Assignment 2

1. Set the sample size `n <- 1000` and the number of variables `k <- 3`. In r, you can simulate a normal random matrix using the command, `x <- matrix(rnorm(n*k),n)`. Verify that the resulting matrix has (approximately) a mean of zero and a variance matrix equal to the identity using the commands `colMeans(x)` and `var(x)`. Next, use the Cholesky decomposition to simulate a $N(\mu,\Omega)$ random variable where $\mu = \begin{bmatrix} 1 \\ -1 \\ 0.5 \end{bmatrix}$ and $\Omega = \begin{bmatrix} 1 & 0.5 & 0.1 \\ 0.5 & 2 & 0.3 \\ 0.1 & 0.3 & 0.5 \end{bmatrix}$ (hint: recall that if $x \sim N(0, I)$, then $\mu + \Omega^{1/2} x \sim N(\mu,\Omega)$ and that the Cholesky decomposition can be computed using `chol(omega)`). Call the new matrix `x2` (the dimensions can be made to work out correctly is you use `x2 <- as.matrix(rep(1,n)) %*% t(as.matrix(mu)) + x %*% chol(omega)`. Remember that in order to perform matrix multiplication in r, you should use `%*%` as `*` will perform component by component multiplication. Check that this worked using the commands `colMeans(x2)` and `var(x2)`. Verify that Ω is indeed positive definite by computing the eigenvalues of omega using the command `eigen(omega)`.

Now, suppose that you were using this to simulate data from a linear regression model. Add a constant term to the matrix x3 using the code,

```
k3 <- k + 1
x3 <- matrix(rep(0,n*k3),n)
x3[1:n,1] <- 1
x3[1:n,2:k3] <- x2[1:n,1:k]
```

Generate the errors vector `eps` from the $N(0, 2.8^2)$ distribution (setting `sigma0 <- 2.8`).

Assume that $\beta_0 = (0.5, -2, 1, 0.7)$. Generate data from a linear regression model with `x3` and the data matrix and `eps` as the error term. Run a linear regression on this data using,

```
lm1 <- lm(y~x3)
summary(lm1)
```

What value of $\hat{\beta}$ do you find and how does this compare to β_0? Is $\hat{\beta}$ very close to β_0, and if so, why? Verify that you get the same answer using `betahat <- solve(t(x3) %*% x3) %*% t(x3) %*% y`.

Now, we will imbed this code in a loop to perform a Monte Carlo simulation. Set the number of replications as `S <- 1000`. Imbed the code you previous wrote in a loop,

```
betahatsim <- matrix(rep(0,S*k3),S)
sigmahatsqrsim <- rep(0,S)
tstat3sim <- rep(0,S)
zstat3sim <- rep(0,S)
for(s in 1:S){
      # code here
}
```

(of course, you may put code like `n <- 1000` before the loop). The top lines will allow you to collect the simulations. You can simulate the main components of the model using,

```
      y <- x3 %*% beta0 + sigma0 * eps
      betahat <- solve(t(x3) %*% x3) %*% t(x3) %*% y
      betahatsim[s,] <- betahat

      epshat <- y - x3 %*% betahat
      sigmahatsqr <- sum(epshat^2) / (n - k3)
      sigmahatsqrsim[s] <- sigmahatsqr

      varbetahat <- sigmahatsqr * solve(t(x3) %*% x3)
      beta3se <- varbetahat[3,3]^.5
      tstat3sim[s] <- (betahat[3] - beta0[3]) / beta3se

      robustmeat <- matrix(rep(0,16),4)
```

```
for(i in 1:n) {
      robustmeat <- robustmeat + epshat[i]^2*x3[i,]%*%t(x3[i,])
}
robustmeat <- robustmeat / n
robustbread <- solve(t(x3) %*% x3 / n)
varbetahatrobust <- robustbread %*% robustmeat %*% robustbread / n
beta3serobust <- varbetahatrobust[3,3]^.5
zstat3sim[s] <- (betahat[3] - beta0[3]) / beta3serobust
```

inside the loop. Notice that $\hat{\beta}$ is computed as $(X'X)^{-1}X'y$, $\hat{\sigma}_{\varepsilon}^2$ is computed as $\frac{1}{N-K}\hat{\varepsilon}'\hat{\varepsilon}$ where $\hat{\varepsilon} = y - X\hat{\beta}$, the variance of $\hat{\beta}$ is estimated to be $\hat{\sigma}_{\varepsilon}^2(X'X)^{-1}$, the sandwich estimator of the variance is computed as $\left(\frac{1}{N}\sum_{n=1}^{N} x_n x_n'\right)^{-1}\left(\frac{1}{N}\sum_{n=1}^{N} x_n x_n'\hat{\varepsilon}_n^2\right)\left(\frac{1}{N}\sum_{n=1}^{N} x_n x_n'\right)^{-1}$, a t-statistic for the test that $\beta_{03} = 1$ is computed based on the classical standard error, and a z-statistic for the test that $\beta_{03} = 1$ is computed based on the robust standard error. You can then evaluate the performance using,

```
# OLS Unbiased / Consistent (LLN)
colMeans(betahatsim)
beta0

mean(sigmahatsqrsim)
sigma0^2

# tstat
mean(abs(tstat3sim) > qt(0.975,n-k3))

# zstat w/ robust standard errors (CLT)
mean(abs(zstat3sim) > qnorm(0.975))
```

Simulate the model with `n <- 10`, `n <- 100`, and `n <- 1000`. Repeat the same exercise with the following heteroskedastic error terms,

```
eps <- rep(0,n)
for(i in 1:n) {
      eps[i] <- rnorm(1) * x3[i,3]^2
}
```

For both cases, and for all three sample sizes, report whether OLS appears to be unbiased and consistent with reference to the simulations. Report whether the t-test and the z-test appear to have the correct size. Include your r code and output in your homework assignment.

Which assumptions from (A1)-(A7) and (B1)-(B8) does the data you generate satisfy? How do the results of the simulations compare to properties we derived for the OLS estimator?

2. Present a heuristic proof of each of the following. Indicate the set of assumptions you are using (and assume that the X's are stochastic).

(a) OLS is unbiased.

(b) OLS is consistent.

(c) OLS is asymptotically normal with mean β_0 and variance covariance matrix $V = E[x_n x_n']^{-1} Var(x_n \varepsilon_n) E[x_n x_n']^{-1}$.