{Y} - b_1\bar{X}$$

Where b_1 is expressed as:

$$b_1 = \frac{\sum(X_i - \bar{X})Y_i}{\sum(X_i - \bar{X})^2}$$

$$b_0 = \frac{1}{N}\sum_{i=0}^{N} Y_i - \bar{X}\sum_{i=0}^{N} C_i Y_i$$

Where C_i is expressed as:

$$C_i = \frac{X_i - \bar{X}}{\sum_{i=0}^{N}(X_i - \bar{X})^2}$$

$$b_0 = \sum_{i=0}^{N}(\frac{1}{N} - \bar{X}C_i)Y_i$$

(b.) $\bar{Y} = \beta_0 + \beta_1\bar{X} + \bar{\epsilon}$ Plug in $Y_i = \beta_0 + \beta_1 X_i + \epsilon_i$ into b_1

$$b_1 = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(\beta_0 + \beta_1 X_i + \epsilon_i - \beta_0 - \beta_1\bar{X} - \bar{\epsilon})}{\sum_{i=1}^{n}(X_i - \bar{X})^2}$$

$$= \frac{\sum_{i=1}^{n}(X_i - \bar{X})(\beta_1(X_i - \bar{X}) - \bar{\epsilon} + \epsilon_i)}{\sum_{i=1}^{n}(X_i - \bar{X})^2}$$

$$= \frac{\sum_{i=1}^{n}(X_i - \bar{X})^2\beta_1 + \sum_{i=1}^{n}(X_i - \bar{X})(\epsilon_i - \bar{\epsilon})}{\sum_{i=1}^{n}(X_i - \bar{X})^2}$$

$$= \beta_1 + \frac{\sum_{i=1}^{n}(X_i - \bar{X})(\epsilon_i - \bar{\epsilon})}{\sum_{i=1}^{n}(X_i - \bar{X})^2}$$

$\bar{\epsilon} = 0$ since it is normally distributed with mean 0 Since $E[\epsilon_i] = 0$ it follows that

$$E[b_1] = \beta_1$$

Hence,

$$E[b_0] = E(\bar{Y} - b_1\bar{X}) = \beta_0 + \beta_1\bar{X} - E[b_1]\bar{X} = \beta_0 + \beta_1\bar{X} - \beta_1\bar{X} = \beta_0$$

(c.)

$$b_0 = \sum_{i=0}^{N}(\frac{1}{N} - \bar{X}C_i)Y_i$$

$$Y_i = \beta_0 + \beta_1 X_i + \epsilon_i$$

$$= \sum_{i=0}^{n}(\frac{1}{N} - \bar{X}C_i)(\beta_0 + \beta_1 X_i) + \sum_{i=0}^{n}(\frac{1}{N} - \bar{X}C_i)\epsilon_i$$

Since $\sum_{i=0}^{n}(\frac{1}{N} - \bar{X}C_i)(\beta_0 + \beta_1 X_i)$ is a constant

$$var(b_0) = var(\sum_{i=0}^{n}(\frac{1}{N} - \bar{X}C_i)\epsilon_i)$$

$$= \sum_{i=0}^{n}(\frac{1}{N} - \bar{X}C_i)^2 var(\epsilon_i)$$

$$= \sigma^2 \sum_{i=0}^{n}(\frac{1}{N} - \bar{X}C_i)^2$$

$$= \sigma^2 \sum_{i=0}^{n}(\frac{1}{N^2} - \frac{2}{N}\bar{X}C_i + \bar{X}^2 C_i^2)$$

$\sum_{i=0}^{n} C_i = 0$ and $\sum_{i=0}^{n} C_i^2 = \frac{1}{\sum_{i=0}^{n}(X_i - \bar{X})^2}$

$$= \sigma^2(\sum_{i=0}^{n}\frac{1}{N^2} - \frac{2}{N}\bar{X}\sum_{i=0}^{n} C_i + \bar{X}^2\sum_{i=0}^{n} C_i^2)$$

$$= \sigma^2[\frac{1}{N} + \frac{\bar{X}^2}{\sum_{i=0}^{n}(X_i - \bar{X})^2}]$$

$s_X^2 = \frac{\sum_{i=0}^{n}(X_i - \bar{X})^2}{N-1}$ Hence

$$var(b_0) = \sigma^2[\frac{1}{N} + \frac{\bar{X}^2}{(N-1)s_X^2}]$$

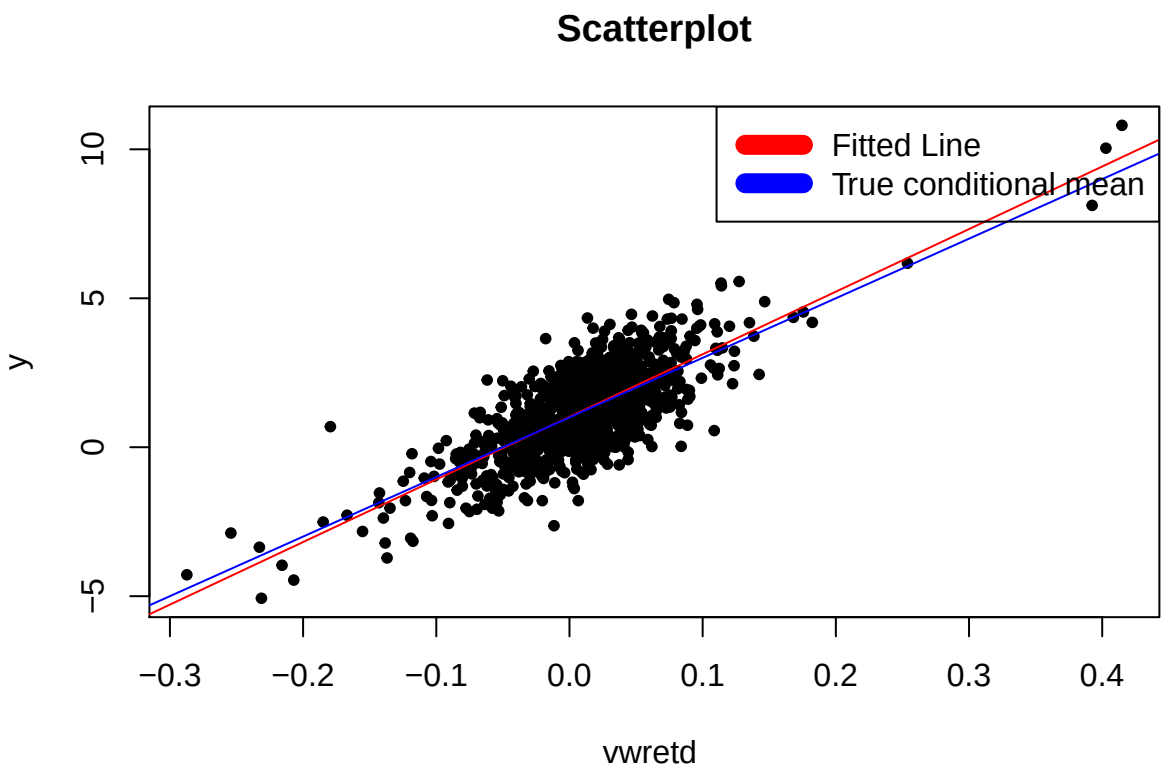
Question 2

(a)

```
simple_linear_regression <- function(beta_0, beta_1, x, sigma){
  return(beta_0 + beta_1 * x + rnorm(length(x),0, sigma))
}
```

(b)

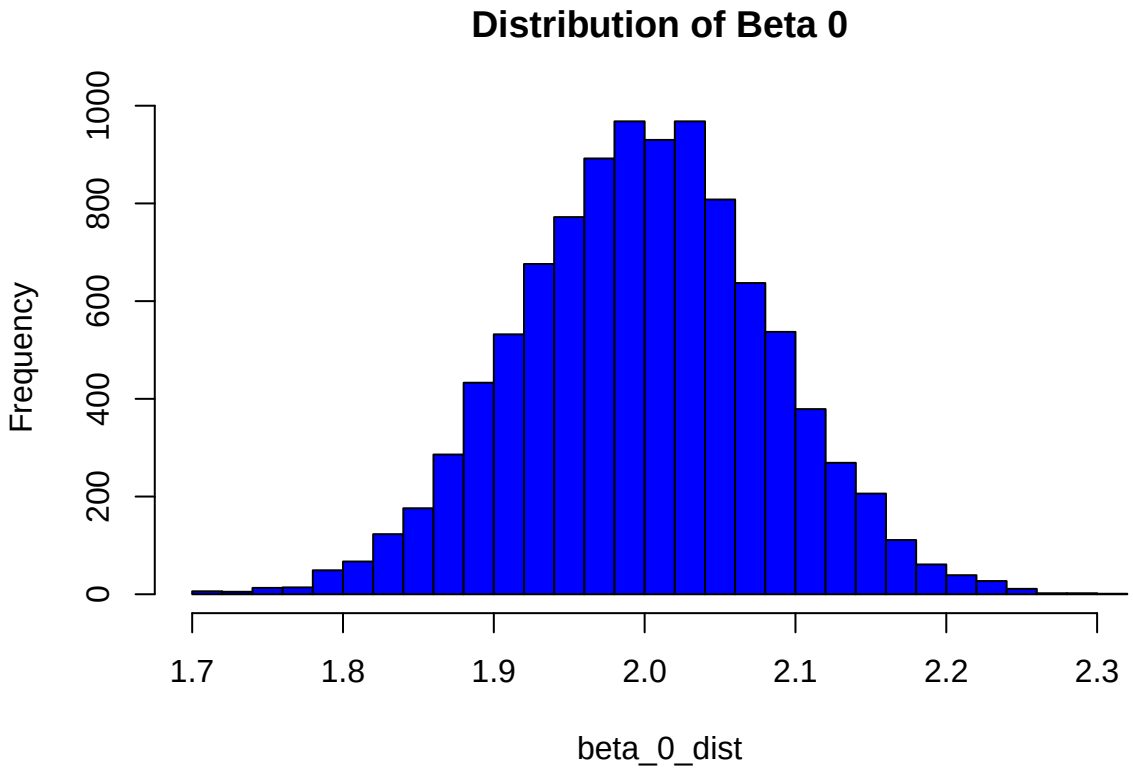
```
library(DataAnalytics)
data(marketRf)
vwretd=marketRf$vwretd
y <- simple_linear_regression(beta_0 = 1, beta_1 = 20, vwretd, sigma=1)
plot(vwretd,y,main="Scatterplot",xlab="vwretd ", ylab="y", pch=20)
abline(lm(y~vwretd), col="red")
abline(a = 1, b =20,col="blue")
legend("topright", c("Fitted Line", "True conditional mean"), col=c("red", "blue"), lwd=10)
```



Question 3

(a) Randomly select a sample of 300 from vwretd

```
nsample =10000
beta_0_dist= double(nsample)
set.seed(0903)
sample_x <- sample(vwretd,300, replace = FALSE)
set.seed(1234)
for(i in 1:nsample){
  y =  simple_linear_regression(beta_0 = 2, beta_1 = 0.6, sample_x, sigma=sqrt(2))
  beta_0_dist[i] = lm(y~sample_x)$coef[1]
}
hist(beta_0_dist, breaks = 40, col = "blue", main = "Distribution of Beta 0")
```



(b) Empirical value of $E[b_0]$:

```
mean(beta_0_dist)
```

```
## [1] 1.999248
```

Theoretical value of $E[b_0] = \beta_0 = 2$ According to 1b.

The two values are very close to each other

(c)

Empirical value of $Var[b_0]$:

```
var(beta_0_dist)
```

```
## [1] 0.006858187
```

Using result from 1c:

$$var(b_0) = \sigma^2[\frac{1}{N} + \frac{\bar{X}^2}{\sum_{i=0}^{n}(X_i - \bar{X})^2}]$$

Theoretical value of$Var[b_0]$:

```
s_square <- sum((sample_x - mean(sample_x))^2)
2 * ((1/300) + (mean(sample_x)^2)/s_square)
```

```
## [1] 0.006791164
```

The two values are very close to each other

Question 4

(a)

```
library(reshape2)
data(Vanguard)
VFIAX <- subset(Vanguard, ticker == "VFIAX")
VFIAX_reshaped=dcast(VFIAX,date~ticker,value.var="mret")
vwretd_date <-marketRf[c("date","vwretd")]
#V_reshaped$VFIAX
merged_data <- merge(VFIAX_reshaped,vwretd_date,by="date")
beta_1_hyptest <- lm(VFIAX~vwretd, data = merged_data)$coef[2]
beta_0_hyptest <- lm(VFIAX~vwretd, data = merged_data)$coef[1]

df <- dim(merged_data)[1]

y.hat <- beta_0_hyptest + beta_1_hyptest * merged_data$vwretd
y <- merged_data$VFIAX
x <- merged_data$vwretd
x_bar <- mean(merged_data$vwretd)
var_x <- sum((x - mean(x))^2)
s_square <- sum((y - y.hat)^2)/(df-2)

s_b_1 <- sqrt(s_square)/sqrt(var_x)
t.value_beta1 <- (beta_1_hyptest - 1)/ s_b_1
t.value_beta1
```

```
##   vwretd
## 2.593157
```

```
qt(c(.025, .975), df=df-1)
```

```
## [1] -1.975905  1.975905
```

t value for β_1 is:

```
##   vwretd
## 2.593157
```

Using qt() to find the confidence interval:

```
qt(c(.025, .975), df=df-2)
```

```
## [1] -1.976013  1.976013
```

Hence we reject the null hypothesis.

(b) Find the p-value for β_0:

```
s_b_0 = sqrt(s_square * ((1/df) + (x_bar^2 / var_x )))
t.value_beta0 <- (beta_0_hyptest - 0)/ s_b_0
pvalue <- 2 *pt(-abs(t.value_beta0), df = df -2)
pvalue
```

```
## (Intercept)
##  0.04426054
```

Since the value is bigger than 0.01, We cant reject the null hypothesis

Question 5

(a) Standard error is the approximate standard deviation of a statistical sample population. Standard deviation measures the amount of variation for a subject of data from the mean. Standard error measures how far the sample mean of the data is likely to be from the true population mean.

(b) A sampling error is a statistical error that occurs when the selected sample is not representative of the entire population. Hence the result found in the sample would not represent the result that would be obtained from the entire population. Standard error is a measure of the sampling error.

(c) Steven needs to verify if the parameters obtained is statistically significant to be used as a predicitve model. He could use hypothesis testing to verify if the parameters are statistically significant. Standard errors can be used in this case to help obtain the t-value and p-value to accept or reject the hypothesis testing.

(d)

- t-value: She needs to calculate the the t acceptance level given the significance and degree of freedom. Then she needs to check if the t-value falls in the acceptance range. Reject null hypothesis if the t-value is out of the t acceptance level.
- p-value: She needs to check if the p-value is smaller or bigger than the significance level.e.g. 0.05 or 5%. Reject the null hypothesis if the number if smaller than the significance level.

Queston 6

(a). Find the cofficients using the lm function and plug into the function $\bar{Y} = b_0 + b_1\bar{X}$ where $\bar{X}$ is 0.05 in this case.

```
VGHCX <- subset(Vanguard, ticker == "VGHCX")
VGHCX_reshaped=dcast(VGHCX,date~ticker,value.var="mret")
vwretd_date <-marketRf[c("date","vwretd")]
VGHCX_vwretd <- merge(VGHCX_reshaped,vwretd_date,by="date")
out <- lm(VGHCX~vwretd, data = VGHCX_vwretd)
conditional_mean <- out$coef[1] + out$coef[2] * 0.05
conditional_mean
```

```
## (Intercept)
##  0.04367826
```

(b). Using the following formula to get the conditional SD of the return:

$$Var(\hat{Y}_f) = \sigma^2(\frac{1}{N} + \frac{(X_f - \bar{X})^2}{(N-1)s_X^2})$$

```
y_qn_6 <- VGHCX_vwretd$VGHCX
x_qn_6 <- VGHCX_vwretd$vwretd
x_star <- 0.1
n <- length(y_qn_6)
denominator <- sum((x_qn_6 - mean(x_qn_6))^2)
numerator <- (x_star - mean(x_qn_6))^2
y.hat <- out$coef[1] + out$coef[2] * x_qn_6
df <- length(y_qn_6)
s_2 <- sum((y_qn_6 - y.hat)^2)/(df-2)
conditional_sd <- sqrt((numerator/denominator) + (1/n)) * sqrt(s_2)
conditional_sd
```

```
## [1] 0.003026517
```

(c). Using the formual for s_{pred}:

$$s_{pred} = s\sqrt{1 + \frac{1}{N} + \frac{(X_f - \bar{X})^2}{\sum_{i=0}^{N}(X_i - \bar{X})^2}}$$

```
x_star <- 0.15
denominator <- sum((x_qn_6 - mean(x_qn_6))^2)
numerator <- (x_star - mean(x_qn_6))^2
s_pred <- sqrt((numerator/denominator) + (1/n) + 1) * sqrt(s_2)
s_pred
```

```
## [1] 0.02543671
```