

HW2_6337_2023

2023-02-26

Authors : Mankirat Singh Bhamra MXB220061 - Harikrishna Dev HXD220000 - Sarthak Vajpayee SXV220020 - Krishnan Venkatesan KXV220007

Loading required libraries and cleaning environment

```
rm(list = ls())  
demo = T  
require(psych)
```

```
## Loading required package: psych
```

```
require(data.table)
```

```
## Loading required package: data.table
```

```
require(dplyr)
```

```
## Loading required package: dplyr
```

```
##  
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:data.table':  
##  
##      between, first, last
```

```
## The following objects are masked from 'package:stats':  
##  
##      filter, lag
```

```
## The following objects are masked from 'package:base':  
##  
##      intersect, setdiff, setequal, union
```

```
require(ggplot2)
```

```
## Loading required package: ggplot2
```

```
##  
## Attaching package: 'ggplot2'
```

```
## The following objects are masked from 'package:psych':  
##  
##      %+%, alpha
```

```
require(rlist)
```

```
## Loading required package: rlist
```

```
if(demo) {setwd("~/Library/Mobile Documents/com~apple~CloudDocs/School Work/Sem 2/BUAN 6337/HW/pset3")}
```

1 (R) (Weak) Law of Large Numbers • (a) Consider a continuous random variable $X_i \sim \text{Uniform}[0, 2]$. What is $E[X_i]$ and $\text{Var}(X_i)$?

Ans: The mean (expected value) of a continuous uniform distribution $[a, b]$ is given by:

$$E[X] = \frac{(a + b)}{2}$$

In this case, $a = 0$ and $b = 2$, so we have:

$$E[X_i] = \frac{(0 + 2)}{2} = 1$$

Therefore, the expected value of X_i is 1.

The variance of a continuous uniform distribution $[a, b]$ is given by:

$$\text{Var}[X] = \frac{(b - a)^2}{12}$$

In this case, $a = 0$ and $b = 2$, so we have:

$$\text{Var}[X_i] = \frac{(2 - 0)^2}{12} = \frac{1}{3}$$

Therefore, the variance of X_i is $\frac{1}{3}$.

• (b) Consider the X_i defined above in (a) for $i = 1, 2, \dots, n$, where each $X_i \perp X_j$ whenever $i \neq j$. Consider the sample mean

What is $E[\bar{X}_n]$ and $\text{Var}(\bar{X}_n)$? (The answer might be a function of n).

Ans: The sample mean of n independent and identically distributed (i.i.d.) random variables X_i has the following properties:

$$E[\bar{X}_n] = E\left[\frac{1}{n} \sum X_i\right] = \frac{1}{n} \sum E[X_i] = \frac{1}{n} \times n \times E[X_i] = E[X_i]$$

$$\text{Var}[\bar{X}_n] = \text{Var}\left[\frac{1}{n} \sum X_i\right] = \frac{1}{n^2} \sum \text{Var}[X_i]$$

Since each X_i is uniformly distributed on $[0, 2]$, we know from part (a) that $E[X_i] = 1$ and $\text{Var}[X_i] = \frac{1}{3}$. Therefore:

$$E[\bar{X}_n] = E[X_i] = 1$$

$$\text{Var}[\bar{X}_n] = \text{Var}\left[\frac{1}{n} \sum X_i\right] = \frac{1}{n^2} \sum \text{Var}[X_i] = \frac{1}{n^2} \times n \times \frac{1}{3} = \frac{1}{3n}$$

Therefore, the expected value of the sample mean \bar{X}_n is 1, and the variance of the sample mean \bar{X}_n is $\frac{1}{3n}$.

Now repeat the following (c)-(e) for $n = 1, 2, 3, 5, 10, 50, 100, 1000, 3000$. Use the for loops to execute (c)-(e). • (c) (R) Generate a size n vector of independent Uniform $[0, 2]$ random variables and calculate its sample mean \bar{X}_n . • (d) (R) Take $|\bar{X}_n - E[X_i]|$ and report the value.

• (e) (R) Now consider a continuous transformation $f(x) = 2x^2 - 5x + 1 + \frac{1}{(3x)}$. Take $|f(\bar{X}_n) - f(E[X_i])|$

```
n = c(1, 2, 3, 5, 10, 50, 100, 1000, 3000)

Q1 = function(n){

  # c
  x = runif(n,0,2)
  x_bar = mean(x)

  #d
  e_x = 1
  diff = abs(x_bar - e_x)

  #e
  quad = function(x)
  {
    return (2*x^2 - 5*x + 1 + (1/(3*x)))
  }

  diff_function = abs(quad(x_bar)-quad(e_x))
  return(list(n=n,mean=x_bar,diff=diff,diff_func=diff_function))
}

for(i in n){
  assign(paste0("v",i),Q1(i))
  if(demo){print(get(paste0("v",i)))}
}
```

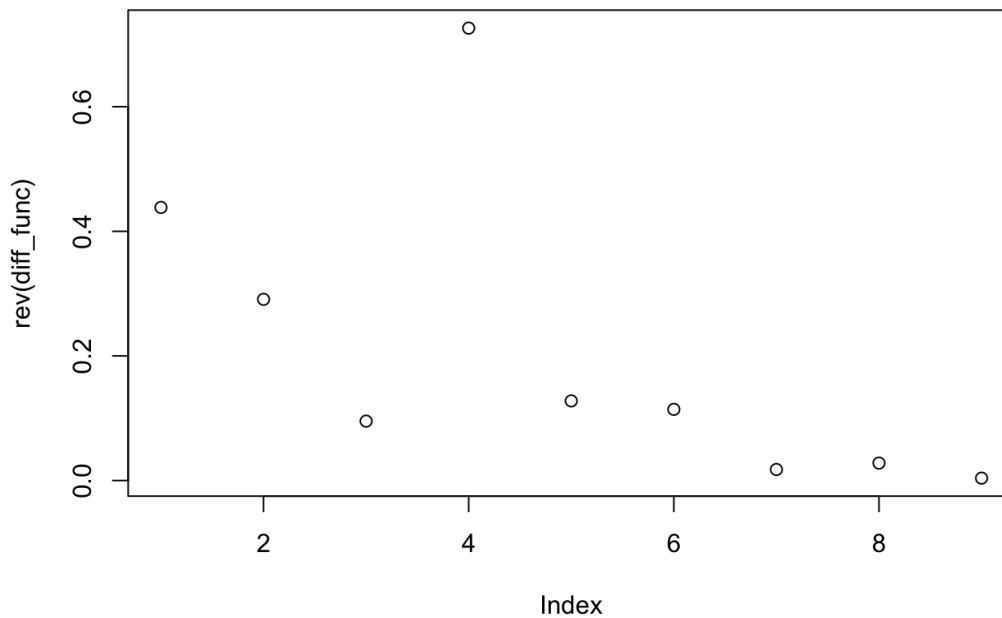
```
## $n
## [1] 1
##
## $mean
## [1] 1.37552
##
## $diff
## [1] 0.3755196
##
## $diff_func
## [1] 0.1844903
##
## $n
## [1] 2
##
## $mean
## [1] 0.6020923
##
## $diff
## [1] 0.3979077
##
## $diff_func
## [1] 0.9348605
##
## $n
## [1] 3
##
## $mean
## [1] 0.7891113
##
## $diff
## [1] 0.2108887
##
## $diff_func
## [1] 0.3889197
##
## $n
## [1] 5
##
## $mean
## [1] 0.8947418
##
## $diff
## [1] 0.1052582
##
## $diff_func
## [1] 0.1666305
##
## $n
## [1] 10
##
## $mean
## [1] 0.9404339
##
## $diff
## [1] 0.05956612
##
## $diff_func
## [1] 0.08777536
##
## $n
## [1] 50
##
## $mean
## [1] 0.964875
##
## $diff
## [1] 0.03512498
##
```

```
## $diff_func
## [1] 0.04972706
##
## $n
## [1] 100
##
## $mean
## [1] 1.188701
##
## $diff
## [1] 0.1887006
##
## $diff_func
## [1] 0.1703998
##
## $n
## [1] 1000
##
## $mean
## [1] 0.9826246
##
## $diff
## [1] 0.01737543
##
## $diff_func
## [1] 0.02387346
##
## $n
## [1] 3000
##
## $mean
## [1] 0.988527
##
## $diff
## [1] 0.01147301
##
## $diff_func
## [1] 0.01560499
```

- (f) What happens to the reported value in (d) and (e) as n increases? Discuss.

Ans: As n increases, the difference between the theoretical value and practical value tends towards 0. This implies larger the sample, the results are better statistically.

```
diff_func = c()
for(i in n){
  diff_func = append((Q1(i)$diff_func),diff_func)
}
plot(rev(diff_func))
```



2 (R) The Central Limit Theorem • (a) Consider a continuous random variable $(X_i \sim \text{Uniform}[0, 2])$. What is $(E[X_i])$ and $(\text{Var}(X_i))$?

$$\text{Mean} = \frac{a+b}{2} = 1 \quad \text{Var} = \frac{1}{3}$$

• (b) Consider the X_i defined above in (a) for $i = 1, 2, \dots, n$, where each $(X_i \perp X_j)$ whenever $(i \neq j)$. Consider the sample mean $\bar{X}_n := \frac{1}{n} \sum_{i=1}^n X_i$. What is $(E[\bar{X}_n])$ and $(\text{Var}(\bar{X}_n))$? (The answer would be a function of n).

Ans:

$$E[\bar{X}_n] = \frac{1}{n} \times n \times 1 = 1 \quad E[\bar{X}_{n=6}] = 1$$

$$\text{Var}[\bar{X}_n] = \frac{1}{\sqrt{n}} \times n \times 0.3333 = 0.041667$$

• (c) Consider the transformation $Y_n := \sqrt{n} (\bar{X}_n - E[X_i])$. What is $E[Y_n]$ and $\text{Var}(Y_n)$?

Ans:

$$E[Y_n] = \sqrt{n} \times 0 = 0$$

$$\text{var} = \frac{1}{3}$$

• (d) Consider the transformation $Z_n := \sqrt{n} (\bar{X}_n - E[X_i])$. What is $E[Z_n]$ and $\text{Var}(Z_n)$?

$$E[Z_n] = 0$$

$$\text{Var}[Z_n] = 1$$

Now repeat the following for $n = 1, 2, 3, 5, 10, 50, 100, 1000, 3000$. Use the for loops to execute (e)-(m). • (e)(R) Generate $t = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, \dots, 2500$ $n \times 1$ vectors of independent $(\text{Uniform}[0, 2])$ random variables and calculate its sample mean (\bar{X}_n) respectively for each t . Denote this size 2500 vector as $v_{2500} \equiv X_1, X_2, \dots, X_{2500}$ for now.

```

n = c(1, 2, 3, 5, 10, 50, 100, 1000, 3000)

Q2 = function(n){

  t = c(1:2500)
  v = replicate(length(t), runif(i, 0, 2))
  if (n == 1) {
    vn_2500 = v
  } else {
    vn_2500 = apply(v, 2, mean)
  }

  vn_2500_bar = mean(vn_2500)
  vn_2500_var = var(vn_2500)

  e_xi = 1
  yn_2500 = sqrt(n)*(vn_2500 - e_xi)

  mean_yn_2500 = mean(yn_2500)
  var_yn_2500 = var(yn_2500)

  var_xi = 1/3
  zn_2500 = sqrt(n)*(vn_2500 - e_xi)/sqrt(var_xi)

  mean_zn_2500 = mean(zn_2500)
  var_zn_2500 = var(zn_2500)

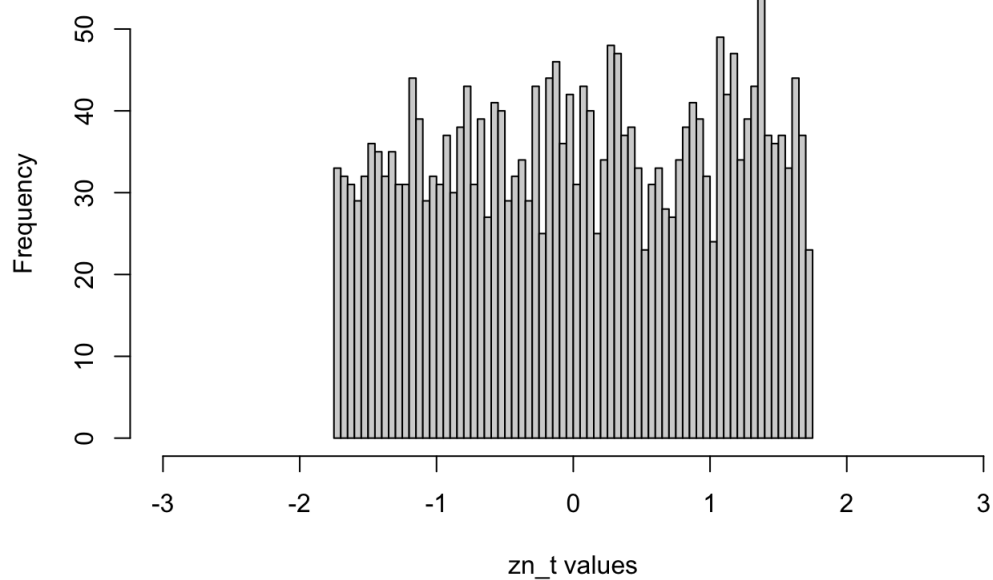
  return(list(n=n, vn = vn_2500, mean_vn = vn_2500_bar, var_vn = vn_2500_var, zn = zn_2500, mean_zn = mean_zn_2500, var_zn = var_zn_2500))
}

for(i in n){
  assign(paste0("f",i),Q2(i))
}

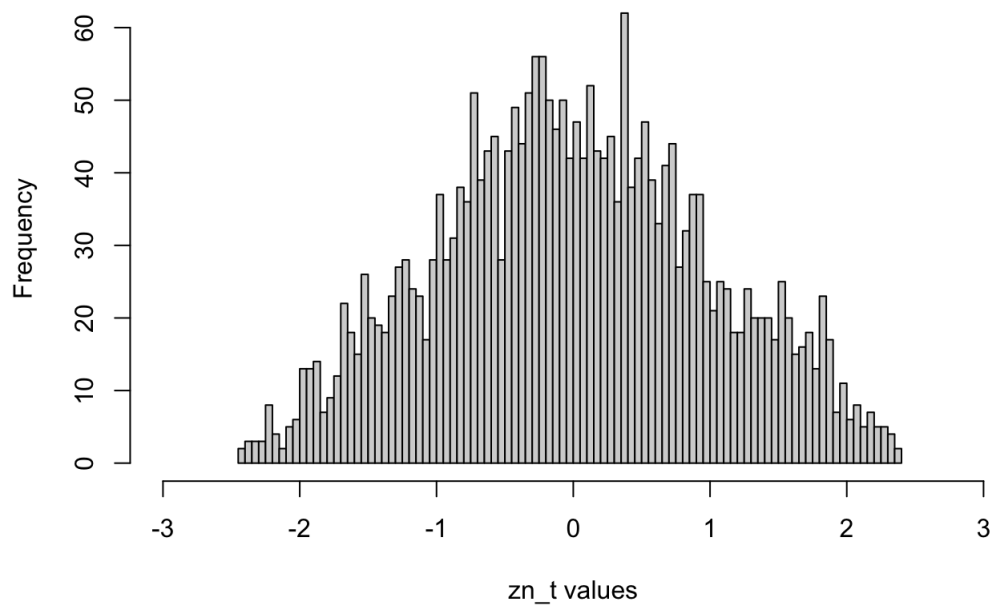
for(i in n){
  hist(Q2(i)$zn, breaks = 100,
       main = paste("Histogram of zn_2500 for n = ", i, sep = ""),
       xlab = "zn_t values",
       xlim = c(-3,3))
}

```

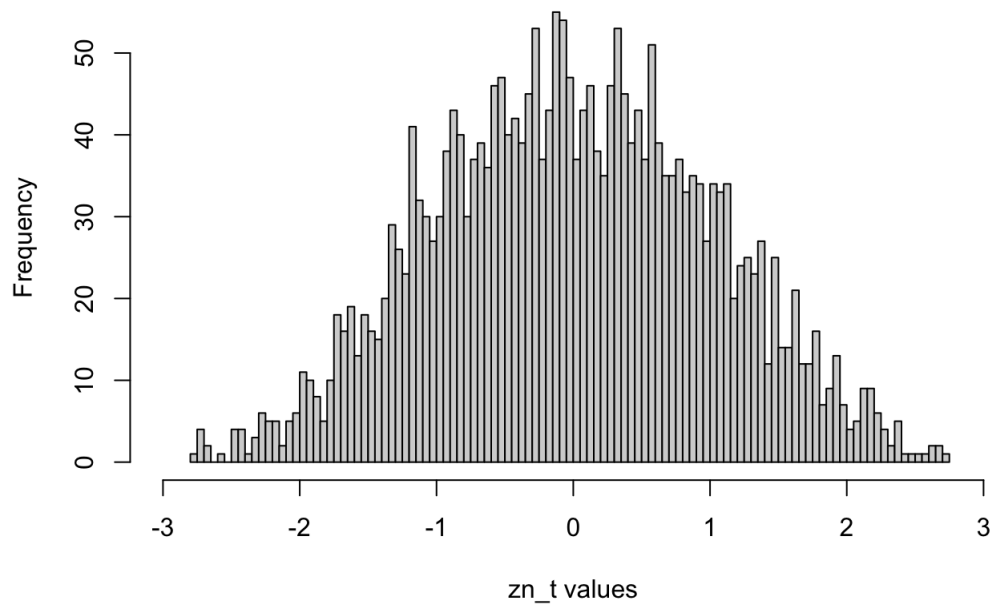
Histogram of zn_2500 for n = 1



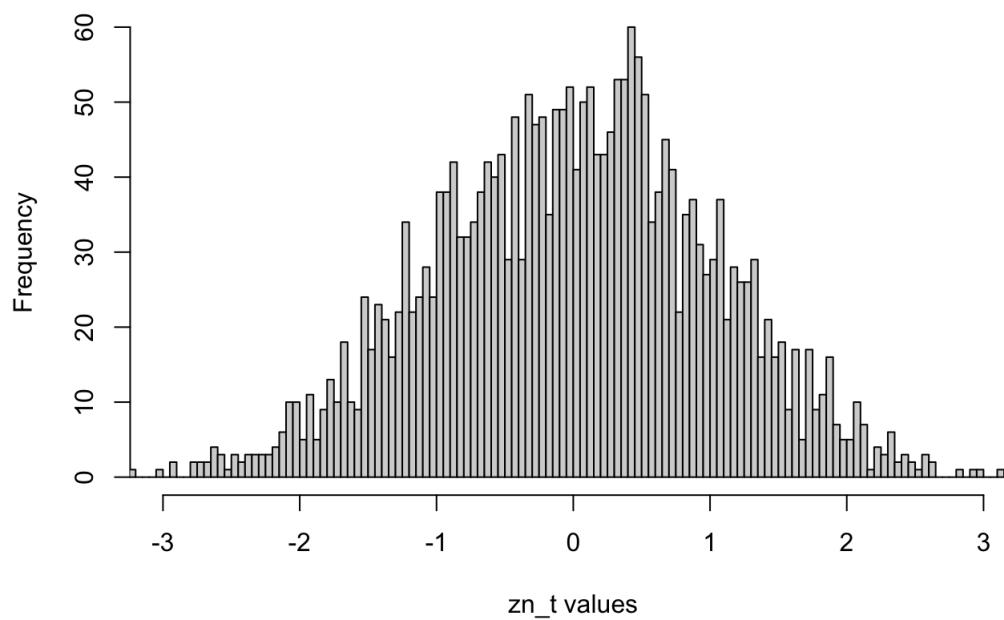
Histogram of zn_2500 for n = 2



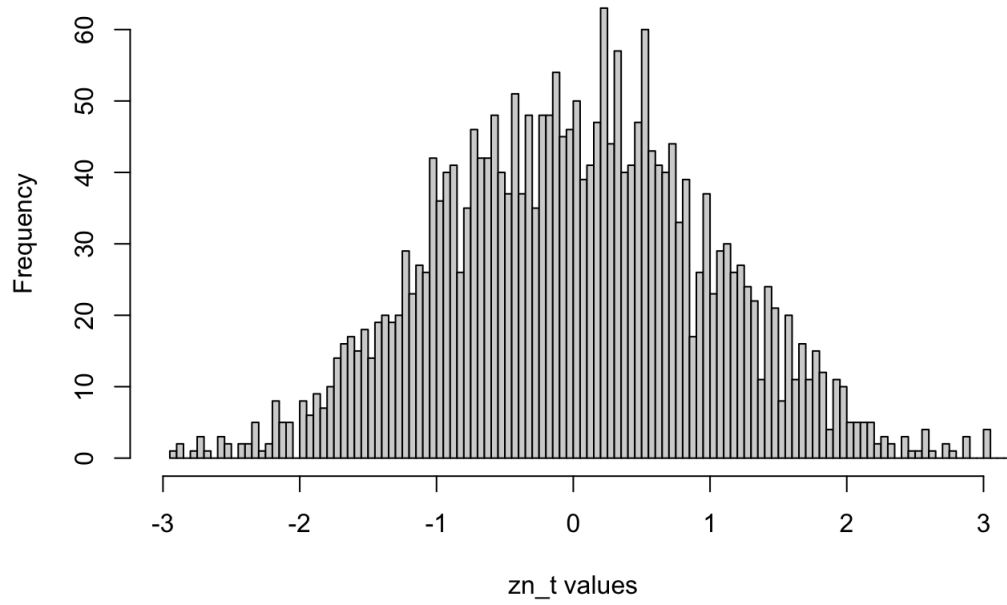
Histogram of zn_2500 for n = 3



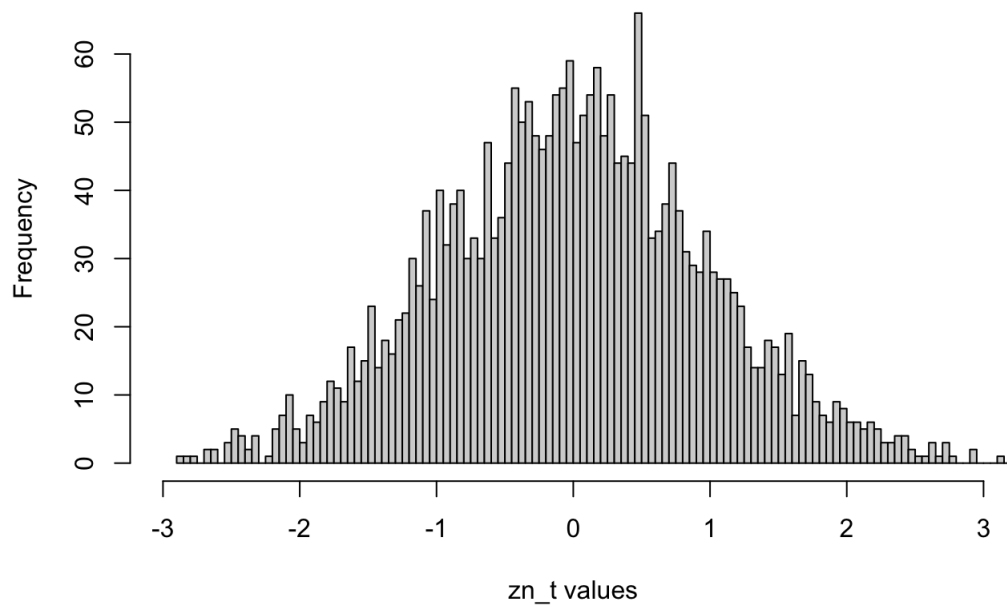
Histogram of zn_2500 for n = 5



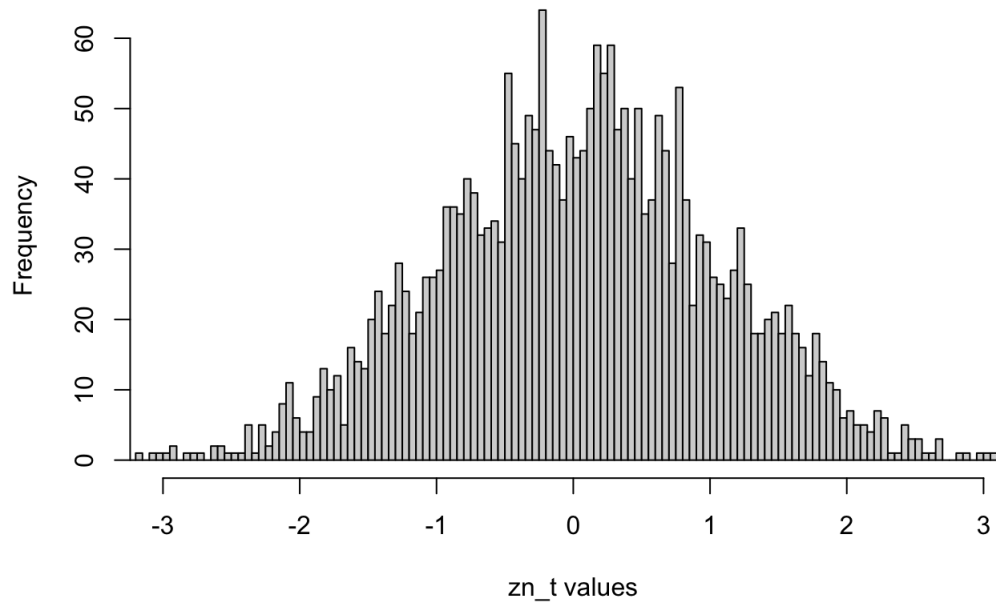
Histogram of zn_2500 for n = 10



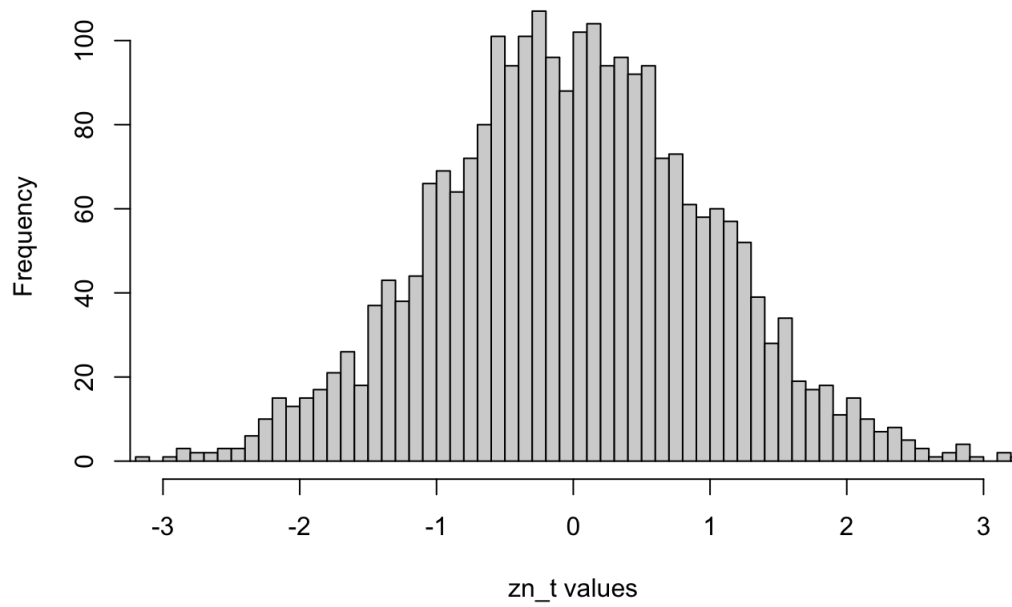
Histogram of zn_2500 for n = 50



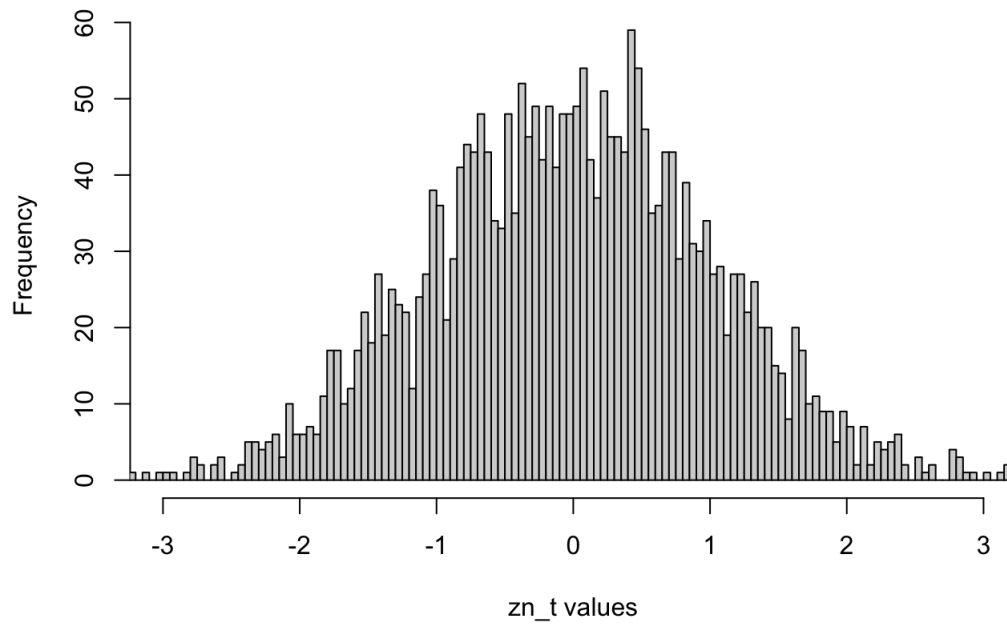
Histogram of zn_2500 for n = 100



Histogram of zn_2500 for n = 1000

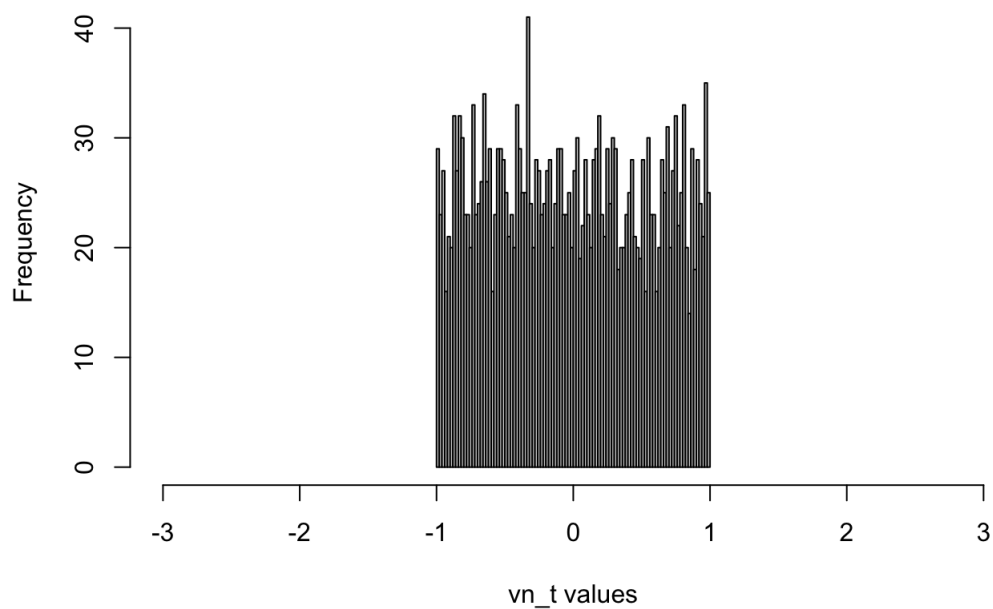


Histogram of zn_2500 for n = 3000

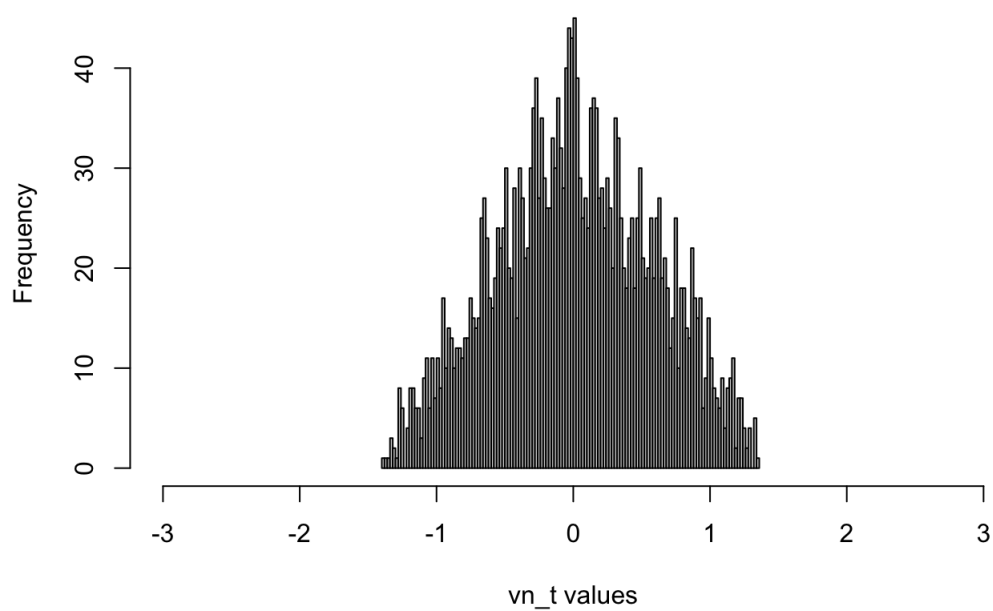


```
for(i in n){  
  hist(Q2(i)$yn, breaks = 100,  
    main = paste("Histogram of zn_2500 for n = ", i, sep = ""),  
    xlab = "zn_t values",  
    xlim = c(-3,3))  
}
```

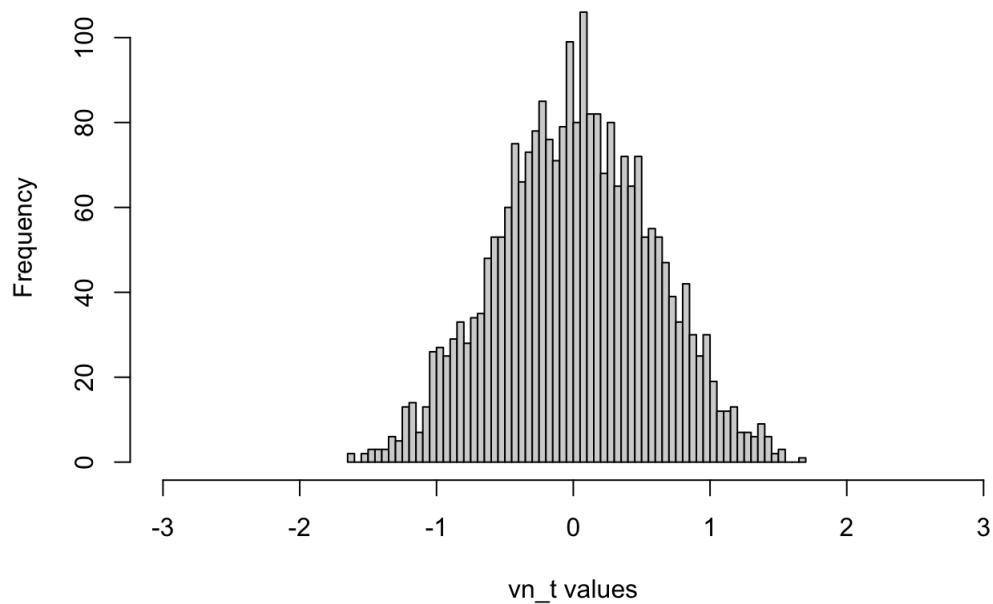
Histogram of zn_2500 for n = 1



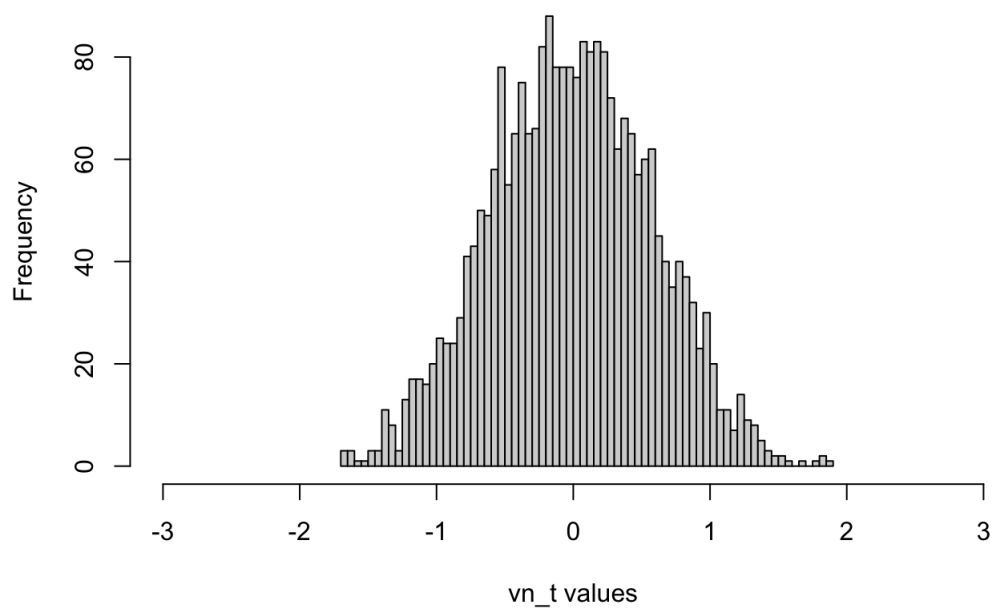
Histogram of zn_2500 for n = 2



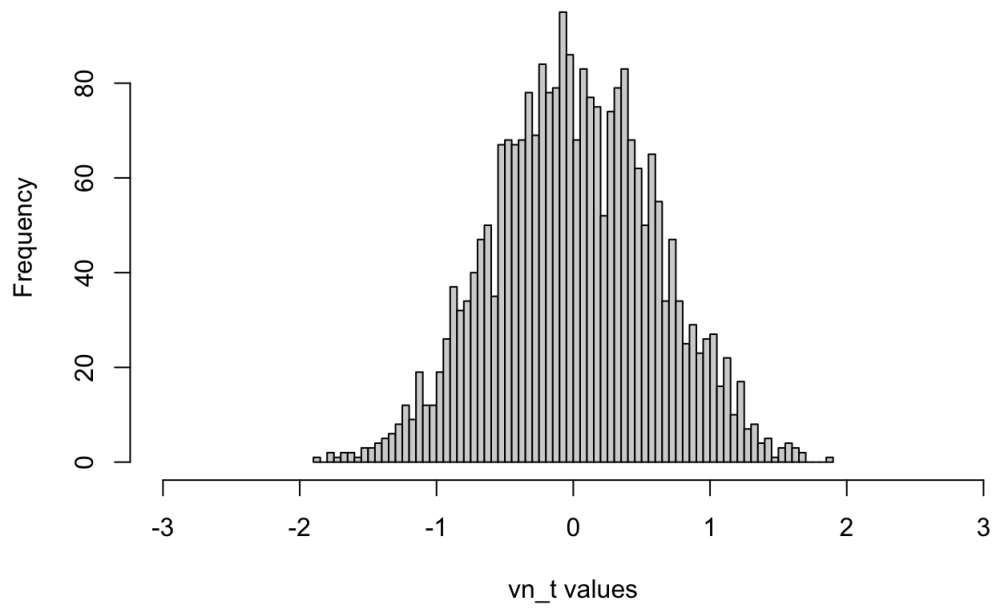
Histogram of zn_2500 for n = 3



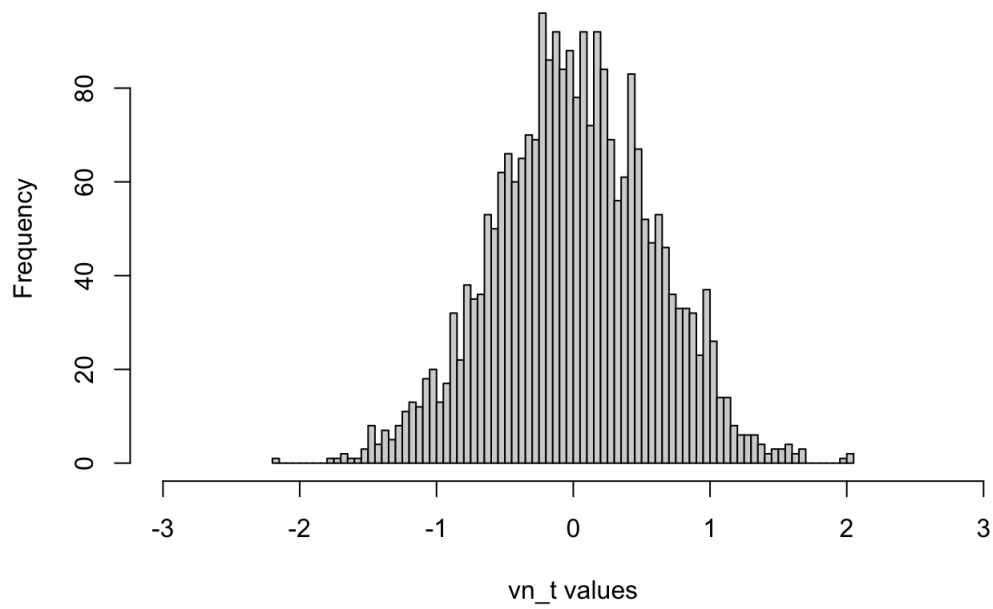
Histogram of zn_2500 for n = 5



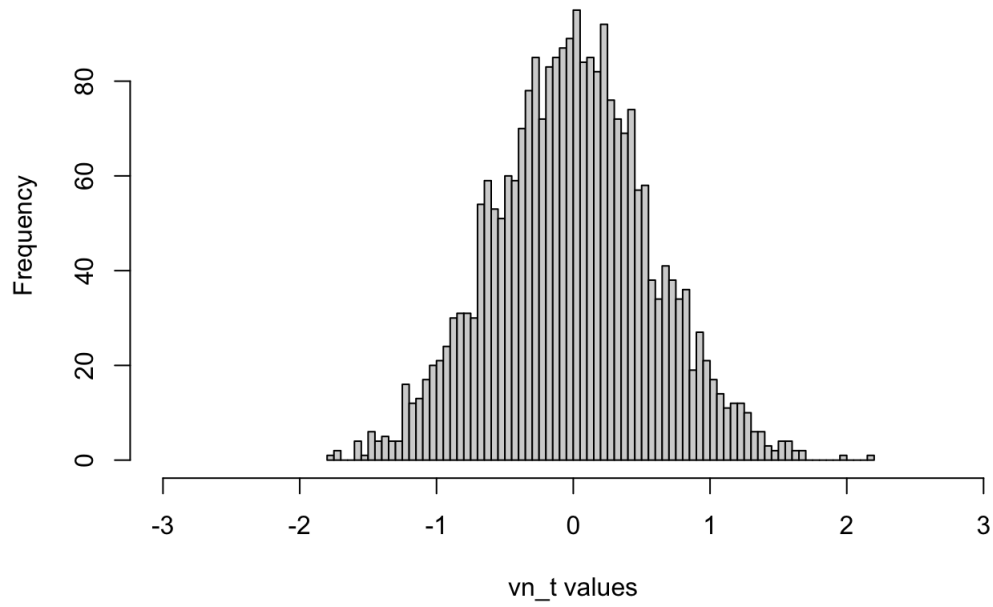
Histogram of zn_2500 for n = 10



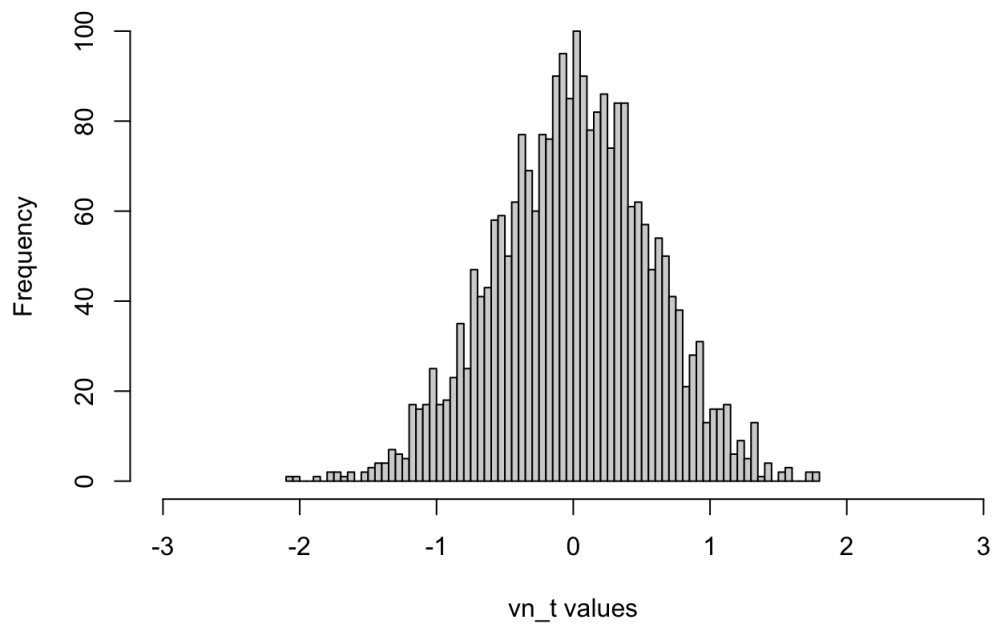
Histogram of zn_2500 for n = 50



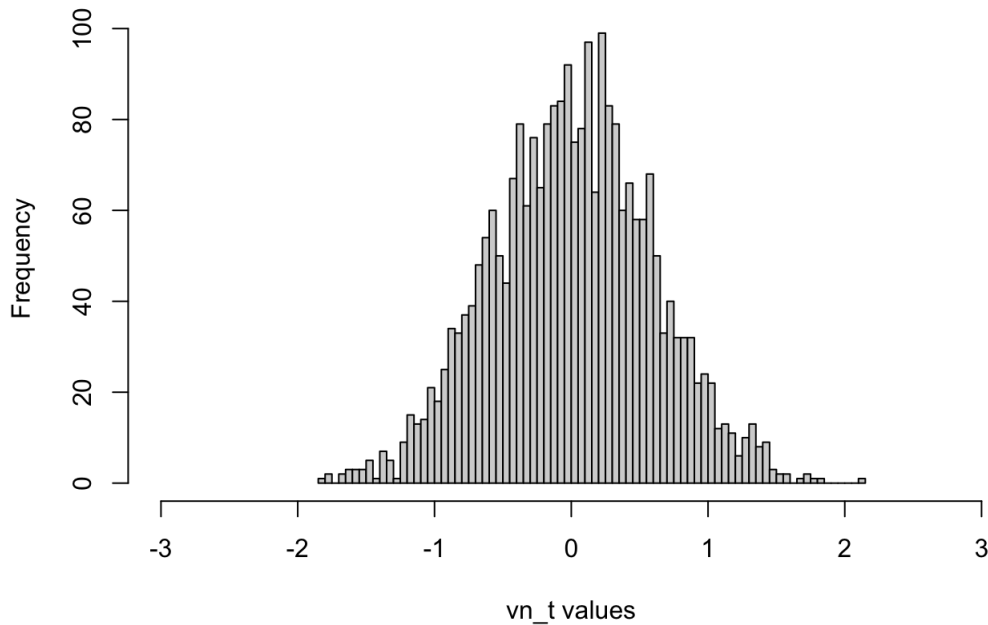
Histogram of zn_2500 for n = 100



Histogram of zn_2500 for n = 1000



Histogram of zn_2500 for n = 3000



- (m) What happens to the reported values in (f), (h), (k) and histograms in (i) and (l) as n increases? Discuss.

3 (R) WLLN with Simple Regression

In this exercise, you will generate datasets for simple regression yourself, and then try to estimate the model parameters to examine the properties of simple regression OLS estimators as the sample size n grows. Repeat the following for $n = 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20, 30, 50, 75, 100, 250, 500, 1000, 2000, 3000$. Use the for loops as necessary.

- (a) (R) Generate a $n \times 1$ column vector of Uniform[0, 12] random variable and denote it as x .
- (b) (R) Generate a $n \times 1$ column vector of Uniform $[-4, 4]$ random variable and denote it as u .
- (c) (R) Generate the y vector using the following formula: $y_i = 3 + 2x_i + u_i$ for each $i = 1, 2, 3, \dots, n$. That is, i 'th row of x and u corresponds to i 'th observation.
- (d) (R) Now you have a Monte-Carlo dataset of size n . Estimate the β in the following model using OLS. (Recall the formula d.) What is the calculated value of $\beta_{OLS,n}$? Report.

```
n = c(2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20, 30, 50, 75, 100, 250, 500, 1000, 2000, 3000)

Q3 <- function(n) {

  x = matrix(runif(n, 0, 12), nrow = n, ncol = 1)

  u = matrix(runif(n, -4, 4), nrow = n, ncol = 1)

  y = 3 + (2 * x) + u

  cov_xy = cov(x, y)
  var_x = var(x)
  hat_beta = cov_xy/var_x

  abs_beta = abs(hat_beta - 2)

  return(list(n=n,x=x,u=u,y=y,cov_xy=cov_xy,var_x=var_x,hat_beta=hat_beta,abs_beta=abs_beta))
}

for(i in n){
  assign(paste0("x",i),Q3(i))
}
```

4 In this exercise, you will generate datasets for simple regression yourself, and then try to estimate the model parameters to examine the properties of simple regression OLS estimators as the sample size n grows. Repeat the following for $n = 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20, 30, 50, 75, 100, 250, 500, 1000, 2000, 3000$. Use the for loops as necessary.

- (a) (R) Generate a $n \times 1$ column vector of Uniform $[0, 12]$ random variable and denote it as x .
- (b) (R) Generate a $n \times 1$ column vector of Uniform $[-4, 4]$ random variable and denote it as u .
- (c) (R) Generate the y vector using the following formula: $y_i = 3 + 2x_i + u_i$ for each $i = 1, 2, 3, \dots, n$. That is, i 'th row of x and u corresponds to i 'th observation.
- (d) (R) Now you have a Monte-Carlo dataset of size n . Estimate the β in the following model $y_i = \alpha + \beta x_i + u_i$
- (e) (R) Repeat (a)-(d) for 2,500 times. You must have 2,500 $\hat{\beta}_{OLS,n}$ estimates in the memory $\hat{\beta}^{(1)} \dots \hat{\beta}^{(2500)}$ at the end of this sub-question. Denote this size 2,500 vector by $b = \hat{\beta}_{OLS,n}^{(1)}, \hat{\beta}_{OLS,n}^{(2)}, \dots, \hat{\beta}_{OLS,n}^{(2500)}$.
- (f) (R) Calculate the variance of b and report.
- (g) (R) Subtract 2 from b and multiply $(c_1, c_2, \dots, c_{2500})$, i.e., n on each element of b and denote this as $c_i = \sqrt{n} (\hat{\beta}_{OLS,n}^{(i)} - 2)$ using OLS. Save it in the memory.

```
n = c(2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20, 30, 50, 75, 100, 250, 500, 1000, 2000, 3000)
```

```
Q4 <- function(n){
  b <- c()
  for (j in c(1:2500)) {

    # a
    x = matrix(runif(n, 0, 12), nrow = n, ncol = 1)

    # b
    u = matrix(runif(n, -4, 4), nrow = n, ncol = 1)

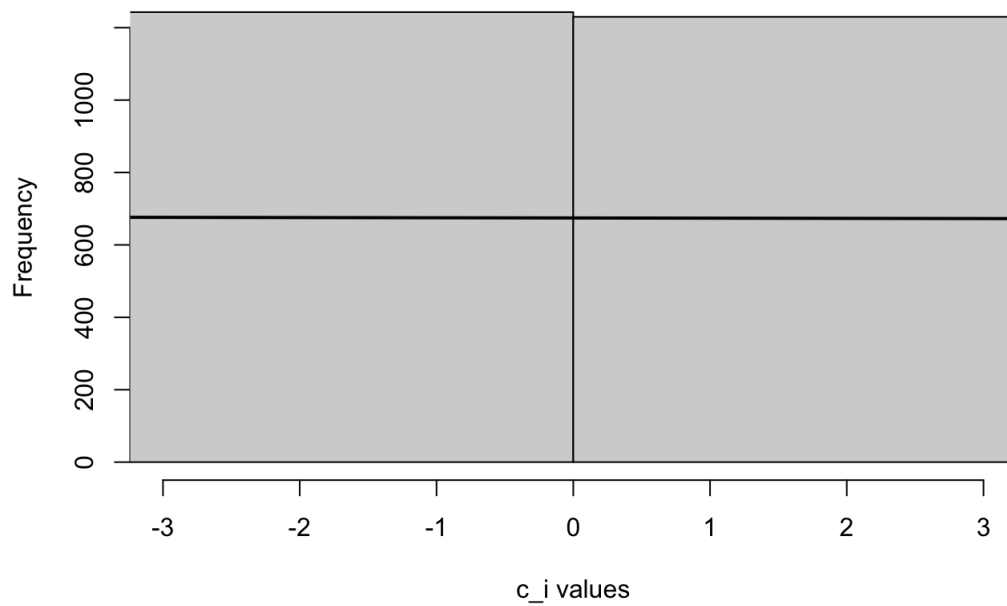
    # c
    y = 3 + (2 * x) + u

    # d
    cov_xy = cov(x, y)
    var_x = var(x)
    beta_hat = cov_xy/var_x
    b = append(b, beta_hat)
  }
  # f
  var_b = var(b)

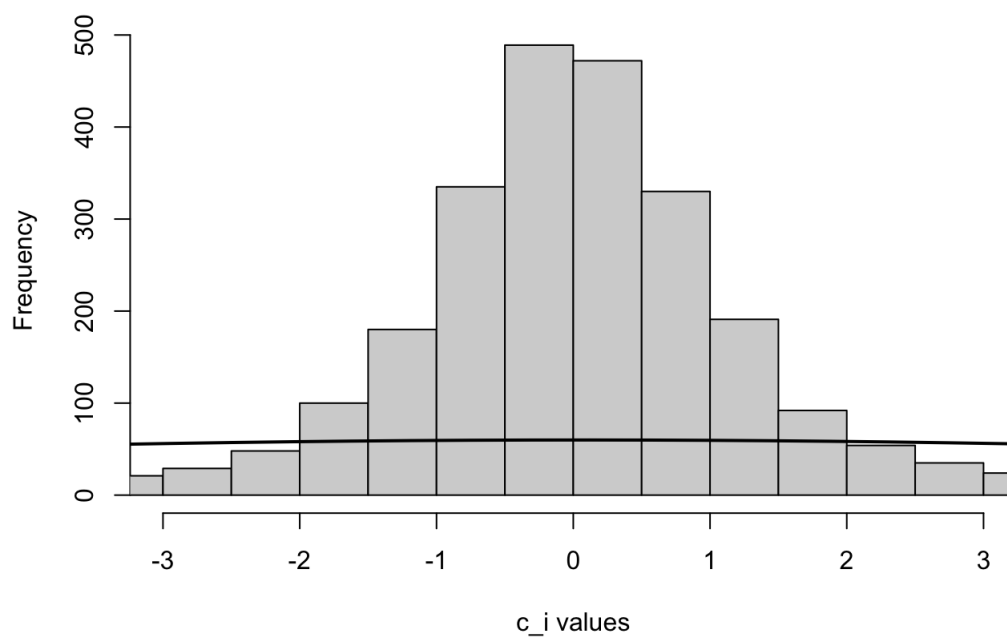
  c = sqrt(n)*(b - 2)
  return(list(n=n,b=b,var_b=var_b,c=c))
}

for(i in n){
  assign(paste0("q",i),Q4(i))
  hist_data <- hist(Q4(i)$c, breaks = 100,
    main = paste("Histogram of c for n = ", i, sep = ""),
    xlab = "c_i values",
    xlim = c(-3,3))
  x_values <- seq(min(Q4(i)$c), max(Q4(i)$c), length = 100)
  y_values <- dnorm(x_values, mean = mean(Q4(i)$c), sd = sd(Q4(i)$c))
  y_values <- y_values * diff(hist_data$mids[1:2]) * length(Q4(i)$c)
  lines(x_values, y_values, lwd = 2)
}
```

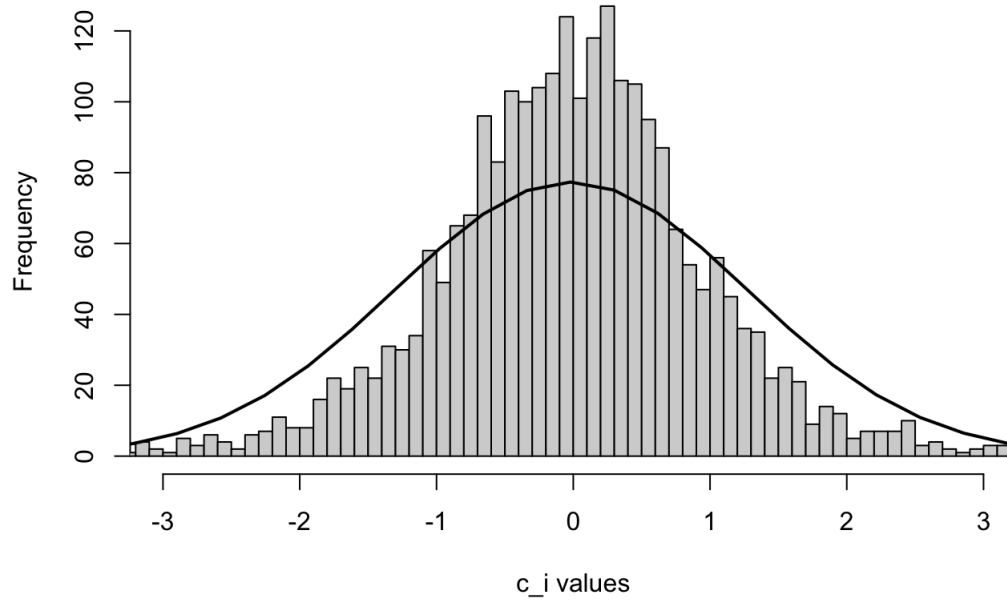
Histogram of c for n = 2



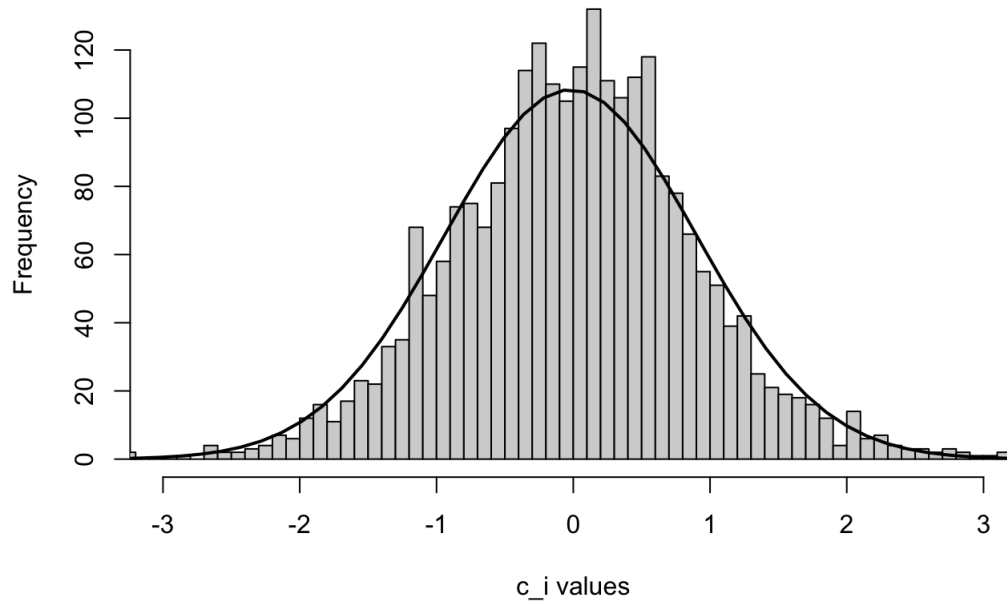
Histogram of c for n = 3



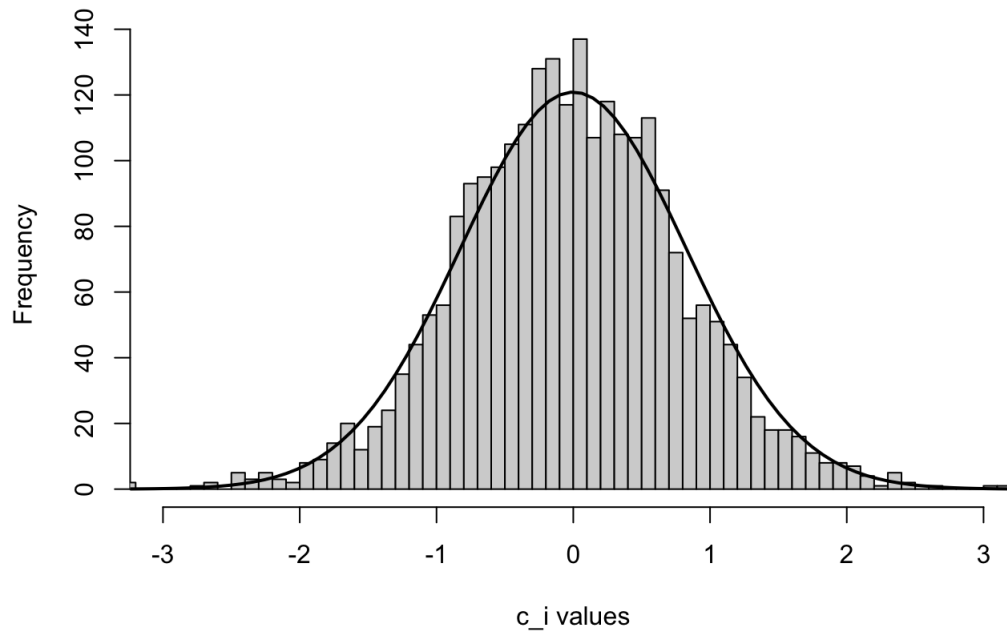
Histogram of c for $n = 4$



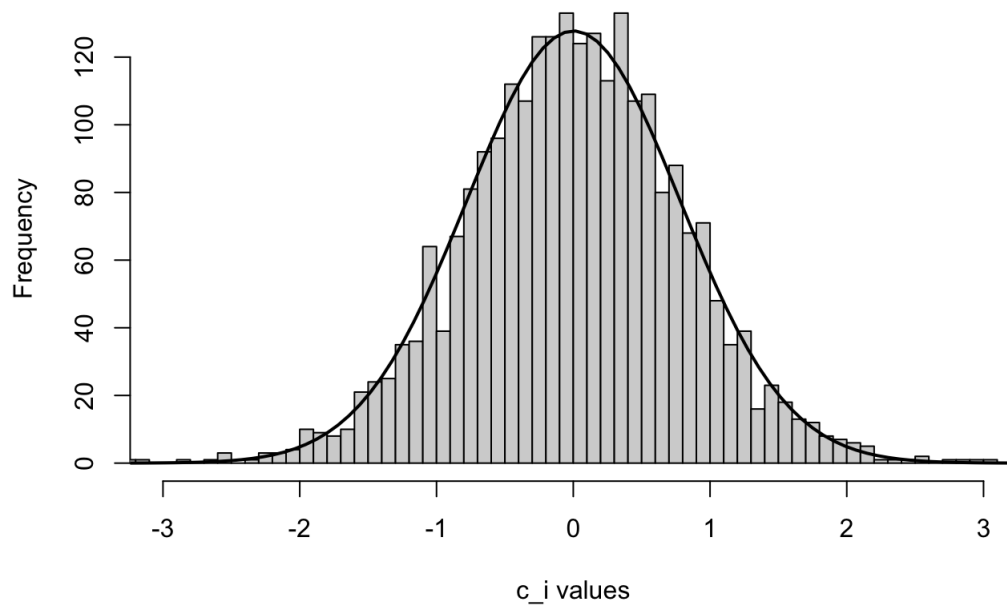
Histogram of c for $n = 5$



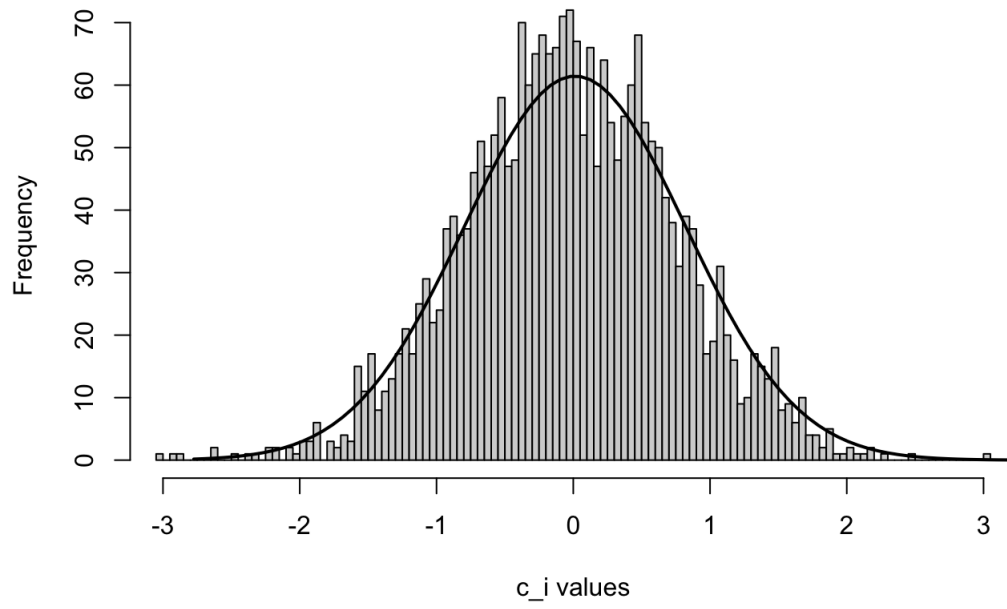
Histogram of c for n = 6



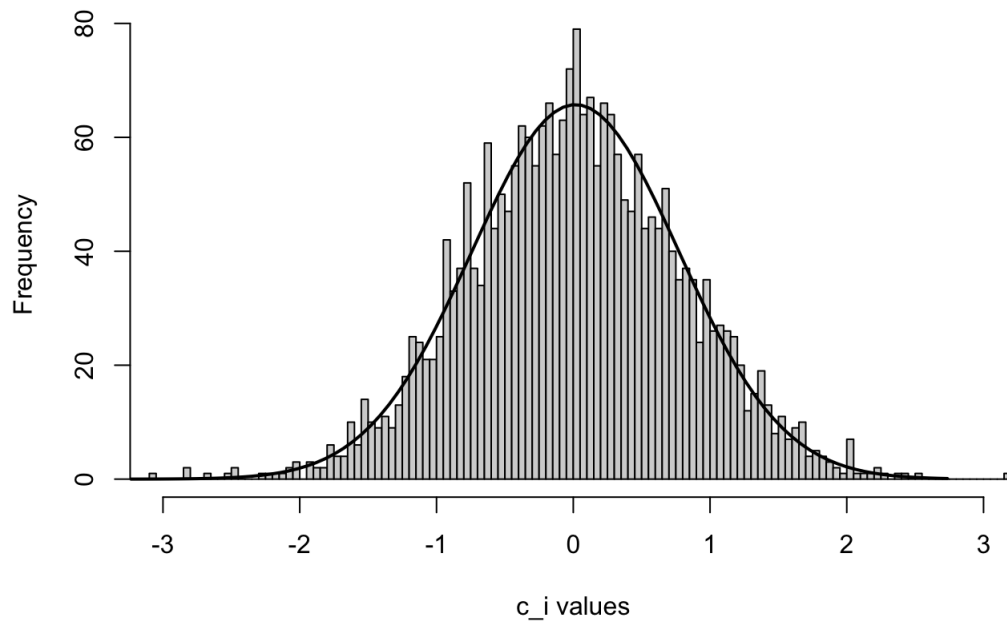
Histogram of c for n = 7



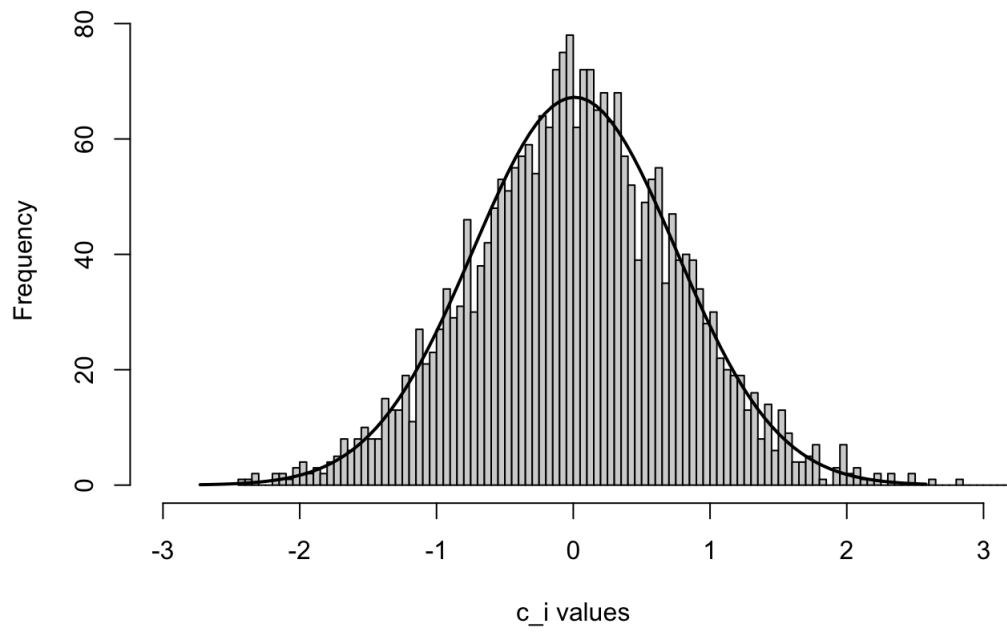
Histogram of c for $n = 8$



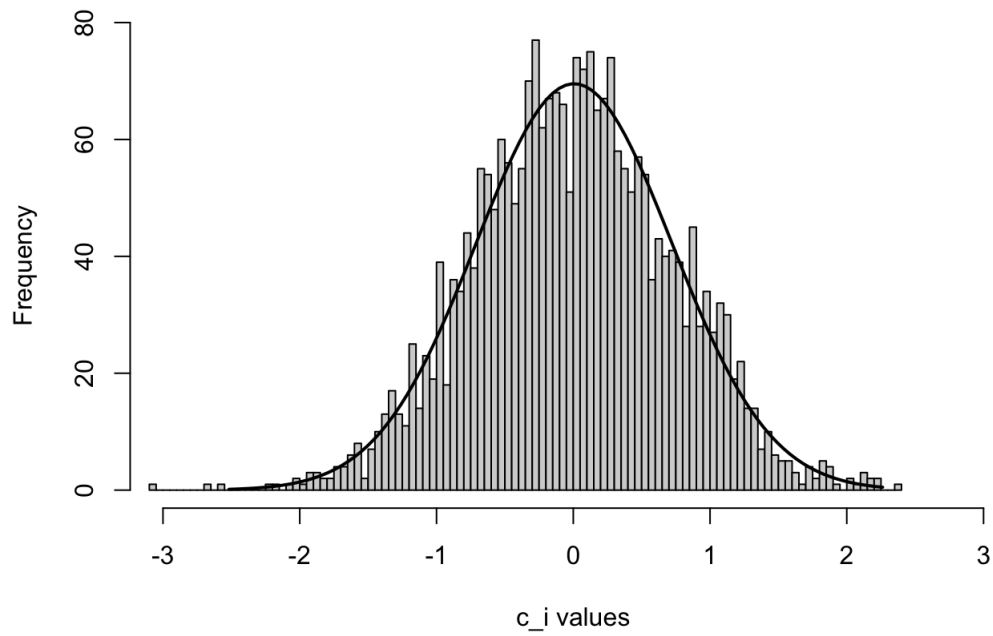
Histogram of c for $n = 9$



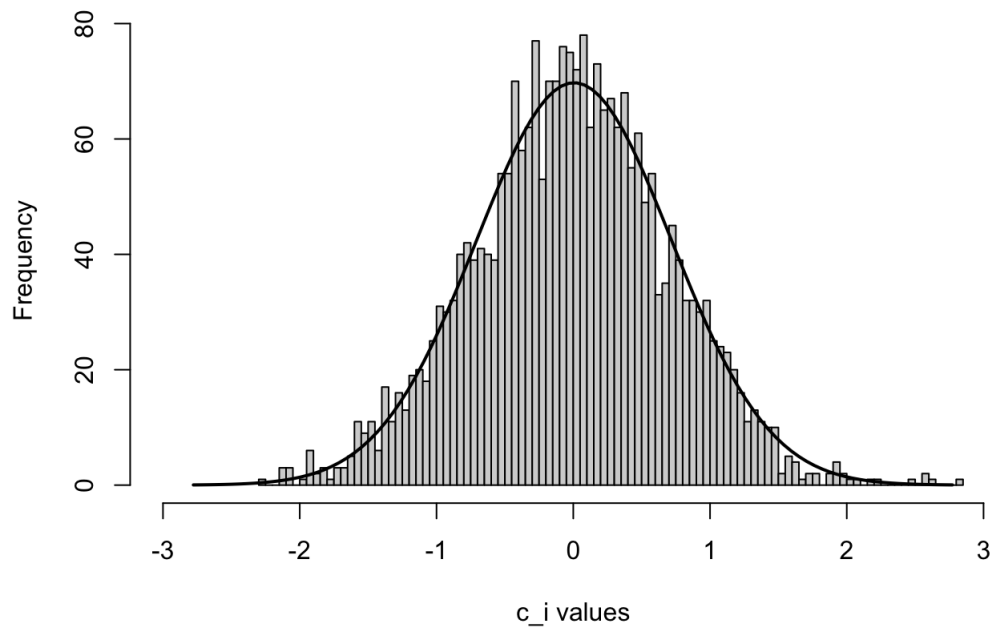
Histogram of c for $n = 10$



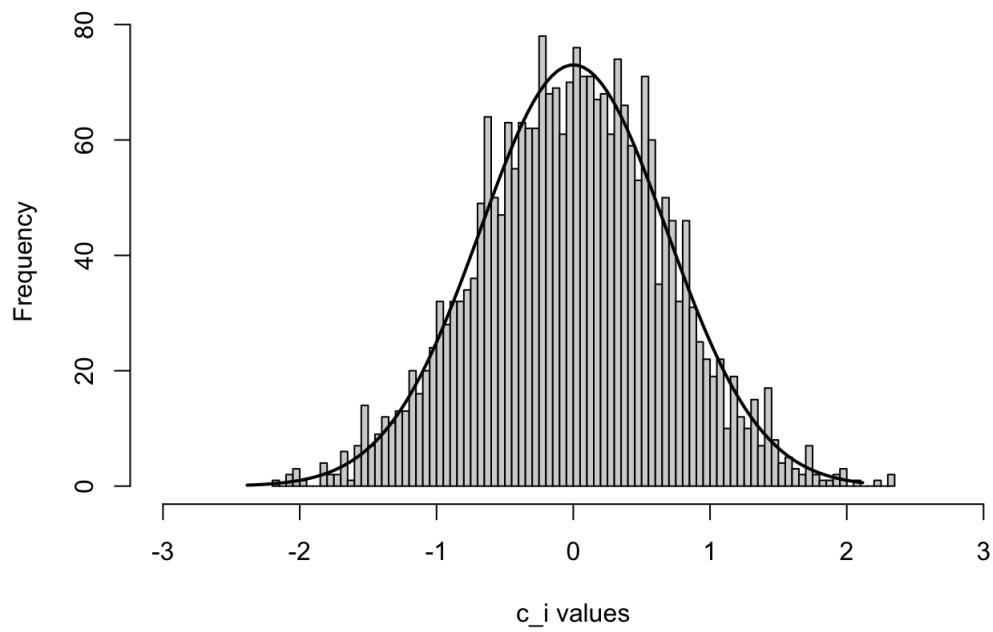
Histogram of c for $n = 15$



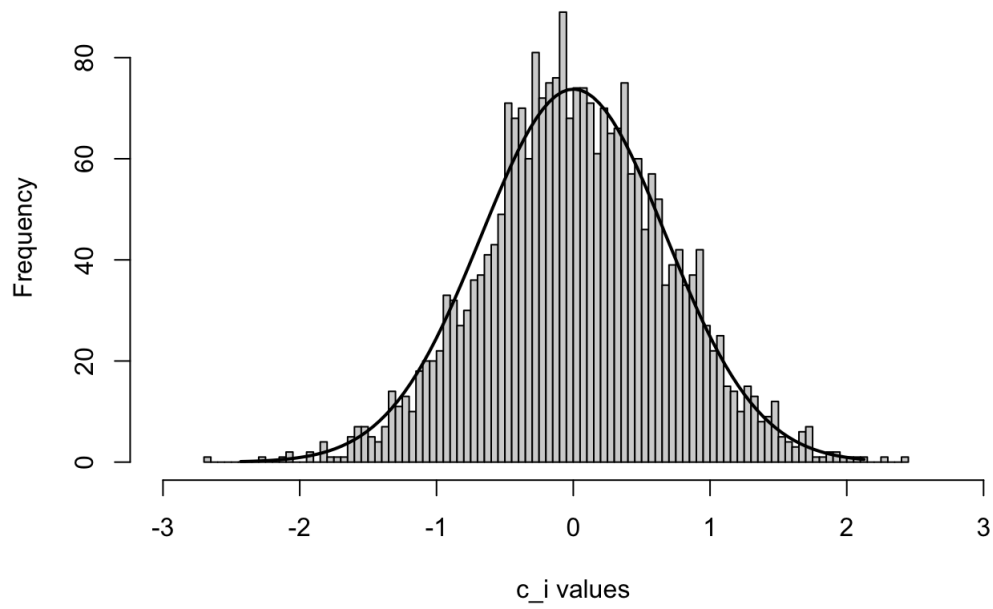
Histogram of c for $n = 20$



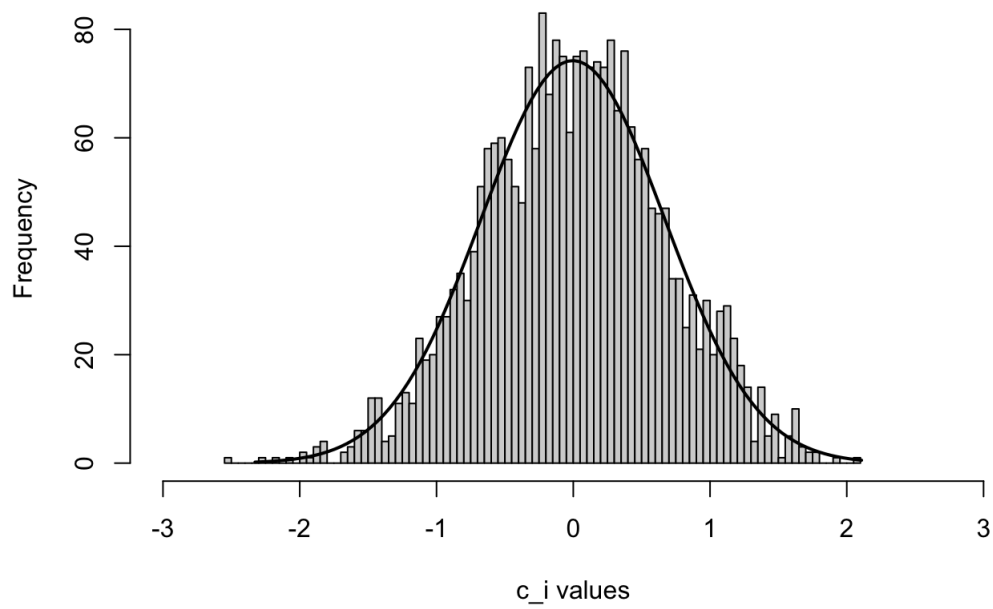
Histogram of c for $n = 30$



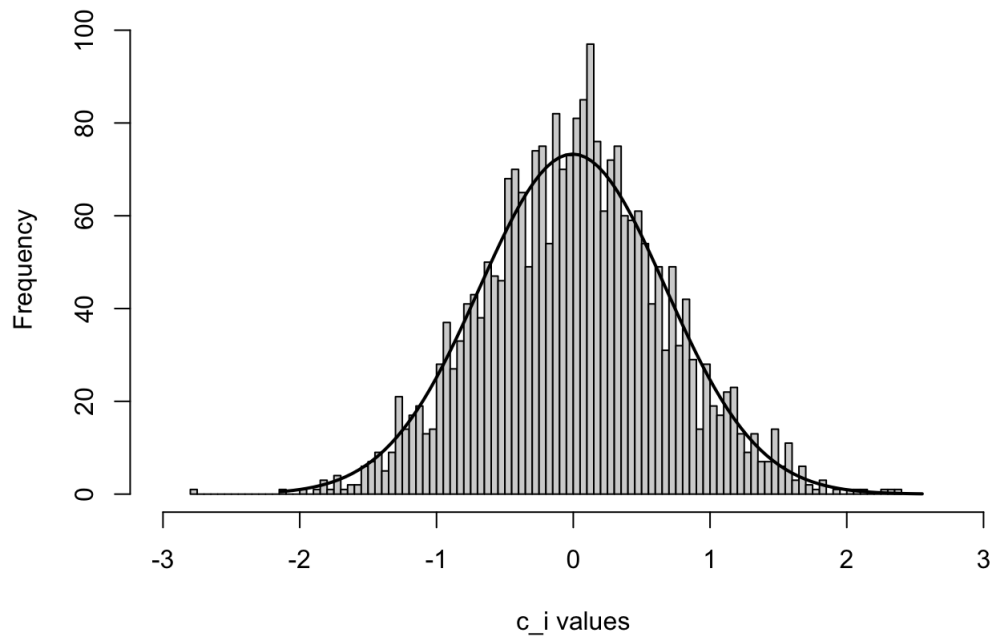
Histogram of c for n = 50



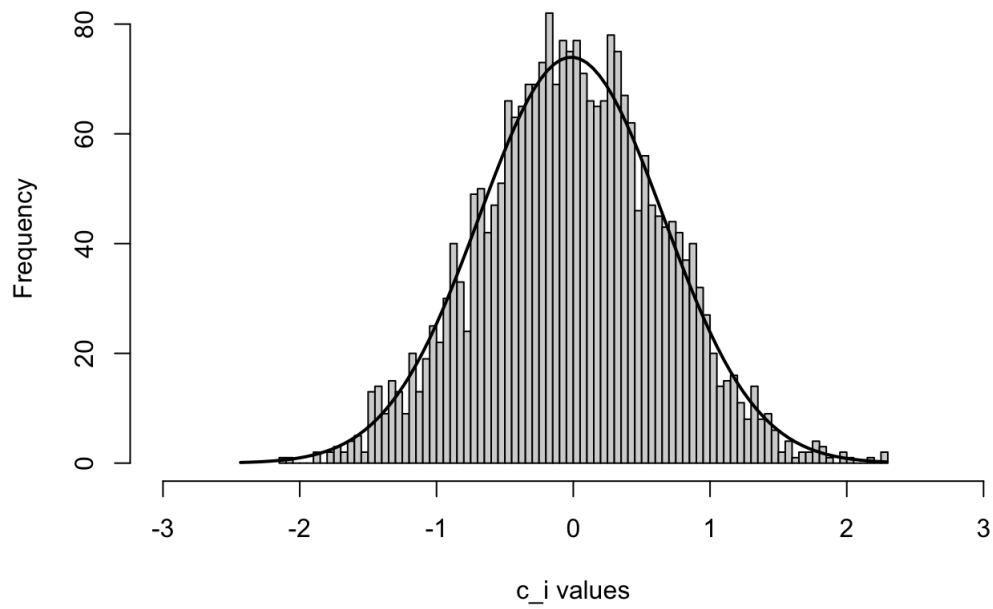
Histogram of c for n = 75



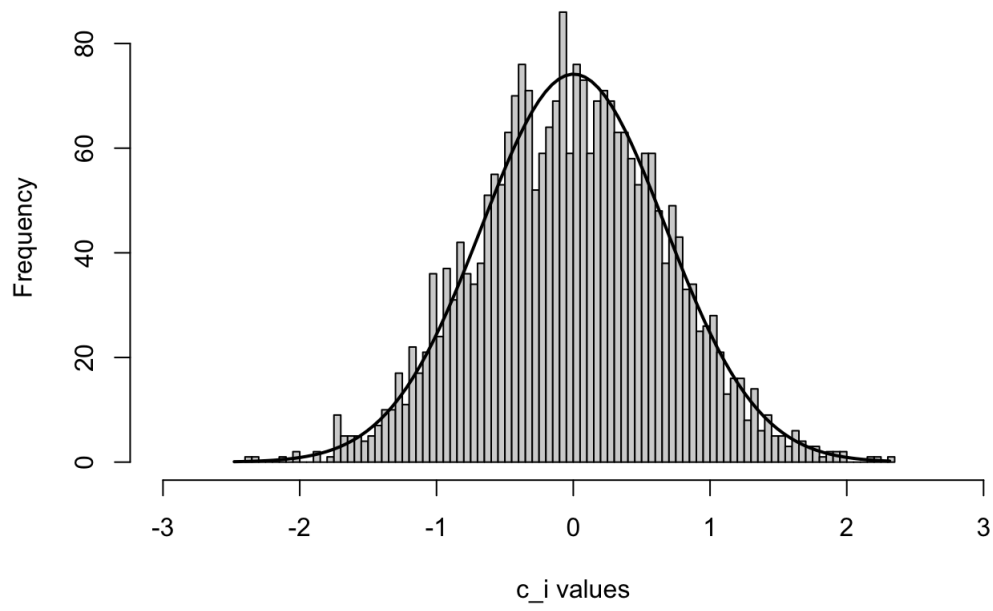
Histogram of c for n = 100



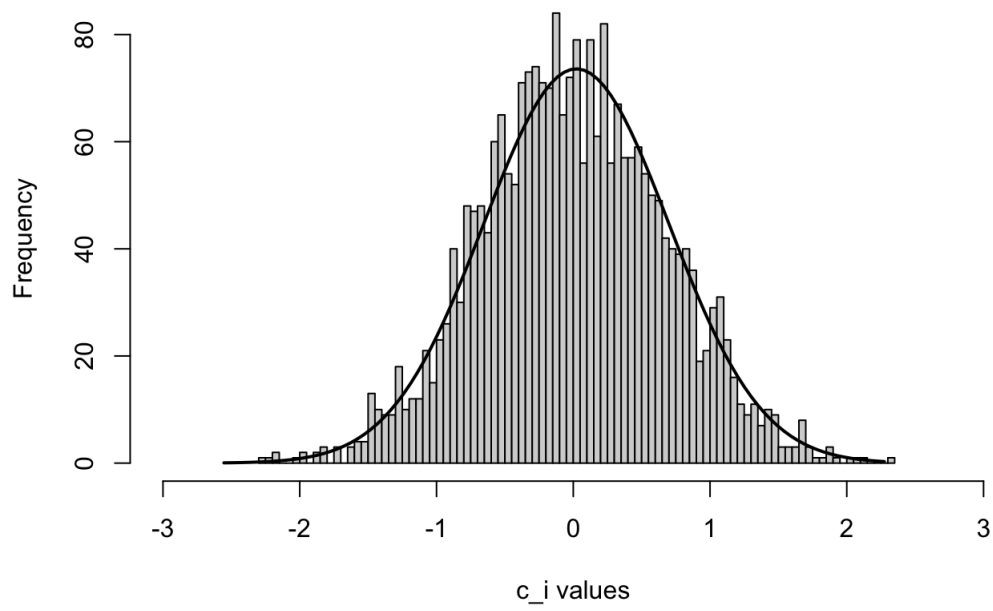
Histogram of c for n = 250



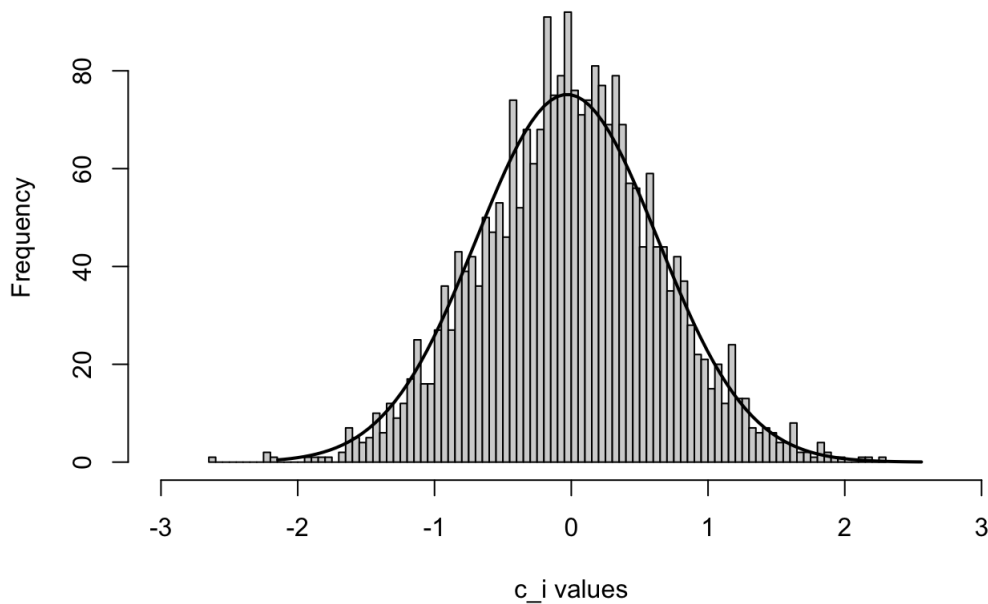
Histogram of c for n = 500



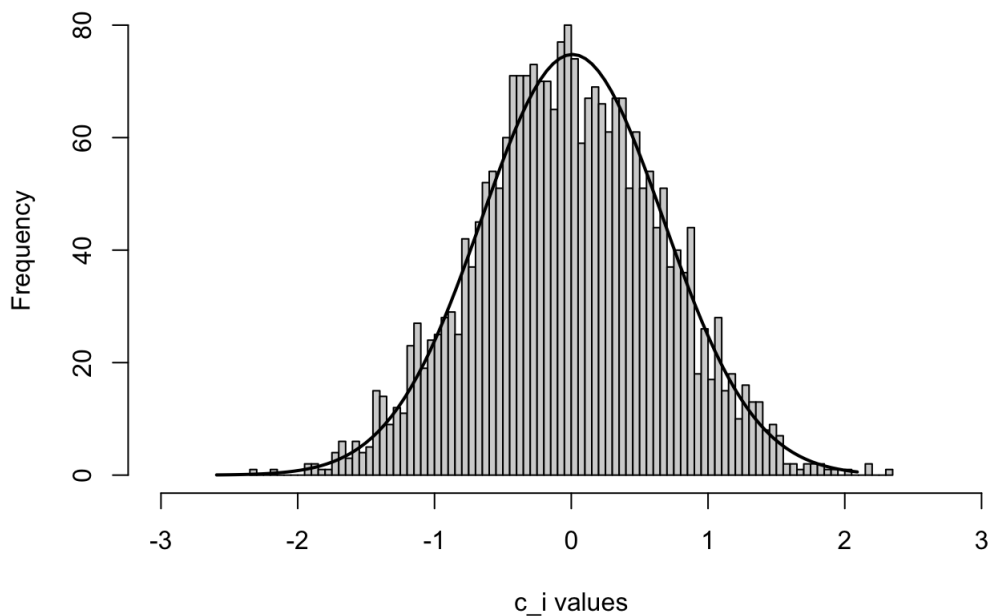
Histogram of c for n = 1000



Histogram of c for n = 2000



Histogram of c for n = 3000



- (h) What happens to the reported values in (f) and the histogram in (g) as n grows large? Discuss.

5. R. Video Game Sales Regression

The dataset for this exercise is available in `VideoGamesSales_Main.csv`. This dataset contains information on the global sales and critic and user review ratings for videogames launched between 2001 and 2012 (from www.vgchartz.com).

- Your task is to develop a regression model (using `lm()`) that links global sales to video game reviews, and explore ways in which the model fit could be improved through suitable changes to the model specification and `vdgm` variables. (a) (R) First, create a frequency table of 3 variables: platform, genre, and rating.

```
vdgm <- fread("videogamesales_main.csv", header = T)
if(demo) {str(vdgm)}
```

```
## Classes 'data.table' and 'data.frame': 4413 obs. of 12 variables:
## $ Name : chr "Madden NFL 11" "Sly Cooper and the Thievius Raccoonus" "The Lord of the Rings:
Conquest" "Red Steel 2" ...
## $ Platform : chr "PS3" "PS2" "X360" "Wii" ...
## $ Genre : chr "Sports" "Platform" "Action" "Shooter" ...
## $ Publisher : chr "Electronic Arts" "Sony Computer Entertainment" "Electronic Arts" "Ubisoft" ...
## $ Developer : chr "EA Tiburon" "Sucker Punch" "Pandemic Studios" "Ubisoft Paris" ...
## $ Rating : chr "E" "E" "T" "T" ...
## $ Global_Sales : num 2.38 1.21 0.63 0.62 1.43 0.09 0.56 0.33 1.52 0.21 ...
## $ Year_of_Release: int 2010 2002 2009 2010 2010 2011 2008 2009 2002 2006 ...
## $ Critic_Score : int 83 86 55 80 61 70 78 75 38 70 ...
## $ Critic_Count : int 36 41 58 73 59 4 47 17 21 9 ...
## $ User_Score : num 6.1 8.6 7 8.6 5.7 7 7.6 7.8 5 6.5 ...
## $ User_Count : int 68 184 110 178 180 4 36 6 26 10 ...
## - attr(*, ".internal.selfref")=<externalptr>
```

```
ptl_frq <- table(vdgm$Platform)
gnr_frq <- table(vdgm$Genre)
rating_frq <- table(vdgm$Rating)
```

- b. R. Create categorical variables for platform, genre, and rating using data.table. Also create a variable for the age of the game relative to year 2013 (Note that these games were released before 2013).

```
vdgm_2013 <- vdgm[,Platform]
vdgm$Action = ifelse(vdgm$Genre == 'Action', 1, 0)
vdgm$adventure = ifelse (vdgm$Genre == 'Adventure',1, 0)
vdgm$fighting = ifelse(vdgm$Genre == 'Fighting', 1, 0)
vdgm$misc = ifelse (vdgm$Genre == 'Misc',1, 0)
vdgm$platform = ifelse(vdgm$Genre == "Platform",1, 0)
vdgm$puzzle = ifelse(vdgm$Genre == "Puzzle", 1, 0)
vdgm$racng = ifelse(vdgm$Genre == 'Racing', 1, 0)
vdgm$Role_playing = ifelse(vdgm$Genre == "Role-Playing", 1, 0)
vdgm$Shooter = ifelse(vdgm$Genre == "Shooter", 1, 0)
vdgm$simulation = ifelse(vdgm$Genre == 'Simulation', 1, 0)
vdgm$sports = ifelse(vdgm$Genre == "Sports", 1, 0)
vdgm$strategy = ifelse(vdgm$Genre == 'strategy',1,0)

vdgm$Action = ifelse(vdgm$Genre == 'Action', 1, 0)
vdgm$adventure = ifelse (vdgm$Genre == 'Adventure',1, 0)
vdgm$fighting = ifelse(vdgm$Genre == 'Fighting', 1, 0)
vdgm$misc = ifelse (vdgm$Genre == 'Misc',1, 0)
vdgm$platform = ifelse(vdgm$Genre == "Platform",1, 0)
vdgm$puzzle = ifelse(vdgm$Genre == "Puzzle", 1, 0)
vdgm$racng = ifelse(vdgm$Genre == 'Racing', 1, 0)
vdgm$Role_playing = ifelse(vdgm$Genre == "Role-Playing", 1, 0)
vdgm$Shooter = ifelse(vdgm$Genre == "Shooter", 1, 0)
vdgm$simulation = ifelse(vdgm$Genre == 'Simulation', 1, 0)
vdgm$sports = ifelse(vdgm$Genre == "Sports", 1, 0)
vdgm$strategy = ifelse(vdgm$Genre == 'strategy',1,0)

vdgm$DS = ifelse (vdgm$Platform == 'DS',1, 0)
vdgm$GBA = ifelse(vdgm$Platform == 'GBA',1, 0)
vdgm$GC = ifelse(vdgm$Platform == 'GC',1,0)
vdgm$Pc = ifelse(vdgm$Platform == 'PC',1,0)
vdgm$Ps2 = ifelse(vdgm$Platform == "PS2",1,0)
vdgm$Ps3 = ifelse(vdgm$Platform == "PS3",1,0)
vdgm$PSP = ifelse (vdgm$Platform == "PSP",1,0)
vdgm$Wii = ifelse (vdgm$platform == "Wii",1,0)
vdgm$X360 = ifelse(vdgm$Platform == "x360",1,0)
vdgm$XB = ifelse (vdgm$Platform == "XB",1,0)
vdgm$DS = ifelse(vdgm$Platform == "DS",1,0)

vdgm$E = ifelse(vdgm$Rating == "E",1,0)
vdgm$`E10+` = ifelse(vdgm$Rating == "E10+",1,0)
vdgm$M = ifelse(vdgm$Rating == 'M' , 1,0)
vdgm$T = ifelse(vdgm$Rating == "T",1,0)
```

c. R. Run a regression with all relevant X variables. Report the adjusted R-squared.

```
lm1 <- lm(Global_Sales ~. - Name - Publisher - Developer, data = vdgm)
summary(lm1)
```

```
##
## Call:
## lm(formula = Global_Sales ~ . - Name - Publisher - Developer,
##     data = vdgm)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -7.931 -0.603 -0.157  0.287 79.609
##
## Coefficients: (26 not defined because of singularities)
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   7.487e+01  3.283e+01   2.280 0.022635 *
## PlatformGBA   -3.655e-01  1.932e-01  -1.891 0.058656 .
## PlatformGC    -5.773e-01  1.738e-01  -3.321 0.000905 ***
## PlatformPC    -1.098e+00  1.564e-01  -7.020 2.57e-12 ***
## PlatformPS2   -1.714e-01  1.429e-01  -1.199 0.230416
## PlatformPS3   -2.784e-01  1.474e-01  -1.889 0.058954 .
## PlatformPSP   -3.523e-01  1.574e-01  -2.238 0.025284 *
## PlatformWii    5.711e-01  1.485e-01   3.846 0.000122 ***
## PlatformX360  -5.233e-01  1.472e-01  -3.555 0.000382 ***
## PlatformXB    -7.815e-01  1.597e-01  -4.893 1.03e-06 ***
## GenreAdventure -2.025e-01  1.695e-01  -1.195 0.232235
## GenreFighting  7.171e-02  1.452e-01   0.494 0.621327
## GenreMisc      3.774e-01  1.375e-01   2.744 0.006099 **
## GenrePlatform  3.196e-02  1.451e-01   0.220 0.825629
## GenrePuzzle    -4.010e-01  2.479e-01  -1.618 0.105776
## GenreRacing     5.483e-02  1.237e-01   0.443 0.657609
## GenreRole-Playing -3.169e-01  1.170e-01  -2.710 0.006762 **
## GenreShooter   -4.054e-02  1.068e-01  -0.380 0.704212
## GenreSimulation 2.353e-01  1.533e-01   1.534 0.125031
## GenreSports    -1.156e-02  1.204e-01  -0.096 0.923500
## GenreStrategy  -2.469e-01  1.635e-01  -1.510 0.131096
## RatingE10+    -3.931e-01  1.069e-01  -3.678 0.000238 ***
## RatingM       -5.186e-01  1.186e-01  -4.374 1.25e-05 ***
## RatingT       -4.798e-01  9.204e-02  -5.213 1.95e-07 ***
## Year_of_Release -3.745e-02  1.635e-02  -2.291 0.022035 *
## Critic_Score   1.649e-02  3.147e-03   5.241 1.67e-07 ***
## Critic_Count   2.981e-02  2.269e-03  13.139 < 2e-16 ***
## User_Score     -5.491e-02  2.779e-02  -1.976 0.048261 *
## User_Count     8.748e-04  6.559e-05  13.337 < 2e-16 ***
## Action          NA          NA      NA      NA
## adventure        NA          NA      NA      NA
## fighting         NA          NA      NA      NA
## misc             NA          NA      NA      NA
## platform         NA          NA      NA      NA
## puzzle           NA          NA      NA      NA
## racing           NA          NA      NA      NA
## Role_playing     NA          NA      NA      NA
## Shooter          NA          NA      NA      NA
## simulation        NA          NA      NA      NA
## sports           NA          NA      NA      NA
## strategy         NA          NA      NA      NA
## DS               NA          NA      NA      NA
## GBA              NA          NA      NA      NA
## GC               NA          NA      NA      NA
## Pc               NA          NA      NA      NA
## Ps2              NA          NA      NA      NA
## Ps3              NA          NA      NA      NA
## PSP              NA          NA      NA      NA
## Wii              NA          NA      NA      NA
## X360             NA          NA      NA      NA
## XB               NA          NA      NA      NA
## E                NA          NA      NA      NA
## `E10+`          NA          NA      NA      NA
## M                NA          NA      NA      NA
## T                NA          NA      NA      NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 1.946 on 4384 degrees of freedom
## Multiple R-squared:  0.1795, Adjusted R-squared:  0.1742
## F-statistic: 34.25 on 28 and 4384 DF,  p-value: < 2.2e-16
```

- d. R. Now, generate natural log of the following variables: global sales, critic_score, critic_count, user_score, user_count as `ln_[original_variable_name]`.

```
vdgm$ln_Global_Sales = log(vdgm$Global_Sales)
vdgm$ln_Critic_Score = log(vdgm$Critic_Score)
vdgm$ln_Critic_Count = log(vdgm$Critic_Count)
vdgm$ln_User_Score = log(vdgm$User_Score)
vdgm$ln_User_Count = log(vdgm$User_Count)
```

- e. R. Run a regression with the log of Y variable and report adjusted R-squared.

```
lm2 <- lm(ln_Global_Sales ~. - Name - ln_Critic_Score - ln_Critic_Count - ln_User_Score - ln_User_Count - Publisher - Developer - Global_Sales, data = vdgm)
summary(lm2)
```

```
##
## Call:
## lm(formula = ln_Global_Sales ~ . - Name - ln_Critic_Score - ln_Critic_Count -
##     ln_User_Score - ln_User_Count - Publisher - Developer - Global_Sales,
##     data = vdgmm)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -4.4449 -0.6634  0.0008  0.6581  4.1020
##
## Coefficients: (26 not defined because of singularities)
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -4.173e+01  1.731e+01  -2.410  0.015989 *
## PlatformGBA    2.952e-02  1.019e-01   0.290  0.772038
## PlatformGC   -1.457e-01  9.166e-02  -1.589  0.112065
## PlatformPC   -1.713e+00  8.246e-02 -20.775 < 2e-16 ***
## PlatformPS2    4.156e-01  7.533e-02   5.517  3.64e-08 ***
## PlatformPS3    2.372e-01  7.770e-02   3.053  0.002281 **
## PlatformPSP   -7.539e-02  8.301e-02  -0.908  0.363857
## PlatformWii    4.616e-01  7.830e-02   5.895  4.03e-09 ***
## PlatformX360  -1.543e-01  7.762e-02  -1.988  0.046896 *
## PlatformXB   -4.391e-01  8.421e-02  -5.214  1.93e-07 ***
## GenreAdventure -4.542e-01  8.937e-02  -5.082  3.89e-07 ***
## GenreFighting  1.823e-02  7.654e-02   0.238  0.811773
## GenreMisc     1.828e-01  7.252e-02   2.520  0.011765 *
## GenrePlatform -3.026e-02  7.650e-02  -0.395  0.692495
## GenrePuzzle   -5.231e-01  1.307e-01  -4.002  6.38e-05 ***
## GenreRacing   -1.086e-01  6.522e-02  -1.665  0.095926 .
## GenreRole-Playing -2.975e-01  6.167e-02  -4.824  1.46e-06 ***
## GenreShooter  -1.345e-01  5.630e-02  -2.389  0.016934 *
## GenreSimulation  2.685e-01  8.085e-02   3.321  0.000905 ***
## GenreSports   -9.037e-02  6.348e-02  -1.423  0.154663
## GenreStrategy -5.684e-01  8.623e-02  -6.592  4.84e-11 ***
## RatingE10+   -2.287e-01  5.636e-02  -4.057  5.05e-05 ***
## RatingM      -4.326e-01  6.253e-02  -6.919  5.19e-12 ***
## RatingT      -3.404e-01  4.853e-02  -7.014  2.67e-12 ***
## Year_of_Release 1.925e-02  8.622e-03   2.233  0.025624 *
## Critic_Score  2.668e-02  1.659e-03  16.081 < 2e-16 ***
## Critic_Count  2.490e-02  1.196e-03  20.809 < 2e-16 ***
## User_Score    -5.077e-02  1.465e-02  -3.464  0.000537 ***
## User_Count    5.051e-04  3.459e-05  14.605 < 2e-16 ***
## Action              NA              NA      NA      NA
## adventure           NA              NA      NA      NA
## fighting            NA              NA      NA      NA
## misc                NA              NA      NA      NA
## platform            NA              NA      NA      NA
## puzzle              NA              NA      NA      NA
## racing              NA              NA      NA      NA
## Role_playing        NA              NA      NA      NA
## Shooter             NA              NA      NA      NA
## simulation          NA              NA      NA      NA
## sports              NA              NA      NA      NA
## strategy            NA              NA      NA      NA
## DS                  NA              NA      NA      NA
## GBA                  NA              NA      NA      NA
## GC                  NA              NA      NA      NA
## Pc                   NA              NA      NA      NA
## Ps2                  NA              NA      NA      NA
## Ps3                  NA              NA      NA      NA
## PSP                  NA              NA      NA      NA
## Wii                  NA              NA      NA      NA
## X360                 NA              NA      NA      NA
## XB                   NA              NA      NA      NA
## E                    NA              NA      NA      NA
## `E10+`              NA              NA      NA      NA
## M                    NA              NA      NA      NA
## T                    NA              NA      NA      NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```



```
##  
## Residual standard error: 1.026 on 4384 degrees of freedom  
## Multiple R-squared:  0.4555, Adjusted R-squared:  0.452  
## F-statistic: 131 on 28 and 4384 DF, p-value: < 2.2e-16
```

- f. R. Run a regression with the log of Y variable as well as log of X variables generated in part d). Report adjusted R-squared.

```
lm3 <- lm(ln_Global_Sales ~. - Global_Sales - Name - Publisher - Developer - Critic_Score - Critic_Count - User_Score - ln_User_Count, data = vdgm)  
summary(lm3)
```

```
##
## Call:
## lm(formula = ln_Global_Sales ~ . - Global_Sales - Name - Publisher -
##     Developer - Critic_Score - Critic_Count - User_Score - ln_User_Count,
##     data = vdgm)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -5.3799 -0.6687  0.0071  0.6726  4.1881
##
## Coefficients: (26 not defined because of singularities)
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -5.970e+01  1.775e+01  -3.363 0.000779 ***
## PlatformGBA     4.049e-02  1.046e-01   0.387 0.698714
## PlatformGC    -1.732e-01  9.380e-02  -1.847 0.064810 .
## PlatformPC    -1.803e+00  8.390e-02 -21.490 < 2e-16 ***
## PlatformPS2     3.656e-01  7.701e-02   4.748 2.12e-06 ***
## PlatformPS3     2.341e-01  7.969e-02   2.937 0.003331 **
## PlatformPSP    -1.171e-01  8.496e-02  -1.378 0.168325
## PlatformWii     4.492e-01  8.011e-02   5.607 2.19e-08 ***
## PlatformX360   -4.764e-02  7.862e-02  -0.606 0.544584
## PlatformXB    -4.573e-01  8.619e-02  -5.305 1.18e-07 ***
## GenreAdventure -4.630e-01  9.146e-02  -5.062 4.32e-07 ***
## GenreFighting  -6.030e-03  7.824e-02  -0.077 0.938575
## GenreMisc       2.037e-01  7.418e-02   2.746 0.006052 **
## GenrePlatform  -1.965e-02  7.827e-02  -0.251 0.801784
## GenrePuzzle    -4.934e-01  1.338e-01  -3.689 0.000228 ***
## GenreRacing    -7.956e-02  6.672e-02  -1.192 0.233173
## GenreRole-Playing -3.034e-01  6.318e-02  -4.803 1.61e-06 ***
## GenreShooter   -1.192e-01  5.761e-02  -2.069 0.038622 *
## GenreSimulation  2.675e-01  8.279e-02   3.231 0.001244 **
## GenreSports    -9.166e-02  6.449e-02  -1.421 0.155272
## GenreStrategy  -5.638e-01  8.829e-02  -6.386 1.88e-10 ***
## RatingE10+    -2.542e-01  5.765e-02  -4.409 1.06e-05 ***
## RatingM       -4.071e-01  6.403e-02  -6.358 2.25e-10 ***
## RatingT       -3.446e-01  4.975e-02  -6.928 4.89e-12 ***
## Year_of_Release 2.552e-02  8.847e-03   2.885 0.003936 **
## User_Count     6.437e-04  3.454e-05  18.640 < 2e-16 ***
## Action          NA          NA      NA      NA
## adventure        NA          NA      NA      NA
## fighting         NA          NA      NA      NA
## misc             NA          NA      NA      NA
## platform         NA          NA      NA      NA
## puzzle           NA          NA      NA      NA
## racing           NA          NA      NA      NA
## Role_playing     NA          NA      NA      NA
## Shooter          NA          NA      NA      NA
## simulation        NA          NA      NA      NA
## sports           NA          NA      NA      NA
## strategy         NA          NA      NA      NA
## DS               NA          NA      NA      NA
## GBA              NA          NA      NA      NA
## GC               NA          NA      NA      NA
## Pc               NA          NA      NA      NA
## Ps2              NA          NA      NA      NA
## Ps3              NA          NA      NA      NA
## PSP              NA          NA      NA      NA
## Wii              NA          NA      NA      NA
## X360             NA          NA      NA      NA
## XB               NA          NA      NA      NA
## E               NA          NA      NA      NA
## `E10+`          NA          NA      NA      NA
## M               NA          NA      NA      NA
## T               NA          NA      NA      NA
## ln_Critic_Score  1.462e+00  9.917e-02  14.748 < 2e-16 ***
## ln_Critic_Count  5.554e-01  3.008e-02  18.461 < 2e-16 ***
## ln_User_Score   -1.711e-01  8.012e-02  -2.136 0.032741 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##  
## Residual standard error: 1.05 on 4384 degrees of freedom  
## Multiple R-squared:  0.4299, Adjusted R-squared:  0.4262  
## F-statistic: 118.1 on 28 and 4384 DF,  p-value: < 2.2e-16
```

g. Which model (out of part c, e, and f) offers the highest adjusted R-squared? What would be the economic reasoning on why that particular model provides the best fit?

Ans: lm2 has the highest R-squared value. It makes sense that this has the best fit as the sales variable usually been increasing exponentially.

h. Interpret the parameter estimates for each of 'genre' in plain English.

Ans: Simulation Games created the highest impact in terms of Global Sales and Sports created the lowest based on the parameter estimates.

i. Interpret the parameter estimate for 'rating' in plain English.

Ans: Simulation Games created the highest impact in terms of Global Sales and Sports created the lowest based on the parameter estimates.

i. Interpret the parameter estimate for 'ln_user_count' in plain English.

Ans: ln_User_Count has a p_value > 0.05. There we cannot reject the null hypothesis i.e. the coefficient of ln_user_count = 0.