Econ 5027 assisgnment one

Tie Ma

2024-02-17

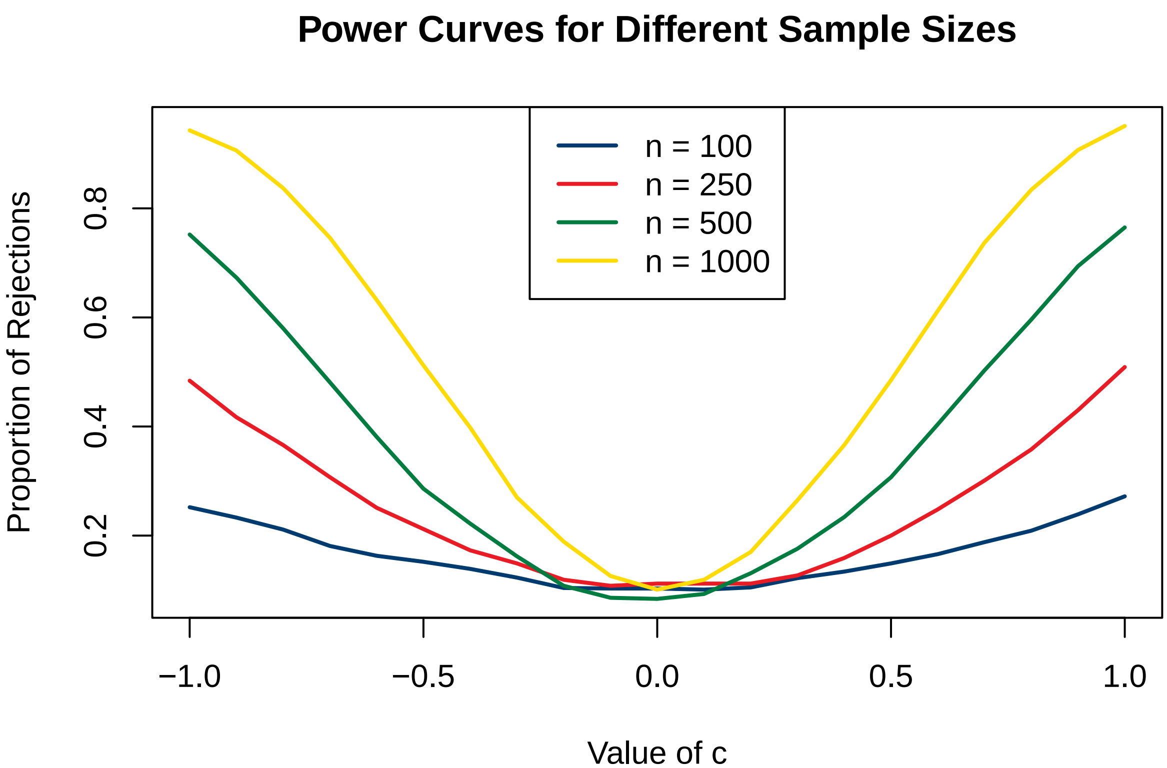
Question 3: Hypothesis Testing

Q3-A

```
#Q 3-A
#clean the environment for the the assignment.
rm(list = ls())

set.seed(1234567)

#set the variance matrix
evil_variance_matrix <- matrix(c(1.0964, -0.5313, -0.5730,
                                  -0.5313, 0.9381, -0.4184,
                                  -0.5730, -0.4184, 1.0228), nrow = 3, ncol = 3, byrow = TRUE)

#set up the mu
mu <- c(1.1141, -0.6768, 3.3521)

# c sequence (from -1 to 1 by 0.1)
c_value <- seq(from = -1, to = 1, by = 0.1)

# Number of simulation rounds
R <- 1000

# Parameters
# the n is for the data generate process within the loop
# the n_value of for the loop to keep going.
n_value <- c(100, 250, 500, 1000)

#set up the beta_ture
beta_true <- c(-0.8, 0, 0, 0.1)

#the t-test with 0.05 degree of freedom (do not change to a you will lost it)
alpha <- 0.05



# The storage matrix (I am uisng three became I have no confidence of my coding skill)

  #final_matrix
    Final_matrix <- matrix(0,ncol = length(c_value), nrow = length(n_value))

#the simulation time
```

```r
for (i in 1:length(n_value)){
  #rest the n
  n <- n_value[i]
  #reset the TF matrix
  TF_matrix <- matrix(NA, ncol=length(c_value), nrow = R)

  #this loop run 1000 times with current value of n
  for (j in 1:R) {

    x <- mvrnorm(n, mu = mu, Sigma = evil_variance_matrix) # its should be "Sigma" not "sigma"

    # Add intercept (we now turn the x from 3x3 matrix to 3x4 matrix)
    x <- cbind(rep(1, n), x)
    # e need to be processed within the loop
    e <- rnorm(n, mean = 0, sd = 1)  #normal distribution
    #now its the time for the calculate the y!
    y <- x %*% beta_true + e

    # OLS estimation
    beta_hat <- solve(t(x) %*% x) %*% (t(x) %*% y)
    eps_hat <- y - x %*% beta_hat

    #the degree of freedom
    dof <- n - length(beta_hat)
    #  the mu
    sigma_hat_sq <- (1 / dof) * sum(eps_hat^2)
    #estimate of the
    v_beta_hat <- solve(t(x) %*% x) * sigma_hat_sq
    # standard error time!
    diag_v_beta_hat <- diag(v_beta_hat)
    sd <- sqrt(diag_v_beta_hat)

    #the T- test time!
    for (k in 1:length(c_value)) {
      #T statistic for the beta3
      t_st <- (beta_hat[3,] - c_value[k])/sd[3]
      #T test P value
      t_p_value <- 1 - pt(abs(t_st), dof) #using the TA's note!
   #the matrix time!
      #count the p_value
      TF_matrix[j, k] <- t_p_value < 0.05
      #this things was crashed my mac for times
      #I spend 2 hours on this line a lone
     }
   }
  Final_matrix[i, ] <- colMeans(TF_matrix)
  #and other hour on this line.
 }

#god tis finally end, I rest in peace.
```

```r
#change the row name of the matrix
rownames(Final_matrix) <- c("n = 100", "n = 250", "n = 500", "n = 1000")


# Assuming Final_matrix and c_value are already defined

# Colors
dw_blue <- "#003b6f"  # Doctor Who blue
cr_red <- "#e91c25"   # Carlton red
ua_green <- "#007c41"  # ualberta green
ua_gold <- "#ffdb05"  # ualberta gold

# Prepare the data
n_labels <- c("n = 100", "n = 250", "n = 500", "n = 1000")
colors <- c(dw_blue, cr_red, ua_green, ua_gold)

# Plotting
plot(c_value, Final_matrix[1, ], type = "l", col = colors[1], ylim = c(min(Final_matrix), max(Final_
lines(c_value, Final_matrix[2, ], col = colors[2], lwd = 2)
lines(c_value, Final_matrix[3, ], col = colors[3], lwd = 2)
lines(c_value, Final_matrix[4, ], col = colors[4], lwd = 2)
legend("top", legend = n_labels, col = colors, lty = 1, lwd = 2)
```

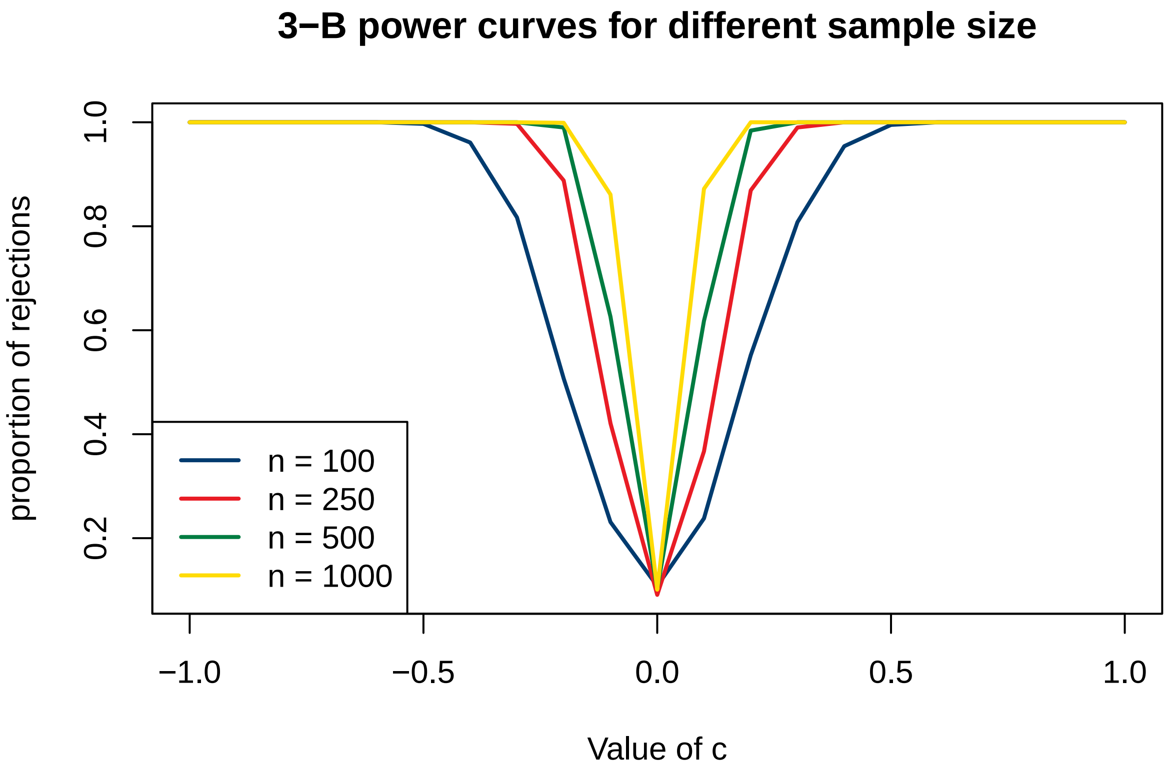


Q3-B

```r
#clean the enviroment and set seed for compare the answer
rm(list = ls())
set.seed(1234567)

#set the variance matrix, we do not have the x1 in the model so we need to remove something from the ma
```

```r
    evil_variance_matrix <- matrix(c(0.9381, -0.4184,
                                     -0.4184, 1.0228), nrow = 2, ncol = 2, byrow = TRUE)

#set up the mu (for question 3-b we do not have the x1)
    mu <- c(-0.6768, 3.3521)

    # c sequence (from -1 to 1 by 0.1)
    c_value <- seq(from = -1, to = 1, by = 0.1)

    # Number of simulation rounds
    R <- 1000

    # Parameters
    # the n is for the data generate process within the loop
    # the n_value of for the loop to keep going.
    n_value <- c(100, 250, 500, 1000)

    #set up the beta_ture
    beta_true <- c(-0.8, 0, 0.1)

    #the t-test with 0.05 degree of freedom (do not change to a you will lost it)
    alpha <- 0.05

    #final_matrix
    Final_matrix <- matrix(0,ncol = length(c_value), nrow = length(n_value))

    #the simulation time
    for (i in 1:length(n_value)){
      #rest the n
      n <- n_value[i]
      #reset the TF matrix
      TF_matrix <- matrix(NA, ncol=length(c_value), nrow = R)

      x <- mvrnorm(n, mu = mu, Sigma = evil_variance_matrix)

      #this loop run 1000 times with current value of n
      for (j in 1:R) {
        x <- mvrnorm(n, mu = mu, Sigma = evil_variance_matrix) # its should be "Sigma" not "sigma"
        # Add intercept (we now turn the x from 3x3 matrix to 3x4 matrix)
       x <- cbind(rep(1, n), x)
        # e need to be processed within the loop
        e <- rnorm(n, mean = 0, sd = 1)  #normal distribution
        #now its the time for the calculate the y!
        y <- x %*% beta_true + e

        # OLS estimation
        beta_hat <- solve(t(x) %*% x) %*% (t(x) %*% y)
        eps_hat <- y - x %*% beta_hat

        #the degree of freedom
        dof <- n - length(beta_hat)
        #  the mu
        sigma_hat_sq <- (1 / dof) * sum(eps_hat^2)
```

```r
    #estimate of the
    v_beta_hat <- solve(t(x) %*% x) * sigma_hat_sq
    # standard error time!
    diag_v_beta_hat <- diag(v_beta_hat)
    sd <- sqrt(diag_v_beta_hat)

    #the T- test time!
    for (k in 1:length(c_value)) {
      #T statistic for the beta2
      t_st <- (beta_hat[2,] - c_value[k])/sd[2]
      #T test P value
      t_p_value <- 1 - pt(abs(t_st), dof) #using the TA's note!
      #the matrix time!
      #count the p_value
      TF_matrix[j, k] <- t_p_value < 0.05
      #this things was crashed my mac for times
      #I spend 2 hours on this line a lone
    }
  }
  Final_matrix[i, ] <- colMeans(TF_matrix)
  #and other hour on this line.
}

#change the row name of the matrix
rownames(Final_matrix) <- c("n = 100", "n = 250", "n = 500", "n = 1000")


# Assuming Final_matrix and c_value are already defined

# Colors
dw_blue <- "#003b6f"  # Doctor Who blue
cr_red <- "#e91c25"   # Carlton red
ua_green <- "#007c41"  # ualberta green
ua_gold <- "#ffdb05"  # ualberta gold

# Prepare the data
n_labels <- c("n = 100", "n = 250", "n = 500", "n = 1000")
colors <- c(dw_blue, cr_red, ua_green, ua_gold)

# Plotting
plot(c_value, Final_matrix[1, ], type = "l", col = colors[1], ylim = c(min(Final_matrix), max(Final_
lines(c_value, Final_matrix[2, ], col = colors[2], lwd = 2)
lines(c_value, Final_matrix[3, ], col = colors[3], lwd = 2)
lines(c_value, Final_matrix[4, ], col = colors[4], lwd = 2)
legend("bottomleft", legend = n_labels, col = colors, lty = 1, lwd = 2)
```



3−B power curves for different sample size
proportion of rejections
0.2
0.4
0.6
0.8
1.0
n = 100
n = 250
n = 500
n = 1000
−1.0
−0.5
0.0
0.5
1.0
Value of c