Pset 4 - Empirical

Romain Fernex

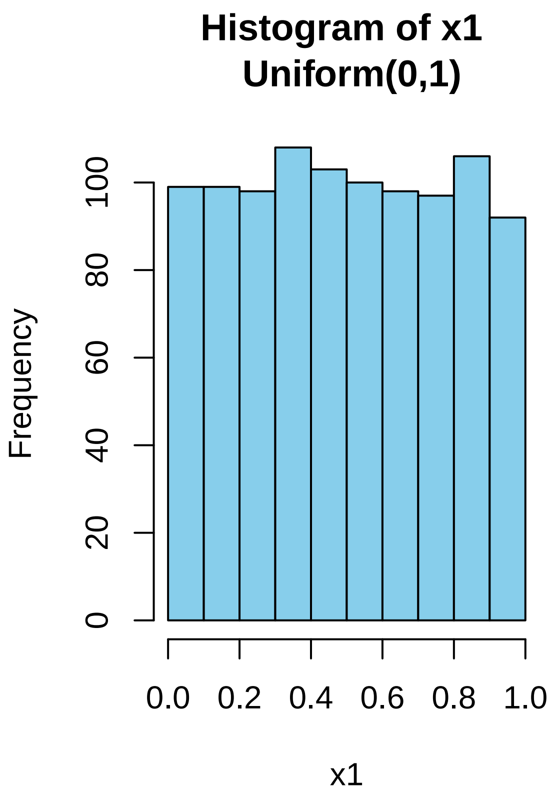
2025-03-16

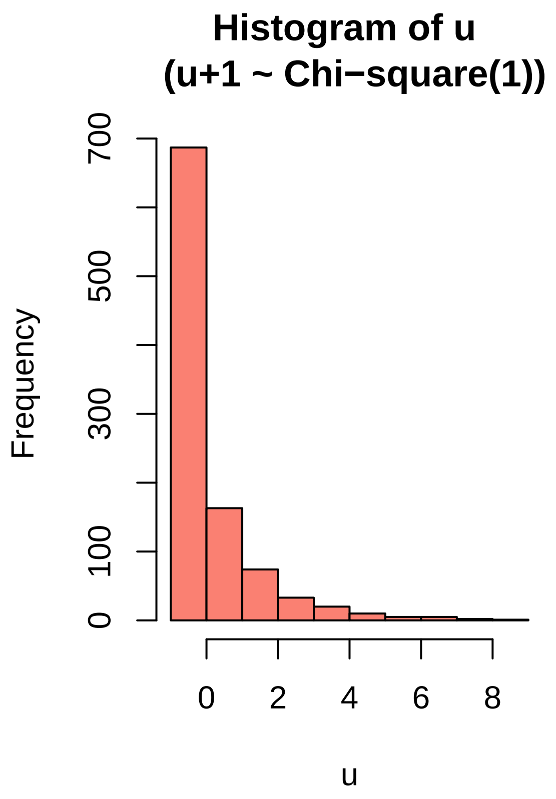
Contents

1 Question 1

```r
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))
```

```
hist(data$x1, main = 'Histogram of x1 \
 Uniform(0,1)', xlab = 'x1', col = 'skyblue', border = 'black')

hist(data$u, main = 'Histogram of u \
 (u+1 ~ Chi-square(1))', xlab = 'u', col = 'salmon', border = 'black')
```





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]) )
}
```

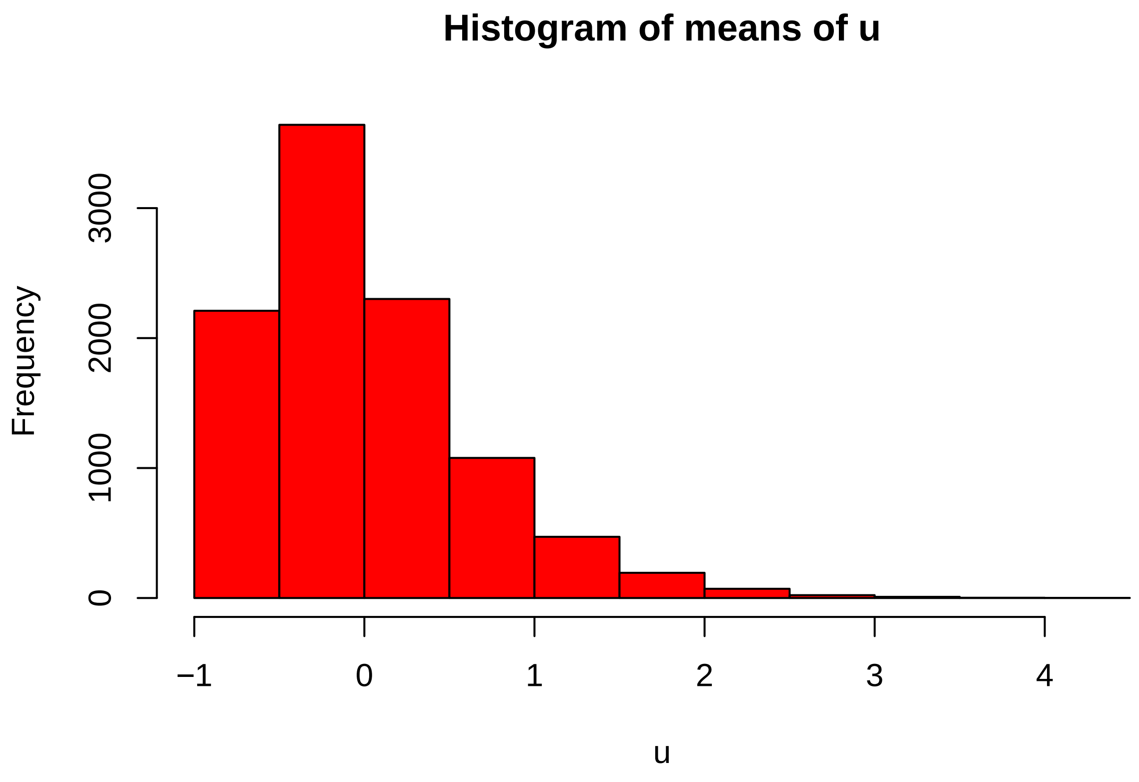
```r
knitr::kable(head(results_df), format = "latex",
             col.names = c("Sample Mean", "Sample SD"),
             caption = "First 6 Rows of Chi-Square Simulation Results (N=5)",
             digits = 4,
             booktabs = TRUE) %>%
  kableExtra::kable_styling(latex_options = c("striped", "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)

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

hist_simulation <- results_df %>%
  pull(sample_mean) %>%
  hist(main = 'Histogram of means of u', xlab = 'u', col = 'red', border = 'black')
```



3 Question 3

```r
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

```r
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)
```

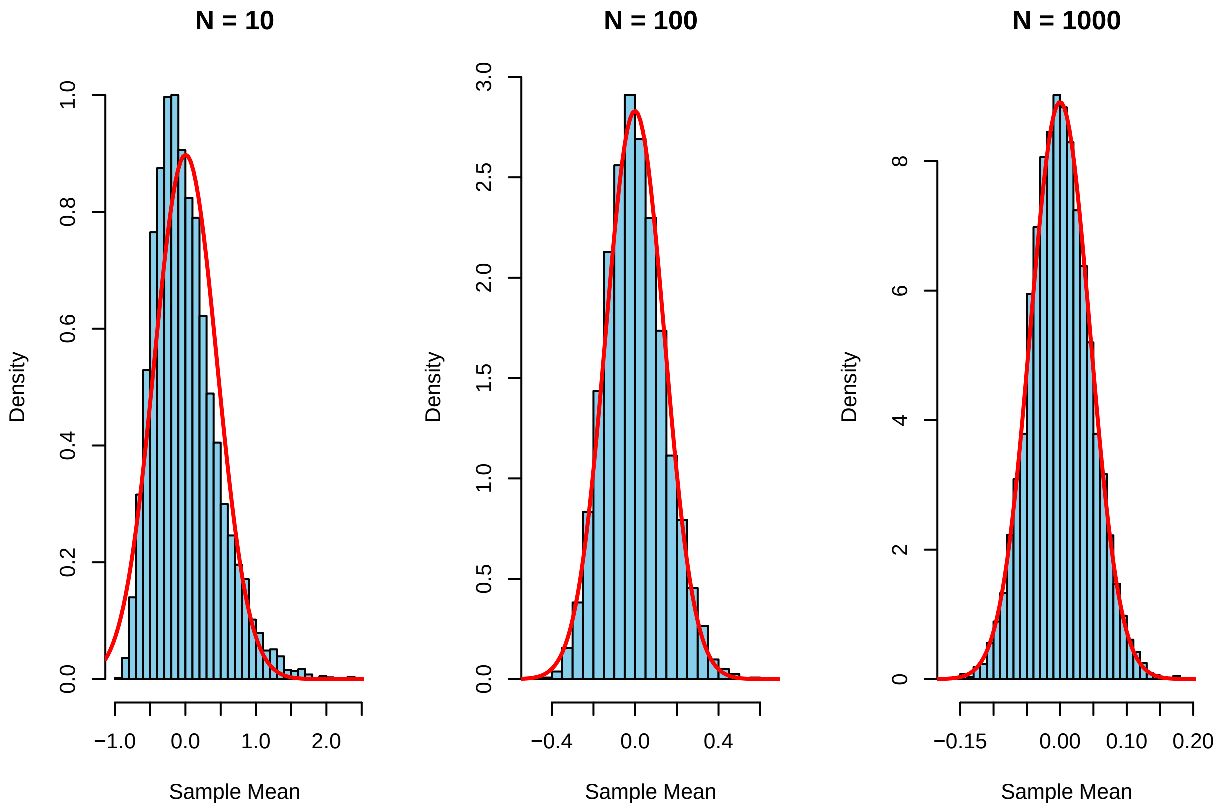
```
Transformed standard deviation (1000) :  0.04466418
```

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)

  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)

```r
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

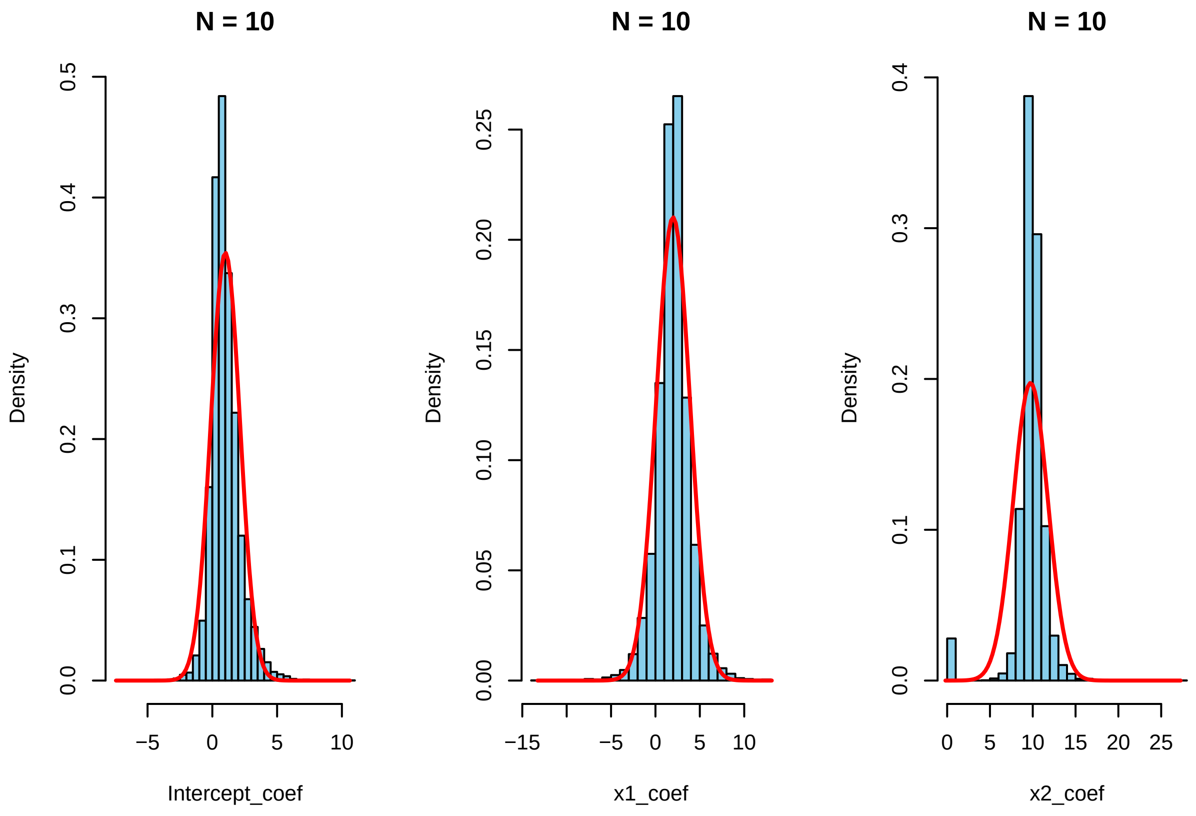
Variable type: numeric

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)
  # Use the range of the data with some padding
  x_range <- range(vec, na.rm = TRUE) + c(-0.1, 0.1) * sd_p
  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)

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

Table 5: Data summary

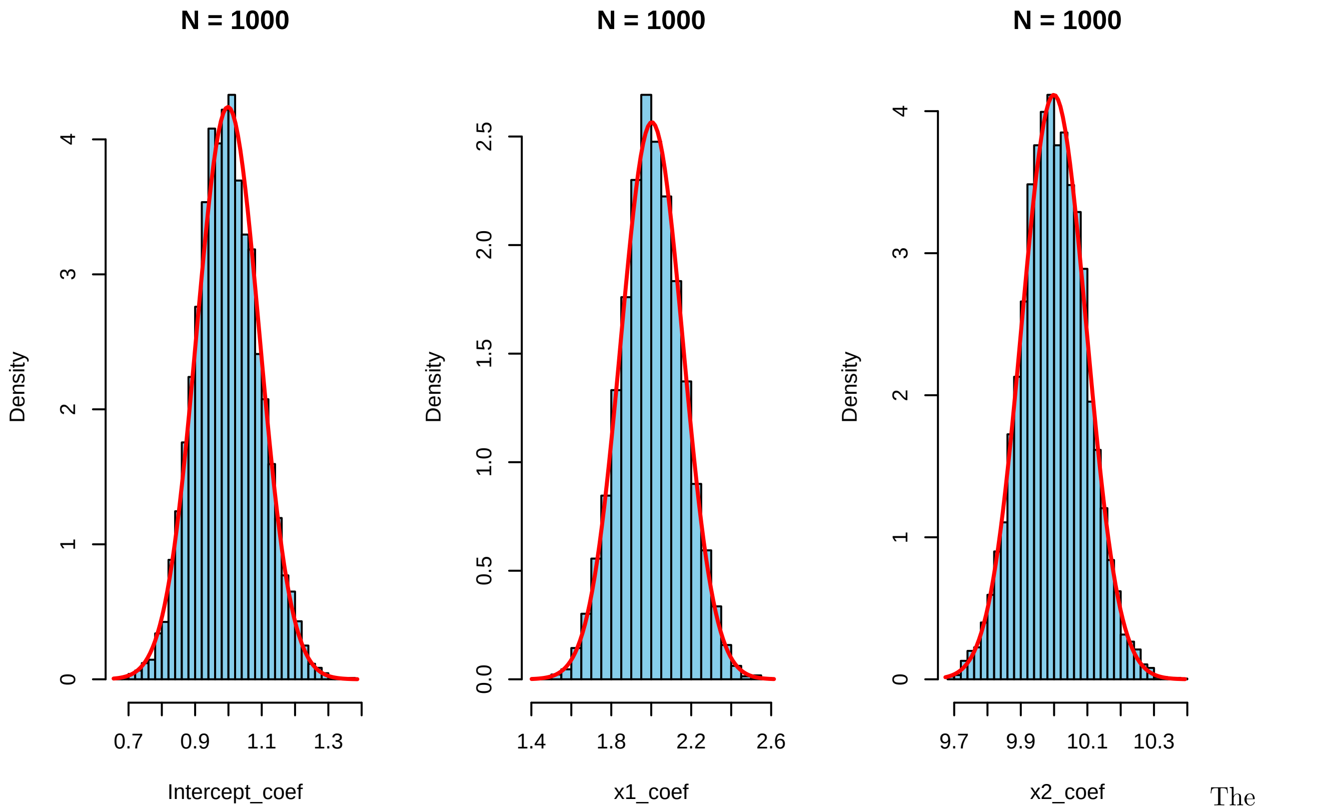
Name	simulation_results_10
Number of rows	10000
Number of columns	6
Column type frequency:	
numeric	6
Group variables	None

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)

```r
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
  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 teh 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).

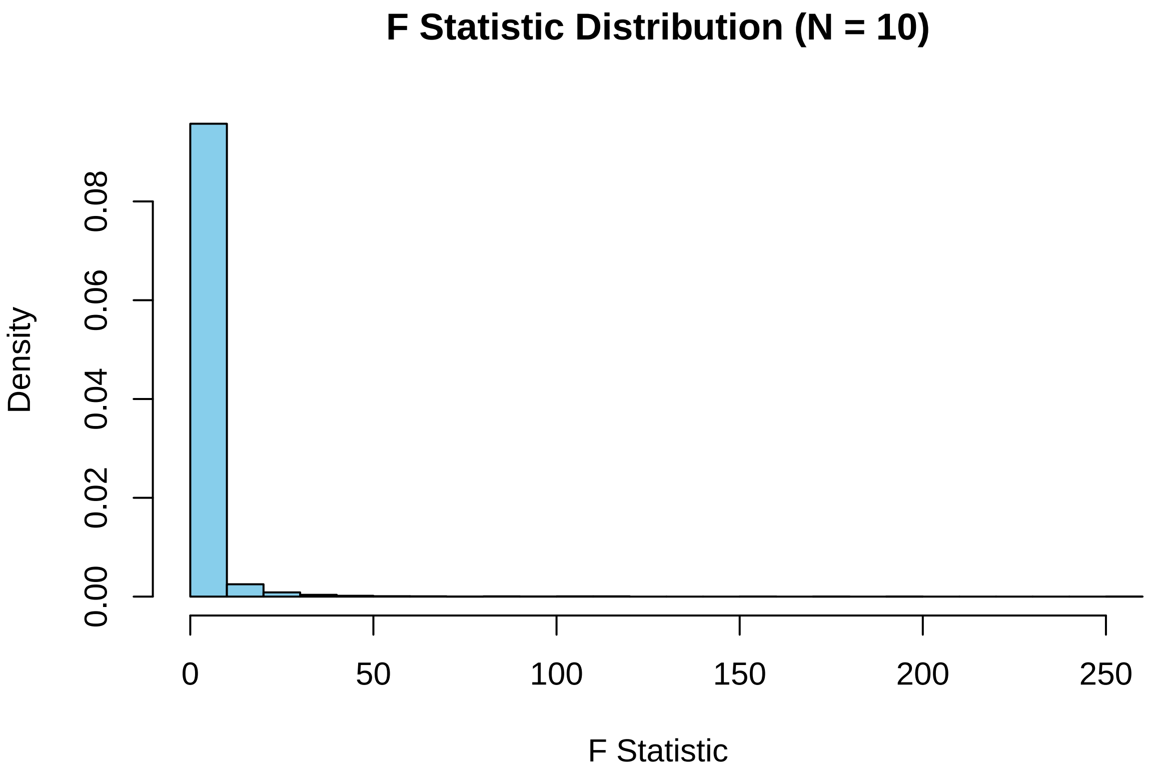
8 Question 8

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

```r
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)

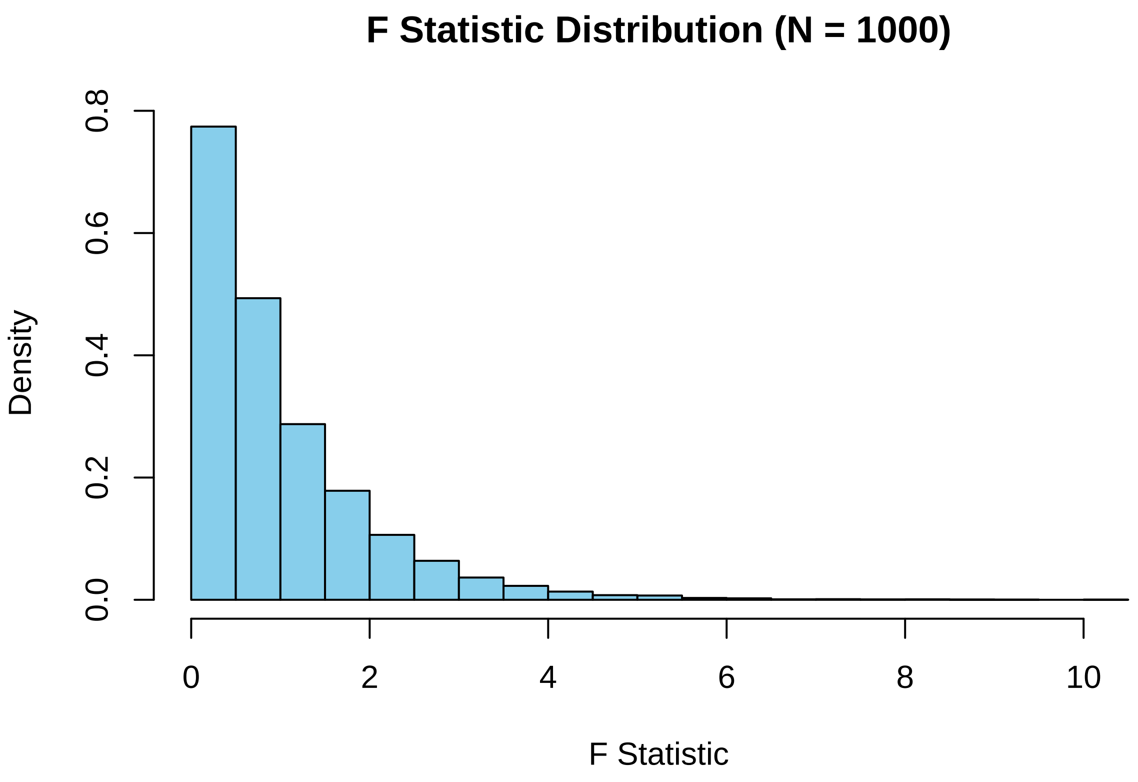
```
F_stat_vec_10 <- simulation2(10)$F_statistic

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
par(mfrow = c(1, 1))
hist(F_stat_vec_10,
     main = "F Statistic Distribution (N = 10)",
     xlab = "F Statistic",
     col = "skyblue",
     breaks = 30,
     freq = FALSE,
     xlim = x_range)
```



8.3 Simulations (N = 1000)

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

```r
 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

# Plot histogram
 hist(F_stat_vec_1000,
      main = "F Statistic Distribution (N = 1000)",
      xlab = "F Statistic",
      col = "skyblue",
      breaks = 30,
      freq = FALSE,
      xlim = x_range)
```



9 Question 9

9.1 9.a)

```r
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)

```r
nb_reject_95_n10 <- sum(F_stat_vec_10 > critval_95_n10)
nb_reject_99_n10 <- sum(F_stat_vec_10 > critval_99_n10)
```

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

```r
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),
  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")
```

9.3 9.c)

If the distribution of the error terms had been normal, we would expect that at the 95% (99%) 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 implication of the central limit theorem : a larger sample size improves the approximation to normality.