ECON 7201
Applied Econometrics
Feremusu R. Koroma

Assignment 2

Due Date

Sunday October 5, 2025 at 11:59 PM

Directions

Answer all questions. Submit both a PDF and Quarto file to the nexus assignment portal.

Git and GitHub

1. (a) Create a new R project in your **econ_3201** directory called **assignment_2**.
   (b) Download the assignment PDF and Quarto file the **assignment_2** folder.
   (c) Commit and push the changes to your **econ_3201** repository on GitHub.com.

LaTeX

Matrices are created in LaTeX using the `\begin{bmatrix}...\end{bmatrix}` command. To separate entries along the same row, use `&`. To end a line, use `\\`. To make vertical elipses (:), use `\vdots`. Practice writing the following matrices and vectors in LaTeX. Write the following matrices in LaTeX.

2. (a) $$X'X = \begin{bmatrix} n & \sum_{i=1}^{n} x_{1i} & \sum_{i=1}^{n} x_{2i} \\ \sum_{i=1}^{n} x_{1i} & \sum_{i=1}^{n} x_{1i}^2 & \sum_{i=1}^{n} x_{1i}x_{2i} \\ \sum_{i=1}^{n} x_{2i} & \sum_{i=1}^{n} x_{1i}x_{2i} & \sum_{i=1}^{n} x_{2i}^2 \end{bmatrix}$$

   (b) $$\Omega = \begin{bmatrix} \sigma_1^2 & 0 & 0 & 0 \\ 0 & \sigma_2^2 & 0 & 0 \\ 0 & 0 & \sigma_3^2 & 0 \\ 0 & 0 & 0 & \sigma_4^2 \end{bmatrix}$$

R

3. In this question we compare standard errors based on (incorrect) asymptotic assumptions with those based on alternate (appropriate) estimator (White). Consider one sample drawn from the following data generating process (DGP) which we will simulate in R:

```
set.seed(123)
n <- 25
x <- rnorm(n,mean=0.0,sd=1.0)
beta0 <- 1
beta1 <- 0
## x is irrelevant in this model, the data generating process is as follows:
dgp <- beta0 + beta1*x
## The residual is heteroskedastic by construction
e <- x^2*rnorm(n,mean=0.0,sd=1.0)
y <- dgp + e
```

(a) Compute the OLS estimator of β_2 and its standard error using the `lm()` command in R for the model $y_i = \beta_1 + \beta_2 x_i + \epsilon_i$ based on the DGP given above.

```
  # (a) Estimating the OLS model
model1 <- lm(y ~ x)
summary(model1)
```

```
Call:
lm(formula = y ~ x)

Residuals:
    Min      1Q  Median      3Q     Max
-3.8764 -0.1604  0.0928  0.5748  2.1268

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.8890     0.2379   3.737  0.00108 **
x             0.4610     0.2563   1.799  0.08520 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.189 on 23 degrees of freedom
Multiple R-squared:  0.1233,    Adjusted R-squared:  0.0852
```

```
F-statistic: 3.235 on 1 and 23 DF,  p-value: 0.0852
```

```
se_beta2_lm <- summary(model1)$coefficients["x", "Std. Error"]
```

(b) Next, compute the standard error of $\hat{\beta}_2$ by computing $\hat{\sigma}^2(X'X)^{-1}$ in R using matrix commands, and verify that the two standard error estimates are identical.

```
  # (b) standard error for beta2_hat
X <- cbind(1, x)

e <- model1$residuals
sigmasq_hat <- as.numeric(t(e) %*% e) / (n - 2)


var_cov <- sigmasq_hat * solve(t(X) %*% X) # variance-co variant matrix
se_beta2_matrix <- sqrt(var_cov[2, 2]) # std error for beta2

cat("Estimated sigma^2:", sigmasq_hat, "\n\n")
```

```
Estimated sigma^2: 1.413163
```

```
print(var_cov)
```

```
                        x
  0.056599489 0.002189604
x 0.002189604 0.065694112
```

```
cat("\nStandard Error for  2 (slope):", se_beta2_matrix, "\n")
```

```
Standard Error for  2 (slope): 0.2563086
```

```
se_beta1 <- sqrt(var_cov[1, 1])
cat("Standard Error for  1 (intercept):", se_beta1, "\n")
```

```
Standard Error for  1 (intercept): 0.2379065
```

```
## verifying similarity
round(se_beta2_lm,4) == round(se_beta2_matrix,4) # TRUE, implying that they are identical
```

```
[1] TRUE
```

(c) Compute White's heteroskedasticity consistent covariance matrix estimator using matrices in R and report the White estimator of the standard error of $\hat{\beta}_2$. Compare this with that from 3 (a) above.

```
# (c) computing white covariance matrix

# ensure that the residuals are numeric
e <- as.numeric(model1$residuals)

# construct a diagonal matrix of the squared residuals
Omega <- diag(e^2)

# computing the matrix formula
S <- t(X) %*% Omega %*% X

# calculate the White covariance matrix
white_var_cov <- solve(t(X) %*% X) %*% S %*% solve(t(X) %*% X)

# extract the standard error
white_se_beta1 <- sqrt(white_var_cov[1, 1])
white_se_beta2 <- sqrt(white_var_cov[2, 2])

# print results
cat("\n--- (c) White (HC0) heteroskedasticity-consistent results ---\n")
```

```
--- (c) White (HC0) heteroskedasticity-consistent results ---
```

```
cat("White Variance-Covariance Matrix:\n")
```

```
White Variance-Covariance Matrix:
```

```
print(white_var_cov)
```

```
                      x
   0.04985383 -0.02884278
x -0.02884278  0.20513402
```

```
cat("\nWhite SE for  1 (intercept):", white_se_beta1, "\n")
```

```
White SE for  1 (intercept): 0.2232797
```

```
cat("White SE for  2 (slope):", white_se_beta2, "\n")
```

```
White SE for  2 (slope): 0.4529172
```

```
## comparing the white se with ordinary se
sign(se_beta2_lm - white_se_beta2)  # return -1 implying that se_beta1_lm < white_se)
```

```
[1] -1
```

4. Let $\hat{\theta}$ be an estimator for the population parameter θ. $\hat{\theta}$ is said to be unbiased if $E(\hat{\theta}) = \theta$. That is, if the mean of the sampling distribution of $\hat{\theta}$ is equal to the true population value.

   Consider the model

   $$y_i = \beta_0 + \beta_1 x_{1,i} - \beta_2 x_{2,i} + \epsilon_i.$$

   Lets provide empirical evidence that the ordinary least squares estimators $\hat{\beta}_0$, $\hat{\beta}_1$, and $\hat{\beta}_2$ are unbiased estimators of β_0, β_1, β_2, respectively, using R.

   (a) Set the seed to 1, i.e., `set.seed(1)`.

   ::: {.cell}

   ```
    # (a) setting seed to 1
   set.seed(1)
   ```

   :::

   (b) Set the number of observations $n = 100$

   ::: {.cell}

```
# (b) sample size
n <- 100
```

:::

(c) Generate the following model

$$y_i = 2 + 3.5x_{1,i} - 9.2x_{2,i} + \epsilon_i,$$

where $x_1 \sim N(3, 6)$, $x_2 \sim N(2, 4)$, and $\epsilon \sim N(0, 100)$. To create a normally distributed variable, use the `rnorm(n, mean, sd)` command in R.

::: {.cell}

```
# (c) generating the variable
x1 <- rnorm(n, mean = 3, sd = sqrt(6))
x2 <- rnorm(n, mean = 2, sd = sqrt(4))
epsilon <- rnorm(n, mean = 0, sd = sqrt(100))
y <- 2 + 3.5*x1 - 9.2*x2 + epsilon
```

:::

(d) Estimate the model coefficients using the `lm()` command. (Search `?lm()` in the console for more info).

::: {.cell}

```
 # (d) model estimation
model2 <- lm(y ~ x1 + x2)
summary(model2)
```

::: {.cell-output .cell-output-stdout}

```

Call:
lm(formula = y ~ x1 + x2)

Residuals:
     Min       1Q   Median       3Q      Max 
-29.4359  -4.3645   0.0202   6.3692  26.3941 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   2.5297     2.1508   1.176    0.242    
x1            3.5862     0.4766   7.524 2.72e-11 ***
x2           -9.4673     0.5474 -17.295  < 2e-16 ***
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 10.43 on 97 degrees of freedom
Multiple R-squared:  0.7859,    Adjusted R-squared:  0.7815
F-statistic:   178 on 2 and 97 DF,  p-value: < 2.2e-16
```

::: :::

(e) Using a `for()` loop, replicate the model above $M = 1000$ times and save the coefficient estimates from each iteration.

::: {.cell}

```
  # (e) replication
set.seed(1)
M <- 1000
n <- 100

beta0 <- 2
beta1 <- 3.5
beta2 <- -9.2

coef_mat <- matrix(NA, nrow = M, ncol = 3)
colnames(coef_mat) <- c("(Intercept)", "x1", "x2")

for (i in 1:M) {
  # generate new random data for each replication
  x1 <- rnorm(n, mean = 3, sd = 6)
  x2 <- rnorm(n, mean = 2, sd = 4)
  eps <- rnorm(n, mean = 0, sd = 10)
  y <- beta0 + beta1 * x1 + beta2 * x2 + eps

  model_rep <- lm(y ~ x1 + x2)
  coef_mat[i, ] <- coef(model_rep)
}

# display results
cat("\n------\n")
```

::: {.cell-output .cell-output-stdout}

```
------
```

:::

```
print(colMeans(coef_mat))
```

::: {.cell-output .cell-output-stdout}

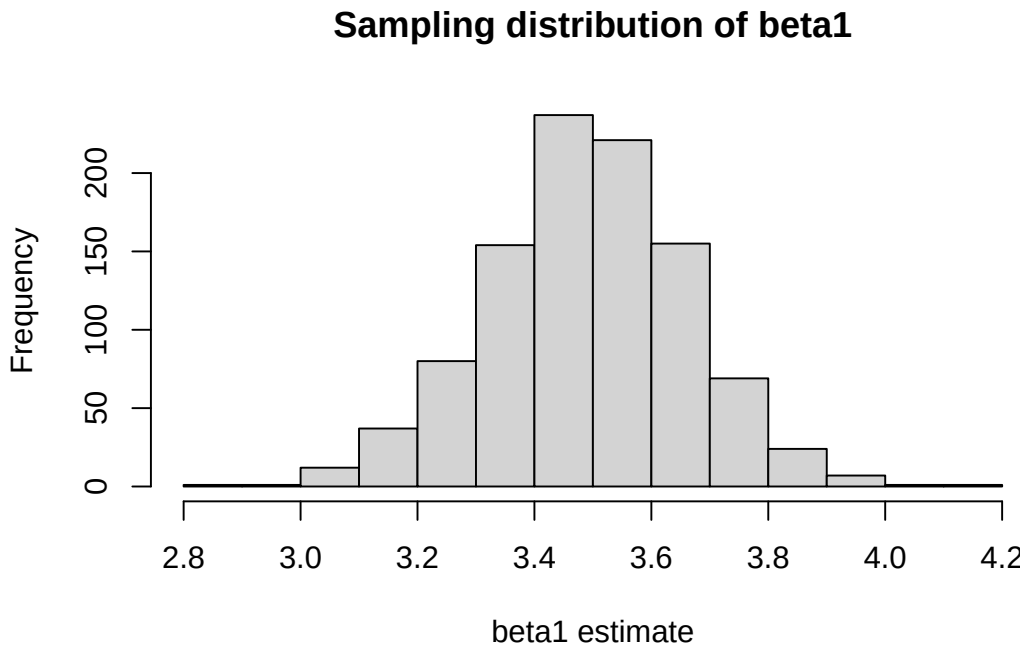
```
(Intercept)          x1          x2
   1.983581    3.491256   -9.193145
```

::: :::

(f) Using `hist()`, plot the sampling distributions of the coefficient estimates, β_1 and β_2.

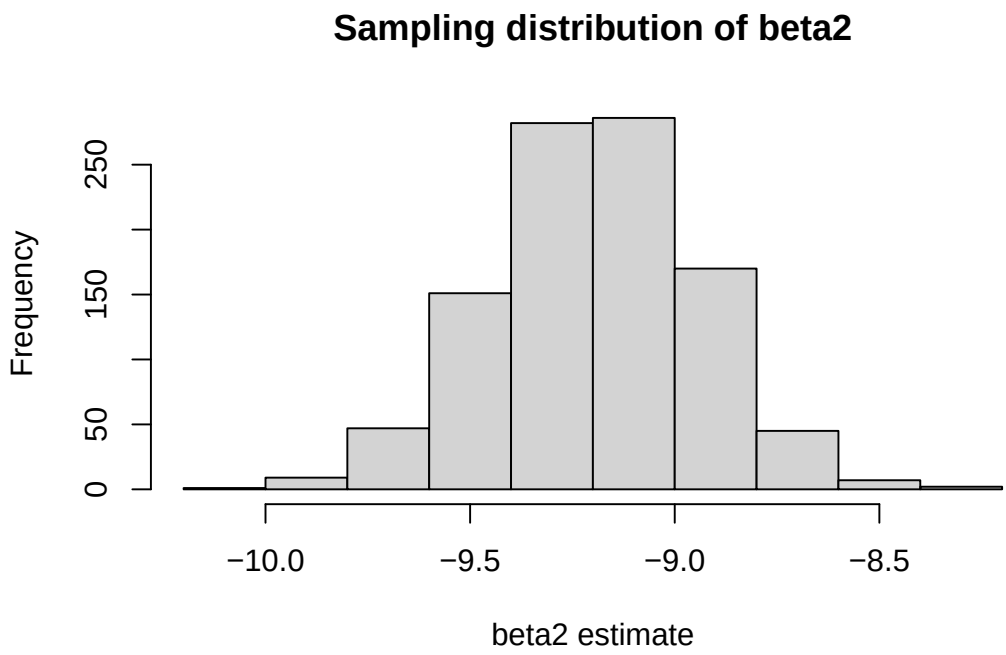
::: {.cell}

```
   # (f)
hist(coef_mat[, "x1"],
 main = "Sampling distribution of beta1",
 xlab = "beta1 estimate")
```



::: {.cell-output-display}
:::

```
hist(coef_mat[, "x2"],
 main = "Sampling distribution of beta2",
 xlab = "beta2 estimate")
```

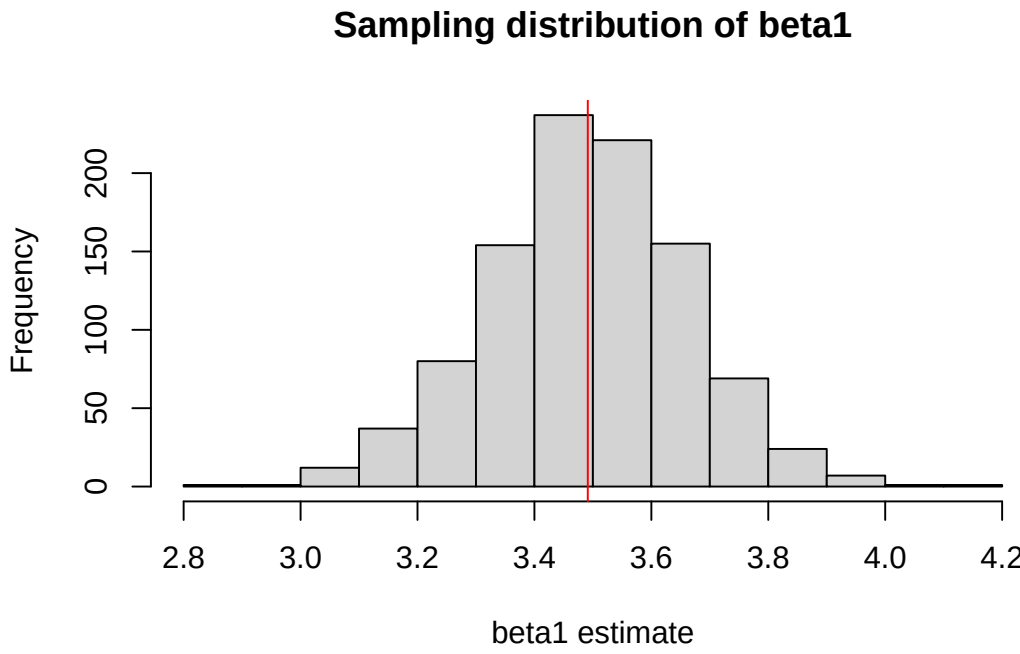


::: {.cell-output-display}
::: :::

(g) Add a vertical line to each figure at the mean of the respective variable. Search `?abline()` in your console.

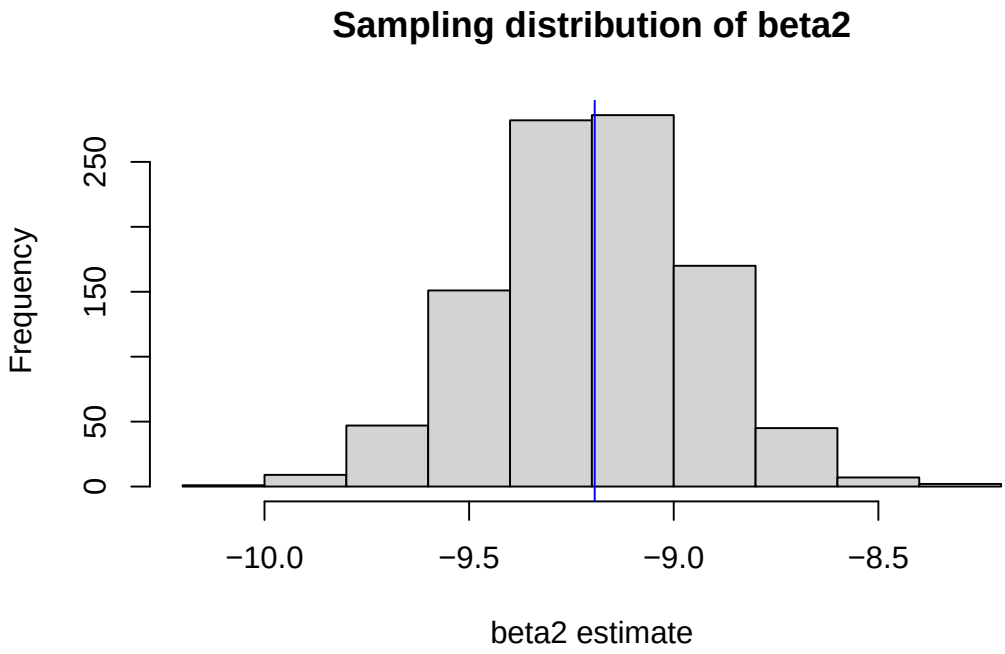
::: {.cell}

```
  #  (g)
hist(coef_mat[, "x1"],
 main = "Sampling distribution of beta1",
 xlab = "beta1 estimate")
abline(v = colMeans(coef_mat)[2], col = "red")
```



::: {.cell-output-display}
:::

```
hist(coef_mat[, "x2"],
 main = "Sampling distribution of beta2",
 xlab = "beta2 estimate")
abline(v = colMeans(coef_mat)[3], col = "blue")
```



::: {.cell-output-display}
::: :::