HW4, Solutions

Problem #1

a. $E[Y_i] = E[\beta_0 + \epsilon_i] = \{E[aX + b] = aE[X] + b\} = \beta_0 + E[\epsilon_i] = \beta_0 + 0 = \beta_0$.

In plain English: "If randomly drawing many Y values from the population, the **average** of them would be equal to β_0".

b. $V[Y_i] = V[\beta_0 + \epsilon_i] = \{V[aX + b] = a^2 V[X]\} = V[\epsilon_i] = \sigma^2$

In plain English: "If randomly drawing many Y values from the population, their variance would be σ^2."

c. As Y_i is simply a shifted version of ϵ_i (shift of $+\beta_0$), which is normally distributed ($\epsilon_i \sim N$), Y_i is itself normally distributed.

Therefore, we get

$$Y_i \sim N(\beta_0, \sigma^2)$$

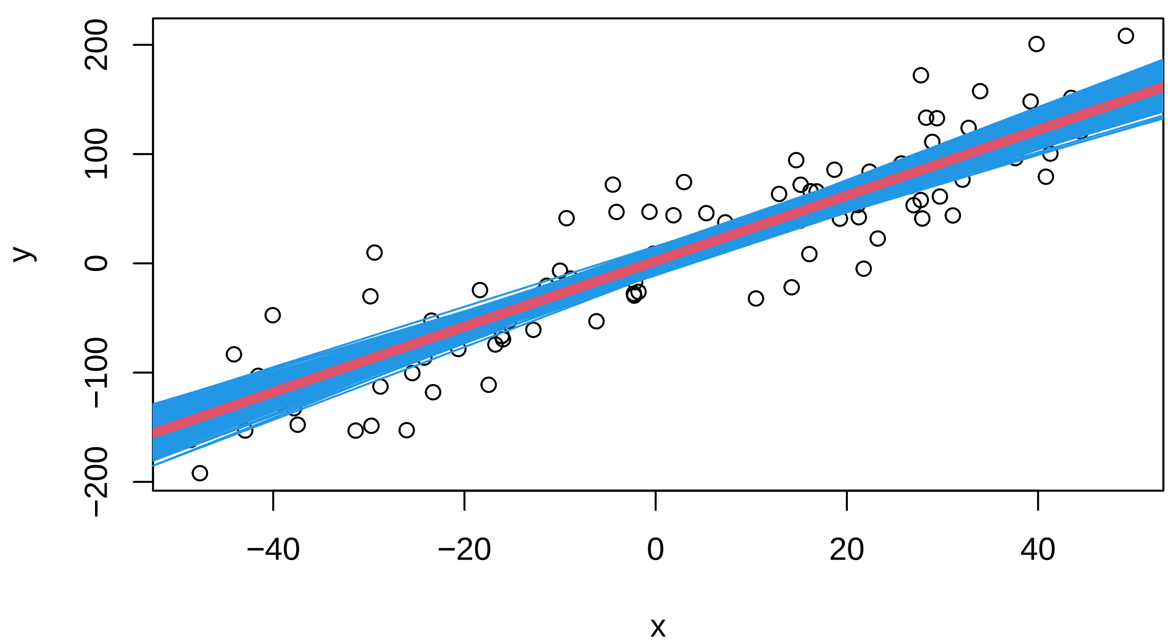
Problem #2

Generating and fixing explanatory variables x, with subsequently generating y as

$$y = 2 + 3x + \epsilon, \ \epsilon \sim N(0, 40^2)$$

```r
## Generate a 100 values for explanatory variable x,
## uniformly distributed from -50 to 50.
set.seed(1)
x <- runif(100, -50,50)

## Generate y from the model
#        y = 3 + 2*x + eps, eps ~ N(0, sigma^2),
## where sigma=40.
sigma <- 40
eps <- rnorm(100, 0, sigma)
y <- 2 + 3*x + eps

## Fit least squares regression y ~ x.
lm.obj <- lm(y ~ x)
```

```r
## Plot the resulting fitted line AND the true population line.
plot(y ~ x)
abline(lm.obj)
abline(2,3, col=2, lwd=3)

## Conduct 1,000 simulations of the following:
##      1. Generate y from the model:
##            y = 2 + 3*x + eps, eps ~ N(0, sigma^2),
##        where sigma = 40.
##      2. Calculate least squares estimates for y ~ x regression,
##        record the beta.hat & alpha.hat estimate (KEEP TRACK of them).
##      3. Add the fitted line to existing plot.
n.sim <- 1000
beta0.hat <- beta1.hat <- NULL

for (j in 1:n.sim){
  eps <- rnorm(100, 0, sigma)
  y <- 2+3*x + eps
  lm.obj <- lm(y ~ x)
  beta0.hat <- c(beta0.hat, coef(lm.obj)[1])
  beta1.hat <- c(beta1.hat, coef(lm.obj)[2])
  abline(lm.obj, col=4)
}

## Overlay a thick red population regression line over.
abline(2,3, col=2, lwd=5)
```



First, we take care of $\hat{\beta}_1$ estimates:

1. Practical and theoretical expected values of $\hat{\beta}_1$ approximately match, pointing to unbiasedness of least squares estimator.

Practical:

```
mean(beta1.hat)
```

```
## [1] 2.994051
```

Theoretical: $E[\hat{\beta}] = \beta = 3$.

2. Practical and theoretical sampling variance of $\hat{\beta}_1$ appear to be approximately matching.

Practical:

```
var(beta1.hat)
```

```
## [1] 0.02258858
```

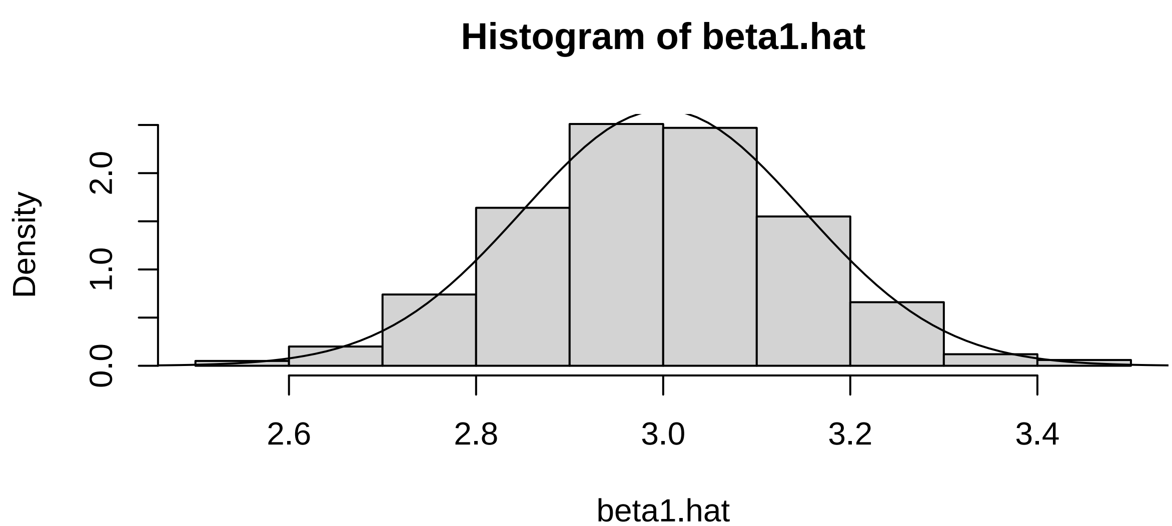
Theoretical: $V[\hat{\beta}_1] = \frac{\sigma^2}{\sum_i (x_i - \bar{x})^2} = 0.02258858$.

```
theor.var <- sigma^2/sum((x - mean(x))^2)
theor.var
```

```
## [1] 0.02257158
```

3. Practical (histogram) and theoretical ($\hat{\beta}_1 \sim N(\beta, V[\hat{\beta}_1])$, overlayed density curve) sampling distributions appear to approximately match, as expected.

```
hist(beta1.hat, freq=F)
my.dnorm <- function(z) dnorm(z, 3, sqrt(theor.var))
curve(my.dnorm, from=2.40, to =3.6, add=T)
```



Second, we take care of $\hat{\beta}_0$ estimates:

1. Practical and theoretical expected values of $\hat{\beta}_0$ approximately match, pointing to unbiasedness of least squares estimator.

Practical:

```
mean(beta0.hat)
```

```
## [1] 1.922462
```

Theoretical: $E[\hat{\beta}_0] = \beta_0 = 2$.

2. Practical and theoretical sampling variance of $\hat{\beta}_0$ appear to be approximately matching.

Practical:

```
var(beta0.hat)
```

```
## [1] 15.51018
```

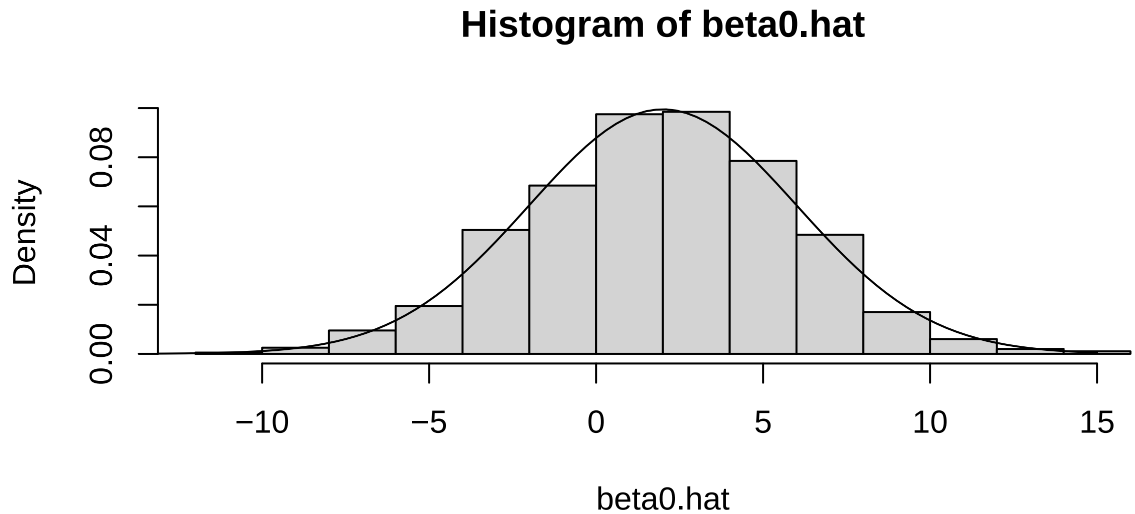
Theoretical: $V[\hat{\beta}_0] = \frac{\sigma^2 \sum_i x_i^2}{n \sum_i (x_i - \bar{x})^2} = 16.07189$.

```
n <- 100
theor.var <- sigma^2*(sum(x^2))/(n*sum((x - mean(x))^2))
theor.var
```

```
## [1] 16.07189
```

3. Practical (histogram) and theoretical ($\hat{\beta}_0 \sim N(\beta_0, V[\hat{\beta}_0])$, overlayed density curve) sampling distributions appear to approximately match, as expected.

```
hist(beta0.hat, freq=F)
my.dnorm <- function(z) dnorm(z, 2, sqrt(theor.var))
curve(my.dnorm, from=-15, to =15, add=T)
```



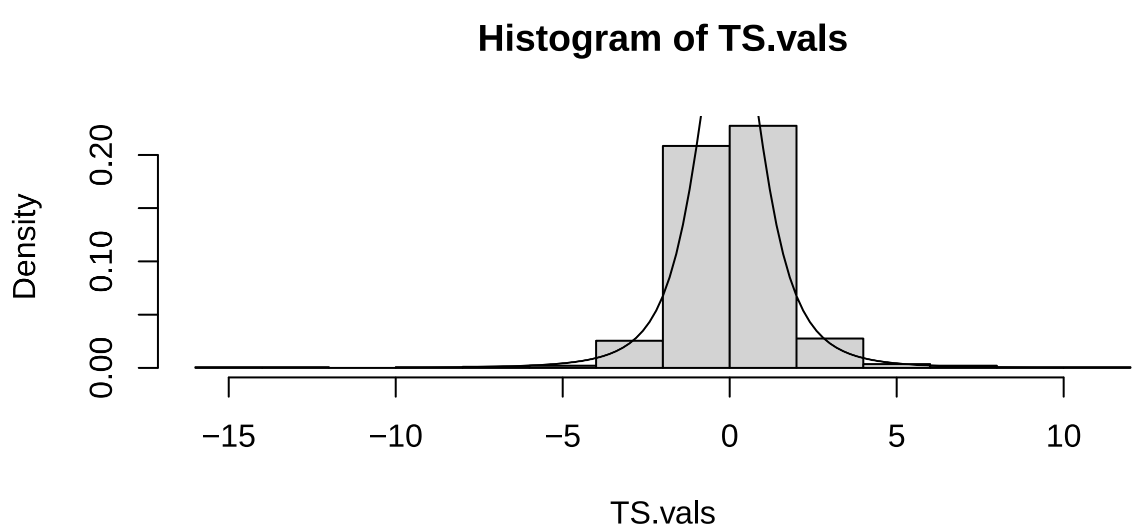
Problem #3

Code below:

```
set.seed(1)
n.rep <- 1000
samp.size <- 5
TS.vals <- numeric(n.rep)

X <- runif(samp.size, -50, 50)

for (i in 1:n.rep){
  Y <- 2 + 3*X + rnorm(samp.size, 0, 10)
  lm.obj <- lm(Y~X)
  beta1.hat <- coef(lm.obj)[2]   # Extracting beta1^hat
  SE.beta1.hat <- coef(summary(lm.obj))[2, "Std. Error"]  # Extracting st error of beta1^hat
  TS.vals[i] <- (beta1.hat - 3)/SE.beta1.hat  # Calculating TS = (beta1.hat - beta1)/SE(beta1.hat),
                                              # where beta1=3, because true relationship
                                              # is Y= 2 + 3*X = beta0 + beta1*X
}

mean(TS.vals)
```

```
## [1] 0.06513587
```

```
hist(TS.vals, freq=F)

my.dt <- function(x) dt(x, samp.size-2)
curve(my.dt, -10, 10, add=T)
```



Mean of the sampling distribution is close to 0 ($\equiv E[t_3]$), with practical (histogram) and theoretical (bell-shaped curve of $t_{5-2} = t_3$, centered at 0) distributions overlaying smoothly.

Problem #4

```r
set.seed(2)

n <- 200
X <- rnorm(n, mean=0, sd=1)

n.rep <- 1000
conf_int_b0 <- matrix(0, nrow=n.rep, ncol=2)
conf_int_b1 <- matrix(0, nrow=n.rep, ncol=2)

for (r in 1:n.rep){
  eps <- rnorm(n, mean=0, sd=4)
  Y <- 2 + 3*X + eps

  lm.obj <- lm(Y~X)

  conf_int_b0[r,] <- confint(lm.obj, level=0.90)[1,]
  conf_int_b1[r,] <- confint(lm.obj, level=0.90)[2,]
}

print(mean(conf_int_b0[,1] < 2 & 2 < conf_int_b0[,2]))
```

```
## [1] 0.887
```

```r
print(mean(conf_int_b1[,1] < 3 & 3 < conf_int_b1[,2]))
```

```
## [1] 0.9
```

We had the true value of β_0 inside the resulting confidence intervals 89.1% of the time, β_1 - 88.5% of the time. By the practical definition of 90% confidence interval, we expect the true parameter to land within the interval 90% of the time, which is roughly what we've got.