MINI PROJECT #1

GROUP NO. 28

Members

1. JEEVAN DSOUZA
2. SAMARTH SAIRAM

Contribution

Both the members of the project have rendered their contributions equally regarding problem solving as well as the implementation of code by means of R.

Section 1 (Solution)

1) To prove pdf & $E(T) = 15$ years

$E(X) = \frac{1}{\lambda}$ (for exponential distribution)

$10 = \frac{1}{\lambda}$

$\lambda = \frac{1}{10} = \underline{0.1}$ (For Both X_A & X_B $\lambda = 0.1$)

The satellite will work until ~~both~~ either one works and whichever works longest can be found out by.

$\max(T_{XA}, T_{XB})$

Exponential cdf is $1 - e^{-\lambda x}$

T_{XA} & T_{XB} independent

$\therefore (1 - e^{-0.1x})(1 - e^{-0.1}x)$

$= \underline{1 - 2e^{-0.1x} + e^{-0.2x}}$ $x > 0$ as per eqn $\textcircled{4}$

cdf differentiation gives pdf $\rightarrow \underline{0.2e^{-0.1t} - 0.2e^{-0.2t}}$ $\hookrightarrow \textcircled{1}$

$E(T) = \int_0^\infty t f(t)\, dt$

$= \int_0^\infty 0.2t e^{-0.1t} - 0.2te^{-0.2t}$

Integrating by parts $\int uv\,dv = uv - \int v\,du$

$= 0.2 \int_0^\infty te^{-0.1t} - \int_0^\infty te^{-0.2t}$ — $\textcircled{2}$

$\int_0^\infty x^{\alpha-1} e^{-\lambda x} dx = \frac{\Gamma(\alpha)}{\lambda^\alpha}$ — $\textcircled{3}$

Comparing eqn $\textcircled{3}$ with $\textcircled{2}$ $\lambda = 0.1$ $\alpha = 2$

$0.2 \left[\frac{\Gamma(2)}{(0.1)^2} - \frac{\Gamma(2)}{(0.2)^2} \right]$ $\Gamma(n) = (n-1)!$

$n = 2$ $\Gamma(2) = 1$

$0.2 \left[\frac{[illegible]}{100} - \frac{[illegible]}{25} \right]$

$\underline{E(T) = 15}$

1 a) As per eqn ①

Pdf is $0.2e^{-0.1t} - 0.2e^{-0.2t}$

we need probability that T satellite's lifetime is greater than 15 years

$P(T > 15)$ - ⓐ

we can use cdf here to know eqnⓐ

$1 - P(T \leq 15)$

cdf of exponential funtion is

$\underline{1 - e^{-\lambda t}}$

As $E(T) = 15$ years

$E(T) = \frac{1}{\lambda}$

$\underline{\lambda = 0.1}$

$\lambda = \frac{1}{15} = \underline{0.1}$

$1 - [(1 - e^{-\lambda t})(1 - e^{-\lambda t})]$

$1 - [(1 - e^{-\lambda t})^2]$

$\underline{t = 15}$

$1 - [1 - e^{-0.1 \times 15}]^2$

$1 - [1 - e^{-1.5}]^2$

$1 - [1 - 0.223]^2$

$= \underline{\underline{0.396}}$

<u>Section 2</u>

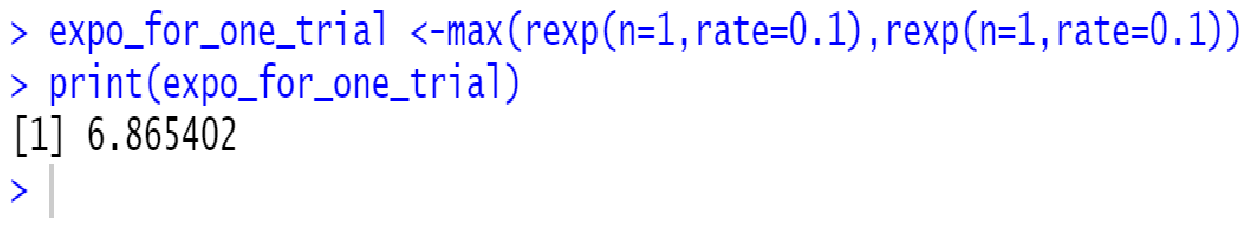
1. b)

i)

one draw of Xa and Xb to get T, we have rexp formula to get the random deviates.
Taking one trial of simulation. rexp takes n i,e observations and rate is the frequency that is 1/mean we get 0.1 as mean is 10.
we use max function so as to get the max life of Xa or Xb as both are independent it would be a product.

```
expo_for_one_trial <-max(rexp(n=1,rate=0.1),rexp(n=1,rate=0.1))
print(expo_for_one_trial)
```

#Result->As we do this, we get fluctuating results

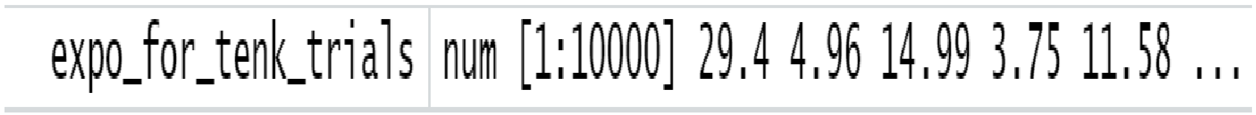


```
> expo_for_one_trial <-max(rexp(n=1,rate=0.1),rexp(n=1,rate=0.1))
> print(expo_for_one_trial)
[1] 6.865402
> 
```

ii)
Avoiding for loop and using replicate function and writing in one line

Replicate will do for loop job and using the same above equation used for one trial

```
expo_for_tenk_trials<-
replicate(10000,max(rexp(n=1,rate=0.1),rexp(n=1,rate=0.1)))
```



```
expo_for_tenk_trials num [1:10000] 29.4 4.96 14.99 3.75 11.58 ...
```

iii)

Histogram is plotted using hist function.
As this is probability we put probability=True, as by default hist calculates frequency, here we get total sum of probabilities times the width should equal to 1.

```
hist(expo_for_tenk_trials,probability = TRUE,xlab='Lifetime T',ylab='Prob_freq', main = "")
```

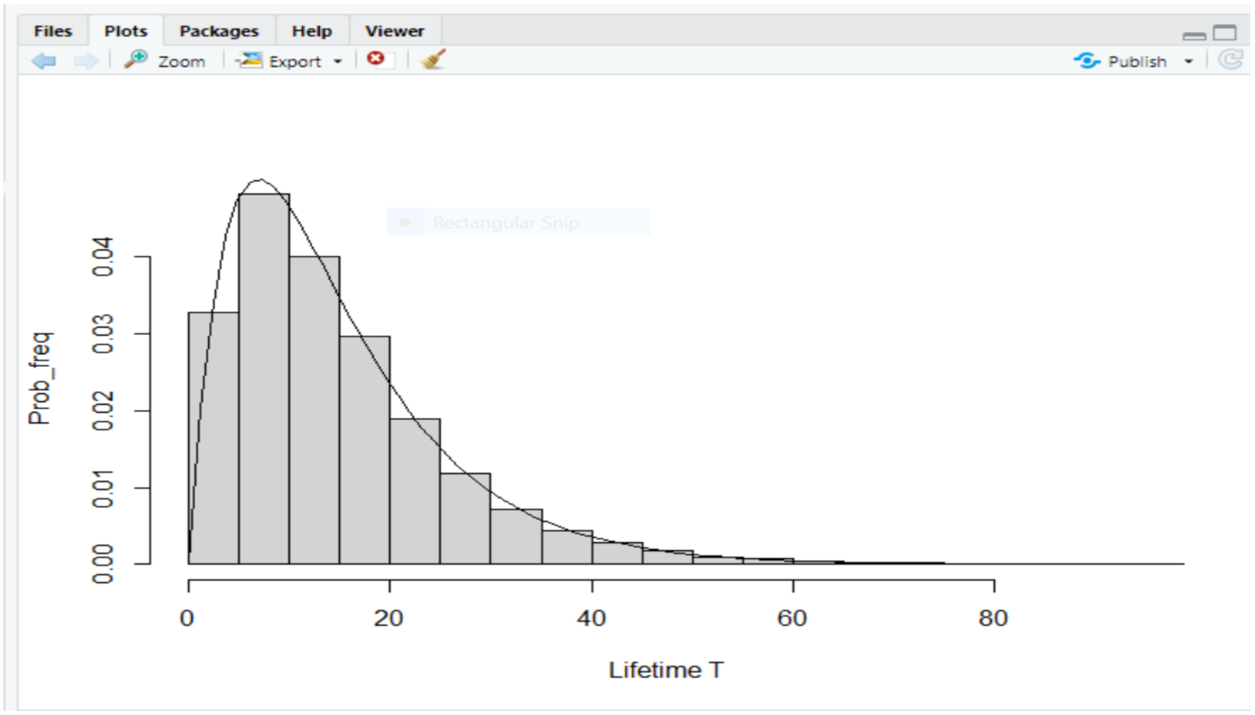
Below code is to find the probability density of exponential distribution and to check for a random example

```
expo_distribution <- function(t){return (0.2*exp(-0.1*t) - 0.2*exp(-0.2*t))}
```

#curve function is used to draw a curve for the equation specified in the argument
#If we put add = True it superimposes on the already existing plot with the default limiters

```
curve(expo_distribution, from = 0, to = max(x), add = TRUE)
```

Result -> We see that both the simulation and the density plot go hand in hand are almost identical in nature.



iv)

We use mean function to get mean of 10k trials and compare.

mean_of_10k_trials <- mean(expo_for_tenk_trials)

print(mean_of_10k_trials)

Result-> We see that the simulated mean and analytically computed mean are very close though not exact

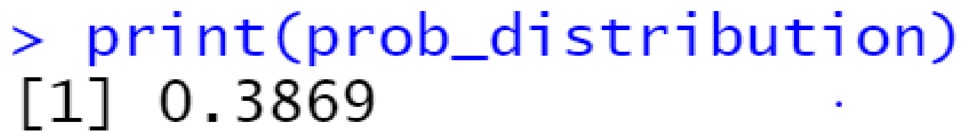
```
> #We use mean function to get mean of 10k trials and compare
> mean_of_10k_trials <- mean(expo_for_tenk_trials)
> print(mean_of_10k_trials)
[1] 14.80413
```

v)

We have 10k trials and we have to check how many are greater than 15 and take average, so sum the values greater than 15 and divide by 10000, that would just be the mean of trials greater than 15

```
prob_distribution <-mean(expo_for_tenk_trials > 15)
print(prob_distribution)
```

Result-> we see that both in a) and the above simulation it is almost the same

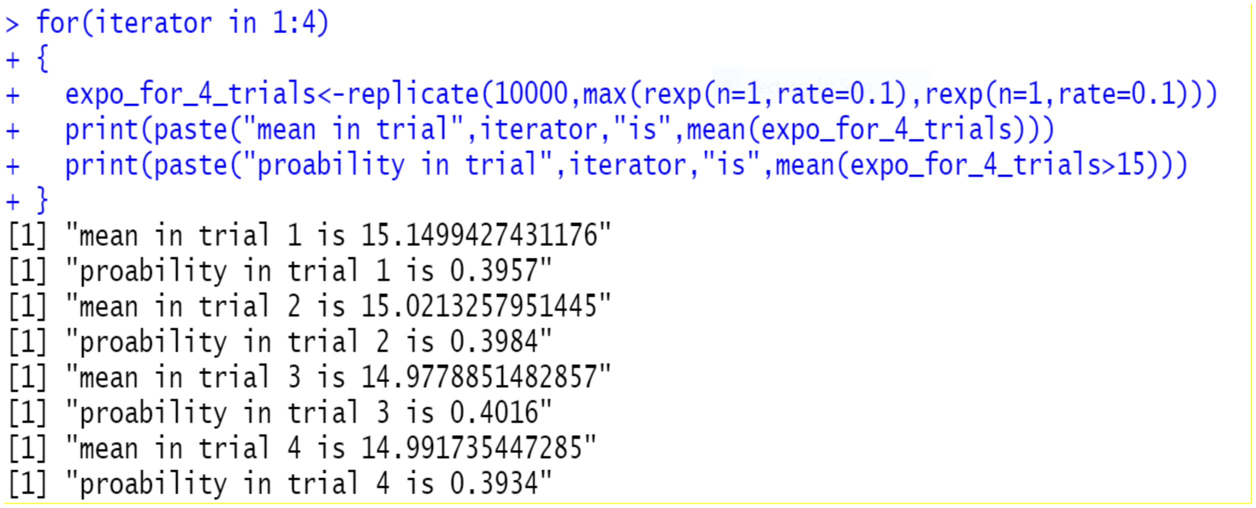


```
> print(prob_distribution)
[1] 0.3869
```

vi)

4 more trials
we use for loop to iterate through this. print and paste function to print result

```
for(iterator in 1:4)
{
  expo_for_4_trials<-
replicate(10000,max(rexp(n=1,rate=0.1),rexp(n=1,rate=0.1)))
print(paste("mean in
trial",iterator,"is",mean(expo_for_4_trials)))
print(paste("probability in
trial",iterator,"is",mean(expo_for_4_trials>15)))
}
```

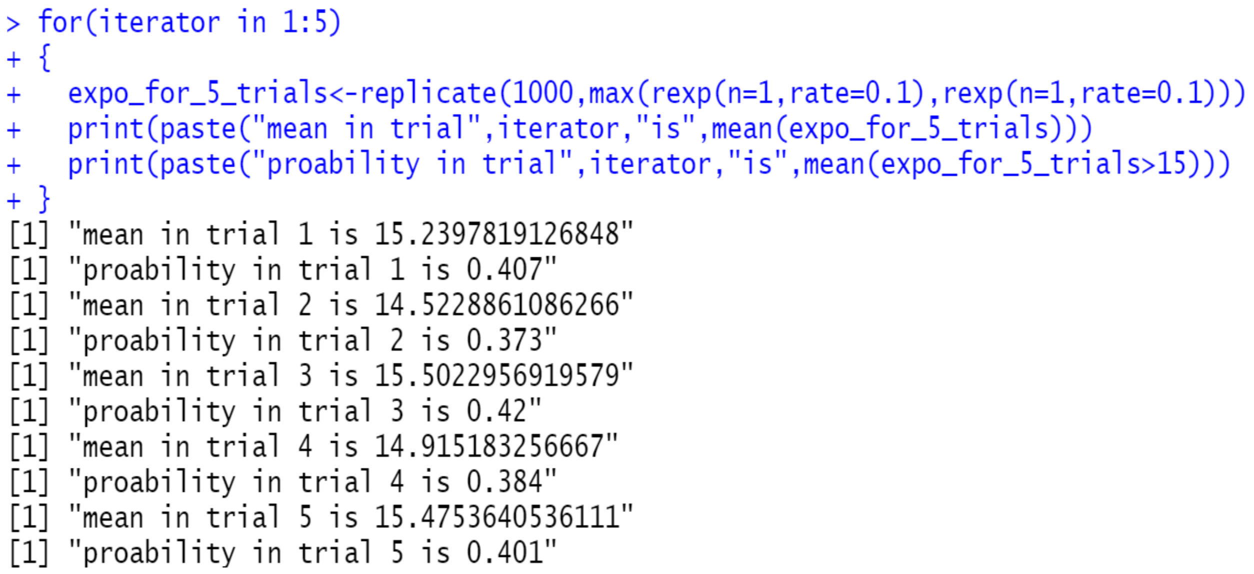


```
> for(iterator in 1:4)
+ {
+   expo_for_4_trials<-replicate(10000,max(rexp(n=1,rate=0.1),rexp(n=1,rate=0.1)))
+   print(paste("mean in trial",iterator,"is",mean(expo_for_4_trials)))
+   print(paste("proability in trial",iterator,"is",mean(expo_for_4_trials>15)))
+ }
[1] "mean in trial 1 is 15.1499427431176"
[1] "proability in trial 1 is 0.3957"
[1] "mean in trial 2 is 15.0213257951445"
[1] "proability in trial 2 is 0.3984"
[1] "mean in trial 3 is 14.9778851482857"
[1] "proability in trial 3 is 0.4016"
[1] "mean in trial 4 is 14.991735447285"
[1] "proability in trial 4 is 0.3934"
```

c)

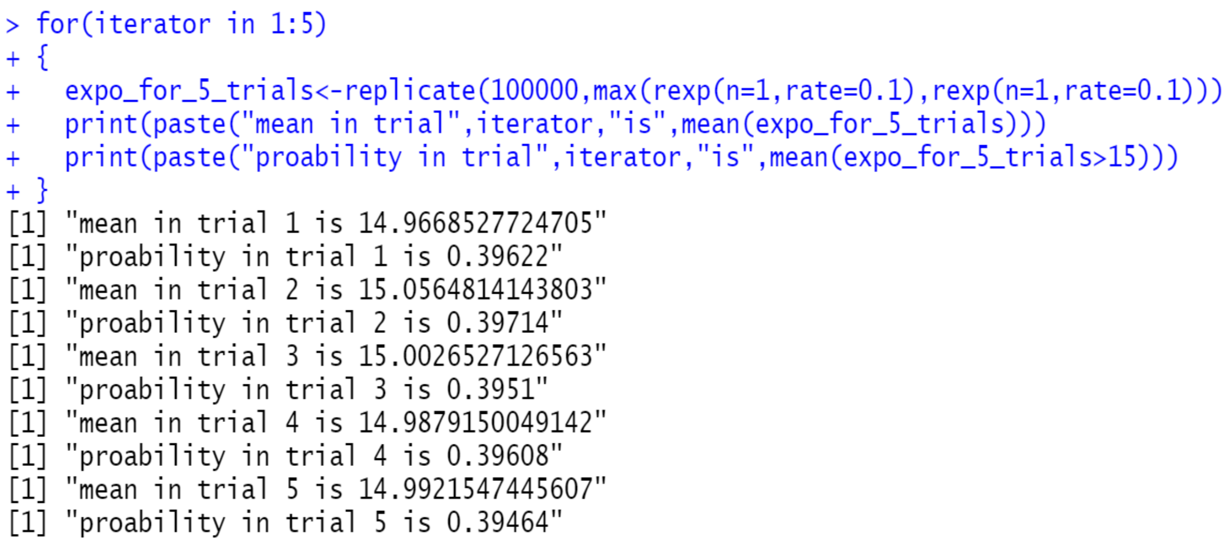
Here we do 5 iterations for both 1k and 100k using the same method as above

```
for(iterator in 1:5)
{
expo_for_5_trials<-
replicate(1000,max(rexp(n=1,rate=0.1),rexp(n=1,rate=0.1)))
print(paste("mean in trial",iterator,"is",mean(expo_for_5_trials)))
print(paste("probability in
trial",iterator,"is",mean(expo_for_5_trials>15)))
}
```



```
> for(iterator in 1:5)
+ {
+   expo_for_5_trials<-replicate(1000,max(rexp(n=1,rate=0.1),rexp(n=1,rate=0.1)))
+   print(paste("mean in trial",iterator,"is",mean(expo_for_5_trials)))
+   print(paste("proability in trial",iterator,"is",mean(expo_for_5_trials>15)))
+ }
[1] "mean in trial 1 is 15.2397819126848"
[1] "proability in trial 1 is 0.407"
[1] "mean in trial 2 is 14.5228861086266"
[1] "proability in trial 2 is 0.373"
[1] "mean in trial 3 is 15.5022956919579"
[1] "proability in trial 3 is 0.42"
[1] "mean in trial 4 is 14.915183256667"
[1] "proability in trial 4 is 0.384"
[1] "mean in trial 5 is 15.4753640536111"
[1] "proability in trial 5 is 0.401"
```

```
for(iterator in 1:5)
{
expo_for_5_trials<-
replicate(100000,max(rexp(n=1,rate=0.1),rexp(n=1,rate=0.1)))
print(paste("mean in trial",iterator,"is",mean(expo_for_5_trials)))
print(paste("probability in
trial",iterator,"is",mean(expo_for_5_trials>15)))
}
```



```
> for(iterator in 1:5)
+ {
+   expo_for_5_trials<-replicate(100000,max(rexp(n=1,rate=0.1),rexp(n=1,rate=0.1)))
+   print(paste("mean in trial",iterator,"is",mean(expo_for_5_trials)))
+   print(paste("proability in trial",iterator,"is",mean(expo_for_5_trials>15)))
+ }
[1] "mean in trial 1 is 14.9668527724705"
[1] "proability in trial 1 is 0.39622"
[1] "mean in trial 2 is 15.0564814143803"
[1] "proability in trial 2 is 0.39714"
[1] "mean in trial 3 is 15.0026527126563"
[1] "proability in trial 3 is 0.3951"
[1] "mean in trial 4 is 14.9879150049142"
[1] "proability in trial 4 is 0.39608"
[1] "mean in trial 5 is 14.9921547445607"
[1] "proability in trial 5 is 0.39464"
```

Trials	1000 Trial		10000 Trial		100000 Trial	
1	E(T)	15.239	E(T)	15.149	E(T)	14.966
	P(T>15)	0.407	P(T>15)	0.395	P(T>15)	0.396
2	E(T)	14.522	E(T)	15.021	E(T)	15.056
	P(T>15)	0.373	P(T>15)	0.398	P(T>15)	0.397
3	E(T)	15.502	E(T)	14.977	E(T)	15.002
	P(T>15)	0.42	P(T>15)	0.401	P(T>15)	0.395
4	E(T)	14.915	E(T)	14.991	E(T)	14.987
	P(T>15)	0.384	P(T>15)	0.3934	P(T>15)	0.396
5	E(T)	15.475	E(T)		E(T)	14.992
	P(T>15)	0.401	P(T>15)		P(T>15)	0.394

#Result-> When we take n =1000 draws we see that both mean and probability vary a bit too much from the actual results but 10k and 100k results are very near to the actual ones and we can infer that as the simulations increases the accuracy increases and after a point the values accuracy doesn't improve even after we significantly improve the results, as we see from 10k and 100k simulations which have the same error from the actual mean and probability.

2nd question

Let's consider a circle with radius r, inscribed inside the square with with radius #2r(a bit bigger than the circle), then area of the circle is

Area of circle = pi x r^2
Area of the square is (2 x r)^2 = 4r^2
The ratio of circle to square is
Ratio = Area of circle/Area of square
Ratio = pi/4

It holds true for all the points, likewise in the 2nd question for unit square with #indices (0,0),(0,1),(1,0),(1,1).Now when we simulate it for let's say 100 trials we #should be doing

*100 * (pi./4)*
As in the question that is given for 10k trials then we should do

*10000 * (pi/4)*

Circle equation is:

(x-p)^2 + (y-q)^2 = r^2,

here r is radius and p,q are center points.

We have a unit circle therefore p and q will be 0.5 each.
For a point to be on the circle it should be either inside the

circle or on the circular diameter therefore the equation becomes

sqrt((x-p)^2 + (y-q)^2) <= r.

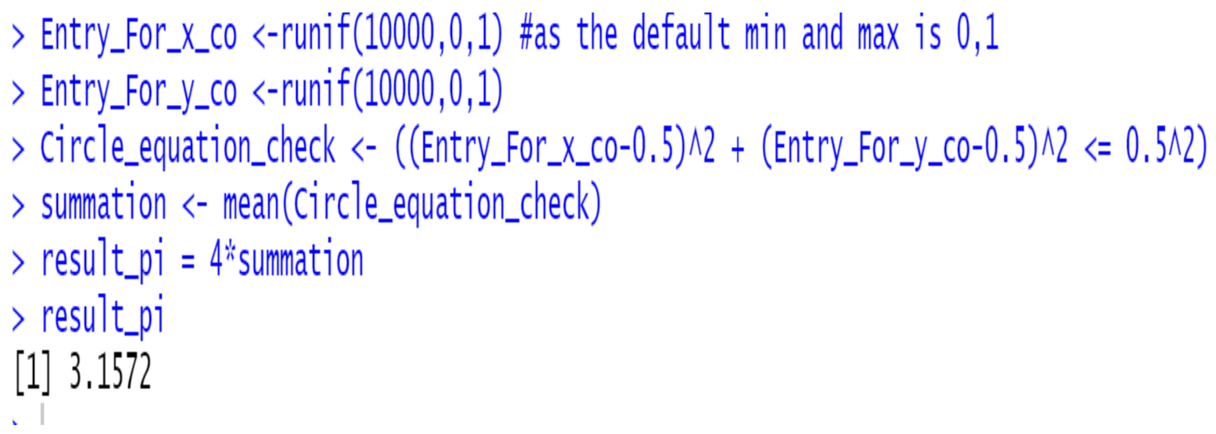
Then after we do the 10k random generations sum it and divide it by 10k to get the mean.

*res = 10000 * (pi/4)*

we can rearrange it as

*pi = (res/10000) * 4.*

```
Entry_For_x_co <-runif(10000,0,1) #as the default min and max is 0,1
Entry_For_y_co <-runif(10000,0,1)
Circle_equation_check <- ((Entry_For_x_co-0.5)^2 + (Entry_For_y_co-
0.5)^2 <= 0.5^2)
summation <- mean(Circle_equation_check)
result_pi = 4*summation
result_pi
```



```
> Entry_For_x_co <-runif(10000,0,1) #as the default min and max is 0,1
> Entry_For_y_co <-runif(10000,0,1)
> Circle_equation_check <- ((Entry_For_x_co-0.5)^2 + (Entry_For_y_co-0.5)^2 <= 0.5^2)
> summation <- mean(Circle_equation_check)
> result_pi = 4*summation
> result_pi
[1] 3.1572
```

#Result-> The value thus obtained(3.1572), is very close to the irrational pi value that is known.

Full Project Code

```
expo_distribution(10)
#from equation in the 1st question for cdf
expo_cdf <- function(t,lam){return (1-exp(-(lam*t)))^2}
print(1- expo_cdf(15,0.1))

#QUESTION 1)b

#i)

#one draw of Xa and Xb to get T, we have rexp formula to get the
random deviates
#Taking one trial of simulation. rexp takes n i,e observations and rate is
the frequency that is 1/mean we get 0.1
#we use max function so as to get the max life of xa or xb as both are
independent it would be a product
expo_for_one_trial <-max(rexp(n=1,rate=0.1),rexp(n=1,rate=0.1))
print(expo_for_one_trial)
#Result->As we do this we get fluctuating results
1-pexp(15,rate=1/mean(expo_for_one_trial))
```

```
#avoiding for loop and using replicate function and writing in one line
#replicate will do for loop job and using the same above equation used
for one trial
expo_for_tenk_trials<-
replicate(10000,max(rexp(n=1,rate=0.1),rexp(n=1,rate=0.1)))
print(expo_for_tenk_trials)
mean(expo_for_tenk_trials) #gives mean not equal to 15, as the trials are
higher it deviates
var(expo_for_tenk_trials) #variance in frequencies is nearly 116


#iii)

#Histogram is plotted using hist function.
#As this is probability we put probability=True,as by default hist
calculates frequency,here we get total sum of probabilites
#times the width should equal to 1.
hist(expo_for_tenk_trials,probability = TRUE,xlab='Lifetime
T',ylab='Prob_freq', main = "")
#Below code is to find the probability density of exponential distribution
and to check for a random example
expo_distribution <- function(t){return (0.2*exp(-0.1*t) - 0.2*exp(-
0.2*t))}
#curve function is used to draw a curve for the equation specified in the
arguement
#If we put add = True it superimposes on the already existing plot with
the default limiters
curve(expo_distribution, from = 0, to = max(x), add = TRUE)
#Result -> We see that both the simulation and the density plot go hand
in hand are almost identical in nature.
```

```
#iv)

#We use mean function to get mean of 10k trials and compare
mean_of_10k_trials <- mean(expo_for_tenk_trials)
print(mean_of_10k_trials)
#Result-> We see that the simulated mean and analytically computed
mean are very close though not exact

# v)
#We have 10k trials and we have to check how many are greater than 15
and take average
#so sum the values grater than 15 and divide by 10000, that would just
be the mean of trials greater than 15
prob_distribution <-mean(expo_for_tenk_trials > 15)
print(prob_distribution)
#Result-> we see that both in a) and the above simulation it is the same
print(1-pexp(15,rate=1/mean(expo_for_tenk_trials)))
x <- replicate(10000, max(rexp(1, 0.1), rexp(1, 0.1)))
# make a histogram
hist(x, prob = T, ylim = c(0, 0.05), xlab = "t", ylab = "probability", main
= "")
# superimpose the density function of T
# a function to compute the density function of T
density.fun <- function(x){
return(0.2*exp(-0.1*x) - 0.2*exp(-0.2*x))
}
curve(density.fun, from = 0, to = max(x), add = T)
#Testing how mean and variance change for another simulation
mean(expo_for_tenk_trials)
1-pexp(15,rate=1/mean(expo_for_tenk_trials))
```

```
#vi)

#4 more trials
#we use for loop to iterate through this. print and paste funtion to print
result
for(iterator in 1:4)
{
expo_for_4_trials<-
replicate(10000,max(rexp(n=1,rate=0.1),rexp(n=1,rate=0.1)))
print(paste("mean in trial",iterator,"is",mean(expo_for_4_trials)))
print(paste("proability in
trial",iterator,"is",mean(expo_for_4_trials>15)))
}
#Result-> We see that mean and probability of the trial hovers very near
to the actual value of 15 for mean and 0.396 for probability
#with a slight error of

#c)

#Here we do 5 iterations for both 1k and 100k using the same method as
above
for(iterator in 1:5)
{
expo_for_5_trials<-
replicate(1000,max(rexp(n=1,rate=0.1),rexp(n=1,rate=0.1)))
print(paste("mean in trial",iterator,"is",mean(expo_for_5_trials)))
print(paste("proability in
trial",iterator,"is",mean(expo_for_5_trials>15)))
}
for(iterator in 1:5)
{
expo_for_5_trials<-
replicate(100000,max(rexp(n=1,rate=0.1),rexp(n=1,rate=0.1)))
print(paste("mean in trial",iterator,"is",mean(expo_for_5_trials)))
print(paste("proability in
trial",iterator,"is",mean(expo_for_5_trials>15)))
}
#we see that mean and variance var a bit but are very close to
analytically computed mean.
expo_for_tenk_trials_2<-
replicate(10000,max(rexp(n=1,rate=0.1),rexp(n=1,rate=0.1)))
expo_for_tenk_trials_2
hist(expo_for_tenk_trials,xlab='t',ylab='Prob_freq')
mean(expo_for_tenk_trials_2)
var(expo_for_tenk_trials_2)
pdf_of_10k <- function(t) {return (0.2*exp(-0.1*t) - 0.2*exp(-0.2*t))}
curve(pdf_of_10k,add=TRUE)



#2nd question

#Let's consider a circle with radius r, inscribed inside the square with
with radius 2r(a bit bigger than the circle), then area
#of the circle is pi x r^2 and area of the square is (2 x r)^2 = 4r^2 and the
ratio of circle to square is pi/4.It holds true
#for all the points, likewise in the 2nd question for unit square with
indices (0,0),(0,1),(1,0),(1,1).Now when I simulate it for let's
#say 100 trials i should be doing 100 x (pi./4) and as in the question that
is given for 10k trials then I should do 10000 x (pi/4)
#Circle equation is (x-p)^2 + (y-q)^2 = r^2, here r is radius and p,q are
center points.
#we have a unit circle therefore p and q will be 0.5 each
#For a point to be on the circle it should be either inside the circle or on
the circular diameter
#therefore the equation becomes sqrt((x-p)^2 + (y-q)^2) <= r
#Then after we do the 10k random generations sum it and divide it by
10k to get the the mean
#as res = 10000 x (pi/4) we can rearrange pi = (res/10000) * 4
```

```
Entry_For_x_co <-runif(10000,0,1) #as the default min and max is 0,1
Entry_For_y_co <-runif(10000,0,1)
Circle_equation_check <- ((Entry_For_x_co-0.5)^2 + (Entry_For_y_co-
0.5)^2 <= 0.5^2)
summation <- mean(Circle_equation_check)
result_pi = 4*summation
result_pi
```