STAT 206 Lab 1_Lihua Xu

Due Monday, October 10, 5:00 PM

General instructions for labs: You are encouraged to work in pairs to complete the lab. Labs must be completed as an R Markdown file. Be sure to include your lab partner (if you have one) and your own name in the file. Give the commands to answer each question in its own code block, which will also produce plots that will be automatically embedded in the output file. Each answer must be supported by written statements as well as any code used.

Agenda: Manipulating data objects; using built-in functions, doing numerical calculations, and basic plots; reinforcing core probabilistic ideas.

Background

The exponential distribution is defined by its cumulative distribution function

$$F(x) = 1 - e^{-\lambda x}$$

The R function `rexp` generates random variables with an exponential distribution.

`rexp(n=10, rate=5)`

produces 10 exponentially-distributed numbers with rate (λ) of 5. If the second argument is omitted, the default rate is 1; this is the "standard exponential distribution".

Part I

1. Generate 200 random values from the standard exponential distribution and store them in a vector `exp.draws.1`. Find the mean and standard deviation of `exp.draws.1`.

```
exp.draws.1 <- rexp(n=200)
mean_value_1 <- mean(exp.draws.1)
mean_value_1
```

```
## [1] 0.9665352
```

```
stadard_deviation_1 <- sd(exp.draws.1)
stadard_deviation_1
```

```
## [1] 0.9596607
```

2. Repeat, but change the rate to 0.1, 0.5, 5 and 10, storing the results in vectors called `exp.draws.0.1`, `exp.draws.0.5`, `exp.draws.5` and `exp.draws.10`.

```
exp.draws.0.1 <- rexp(n=200,rate=0.1)
mean_value_0.1 <- mean(exp.draws.0.1)
mean_value_0.1
```

```
## [1] 9.484534
```

```
stadard_deviation_0.1 <- sd(exp.draws.0.1)
stadard_deviation_0.1
```

```
## [1] 9.375175
```

```
exp.draws.0.5 <- rexp(n=200,rate=0.5)
mean_value_0.5 <- mean(exp.draws.0.5)
mean_value_0.5
```

```
## [1] 1.968341
```

```
stadard_deviation_0.5 <- sd(exp.draws.0.5)
stadard_deviation_0.5
```

```
## [1] 2.327087
```

```
exp.draws.5 <- rexp(n=200,rate=5)
mean_value_5 <- mean(exp.draws.5)
mean_value_5
```

```
## [1] 0.190769
```

```
stadard_deviation_5 <- sd(exp.draws.5)
stadard_deviation_5
```

```
## [1] 0.202486
```

```
exp.draws.10 <- rexp(n=200,rate=10)
mean_value_10 <- mean(exp.draws.10)
mean_value_10
```

```
## [1] 0.1122496
```

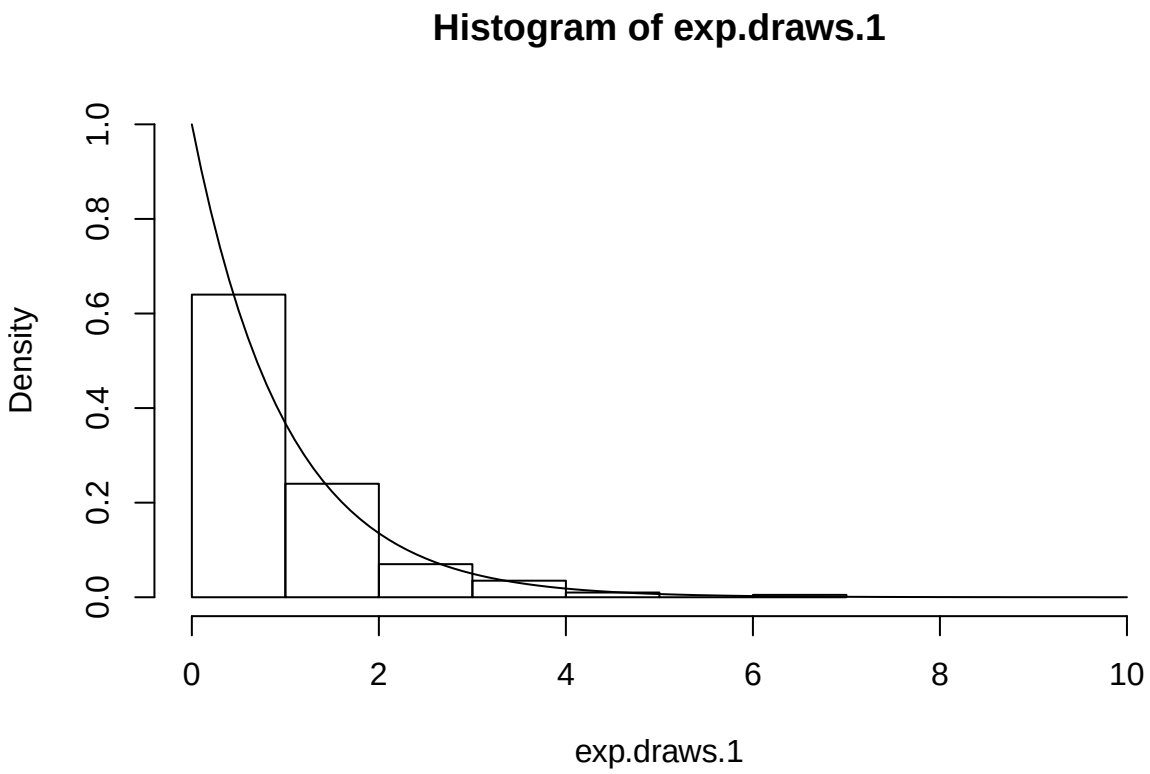
```
stadard_deviation_10 <- sd(exp.draws.10)
stadard_deviation_10
```

```
## [1] 0.1062798
```

3. The function `plot()` is the generic function in R for the visual display of data. `hist()` is a function that takes in and bins data as a side effect. To use this function, we must first specify what we'd like to plot.

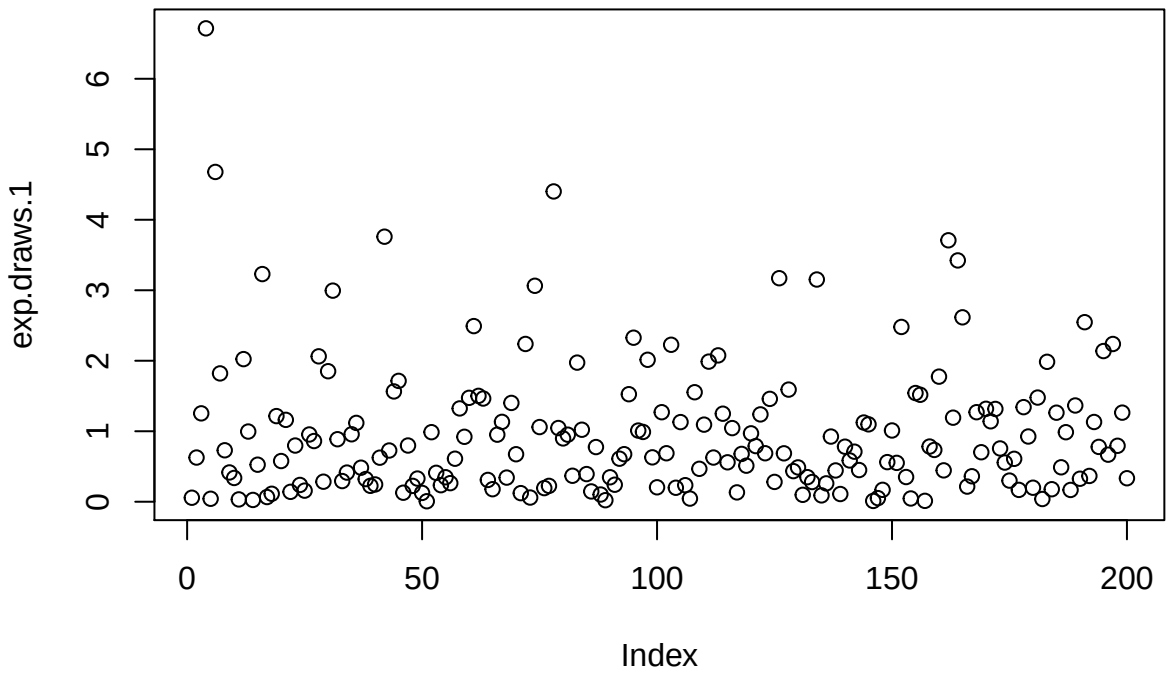
a. Use the `hist()` function to produce a histogram of your standard exponential distribution.

```
hist(exp.draws.1,freq = FALSE,ylim = c(0,1),xlim = c(0,10))
curve(dexp, add = TRUE)
```



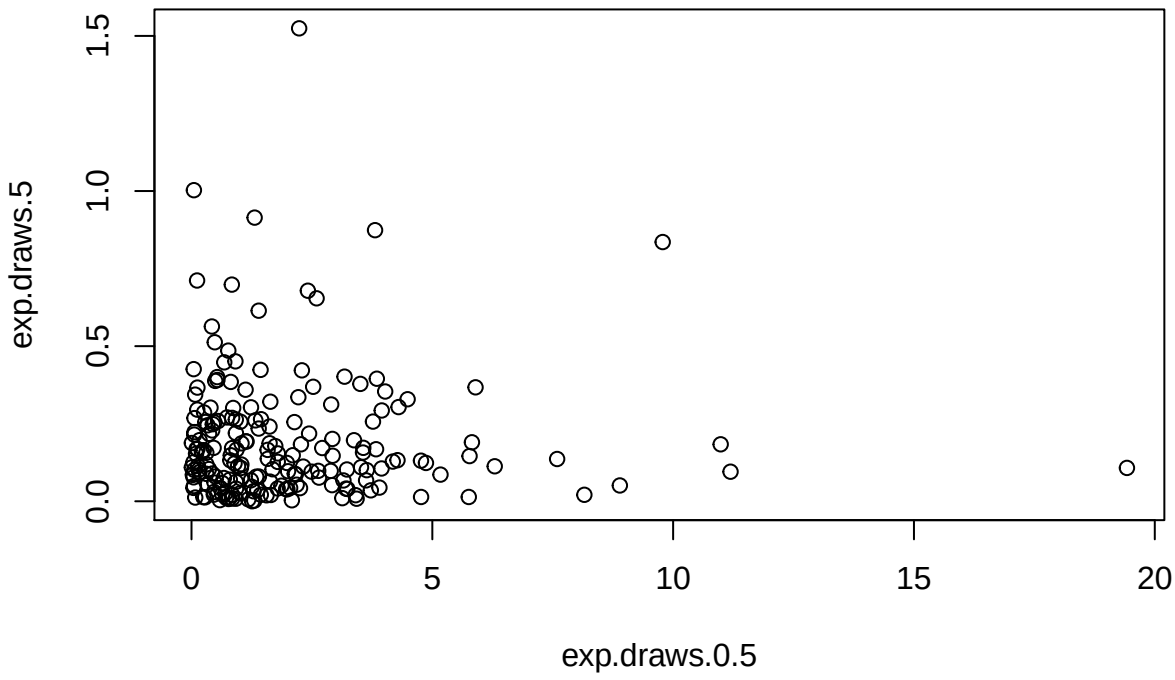
b. Use `plot()` with this vector to display the random values from your standard distribution in order.

```
plot(exp.draws.1)
```



c. Now, use `plot()` with two arguments – any two of your other stored random value vectors – to create a scatterplot of the two vectors against each other.

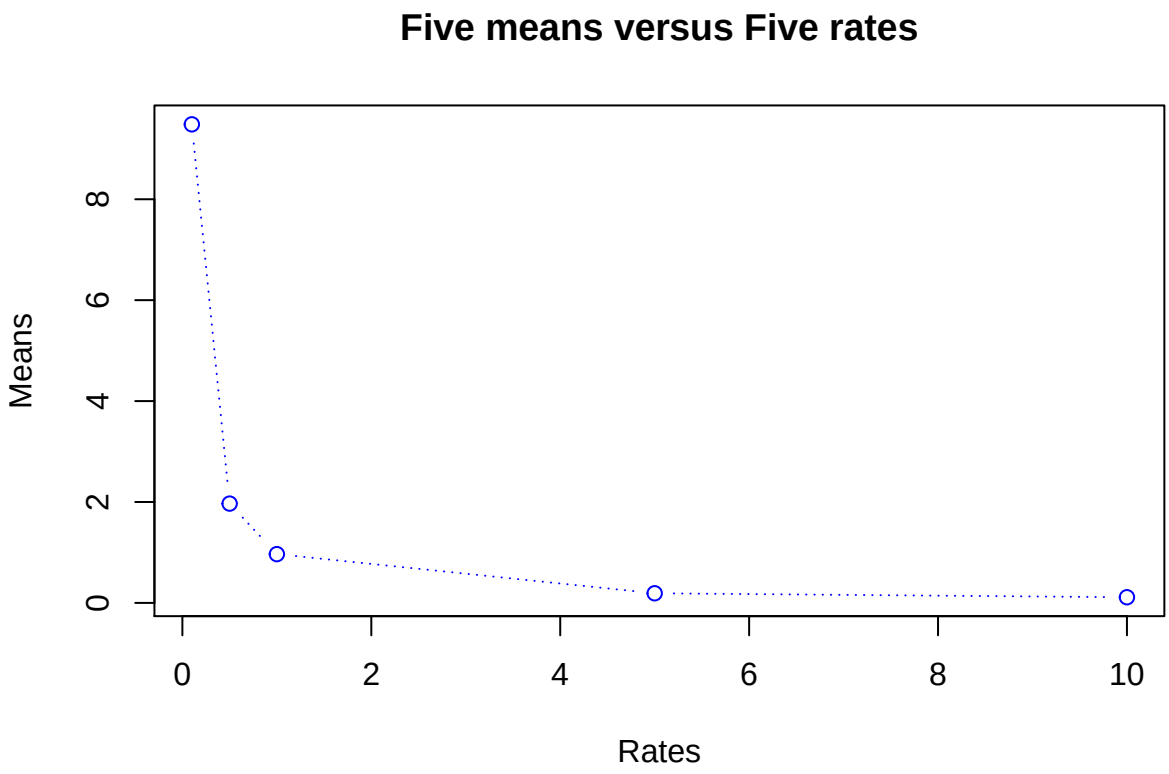
```
plot (exp.draws.0.5,exp.draws.5)
```



4. We'd now like to compare the properties of each of our vectors. Begin by creating a vector of the means of each of our five distributions in the order we created them and saving this to a variable name of your choice. Using this and other similar vectors, create the following scatterplots:

a. The five means versus the five rates used to generate the distribution.

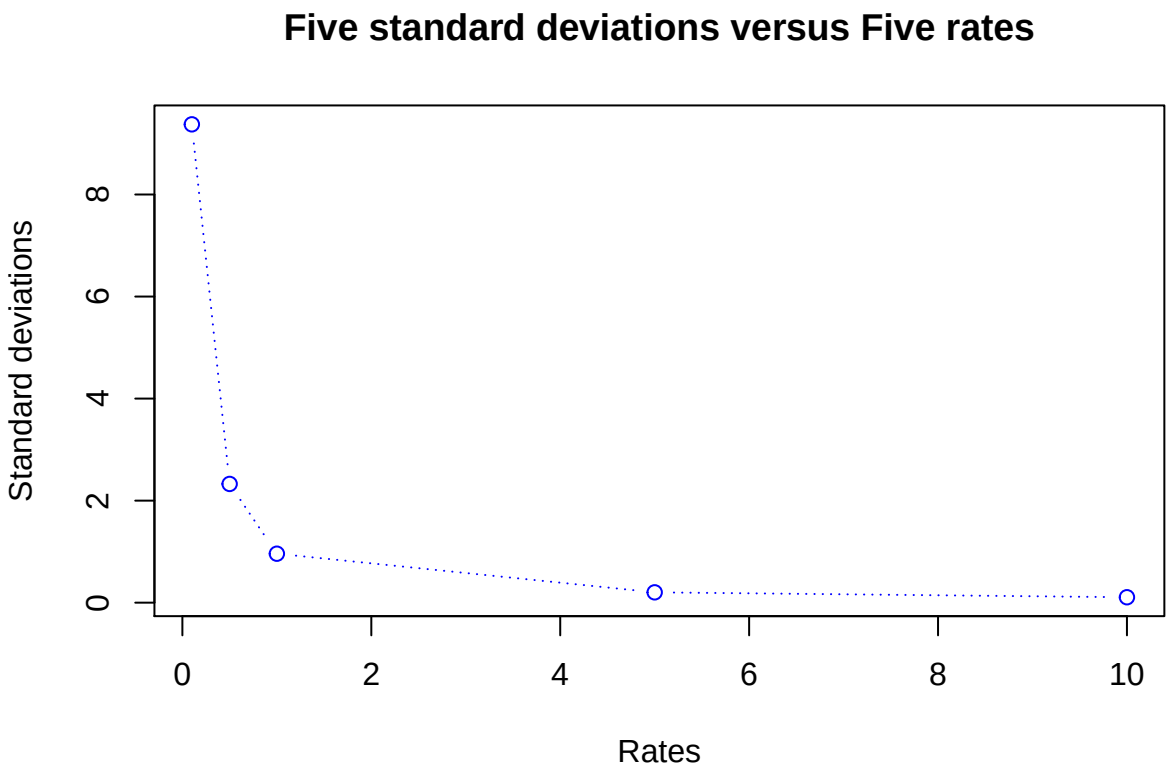
```
mean_vextor <- c(mean_value_0.1,mean_value_0.5,
                 mean_value_1,mean_value_5,mean_value_10)
rate_vector <- c(0.1,0.5,1,5,10)
plot (rate_vector,mean_vextor,type="b",lty=3,
      xlab = "Rates",ylab = "Means",
      main = "Five means versus Five rates",col="blue")
```




```
#From this plot, we can see that when rate decrease, the mean increases.
```

b. The standard deviations versus the rates.

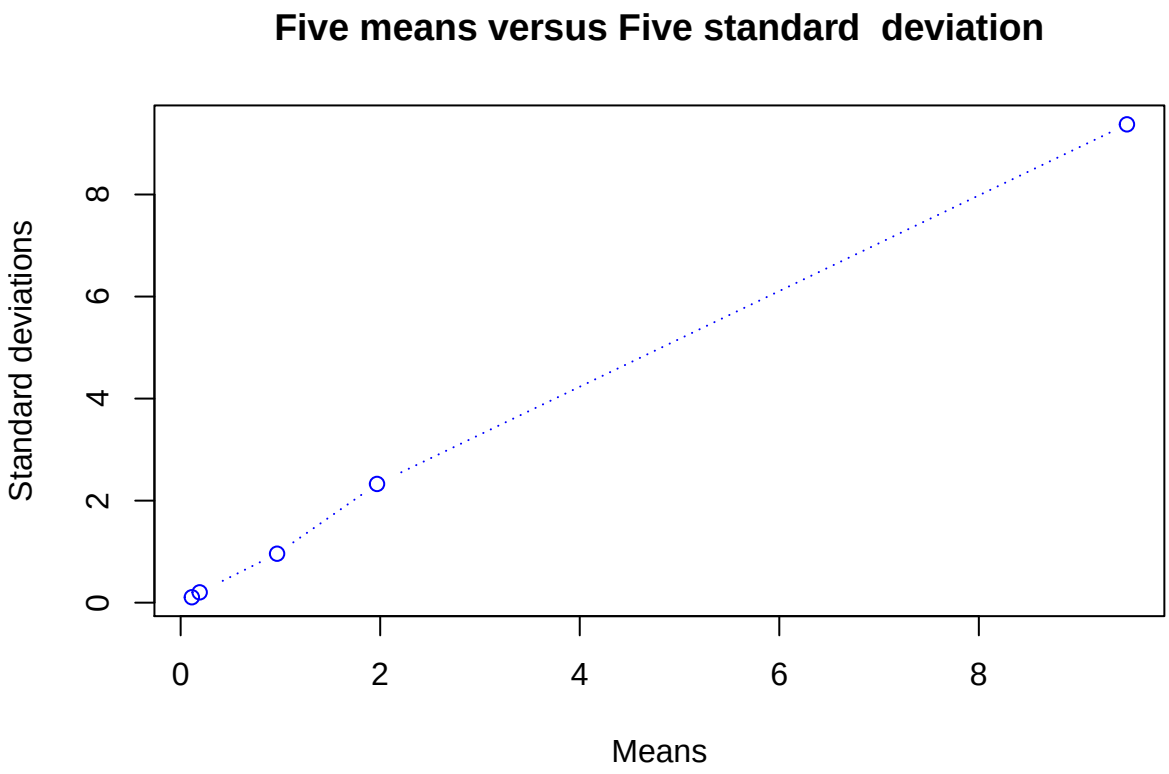
```
sd_vextor <- c(stadard_deviation_0.1,stadard_deviation_0.5,
               stadard_deviation_1,stadard_deviation_5,
               stadard_deviation_10)
rate_vector <- c(0.1,0.5,1,5,10)
plot (rate_vector,sd_vextor,type="b",lty=3,
      xlab = "Rates",ylab = "Standard deviations",
      main = "Five standard deviations versus Five rates",col="blue")
```




```
#The tendency for this plot is similar to that for the former plot.
#When the value of Rate decreases, the standard deviation increases.
```

c. The means versus the standard deviations.

```
plot (mean_vextor,sd_vextor,type="b",lty=3,xlab = "Means",
      ylab = "Standard deviations",
      main = "Five means versus Five standard  deviation",
      col="blue")
```




```
#This plot shows that the standard deviation will increase
#(more likely to be in an linear tendency) when the mean value increases.
```

For each plot, explain in words what's going on.

Part II

5. R's capacity for data and computation is large to what was available 10 years ago.

a. To show this, generate 1.1 million numbers from the standard exponential distribution and store them in a vector called `big.exp.draws.1`. Calculate the mean and standard deviation.

```
big.exp.draws.1 <- rexp(n=1e6+1e5)
mean_value_1e6 <- mean(big.exp.draws.1)
mean_value_1e6
```

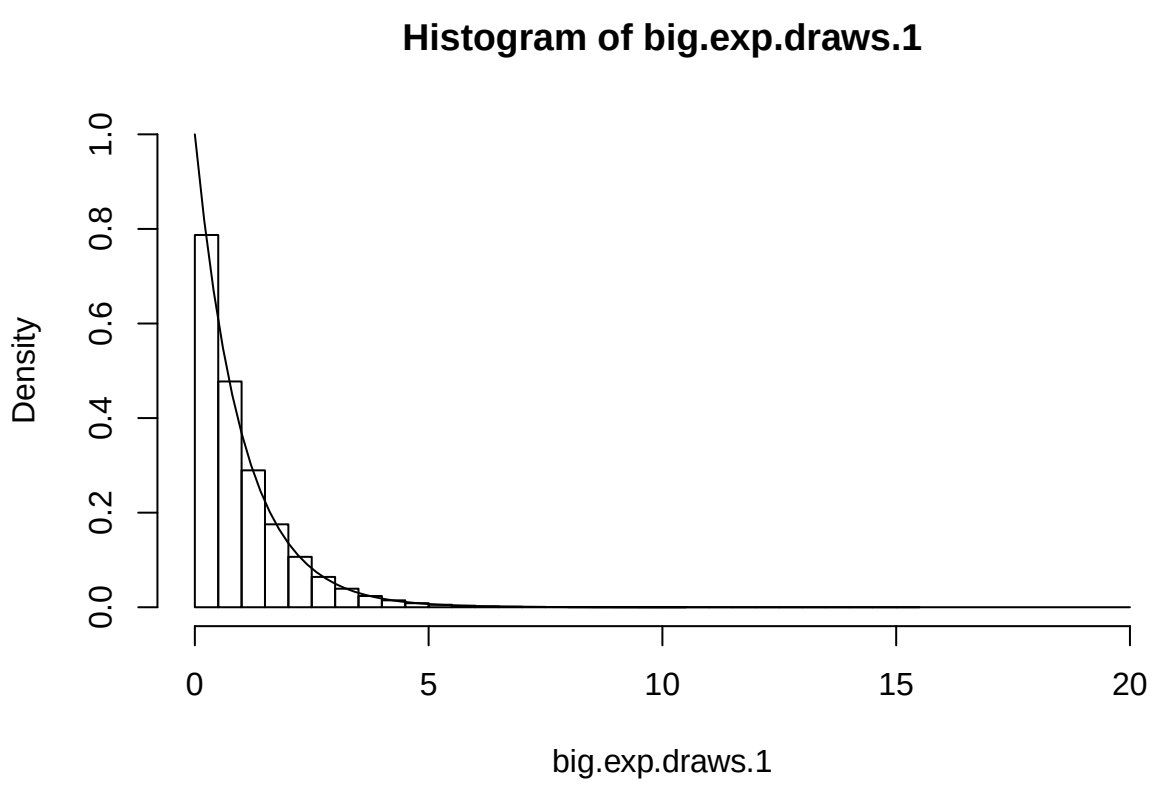
```
## [1] 0.9996993
```

```
stadard_deviation_1e6 <- sd(big.exp.draws.1)
stadard_deviation_1e6
```

```
## [1] 1.000778
```

b. Plot a histogram of `big.exp.draws.1`. Does it match the function $1 - e^{-x}$? Should it?

```
variable_c <- hist(big.exp.draws.1,freq = FALSE,ylim = c(0,1),xlim = c(0,20))
curve(dexp, add = TRUE)
```




```
variable_c
```

```
## $breaks
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5 15.0 15.5
##
## $counts
##  [1] 432956 262617 159233  96471  58630  35254  21561  13076   8032   4699
## [11]   2923   1791   1061    645    396    264    153     90     52     46
## [21]     22      9      9      3      2      2      2      0      0      0
## [31]      1
##
## $density
##  [1] 7.871927e-01 4.774855e-01 2.895145e-01 1.754018e-01 1.066000e-01
##  [6] 6.409818e-02 3.920182e-02 2.377455e-02 1.460364e-02 8.543636e-03
## [11] 5.314545e-03 3.256364e-03 1.929091e-03 1.172727e-03 7.200000e-04
## [16] 4.800000e-04 2.781818e-04 1.636364e-04 9.454545e-05 8.363636e-05
## [21] 4.000000e-05 1.636364e-05 1.636364e-05 5.454545e-06 3.636364e-06
## [26] 3.636364e-06 3.636364e-06 0.000000e+00 0.000000e+00 0.000000e+00
## [31] 1.818182e-06
##
## $mids
##  [1]  0.25  0.75  1.25  1.75  2.25  2.75  3.25  3.75  4.25  4.75  5.25
## [12]  5.75  6.25  6.75  7.25  7.75  8.25  8.75  9.25  9.75 10.25 10.75
## [23] 11.25 11.75 12.25 12.75 13.25 13.75 14.25 14.75 15.25
```

```
##
## $xname
## [1] "big.exp.draws.1"
##
## $equidist
## [1] TRUE
##
## attr(,"class")
## [1] "histogram"
```

```
#It matches the function $1-e^{-x}$.
#As the number of the sampling are huge, the result should approach the function.
```

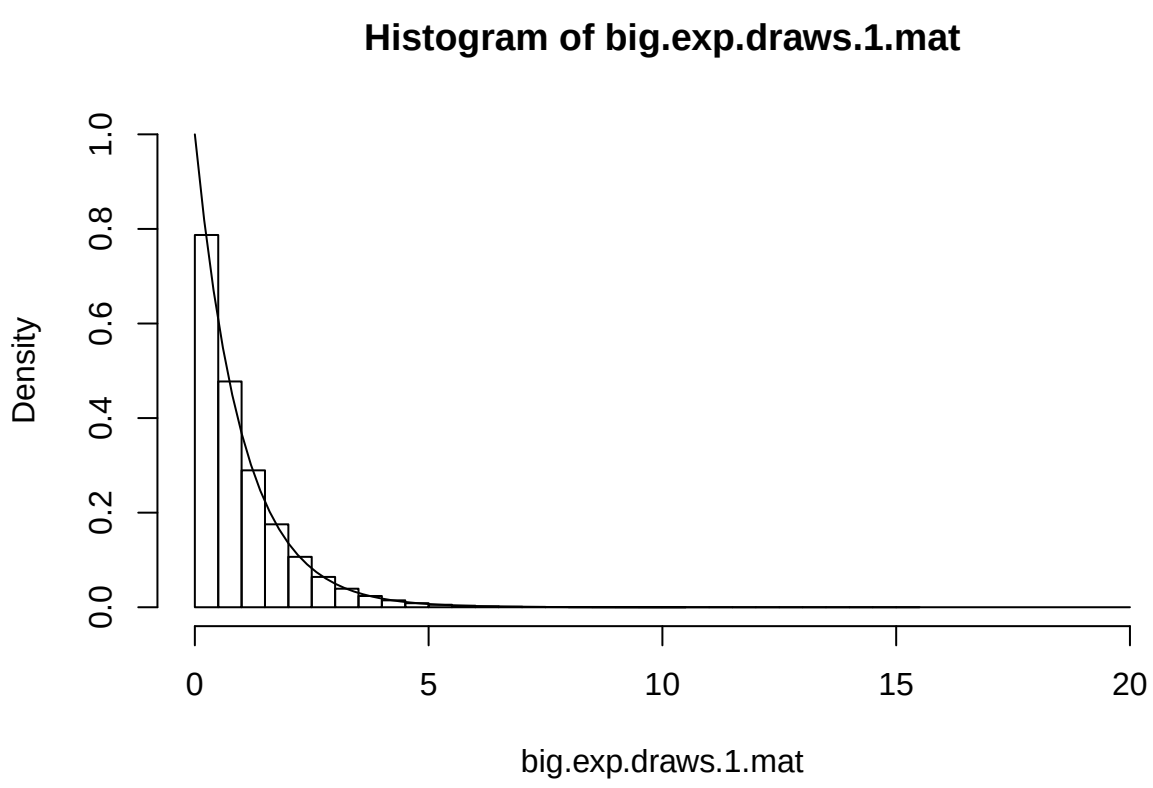
c. Find the mean of all of the entries in `big.exp.draws.1` which are strictly greater than 1. You may need to first create a new vector to identify which elements satisfy this.

```
index <- which(big.exp.draws.1>1)
Greater_1_vector <- big.exp.draws.1[index]
mean_value_greater_1 <- mean(Greater_1_vector)
mean_value_greater_1
```

```
## [1] 2.00051
```

d. Create a matrix, `big.exp.draws.1.mat`, containing the the values in `big.exp.draws.1`, with 1100 rows and 1000 columns. Use this matrix as the input to the `hist()` function and save the result to a variable of your choice. What happens to your data?

```
big.exp.draws.1.mat <- array(big.exp.draws.1,dim=c(1100,1000))
variable_d <- hist(big.exp.draws.1.mat,freq = FALSE,ylim = c(0,1),xlim = c(0,20))
curve(dexp, add = TRUE)
```




```
variable_d
```

```
## $breaks
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5 15.0 15.5
##
## $counts
##  [1] 432956 262617 159233  96471  58630  35254  21561  13076   8032   4699
## [11]   2923   1791   1061    645    396    264    153     90     52     46
## [21]     22      9      9      3      2      2      2      0      0      0
## [31]      1
##
## $density
##  [1] 7.871927e-01 4.774855e-01 2.895145e-01 1.754018e-01 1.066000e-01
##  [6] 6.409818e-02 3.920182e-02 2.377455e-02 1.460364e-02 8.543636e-03
## [11] 5.314545e-03 3.256364e-03 1.929091e-03 1.172727e-03 7.200000e-04
## [16] 4.800000e-04 2.781818e-04 1.636364e-04 9.454545e-05 8.363636e-05
## [21] 4.000000e-05 1.636364e-05 1.636364e-05 5.454545e-06 3.636364e-06
## [26] 3.636364e-06 3.636364e-06 0.000000e+00 0.000000e+00 0.000000e+00
## [31] 1.818182e-06
##
## $mids
##  [1]  0.25  0.75  1.25  1.75  2.25  2.75  3.25  3.75  4.25  4.75  5.25
## [12]  5.75  6.25  6.75  7.25  7.75  8.25  8.75  9.25  9.75 10.25 10.75
## [23] 11.25 11.75 12.25 12.75 13.25 13.75 14.25 14.75 15.25
```

```
##
## $xname
## [1] "big.exp.draws.1.mat"
##
## $equidist
## [1] TRUE
##
## attr(,"class")
## [1] "histogram"
```

```
#It will give the same results in problem c.
```

e. Calculate the mean of the 371st column of `big.exp.draws.1.mat`.

```
#solution 1
co <- big.exp.draws.1.mat[,371]
co_mean <- mean(co)
co_mean
```

```
## [1] 1.033759
```

```
#solution 2
mean_value_371_co <- colMeans(big.exp.draws.1.mat,na.rm = FALSE,dims =1)
mean_value_371_co[371]
```

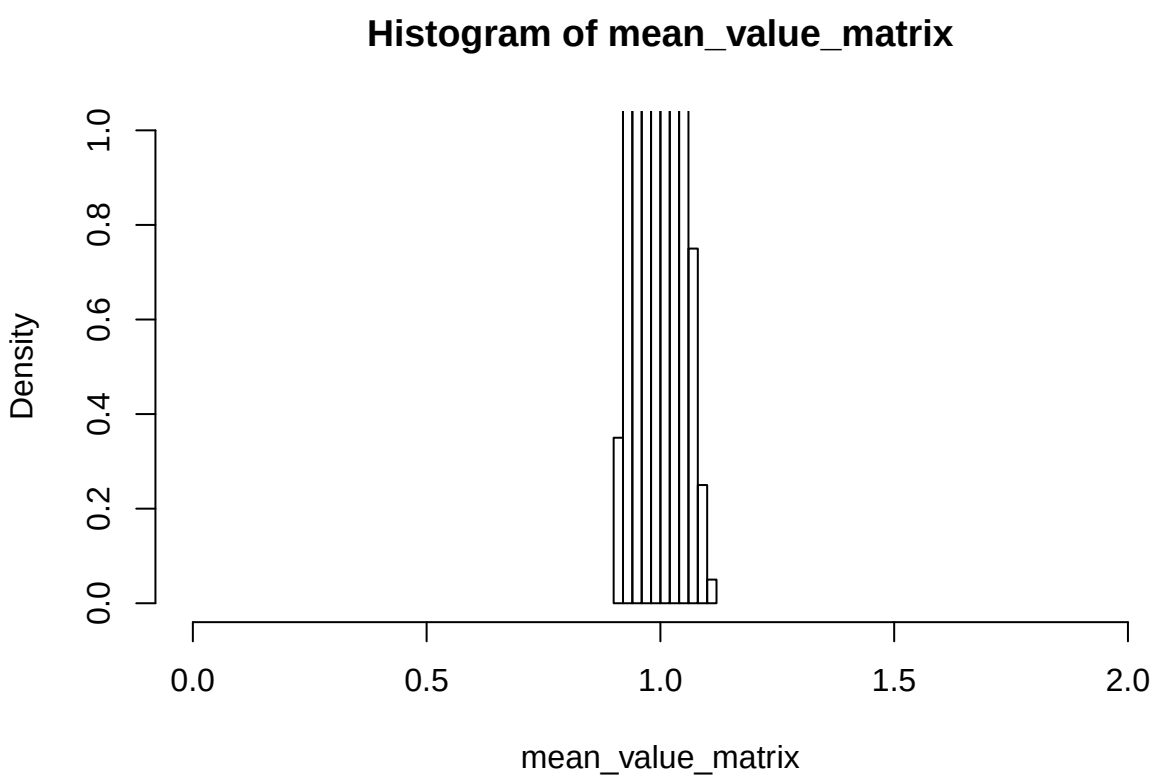
```
## [1] 1.033759
```

```
#solution 3
a <- big.exp.draws.1.mat
vector_value <- apply(a,2,mean)
vector_value[371]
```

```
## [1] 1.033759
```

f. Now, find the means of all 1000 columns of `big.exp.draws.1.mat` simultaneously. Plot the histogram of column means. Explain why its shape does not match the histogram in problem 5b).

```
mean_value_matrix <- colMeans(big.exp.draws.1.mat,na.rm = FALSE,dims =1)
detailed_inf <- hist(mean_value_matrix,freq = FALSE,ylim = c(0,1),xlim = c(0,2))
```




```
detailed_inf
```

```
## $breaks
##  [1] 0.90 0.92 0.94 0.96 0.98 1.00 1.02 1.04 1.06 1.08 1.10 1.12
##
## $counts
##  [1]   7  24  57 157 267 235 161  71  15   5   1
##
## $density
##  [1]  0.35  1.20  2.85  7.85 13.35 11.75  8.05  3.55  0.75  0.25  0.05
##
## $mids
##  [1] 0.91 0.93 0.95 0.97 0.99 1.01 1.03 1.05 1.07 1.09 1.11
##
## $xname
## [1] "mean_value_matrix"
##
## $equidist
## [1] TRUE
##
## attr(,"class")
## [1] "histogram"
```

```
#Its shape does not match the histogram in problem 5b.
#because each value in mean_value_matrix is the colume average in big.exp.draws.1.mat.
#The mean_value_matrix is not standard deviations, each value should be around 1.
#So here, you could see that the distribution is around 1.
```

g. Take the square of each number in `big.exp.draws.1`, and find the mean of this new vector. Explain this in terms of the mean and standard deviation of `big.exp.draws.1`. ***Hint:*** think carefully about the formula R uses to calculate the standard deviation.

```
squre_matrix <- big.exp.draws.1^2
mean_new_vector <- mean (squre_matrix)
mean_new_vector
```

```
## [1] 2.000955
```

Some explainations:

The formula of standard deviation:

$$S = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \bar{x})^2}{N-1}}$$

where mean value:

$$\bar{x} = \sum_{i=1}^{N} x_i/N$$

The mean and the standard deviation for the big.exp.draws.1 vector is shown by the two equations above. Now we do a trick for the standard deviation:

$$S = \sqrt{\frac{(N) * \sum_{i=1}^{N}(x_i - \bar{x})^2}{(N-1) * N}}$$

And the equation of the mean of the squre_matrix:

$$\bar{x^2} = \sum_{i=1}^{N} x_i^2/N =< x_i^2 >$$

So now I will find the relationship for these two different equations: In the standard devision euqation, we have the part

$$(x_i - \bar{x})^2$$

then we separate it, we get

$$(x_i^2 + \bar{x}^2 - 2x_i\bar{x})$$

We known that

$$\frac{\sum_{i=1}^{N}(x_i^2 + \bar{x}^2 - 2x_i\bar{x})}{N} =< x_i^2 > + < \bar{x}^2 > -2 < x_i >< \bar{x} >$$

and

$$\bar{x} =< x_i >=< \bar{x} >$$

The above equation should be written as

$$< x_i^2 > +\bar{x}^2 - 2 < x_i > \bar{x}$$

So then we can get the Standard deviztion of the big.exp.draws.1 vector:

$$S = \sqrt{\frac{(N) * (< x_i^2 > -\bar{x}^2)}{(N-1)}}$$

From this equation: (We know N is the number os the values in the vector)

$$< x_i^2 >$$

refers to the mean of the new vector (the square of each number in `big.exp.draws.1`).

$$\bar{x}^2$$

refers to the squre of the mean of big.exp.draws.1 vector.