MATH 392 Problem Set 8

Sam D. Olson

Warm-up

Using matrix notation and the hat matrix, find $Var(\hat{Y})$.

$Var(\hat{Y}) = Var(X\beta + \epsilon) = Var(X\beta) + Var(\epsilon)$

Note: $\hat{Y} = HY$, where X denotes the hat matrix:

$H = X(X'X)^{-1}X'$

Thus, we have:

$Var(\hat{Y}) = Var(HY)$

Taking advantage of a linear algebra identity, we can say:

$Var(\hat{Y}) = Var(HY) = HVar(Y)H'$

Substituting the definition of the hat matrix gives us:

$Var(\hat{Y}) = HVar(Y)H' = X(X'X)^{-1}X'Var(Y)(X(X'X)^{-1}X')'$

Note: $Var(Y) = \sigma^2 I$, giving us:

$Var(\hat{Y}) = HVar(Y)H' = X(X'X)^{-1}X'(\sigma^2 I)(X(X'X)^{-1}X')'$

As σ^2 is a constant, we can pull it out. Giving us:

$Var(\hat{Y}) = \sigma^2 X(X'X)^{-1}X'(X(X'X)^{-1}X')' = \sigma^2 HH'$

Noting the properties of symmetry and idempotence, we may say:

$Var(\hat{Y}) = \sigma^2 HH' = \sigma^2 H^2 = \sigma^2 H$

Thus we conclude:

$Var(\hat{Y}) = \sigma^2 H$

MLR Simulator

The core of this assignment is the creation of a MLR simulator that you will use to investigate the properties of the method. The model under investigation is the following.

$$Y = X\beta + \epsilon$$

Where Y and ϵ are vectors of length n, X is an $n \times 3$ design matrix (the first column is just ones) and β is a vector of length 3.

You're welcome to select any values that you like for the parameters and any distribution that you like for the x's (you may be want to check out the `rmvnorm()` function in the `mvtnorm` package). The core bit of code should look something like:

```r
library(mvtnorm)
# set params
B0 <- 4
B1 <- 2
B2 <- 1
B <- c(B0, B1, B2)
p <- 3
# Example rmvnorm code
# sigma <- matrix(c(1,0,0,0,2,0,0,0,3), 3, 3)
# mu <- c(0,2,4)
# X <- rmvnorm(100, mean = mu, sigma = sigma)

# complete specification
sigma <- .5
n <- 100
x_0 <- rep(1, n)
x_1 <- rnorm(n, mean = 2, sd = 1)
x_2 <- rnorm(n, mean = 4, sd = .5)
X <- cbind(x_0, x_1, x_2)

# simulate ys (this part inside a for loop)
epsilon <- rnorm(n, mean = 0, sd = sigma)

y <- (X %*% B) + epsilon
```

Part I. Sampling distributions

Use your simulator to create an MC approximation of the true sampling distribution of the estimates of β_1, $E(Y_s)$, and Y_s corresponding to a fixed new observation x_s. How do these empirical distributions compare to their analytical form in terms of center, shape, and spread?

Empirical Notes Regarding the analytical form of the sampling distributions, with an initial specification of normality, we have:

(1): $\hat{\beta} = (X'X)^{-1}X'Y$

$\hat{\beta} \sim N(\beta, \sigma^2(X'X)^{-1})$

$\beta_1 \sim N(\beta_1, \sigma^2 C_{11})$ where $C = (X'X)^{-1}$

(2): $E(Y_s) = E(\beta_0 + \beta_1 x_s + \beta_2 x_s + \epsilon_s) = Y_s$

$E(Y_s) \sim N(Y_s, Var(E(\beta_0 + \beta_1 x_s + \beta_2 x_s + \epsilon_s)))$

Taking advantage of independence, we can evaluate each expected value piecewise and separate elements within the equation, giving us:

$Var(E(\beta_0 + \beta_1 x_s + \beta_2 x_s + \epsilon_s)) = Var(E(\beta_0) + E(\beta_1 x_s) + E(\beta_2 x_s) + E(\epsilon_s)) = Var(\beta_0 + \beta_1 x_s + \beta_2 x_s + 0) = \sigma^2(X'X)^{-1})$

Thus, we're left with:

$E(Y_s) \sim N(Y_s, \sigma^2(X'X)^{-1})$

Alternative Note:

$\hat{y}(x_s) = x_s'\hat{\beta} = \hat{\beta}_0 + \hat{\beta}_1 x_{s1} + \hat{\beta}_2 x_{s2}$

(3): $Y_s = \beta_0 + \beta_1 x_s + \beta_2 x_s + \epsilon_s$

$Y_s \sim N(\beta_0 + \beta_1 x_s + \beta_2 x_s + \epsilon_s, Var(\beta_0 + \beta_1 x_s + \beta_2 x_s + \epsilon_s))$

Taking advantage of (1), and independence, we simplify this to:

$Y_s \sim N(\beta_0 + \beta_1 x_s + \beta_2 x_s + \epsilon_s, \sigma^2(X'X)^{-1} + \sigma^2 I)$, where I denotes the identity matrix.

```
n <- 100
p <- 3
sigma <- 0.5
x0 = rep(1, n)
x1 = sample(seq(1, 10, length = n))
x2 = sample(seq(1, 10, length = n))
X = cbind(x0, x1, x2)
C = solve(t(X) %*% X)
beta_0 <- 4
beta_1 <- 2
beta_2 <- 1
eps      = rnorm(n, mean = 0, sd = sigma)
y        = beta_0 + beta_1 * x1 + beta_2 * x2 + eps
sim_data = data.frame(x1, x2, y)

## Beta
(beta_hat = C %*% t(X) %*% y)
coef(lm(y ~ x1 + x2, data = sim_data))
c(beta_0, beta_1, beta_2)

## Simulation Beta_1
num_sims = 1000
beta_hat_1 = rep(0, num_sims)
for(i in 1:num_sims) {
  eps           = rnorm(n, mean = 0 , sd = sigma)
  sim_data$y    = beta_0 * x0 + beta_1 * x1 + beta_2 * x2 + eps
  fit           = lm(y ~ x1 + x2, data = sim_data)
  beta_hat_1[i] = coef(fit)[2]
}

## Density Plot
hist(beta_hat_1, prob = TRUE, breaks = 20,
     xlab = expression(hat(beta)[1]), main = "", border = "dodgerblue")
curve(dnorm(x, mean = beta_1, sd = sqrt(sigma ^ 2 * C[1 + 1, 1 + 1])),
      col = "darkorange", add = TRUE, lwd = 3)
```

```
## Simulation E(Y_s)
n <- 100
p <- 3
sigma <- 0.5
x0 = rep(1, n)
x1 = sample(seq(1, 10, length = n))
x2 = sample(seq(1, 10, length = n))
X = cbind(x0, x1, x2)
C = solve(t(X) %*% X)
beta_0 <- 4
beta_1 <- 2
```

```r
beta_2 <- 1
eps      = rnorm(n, mean = 0, sd = sigma)
y        = beta_0 + beta_1 * x1 + beta_2 * x2 + eps
sim_data = data.frame(x1, x2, y)

## E(Y_s)
(beta_hat = C %*% t(X) %*% y)
```

```
##         [,1]
## x0 4.0919966
## x1 1.9894312
## x2 0.9918636
```

```r
y_hat = X %*% beta_hat
(s_e = sqrt(sum((y - y_hat) ^ 2) / (n - p)))
```

```
## [1] 0.4837461
```

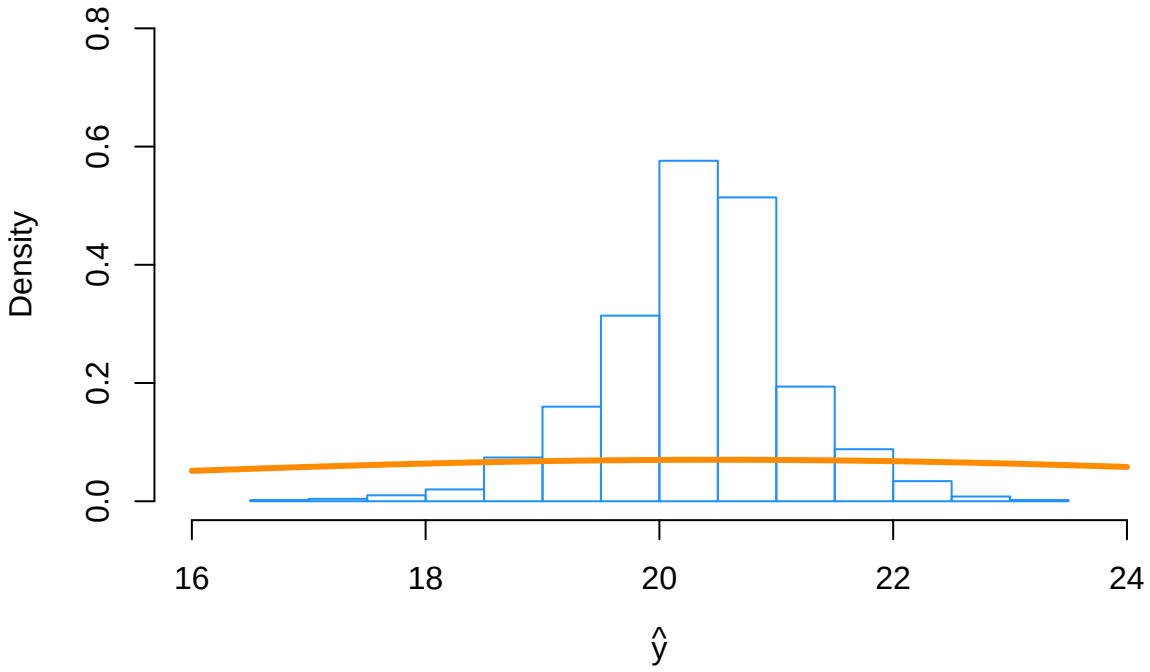
```r
summary(lm(y ~ x1 + x2, data = sim_data))$sigma
```

```
## [1] 0.4837461
```

```r
y_hat_mean <- mean(y_hat)
y_hat_var <- var(y_hat)

## Simulation
num_sims = 1000
y_hat = rep(0, num_sims)
for(i in 1:num_sims) {
  x0 = rep(1, n)
  x1 = sample(seq(1, 10, length = n))
  x2 = sample(seq(1, 10, length = n))
  X = cbind(x0, x1, x2)
  beta_0 <- 4
  beta_1 <- 2
  beta_2 <- 1
  C = solve(t(X) %*% X)
  (beta_hat = C %*% t(X) %*% y)
  y_hat[i] = (X %*% beta_hat)[[1]]
}

# sqrt(sum((y - y_hat) ^ 2) / (n - p))
## Density Plot
hist(y_hat, prob = TRUE, breaks = 20,
     xlab = expression(hat(y)), main = "", border = "dodgerblue", ylim=c(0,0.8), xlim=c(16, 24))
curve(dnorm(x, mean = y_hat_mean, sd = sqrt(y_hat_var)),
      col = "darkorange", add = TRUE, lwd = 3)
```




```r
## Simulation Y_s
n <- 100
p <- 3
sigma <- 0.5
x0 = rep(1, n)
x1 = sample(seq(1, 10, length = n))
x2 = sample(seq(1, 10, length = n))
X = cbind(x0, x1, x2)
C = solve(t(X) %*% X)
beta_0 <- 4
beta_1 <- 2
beta_2 <- 1
eps      = rnorm(n, mean = 0, sd = sigma)
y        = beta_0 + beta_1 * x1 + beta_2 * x2 + eps
y_est = beta_0 + beta_1 * x1 + beta_2 * x2
mean_y <- mean(y_est)
sd_y <- sd(y_est)
sim_data = data.frame(x1, x2, y)

## E(Y_s)
(beta_hat = C %*% t(X) %*% y)
```

```
##         [,1]
## x0 4.2278028
## x1 1.9940864
## x2 0.9714148
```

```r
y_hat = X %*% beta_hat
(s_e = sqrt(sum((y - y_hat) ^ 2) / (n - p)))
```

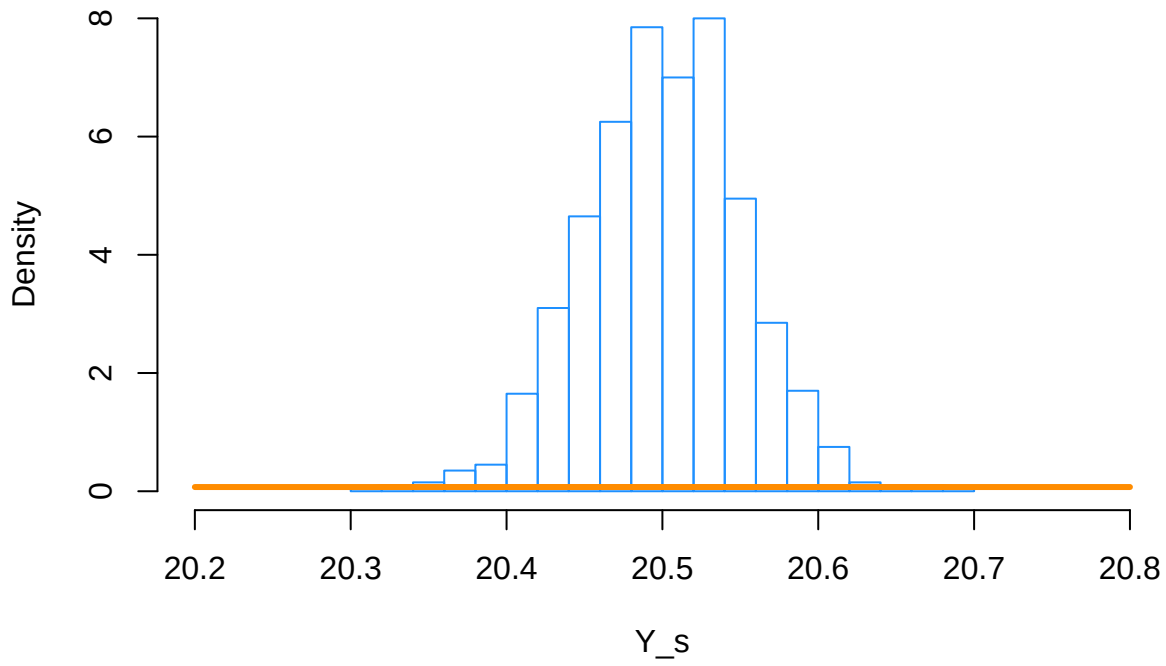
```
## [1] 0.4813752
```

```r
summary(lm(y ~ x1 + x2, data = sim_data))$sigma
```

```
## [1] 0.4813752
```

```r
## Simulation
num_sims = 1000
y_s = rep(0, num_sims)
for(i in 1:num_sims) {
  eps           = rnorm(n, mean = 0 , sd = sigma)
  sim_data$y    = beta_0 * x0 + beta_1 * x1 + beta_2 * x2 + eps
  fit           = lm(y ~ x1 + x2, data = sim_data)
  y_s[i] = mean(sim_data$y)
}

## Density Plot
hist(y_s, prob = TRUE, breaks = 20,
     xlab = expression(Y_s), main = "", border = "dodgerblue", xlim=c(20.2,20.8), freq=FALSE)
curve(dnorm(x, mean = mean_y, sd = sd_y),
      col = "darkorange", add = TRUE, lwd = 3, yaxt="n")
```



```r
beta_hat <- (solve(t(X) %*% X) %*% t(X) %*% y)

y_hat <- X %*% solve(t(X) %*% X) %*% t(X) %*% y

epsilon_hat <- y - y_hat
# Verifying the above epsilon_hat value
sqrt(t(e) %*% e / (n - p))


predict(y, newdata = y1, interval = "confidence", level = 0.95)

## E(Y_s)
x0 = c(1, 2, 4)
x0 %*% beta_hat
# set sample size
n <- 100

# compute the variance of hat_beta_1
var_b1 <- var( ( X - mean(X) ) * epsilon ) / (n * var(X)^2)
```

Part II. A different model

Consider two variations on the model:

1. Change the marginal distribution of the ϵ (though it still should be centered at 0).

```r
library(mvtnorm)
# set params
B0 <- 4
B1 <- 2
B2 <- 1
B <- c(B0, B1, B2)

# Example rmvnorm code
# sigma <- matrix(c(1,0,0,0,2,0,0,0,3), 3, 3)
# mu <- c(0,2,4)
# X <- rmvnorm(100, mean = mu, sigma = sigma)

# complete specification
sigma <- .5
n <- 100
x_0 <- rep(1, n)
x_1 <- rnorm(n, mean = 2, sd = 1)
x_2 <- rnorm(n, mean = 4, sd = .5)
X <- cbind(x_0, x_1, x_2)

# simulate ys (this part inside a for loop)
epsilon <- rnorm(n, mean = 0, sd = sigma*2)

y <- (X %*% B) + epsilon
```

2. Introduce non-zero covariance into the joint distribution of the X (`rvmnorm()` is helpful here).

```r
library(mvtnorm)
# set params
B0 <- 4
B1 <- 2
B2 <- 1
B <- c(B0, B1, B2)

# Example rmvnorm code
n <- 100
sigma <- matrix(c(1,-1,-2,-1,2,-1,-2,-1,3), 3, 3)
mu <- c(0,2,4)
X <- rmvnorm(100, mean = mu, sigma = sigma)
x_0 <- rep(1, n)
x_1 <- X[,2]
x_2 <- X[,3]

X <- cbind(x_0, x_1, x_2)

# simulate ys (this part inside a for loop)
epsilon <- rnorm(n, mean = 0, sd = sigma)

y <- (X %*% B) + epsilon
```

3. Introduce non-zero covariance into the joint distribution of the ϵ.

```r
library(mvtnorm)
# set params
B0 <- 4
B1 <- 2
B2 <- 1
B <- c(B0, B1, B2)

# Example rmvnorm code
n <- 100
sigma <- matrix(c(1,-1,-2,-1,2,-1,-2,-1,3), 3, 3)
mu <- c(0,2,4)
X <- rmvnorm(100, mean = mu, sigma = sigma)
x_0 <- rep(1, n)
epsilon <- X[,1]
x_1 <- X[,2]
x_2 <- X[,3]
X <- cbind(x_0, x_1, x_2)
# simulate ys (this part inside a for loop)
y <- (X %*% B) + epsilon
```

Does the inference change for the statistics investigated in the previous exercise? For this part, you should be able to recycle much of your code from the previous part.

Part III. Variance/Covariance

Generate a scatterplot matrix involving all pairwise relationships between the three simulated regression coefficients in the original model. To be clear, this involves generating a good number fitted betas from

your simulator and plotting them. Based on a visual assessment of these plots, please characterize the joint distribution of these statistics in terms of center, shape, spread, and correlation. Compare the empirical covariance matrix to the analytical form that we derived in class.