# Pset 4 - Empirical

Romain Fernex

2025-03-16

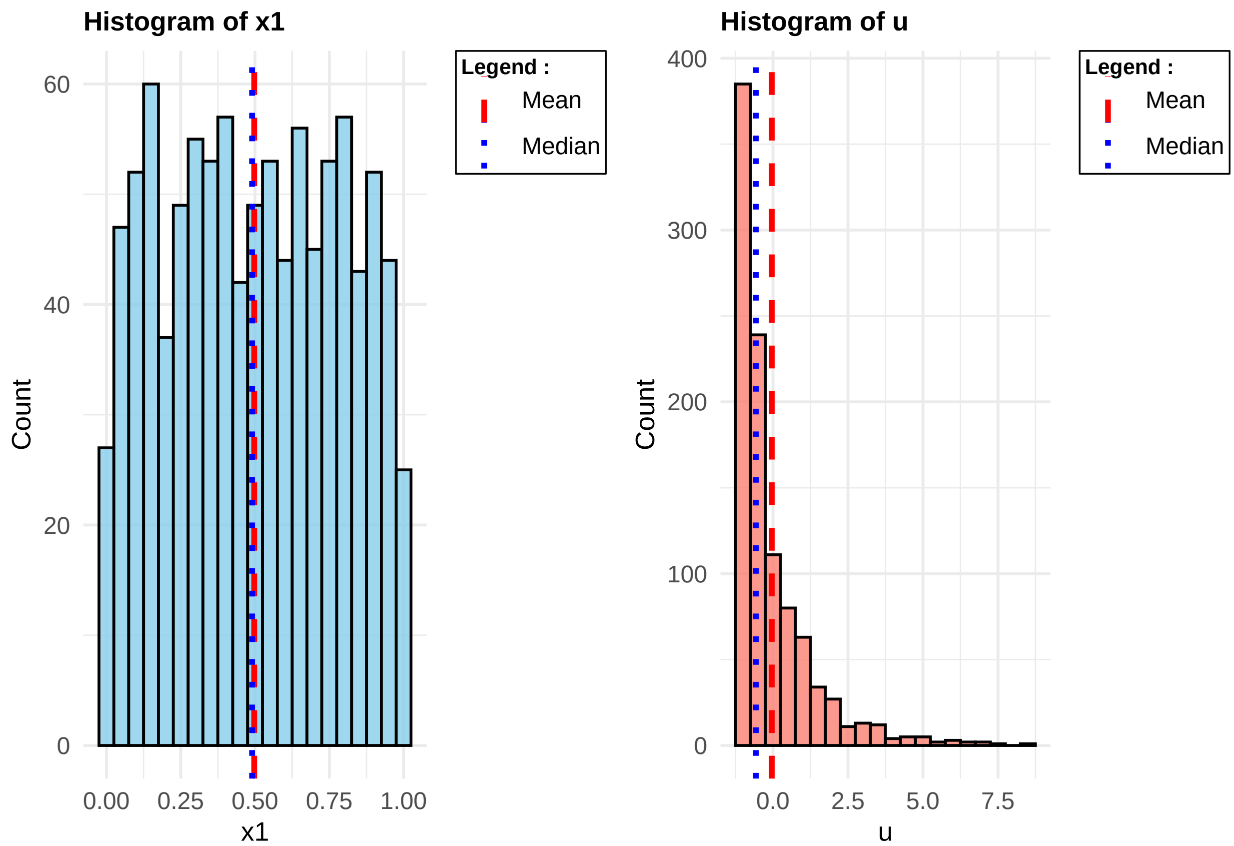
## Contents



# 1 Question 1

```
set.seed(123)
x1  <- runif(1000, min = 0, max = 1)
u <- rchisq(1000, df = 1) - 1
data <- data.frame(x1 = x1, u = u)

par(mfrow = c(1, 2))
```

```r
# histogram for x1
p1 <- ggplot(data, aes(x = x1)) +
  geom_histogram(binwidth = 0.05, fill = 'skyblue', color = 'black', alpha = 0.8) +
  geom_vline(aes(xintercept = mean(x1), color = 'Mean'), linetype = 'dashed', linewidth
   ↪ = 1) +
  geom_vline(aes(xintercept = median(x1), color = 'Median'), linetype = 'dotted',
   ↪ linewidth = 1) +
  scale_color_manual(name = 'Legend :', values = c('Mean' = 'red', 'Median' = 'blue')) +
  labs(title = 'Histogram of x1', x = 'x1', y = 'Count') +
  theme_minimal() +
  theme(
    plot.title = element_text(face = 'bold', hjust = 0, size = 10),
    axis.title = element_text(size = 10),
    legend.position.inside = c(0.95, 0.95),
    legend.justification = c('right', 'top'),
    legend.background = element_rect(color = 'black', fill = 'white', linewidth = 0.3),
    legend.margin = margin(2, 2, 2, 2),
    legend.box.margin = margin(0, 0, 0, 0),
    legend.title = element_text(size = 8, face = 'bold', margin = margin(b = 0))
  )

# histogram for u
p2 <- ggplot(data, aes(x = u)) +
  geom_histogram(binwidth = 0.5, fill = 'salmon', color = 'black', alpha = 0.8) +
  geom_vline(aes(xintercept = mean(u), color = 'Mean'), linetype = 'dashed', linewidth =
   ↪ 1) +
  geom_vline(aes(xintercept = median(u), color = 'Median'), linetype = 'dotted',
   ↪ linewidth = 1) +
  scale_color_manual(name = 'Legend :', values = c('Mean' = 'red', 'Median' = 'blue')) +
  labs(title = 'Histogram of u', x = 'u', y = 'Count') +
  theme_minimal() +
  theme(
    plot.title = element_text(face = 'bold', hjust = 0, size = 10),
    axis.title = element_text(size = 10),
    legend.position.inside = c(0.95, 0.95),
    legend.justification = c('right', 'top'),
    legend.background = element_rect(color = 'black', fill = 'white', linewidth = 0.3),
    legend.margin = margin(2, 2, 2, 2),
    legend.box.margin = margin(0, 0, 0, 0),
    legend.title = element_text(size = 8, face = 'bold', margin = margin(b = 0))
  )

plot_grid(p1, p2, ncol = 2)
```




# 2 Question 2

## 2.1 2.a)

```
chi_mean_std <- function(N) {
  u <- rchisq(N, df = 1) - 1
  mean_u <- mean(u)
  sd_u <- sd(u)
  return(c(mean_u, sd_u))
}
```

## 2.2 2.b)

```
results_df <- data.frame(sample_mean=numeric(),sample_sd=numeric())  |>
   setNames(c("sample_mean", "sample_sd"))
for (i in 1:10000) {
  res <- chi_mean_std(5)
  results_df <- rbind(results_df, data.frame(sample_mean = res[1], sample_sd = res[2]) )
}
 knitr::kable(head(results_df), format = "latex",
             col.names = c("Sample Mean", "Sample SD"),
             caption = "First 6 Rows of Chi-Square Simulation Results (N=5)",
             digits = 4,
             booktabs = TRUE) %>%
   kableExtra::kable_styling(latex_options = c("striped", "hold_position"),
```

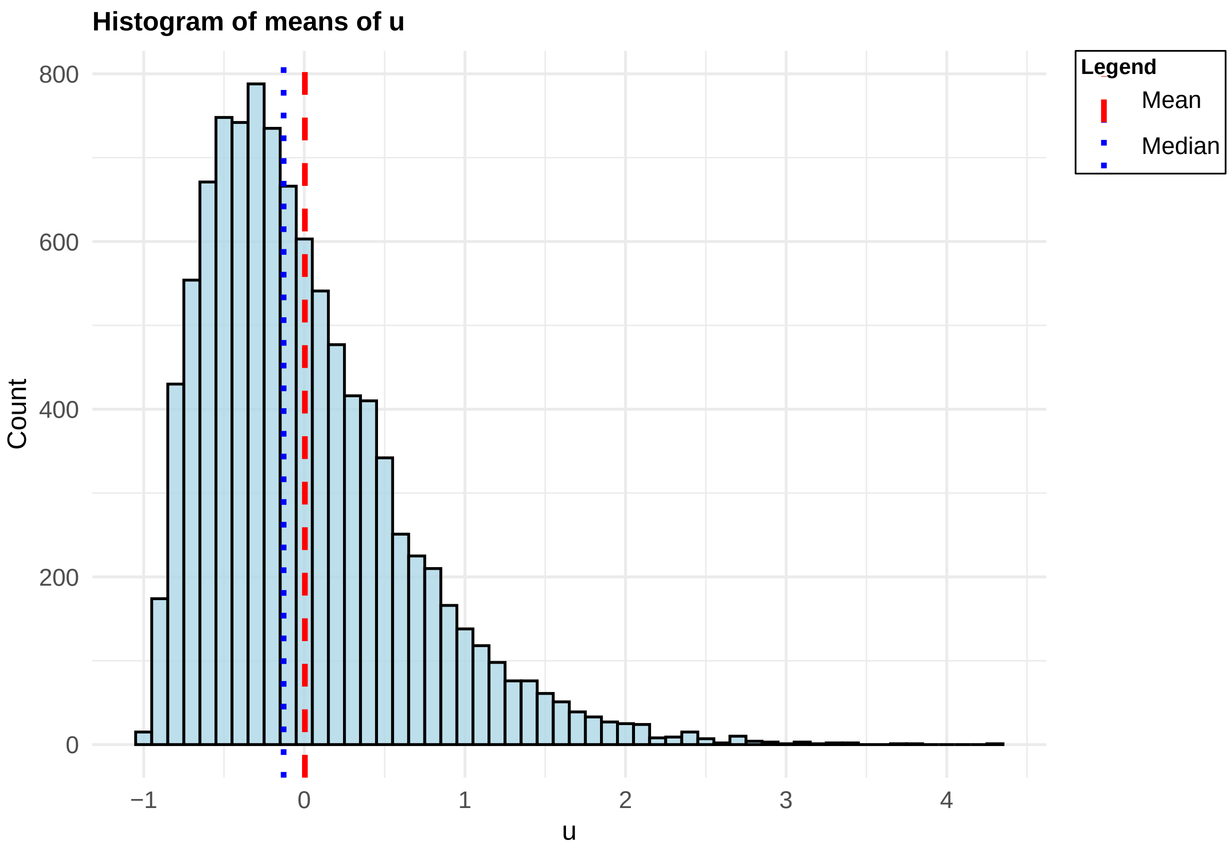
```
                             full_width = FALSE,
                             position = "center")
```

Table 1: First 6 Rows of Chi-Square Simulation Results (N=5)

| Sample Mean | Sample SD |
|---:|---:|
| -0.1054 | 0.6468 |
| 0.4332 | 1.5080 |
| -0.6697 | 0.6611 |
| 1.0986 | 1.6891 |
| -0.4063 | 0.3756 |
| -0.5543 | 0.4684 |

## 2.3 2.c)

```
par(mfrow = c(1, 1))

mean_sample_means <- mean(results_df$sample_mean)
median_sample_means <- median(results_df$sample_mean)

# plots the histogram
p3 <- ggplot(results_df, aes(x = sample_mean)) +
  geom_histogram(binwidth = 0.1, fill = 'lightblue', color = 'black', alpha = 0.8) +
  geom_vline(aes(xintercept = mean_sample_means, color = 'Mean'), linetype = 'dashed',
   ↪  linewidth = 1) +
  geom_vline(aes(xintercept = median_sample_means, color = 'Median'), linetype =
   ↪  'dotted', linewidth = 1) +
  scale_color_manual(name = 'Legend', values = c('Mean' = 'red', 'Median' = 'blue')) +
  labs(title = 'Histogram of means of u', x = 'u', y = 'Count') +
  theme_minimal() +
  theme(
    plot.title = element_text(face = 'bold', hjust = 0, size = 10),
    axis.title = element_text(size = 10),
    legend.position.inside = c(0.95, 0.95),
    legend.justification = c('right', 'top'),
    legend.background = element_rect(color = 'black', fill = 'white', linewidth = 0.3),
    legend.margin = margin(2, 2, 2, 2),
    legend.box.margin = margin(0, 0, 0, 0),
    legend.title = element_text(size = 8, face = 'bold', margin = margin(b = 0))
  )

print(p3)
```




## 3 Question 3

```
sample_mean_stats <- data.frame(sample_size = integer(),
                                mean_of_means = numeric(),
                                sd_of_means = numeric())

num_iterations <- 10000

for (N in c(10, 100, 1000)) {
  results_df <- data.frame(sample_mean = numeric(), sample_sd = numeric())
  for (j in 1:num_iterations) {
    res <- chi_mean_std(N)
    results_df <- rbind(results_df,
                        data.frame(sample_mean = res[1], sample_sd = res[2]))
  }
  sample_mean_stats <- rbind(sample_mean_stats,
                             data.frame(sample_size = N,
                                        mean_of_means = mean(results_df$sample_mean),
                                        sd_of_means = sd(results_df$sample_mean)))
}

knitr::kable(sample_mean_stats, format = "latex",
              col.names = c("Sample Size", "Mean of Means", "SD of Means"),
              caption = "Sample Mean Statistics",
              booktabs = TRUE) %>%
   kableExtra::kable_styling(latex_options = c("striped", "hold_position"),
```

```
                              full_width = FALSE,
                              position = "center")
```

Table 2: Sample Mean Statistics

| Sample Size | Mean of Means | SD of Means |
|---:|---:|---:|
| 10 | 0.0018323 | 0.4466418 |
| 100 | 0.0007109 | 0.1419020 |
| 1000 | -0.0004091 | 0.0441187 |

We can see that, as the sample size increases, the mean and standard deviation of sample means goes towards 0.

# 4 Question 4

```
modified_sd <- sample_mean_stats %>%
  pull(sd_of_means) %>%
  first() %>%
  {list(
      m100 = . * (sqrt(10) / sqrt(100)),
      m1000 = . * (sqrt(10) / sqrt(1000))
    )
  }

cat("Transformed standard deviation (100) : ", modified_sd$m100)
```

```
Transformed standard deviation (100) :  0.1412405
```

```
cat("Transformed standard deviation (1000) : ", modified_sd$m1000)
```

```
Transformed standard deviation (1000) :  0.04466418
```

We clearly see that the standard deviation goes closer to 0 as we increase the denominator (which makes sense).By applying these transformations we get the predicted standard error of the mean of u for sample sizes 100 or 1000 respectively.

We notice that these predicted values are very close from those we computed in 3 for samples of size 100 and 10 000 respectively.

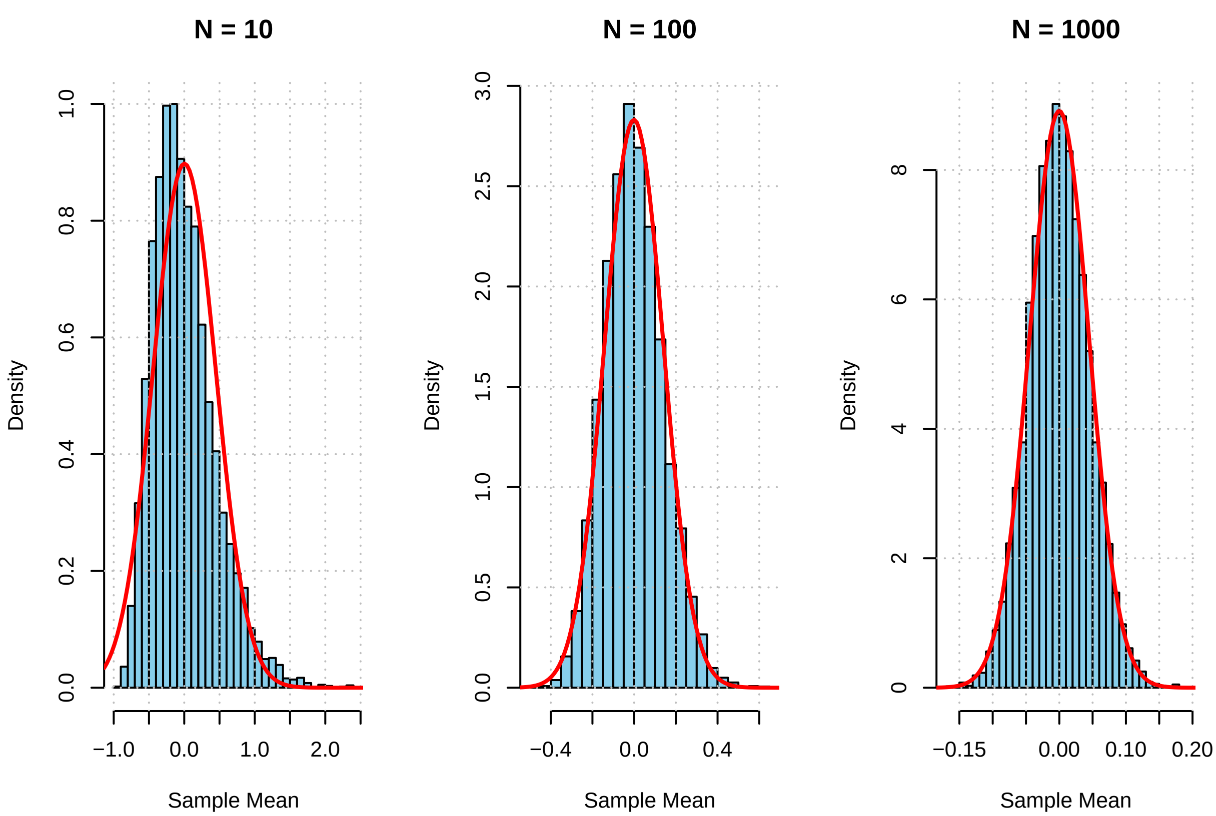
We solve for N using the following equation : $sd_n = sd_{10} * \sqrt{\frac{10}{N}}$ with $sd_{10} \approx 0.44$ and $sd_n = 0.001$.

- This gives us $N \approx 2.10^6$.

# 5 Question 5

```
num_iterations <- 10000

par(mfrow = c(1, 3))

for (N in c(10, 100, 1000)) {
  results_df <- data.frame(sample_mean = numeric(), sample_sd = numeric())
  for (j in 1:num_iterations) {
```

```
    res <- chi_mean_std(N)
    results_df <- rbind(results_df,
                        data.frame(sample_mean = res[1], sample_sd = res[2]))
  }
  mean_of_means <- mean(results_df$sample_mean)
  sd_of_means <- sd(results_df$sample_mean)

  hist(results_df$sample_mean,
       main = paste("N =", N),
       xlab = "Sample Mean",
       col = "skyblue",
       breaks = 30,
       freq = FALSE)

  grid(nx = NULL, ny = NULL, col = "gray", lty = "dotted", lwd = 1)
  x_range <- par("usr")[1:2]

  # Overlay normal curve with the calculated mean and sd
  curve(dnorm(x, mean = mean_of_means, sd = sd_of_means),
        add = TRUE,
        col = "red",
        lwd = 2,
        from = x_range[1],
        to = x_range[2])
}
```




In accordance with the central limit theorem, we can see that the distribution of the sample mean converges to a normal distribution.

# 6 Question 6

## 6.1 Definition of the Function (a,b,c,d)

```
reg_function <- function(N) {
  x1 <- runif(N, min = 0, max = 1)
  x2 <- rbinom(N, size = 1, prob = 0.3)
  u  <- rchisq(N, df = 1) - 1
  y  <- 1 + 2*x1 + 10*x2 + u
  data_df <- data.frame(x1 = x1, x2 = x2, y = y)
  model <- lm(y ~ x1 + x2, data = data_df)
  return(list(model=model, data_df=data_df))
}

simulation <- function(N){
  results_df <- as.data.frame(matrix(numeric(0), ncol = 6))
  colnames(results_df) <- c("Intercept_coef", "Intercept_se", "x1_coef", "x1_se",
   ↪  "x2_coef", "x2_se")
  for (i in 1:10000) {
    outcome <- reg_function(N)
    model <- outcome$model
    data_df <- outcome$data_df
    coefs <- summary(model)$coefficients
    # Check if x2 has no variation in data
    if(var(data_df$x2) == 0) {
      # We assign x2_coef = 0 and x2_se = 0 because x2 is essentially constant
      x2_coef_val <- 0
      x2_se_val <- 0
    } else {
      x2_coef_val <- coefs["x2", "Estimate"]
      x2_se_val <- coefs["x2", "Std. Error"]
    }
    results_df <- rbind(results_df, data.frame(
      Intercept_coef = coefs["(Intercept)", "Estimate"],
      Intercept_se   = coefs["(Intercept)", "Std. Error"],
      x1_coef        = coefs["x1", "Estimate"],
      x1_se          = coefs["x1", "Std. Error"],
      x2_coef        = x2_coef_val,
      x2_se          = x2_se_val
    ))
    }
  return(results_df)
}
```

## 6.2 Simulations (N=10)

```
simulation_results_10 <-simulation(10)
skim(simulation_results_10)
```

Table 3: Data summary

| | |
|---|---|
| Name | simulation_results_10 |
| Number of rows | 10000 |

| | |
|---|---|
| Number of columns | 6 |
| Column type frequency: | |
| numeric | 6 |
| Group variables | None |

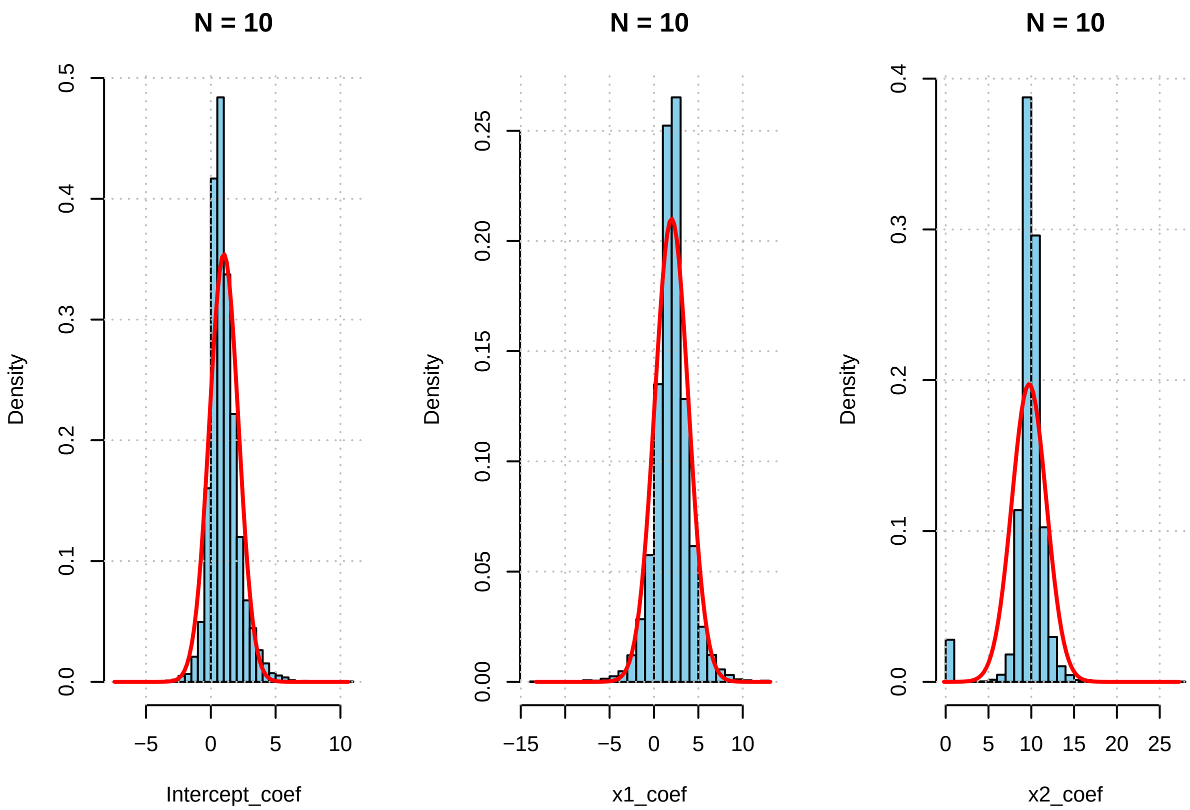
**Variable type: numeric**

| skim_variable | n_missing | complete_rate | mean | sd | p0 | p25 | p50 | p75 | p100 | hist |
|---|---|---|---|---|---|---|---|---|---|---|
| Intercept_coef | 0 | 1 | 1.01 | 1.13 | -7.32 | 0.33 | 0.83 | 1.53 | 10.50 | |
| Intercept_se | 0 | 1 | 0.95 | 0.59 | 0.06 | 0.54 | 0.82 | 1.21 | 8.70 | |
| x1_coef | 0 | 1 | 1.99 | 1.90 | -13.09 | 1.03 | 2.01 | 2.94 | 12.92 | |
| x1_se | 0 | 1 | 1.61 | 0.97 | 0.10 | 0.93 | 1.41 | 2.06 | 13.23 | |
| x2_coef | 0 | 1 | 9.74 | 2.02 | 0.00 | 9.29 | 9.88 | 10.54 | 27.07 | |
| x2_se | 0 | 1 | 0.96 | 0.60 | 0.00 | 0.56 | 0.85 | 1.24 | 6.10 | |

# 7 Question 7

## 7.1 Histogram : Simulations (N=10)

```r
par(mfrow = c(1, 3))
cols <- c("Intercept_coef", "x1_coef", "x2_coef")

for(cl in cols){
  vec <- simulation_results_10[[cl]]
  hist(vec,
       main = paste("N =", 10),
       xlab = cl,
       col = "skyblue",
       breaks = 30,
       freq = FALSE)
  mean_p <- mean(vec, na.rm = TRUE)
  sd_p <- sd(vec, na.rm = TRUE)
  x_range <- range(vec, na.rm = TRUE) + c(-0.1, 0.1) * sd_p
   grid(nx = NULL, ny = NULL, col = "gray", lty = "dotted", lwd = 1)
  curve(dnorm(x, mean = mean_p, sd = sd_p),
        add = TRUE,
        col = "red",
        lwd = 2,
        from = x_range[1],
        to = x_range[2])
}
```




## 7.2 Simulations (N=1000)

```
simulation_results_1000 <-simulation(1000)
skim(simulation_results_10)
```

Table 5: Data summary

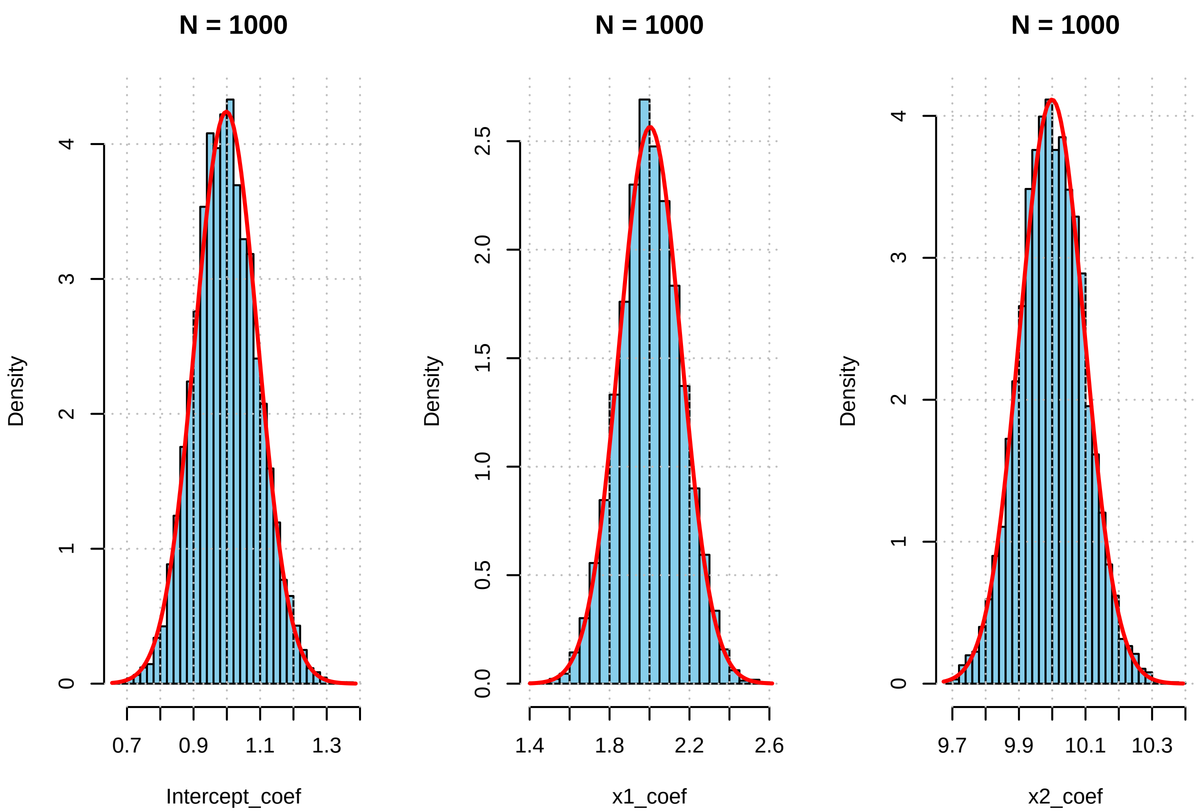
| | |
|---|---|
| Name | simulation_results_10 |
| Number of rows | 10000 |
| Number of columns | 6 |
| Column type frequency: | |
| numeric | 6 |
| Group variables | None |

**Variable type: numeric**

| skim_variable | n_missing | complete_rate | mean | sd | p0 | p25 | p50 | p75 | p100 | hist |
|---|---|---|---|---|---|---|---|---|---|---|
| Intercept_coef | 0 | 1 | 1.01 | 1.13 | -7.32 | 0.33 | 0.83 | 1.53 | 10.50 | |
| Intercept_se | 0 | 1 | 0.95 | 0.59 | 0.06 | 0.54 | 0.82 | 1.21 | 8.70 | |
| x1_coef | 0 | 1 | 1.99 | 1.90 | -13.09 | 1.03 | 2.01 | 2.94 | 12.92 | |
| x1_se | 0 | 1 | 1.61 | 0.97 | 0.10 | 0.93 | 1.41 | 2.06 | 13.23 | |
| x2_coef | 0 | 1 | 9.74 | 2.02 | 0.00 | 9.29 | 9.88 | 10.54 | 27.07 | |
| x2_se | 0 | 1 | 0.96 | 0.60 | 0.00 | 0.56 | 0.85 | 1.24 | 6.10 | |

### 7.3 Histogram : Simulations (N=1000)

```
par(mfrow = c(1, 3))
cols <- c("Intercept_coef", "x1_coef", "x2_coef")

for(col in cols){
  vec <- simulation_results_1000[[col]]
  hist(vec,
       main = paste("N =", 1000),
       xlab = col,
       col = "skyblue",
       breaks = 30,
       freq = FALSE)
  mean_p <- mean(vec)
  sd_p   <- sd(vec)

  # Define x-range based on data (with a little margin)
  x_range <- range(vec, na.rm = TRUE) + c(-1, 1)*sd_p*0.1
  grid(nx = NULL, ny = NULL, col = "gray", lty = "dotted", lwd = 1)
  curve(dnorm(x, mean = mean_p, sd = sd_p),
        add = TRUE, col = "red", lwd = 2,
        from = x_range[1], to = x_range[2])
}
```




The more we increase the sample size (N), the more the distribution of the sample mean tends towards a normal distribution which is conform to the central limit theorem. Thus the coefficients converge to the population value.

The mean and standard deviation of the standard error of coefficients tend towards zero as the sample size

(N) increases, thus we can safely assert that this estimator is consistent (its variance is asymptotically null).

# 8 Question 8

## 8.1 Definition of the Function (a,b,c,d)

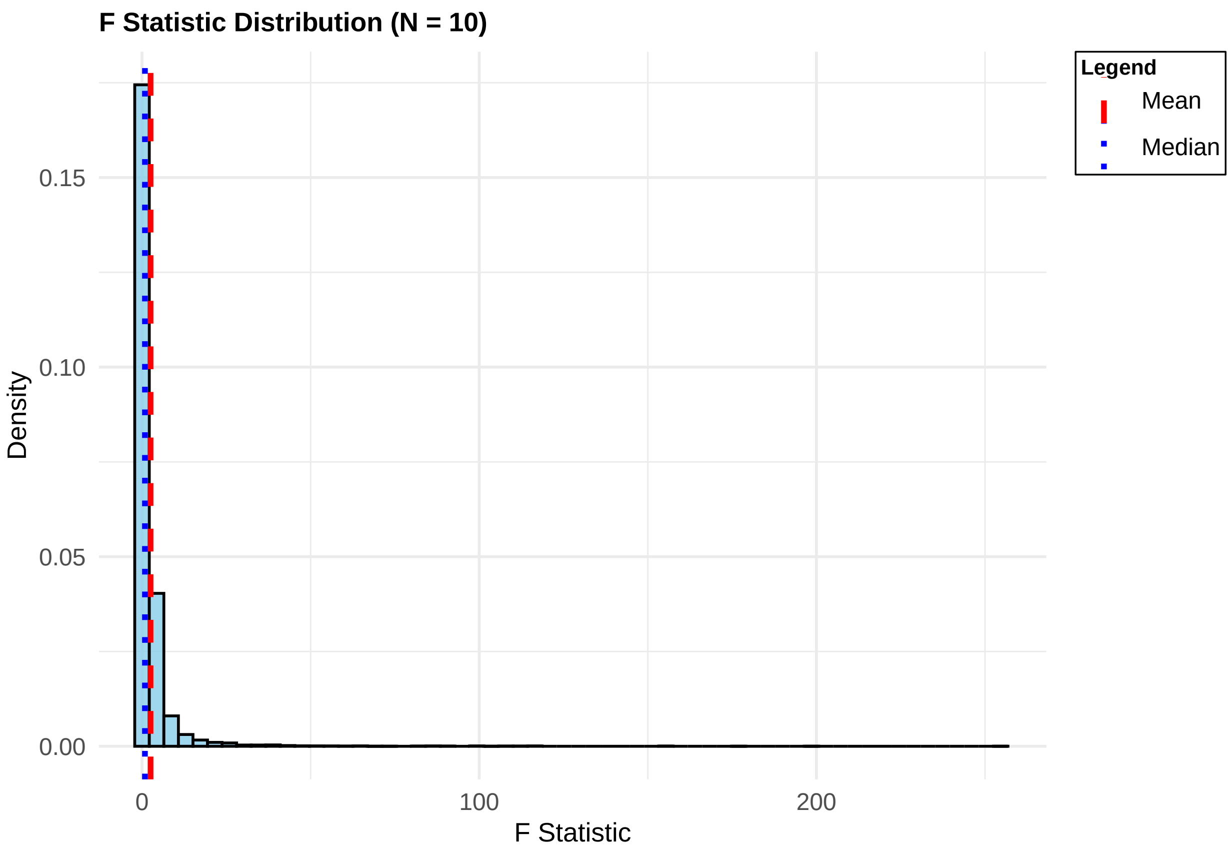
```
simulation2 <- function(N){
  test_results <- data.frame(F_statistic = numeric(0))
  for (i in 1:10000) {
    result <- tryCatch({
      outcome <- reg_function(N)
      model <- outcome$model
      data_df <- outcome$data_df
      # Check for aliased coefficients
      if(any(is.na(coef(model)))) {
        next
      }
      joint_test <- linearHypothesis(model, c("(Intercept) = 1", "x1 = 2"))
      f_stat <- joint_test$F[2]
    }, error = function(e) {
      NA
    })
    if(!is.na(result)) {
      test_results <- rbind(test_results, data.frame(F_statistic = result))
    }
  }
  return(test_results)
}
```

## 8.2 Simulations (N = 10)

```
F_stat_vec_10 <- simulation2(10)$F_statistic

sd_p_10 <- sd(F_stat_vec_10, na.rm = TRUE)
x_range <- range(F_stat_vec_10, na.rm = TRUE)
x_range[1] <- max(0, x_range[1] - 0.1 * sd_p_10)
x_range[2] <- x_range[2] + 0.1 * sd_p_10

# Plot histogram
 df10 <- data.frame(F_stat = F_stat_vec_10)

 p10 <- ggplot(df10, aes(x = F_stat)) +
   geom_histogram(aes(y = ..density..), bins = 60, fill = "skyblue", color = "black",
    ↪  alpha = 0.8) +
   geom_vline(aes(xintercept = mean(F_stat), color = "Mean"), linetype = "dashed",
    ↪  linewidth = 1) +
   geom_vline(aes(xintercept = median(F_stat), color = "Median"), linetype = "dotted",
    ↪  linewidth = 1) +
   scale_color_manual(name = "Legend", values = c("Mean" = "red", "Median" = "blue")) +
   labs(title = "F Statistic Distribution (N = 10)", x = "F Statistic", y = "Density") +
   coord_cartesian(xlim = x_range) +
   theme_minimal() +
   theme(
```

```r
    plot.title = element_text(face = "bold", hjust = 0, size = 10),
    axis.title = element_text(size = 10),
    legend.position.inside = c(0.95, 0.95),
    legend.justification = c("right", "top"),
    legend.background = element_rect(color = "black", fill = "white", linewidth = 0.3),
    ↪
    legend.margin = margin(2, 2, 2, 2),
    legend.box.margin = margin(0, 0, 0, 0),
    legend.title = element_text(size = 8, face = "bold", margin = margin(b = 0))
  )
print(p10)
```

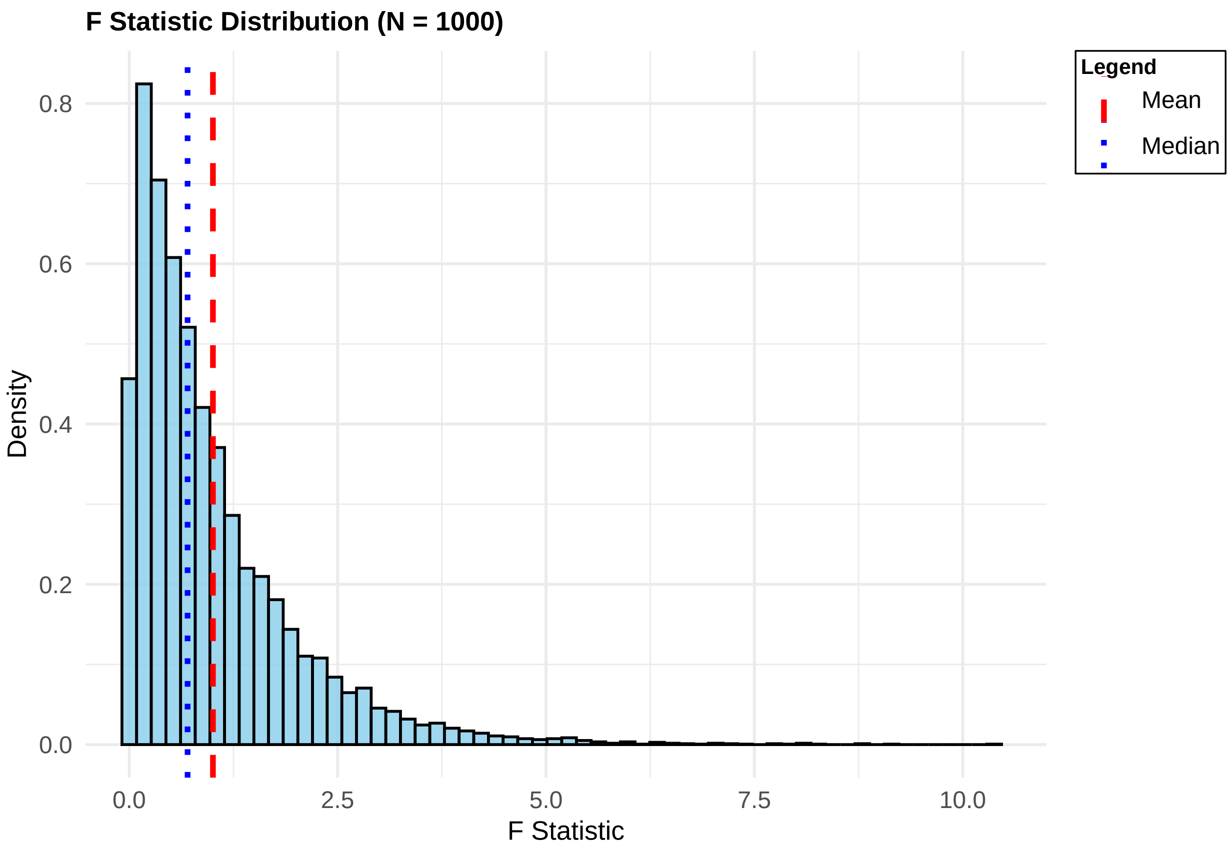
```
Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.
i Please use `after_stat(density)` instead.
This warning is displayed once every 8 hours.
Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
generated.
```




## 8.3 Simulations (N = 1000)

```r
F_stat_vec_1000 <- simulation2(1000)$F_statistic

sd_p_1000 <- sd(F_stat_vec_1000, na.rm = TRUE)
x_range <- range(F_stat_vec_1000, na.rm = TRUE)
x_range[1] <- max(0, x_range[1] - 0.1 * sd_p_1000)
x_range[2] <- x_range[2] + 0.1 * sd_p_1000
```

```r
# Plot histogram
df1000 <- data.frame(F_stat = F_stat_vec_1000)

p1000 <- ggplot(df1000, aes(x = F_stat)) +
  geom_histogram(aes(y = ..density..), bins = 60, fill = "skyblue", color = "black",
   ↪  alpha = 0.8) +
  geom_vline(aes(xintercept = mean(F_stat), color = "Mean"), linetype = "dashed",
   ↪  linewidth = 1) +
  geom_vline(aes(xintercept = median(F_stat), color = "Median"), linetype = "dotted",
   ↪  linewidth = 1) +
  scale_color_manual(name = "Legend", values = c("Mean" = "red", "Median" = "blue")) +
  labs(title = "F Statistic Distribution (N = 1000)", x = "F Statistic", y = "Density")
   ↪  +
  coord_cartesian(xlim = x_range) +
  theme_minimal() +
  theme(
    plot.title = element_text(face = "bold", hjust = 0, size = 10),
    axis.title = element_text(size = 10),
    legend.position.inside = c(0.95, 0.95),
    legend.justification = c("right", "top"),
    legend.background = element_rect(color = "black", fill = "white", linewidth = 0.3),
     ↪
    legend.margin = margin(2, 2, 2, 2),
    legend.box.margin = margin(0, 0, 0, 0),
    legend.title = element_text(size = 8, face = "bold", margin = margin(b = 0))
  )
print(p1000)
```




# 9 Question 9

## 9.1 9.a)

```
critval_95_n10 <- qf(0.95, df1 = 2, df2 = 7)
critval_99_n10 <- qf(0.99, df1 = 2, df2 = 7)
critval_95_n1000 <- qf(0.95, df1 = 2, df2 = 997)
critval_99_n1000 <- qf(0.99, df1 = 2, df2 = 997)
```

## 9.2 9.b)

```
nb_reject_95_n10 <- sum(F_stat_vec_10 > critval_95_n10)
nb_reject_99_n10 <- sum(F_stat_vec_10 > critval_99_n10)
nb_reject_95_n1000 <- sum(F_stat_vec_1000 > critval_95_n1000)
nb_reject_99_n1000 <- sum(F_stat_vec_1000 > critval_99_n1000)

pct_reject_95_n10 <- (nb_reject_95_n10 / length(F_stat_vec_10)) * 100
pct_reject_99_n10 <- (nb_reject_99_n10 / length(F_stat_vec_10)) * 100
pct_reject_95_n1000 <- (nb_reject_95_n1000 / length(F_stat_vec_1000)) * 100
pct_reject_99_n1000 <- (nb_reject_99_n1000 / length(F_stat_vec_1000)) * 100

results_df <- data.frame(
  Distribution = c("F(2,10)", "F(2,10)", "F(2,1000)", "F(2,1000)"),
  Significance_Level = c("95%", "99%", "95%", "99%"),
  Critical_Value = c(critval_95_n10, critval_99_n10, critval_95_n1000, critval_99_n1000),
```

```r
  Rejections_Count = c(nb_reject_95_n10, nb_reject_99_n10, nb_reject_95_n1000,
   ↪  nb_reject_99_n1000),
  Rejections_Percentage = c(pct_reject_95_n10, pct_reject_99_n10, pct_reject_95_n1000,
   ↪  pct_reject_99_n1000)
)

results_df$Critical_Value <- round(results_df$Critical_Value, 4)
results_df$Rejections_Percentage <- round(results_df$Rejections_Percentage, 2)
```

```r
kable(results_df, format = "latex",
      col.names = c("Distribution", "Significance Level",
                    "Critical Value", "Rejections Count", "Rejections (%)"),
      caption = "F-Test Rejection Results"
) %>%
  kable_styling(bootstrap_options = c("striped", "hold_position"),
                full_width = FALSE,
                position = "center",
                latex_options = "hold_position")
```

Table 7: F-Test Rejection Results

| Distribution | Significance Level | Critical Value | Rejections Count | Rejections (%) |
|---|---|---|---|---|
| F(2,10) | 95% | 4.7374 | 1039 | 10.70 |
| F(2,10) | 99% | 9.5466 | 444 | 4.57 |
| F(2,1000) | 95% | 3.0048 | 482 | 4.82 |
| F(2,1000) | 99% | 4.6265 | 109 | 1.09 |

## 9.3 9.c)

If the distribution of the error terms had been normal, we would expect that at the 95% (99%) confidence level we would reject $H_0$ for 5%(1%) of the samples tested.

We notice that, based on our simulation, the observed rejection rates were much higher when the sample size is 10 (around 11% of rejection at the 95% confidence level and around 5% for the 99% confidence level)

When we look at samples of larger sizes (1000), however, we notice observed results are much closer to the expected proportions !

- This is a direct illustration of the central limit theorem : a larger sample size improves the approximation to normality.

# Pset 4 - Theory

Romain Fernex

March 2025

## 1 Problem 1

Let's show that $\hat{\alpha} = \bar{y} - \hat{\beta}\bar{x} \xrightarrow{P} \alpha$ : We know that $\hat{\beta} \xrightarrow{P} \beta$ (consistent estimator) Using the Law of Large Number, we obtain :

$$\bar{y} = \frac{1}{N}\sum_{i=1}^{N} y_i \xrightarrow{P} E(y_i) = E(\alpha + \beta x_i + u_i) \tag{1}$$

Similarly, based on the Law of Large Number, we also get :

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{n} x_i \xrightarrow{P} E(x_i) \tag{2}$$

Considering result (2) and the consistency of $\hat{\beta}$, we can use the Slutsky theorem to assert that :

$$\hat{\beta}\bar{x} \xrightarrow{P} \beta E(x_i) \tag{3}$$

Lastly, using Slutsky with results (1) and (3) this time, we get :

$$\hat{\alpha} \xrightarrow{P} E(\alpha + \beta x_i + u_i) - \beta E(x_i) = \alpha + E(u_i) \tag{4}$$

As $E(u_i|x_i) = 0$ is a condition for $\hat{\beta}$ to be consistent and since $E(u_i|x_i) = 0 \Longrightarrow E(u_i) = 0$, we have :

$$\hat{\alpha} \xrightarrow{P} \alpha \tag{5}$$

So $\hat{\alpha}$ is a consistent estimator of $\alpha$ !

## 2 Problem 2

We start by proving that :

$$\hat{u}_i = (u_i - \bar{u}) - \sum_{j=1}^{k}(\hat{\beta}_j - \beta_j)(x_{ij} - \hat{x}_j) \tag{6}$$

We know that :

$$
\begin{aligned}
\hat{u}_i &= y_i - \hat{y}_i \\
&= (\beta_0 + \sum_{j=1}^{k} \beta_j x_{ij} + u_i) - (\hat{\beta}_0 + \sum_{j=1}^{k} \hat{\beta}_j x_{ij}) \\
&= (\beta_0 - \hat{\beta}_0) + \sum_{j=1}^{k} (\beta_j - \hat{\beta}_j) x_{ij} + u_i
\end{aligned} \tag{7}
$$

Now that we have (7), we can take the sum over all i and divide by n:

$$
\begin{aligned}
\frac{1}{n}\sum_{i=1}^{k} \hat{u}_i &= (\beta_0 - \hat{\beta}_0) + \sum_{j=1}^{k} (\beta_j - \hat{\beta}_j)[\frac{1}{n}\sum_{i=1}^{n} x_{ij}] + \bar{u} \\
\underbrace{\bar{\hat{u}}}_{=0} &= (\beta_0 - \hat{\beta}_0) + \sum_{j=1}^{k} (\beta_j - \hat{\beta}_j)\bar{x_j} + \bar{u} \\
\hat{u}_i - 0 &= \hat{u}_i - (\beta_0 - \hat{\beta}_0) - \sum_{j=1}^{k} (\beta_j - \hat{\beta}_j)\bar{x_j} - \bar{u} \\
\hat{u}_i &= (\beta_0 - \hat{\beta}_0) + \sum_{j=1}^{k} (\beta_j - \hat{\beta}_j) x_{ij} + u_i - (\beta_0 - \hat{\beta}_0) - \sum_{j=1}^{k} (\beta_j - \hat{\beta}_j)\bar{x_j} - \bar{u} \\
\hat{u}_i &= (u_i - \bar{u}) - \sum_{j=1}^{k} (\hat{\beta}_j - \beta_j)(x_{ij} - \bar{x}_j)
\end{aligned} \tag{8}
$$

Now, using result (8) :

$$
\hat{\sigma}^2 = \frac{1}{n-k-1} \sum_{i=1}^{n} [(u_i - \bar{u}) - \sum_{j=1}^{k} (\hat{\beta}_j - \beta_j)(x_{ij} - \bar{x}_j)]^2 \tag{9}
$$

Let's expand this squared sum to better identify which terms tends to what :

$$
\hat{\sigma}^2 = \frac{1}{n-k-1} \sum_{i=1}^{n} \left[ (u_i - \bar{u})^2 - 2(u_i - \bar{u}) \sum_{j=1}^{n} (\hat{\beta}_j - \beta_j)(x_{ij} - \bar{x}_j) + \left( \sum_{j=1}^{k} (\hat{\beta}_j - \beta_j)(x_{ij} - \bar{x}_j) \right)^2 \right] \tag{10}
$$

We know that $E[x_i u_i] = 0$ and that $E[x_i x_i']$ is non singular. Thus, the conditions are satisfied for the OLS estimator to be consistent. In other words, we can assert that :

$$
\forall j, \hat{\beta}_j \xrightarrow{P} \beta_j \tag{11}
$$

Using Slutsky with result (11), we can now state :

$$
\sum_{j=1}^{k} (\hat{\beta}_j - \beta_j)(x_{ij} - \bar{x}_j) \xrightarrow{P} \sum_{j=1}^{k} (\beta_j - \beta_j)(x_{ij} - \bar{x}_j) = 0 \tag{12}
$$

Similarly, using the continuous mapping theorem with $g : x \mapsto x^2$, we get :

$$\left( \sum_{j=1}^{k} (\hat{\beta}_j - \beta_j)(x_{ij} - \bar{x}_j) \right)^2 \xrightarrow{P} 0^2 = 0 \tag{13}$$

Finally, apllying the Law of Large number to the leftmost term we get :

$$\frac{1}{n-k-1} \sum_{i=1}^{n} (u_i - \bar{u})^2 \approx \frac{1}{n} \sum_{i=1}^{n} (u_i - \bar{u})^2 \xrightarrow{P} E[(u_i - \bar{u})^2] = Var(u_i) \tag{14}$$

Thus, applying the Slutsky theorem with results (10), (12), (13) and (14), we get :

$$\hat{\sigma}^2 \xrightarrow{P} Var(u_i) + 0 + 0 = \sigma^2 \tag{15}$$

Thus we can safely state that $\hat{\sigma}^2$ is a consistent estimator for $\sigma^2$.