Homework 5: The Normal Distribution and the CLT

NAME: Andres Castano

NETID: ac986

DUE DATE: October 19, 2016 by 1:00pm

For this homework, it will be helpful to have a copy of the knitted version of this document to answer the questions as much of it is written using mathematical notation that may be difficult to read when the document is not knitted.

Instructions

For this homework:

1. All calculations must be done within your document in code chunks. Provide all intermediate steps.
2. Include any mathematical formulas you are using for a calculation. Surrounding mathematical expresses by dollar signs makes the math look nicer and lets you use a special syntax (called latex) that allows for Greek letters, fractions, etc. Note that this is not R code and therefore should not be put in a code chunk. You can put these immediately before the code chunk where you actually do the calculation.

Some Notation

Your solutions to the problems below must include the formula used for each calculation. To get you started, here is some mathematical expressions written in latex that you may find helpful when writing out the math in your answers. You can copy, paste, and edit these expressions as needed.

For $X \sim N(\mu, \sigma)$ and real numbers a and b:

1) $P(X \leq b) = P(Z \leq (b - \mu)/\sigma)$
2) $P(X \geq a) = P(Z \geq (a - \mu)/\sigma)$
3) $P(a \leq X \leq b) = P((a - \mu)/\sigma \leq Z \leq (b - \mu)/\sigma)$

In this homework we will explore the normal distribution and the Central Limit Theorem.

For X $N(\mu, \sigma)$ and an interval (a, b) on the real line,

$$P(X \in (a, b)) = \int_a^b \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(x-\mu)^2} dx$$

(i.e., area under the pdf between a and b). As noted in lecture, this cannot be computed in closed form; however, in R this can be computed numerically.

For $X \sim N(\mu, \sigma)$, the probability of getting a value in any interval on the real line can be expressed solely in terms of the cumulative distribution function, $P(X \leq x)$. For any real numbers, a and b, $a < b$:

1. $P(X < b) = P(X \leq b)$, the probability of getting a value less than (or equal to) b
2. $P(X > a) = P(X \geq a) = 1 - P(X < a)$, the probability of getting a value greater than (or equal to) a
3. $P(a < X < b) = P(X < b) - P(X < a)$, the probability of getting a value between a and b
4. $P(X = a) = 0$, the probability of getting the value a

In R, the `pnorm`(x,μ,σ) function is the cumulative probability distribution function for the normal distribution with mean, μ and standard deviation, σ, evaluated at x, i.e. $P(X \leq x)$ =`pnorm`(x,μ,σ) for $X \sim N(\mu, \sigma)$.

Calculating Probabilities Associated with The Normal Distribution Using z-scores

Every normal distribution, $N(\mu, \sigma)$, can be seen as a shifted and scaled standard normal distribution, $N(0, 1)$.

Assume, $X \sim N(\mu, \sigma)$ and $Z \sim N(0, 1)$. Then

$\frac{X-\mu}{\sigma} \sim Z$.

Thus, every quantile in the sample space of X has a corresponding "standardized" quantile in the sample space of Z (called the z-score). If b is an outcome in the sample space of X, the corresponding standardized value of b in the sample space of Z is

$\frac{b-\mu}{\sigma}$ = z-score for b.

Using z-scores, probabilities for X can be determined by transforming each quantile of X into a standardized quantile and using the probability distribution function for the standard normal distribution. For example, for any real b,

$P(X \leq b) = P(Z \leq \frac{b-\mu}{\sigma})$

In R, the `pnorm(x)` function without a mean or standard deviation specified is the cumulative probability distribution function for the standard normal distribution evaluated at x, i.e. $P(Z \leq z)$ =`pnorm(z)`.

Problem 1

The daily milk production of a Guernsey cow has a normal distribution with $\mu = 70$ pounds and $\sigma = 13$. A Guernsey cow is chosen at random. Let X = Milk production in one day for a Guernsey cow.

For (a) - (c) answer each question in two ways:

1) Using the `pnorm()` function with the mean and standard deviation for X specified.

2) By converting all probabilities in terms of the standard normal distribution and using the `pnorm()` function without the mean and standard deviation specified.

For both (1) and (2), the formula you are using to calculate each probability must be included before the code chunk where the answer is evaluated.

a) What is the probability that a Guernsey cow chosen at random produces more than 90 pounds of milk in a given day?

$P(X > 90) = P(X \geq 90) = 1 - P(X \leq 90)$

Method 1:

```
x <-90
u <-70
sigma <-13
probx <- 1 - pnorm(x, mean=u,sd=sigma)
probx
```

```
## [1] 0.0619679
```

Method 2: requires find the z-score associated with $X \leq 90$

$P(X \leq 90) = P(Z \leq \frac{90-70}{13})$

```
x <-90
u <-70
sigma <-13
z <- (90-70)/(13)
z
```

```
## [1] 1.538462
```

```
prz <- 1 - pnorm(z)
prz
```

```
## [1] 0.0619679
```

b) What is the probability that a Guernsey cow chosen at random produces between 85 and 100 pounds of milk in a given day?

Here we are interested in: $P(85 < X < 100) = P(X < 100) - P(X < 85)$;

Method 1:

```
a <-100
b <-85
u <-70
sigma <-13
proba <- pnorm(a, mean=u,sd=sigma)
proba
```

```
## [1] 0.9894919
```

```
probb <- pnorm(b, mean=u,sd=sigma)
probb
```

```
## [1] 0.8757184
```

```
probx <- proba - probb
probx
```

```
## [1] 0.1137735
```

Method 2: first, we need to fin the z-scores associated with $X \leq 100$ and $X \leq 85$

$P(X \leq 100) = P(Z \leq \frac{100-70}{13})$

$P(X \leq 85) = P(Z \leq \frac{85-70}{13})$

```
a <-100
b <-85
u <-70
sigma <-13
za <- (100-70)/(13)
zb <- (85-70)/(13)
prz <-  pnorm(za) - pnorm(zb)
prz
```

```
## [1] 0.1137735
```

c) What is the probability that the quantity of milk produced by a Guernsey cow chosen at random is within 1.5 standard deviations of the mean number of pounds of milk produced by a Guernsey cow?

Method 1:

If X is between 1.5 standard deviations then we are interested in $P(X < (\mu + 1.5\sigma)) = P(X < (70 + 1.5 * (13)))$

```
u <- 70
sigma <- 13
x <- (u) + 1.5*(sigma)
x
```

```
## [1] 89.5
```

```
prx <- pnorm(x, mean=u,sd=sigma)
prx
```

```
## [1] 0.9331928
```

Method 2: here we need to find the z score associate with X=89.5.

$P(X \leq 89.5) = P(Z \leq \frac{89.5-70}{13})$

```
x <- 89.5
u <- 70
sigma <- 13
zx <- (x-u)/(sigma)
przx <-  pnorm(zx)
przx
```

```
## [1] 0.9331928
```

Problem 2

As we have seen for other probability calculations, we can simulate from a $N(70, 13)$ distribution to estimate the probabilities computed above in Problem 1.

The `rnorm`(n,μ,σ) function in R will simulate n draws from a normal distribution with mean, μ and standard deviation, σ.

i) Include a code chunk here that simulates 10,000 draws from the N(70, 13) distribution to estimate the probability computed in 1c). As in previous code for estimating probabilities through simulation, you will need to count the number of simulated draws that meet the criterion associated with the event defined in 1c) to be able to estimate this probability.

In the question 1.c we were interested in the $P(X \leq 89.5)$, we can do this we the following simulation:

```
set.seed(2)
u <- 70
sigma <- 13
counter = 0
simulations=10000
for (i in 1:simulations) {
  x = rnorm(1,u,sigma)
  if (x  < 89.5) {
    counter=counter+1
  }
}
probxa = counter/simulations
probxa
```

```
## [1] 0.934
```

As we can see with 10000 simulations the probability of get a X<89.5 is around 93.34%, which is pretty close of the original result 93.33%.

Another Way to Simulate Random Draws from $N(70, 13)$

Randomly drawing from $X \sim N(70, 13)$ using `rnorm(n,70,13)` is equivalent to:

1) Drawing randomly from a N(0,1) distribution
2) Multiplying (1) by the standard deviation of X
3) Adding the mean of X to (2)

So, instead of using `rnorm(n,70,13)` to draw from $X \sim N(70, 13)$, you can also use 13 × `rnorm(n)` + 70 to draw from $N(70, 13)$.

ii) Include a code chunk here that estimates the probability of 1c) through simulating 10,000 draws from the standard normal distribution and using the criterion for counting events associated with 1c) determined in (i).

```
set.seed(3)
u <- 70
sigma <- 13
counter = 0
simulations=10000
for (i in 1:simulations) {
  x = (sigma*rnorm(1)) + u
  if (x  < 89.5) {
    counter=counter+1
  }
  }
probxb = counter/simulations
probxb
```

```
## [1] 0.9324
```

Using the second strategy we also found a pretty close probability (93.24%) to the probability estimated in 1.c (93.33%)

Problem 3

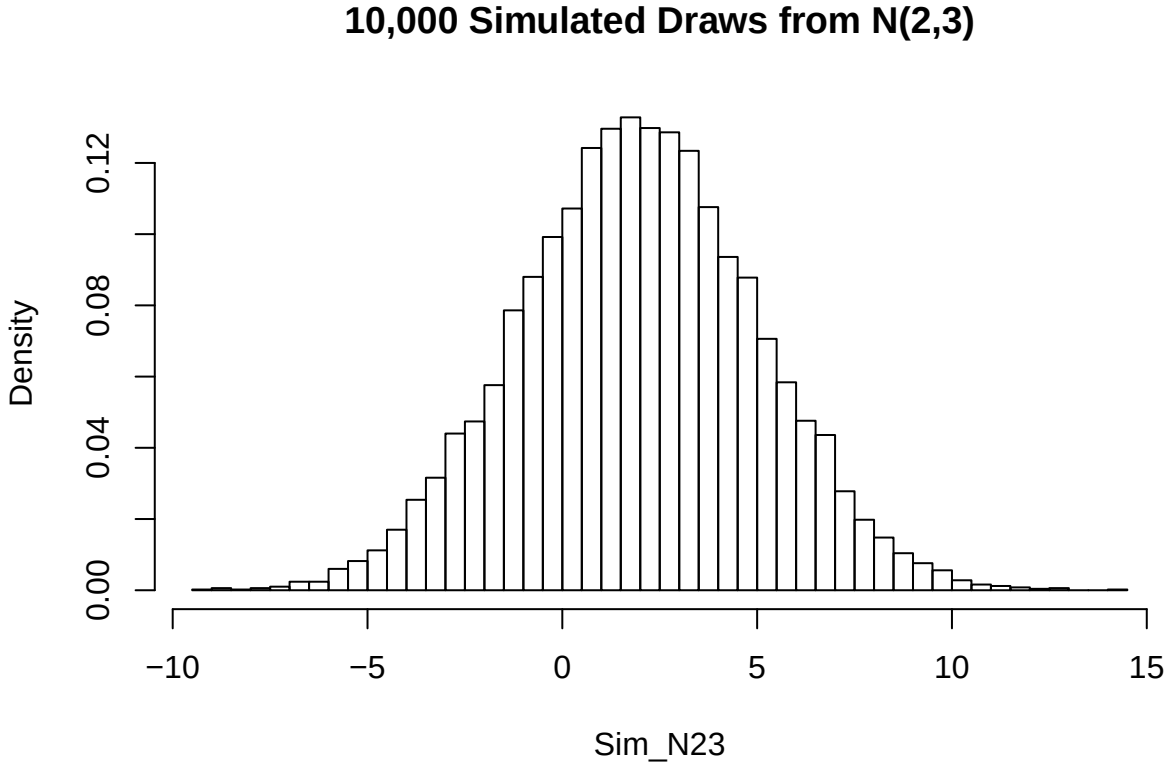
Here we will take a look at how an entire probability density function can be approximated by simulation.

i) In a code chunk here simulate 10,000 draws from a $N(2, 3)$ distribution. Call these simulated values, `Sim_N23`.

```
u <- 2
sigma <- 3
n <- 10000
Sim_N23 = rnorm(n,u,sigma)
```

ii) In another code chunk, create a probability histogram of `Sim_N23` using `hist()`. Remember to set `freq=FALSE`. Set `breaks = 75` and `main= '10,000 Simulated Draws from N(2,3)'`.

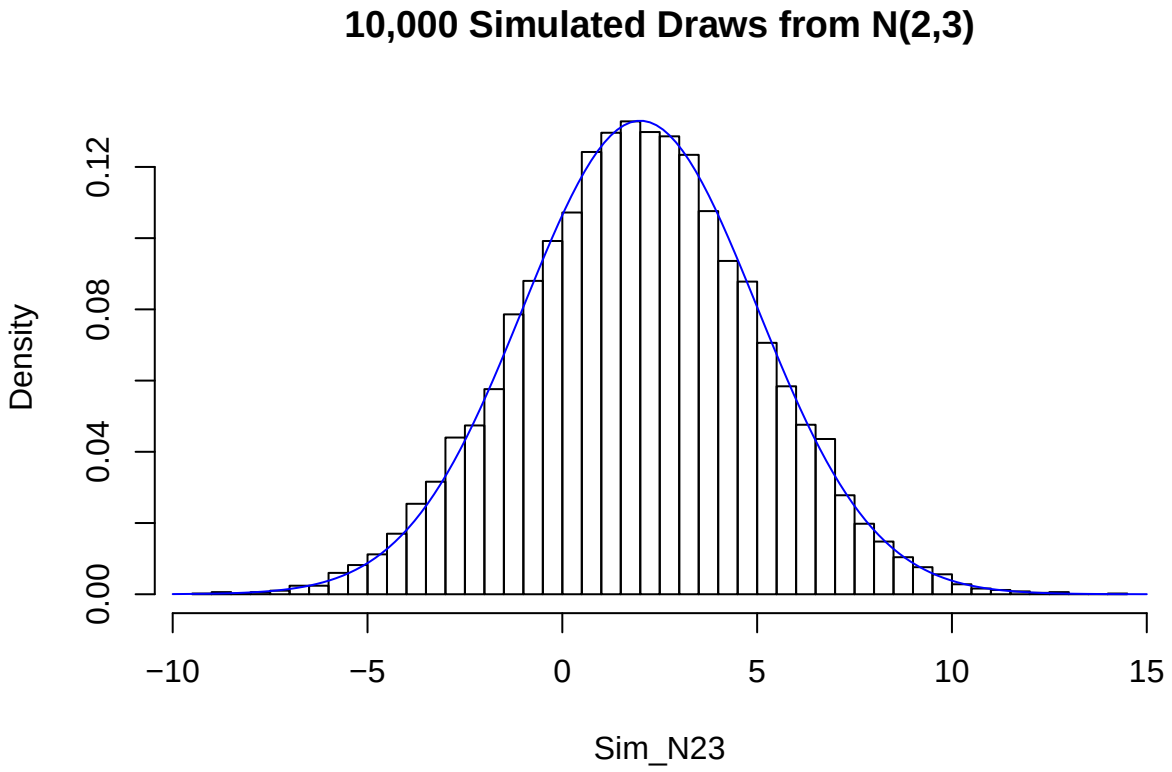
```
hist(Sim_N23, breaks = 75, freq=FALSE,
main= "10,000 Simulated Draws from N(2,3)")
```



iii) The R function `lines(x,y)` will add a line to an existing plot. The arguments, `x` and `y`, are vectors of x and y coordinates that together define all the points the line must run through. In the code chunk you used to create the histogram above, we will now add code to overlay the histogram with the probability density function for the $N(2,3)$ distribution. You can do this in three lines of code:

   a) Define the x coordinates as `xvalues=seq(-10, 15, length=150)`. `xvalues` will include an evenly spaced grid of x coordinates.
   b) Assign the output of `dnorm(xvalues, 2, 3)` to a vector named `yvalues`.
   c) Use the `lines()` function to overlay the histogram created in (ii) with the probability density function for $N(2,3)$. In particular `lines(xvalues, yvalues)` should do the trick.

```
hist(Sim_N23, breaks = 75, freq=FALSE,  main= "10,000 Simulated Draws from N(2,3)")
xvalues=seq(-10, 15, length=150)
yvalues=dnorm(xvalues, 2, 3)
lines(xvalues,yvalues, col="blue")
```



Problem 4

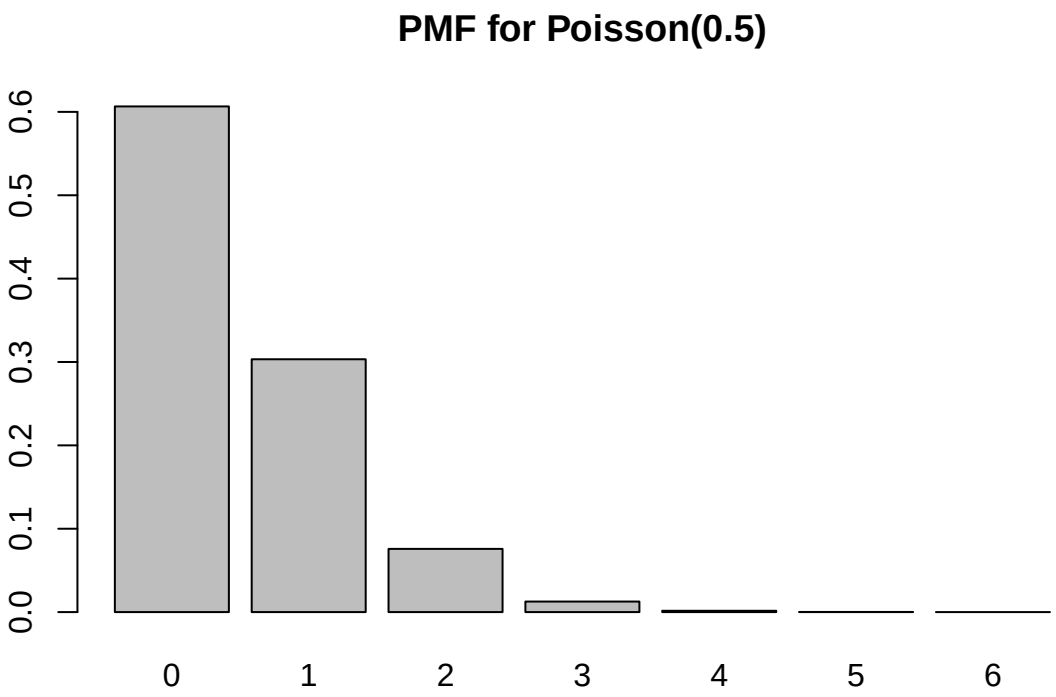
In lecture we saw that the sampling distribution of the sample mean when you draw simple random samples seemed to look like a normal. This is a general phenomenon, known as the Central Limit Theorem. In particular, if $X_i, i = 1, \ldots, n$ is an independent random sample from just about *any* distribution with mean, μ and standard deviation σ, then for large enough n,

$$\bar{X}_n \approx N(\mu, \sigma/\sqrt{n}).$$

Here we will illustrate the Central Limit Theorem by simulating independent draws from the X $\sim$ Poisson(λ = .5) distribution.

a) We start by looking at the PMF of a Poisson(0.5). Change the code from `eval=FALSE` to `eval=TRUE` once you make sure you understand every line of the code chunk.

```
probs = dpois(c(0:6), 0.5)
barplot(probs,names.arg = c(0,1,2,3,4,5,6),main='PMF for Poisson(0.5)')
```



b) Using one or two lines in a code chunk, generate a realization of the sample mean of 4 independent draws from Poisson(.5). You can generate the sample using, `rpois(4,.5)`.

```
draws=rpois(4,.5)
x4mean=mean(draws)
```

c) To simulate draws from the sampling distribution of $\bar{X}_4$, we want to repeat (b) many times, storing the results in a vector.

i) In a code chunk, define the vector of sample means as `xbar.realizations = rep(NA,num.simulations)` where `num.simulations = 10000`.

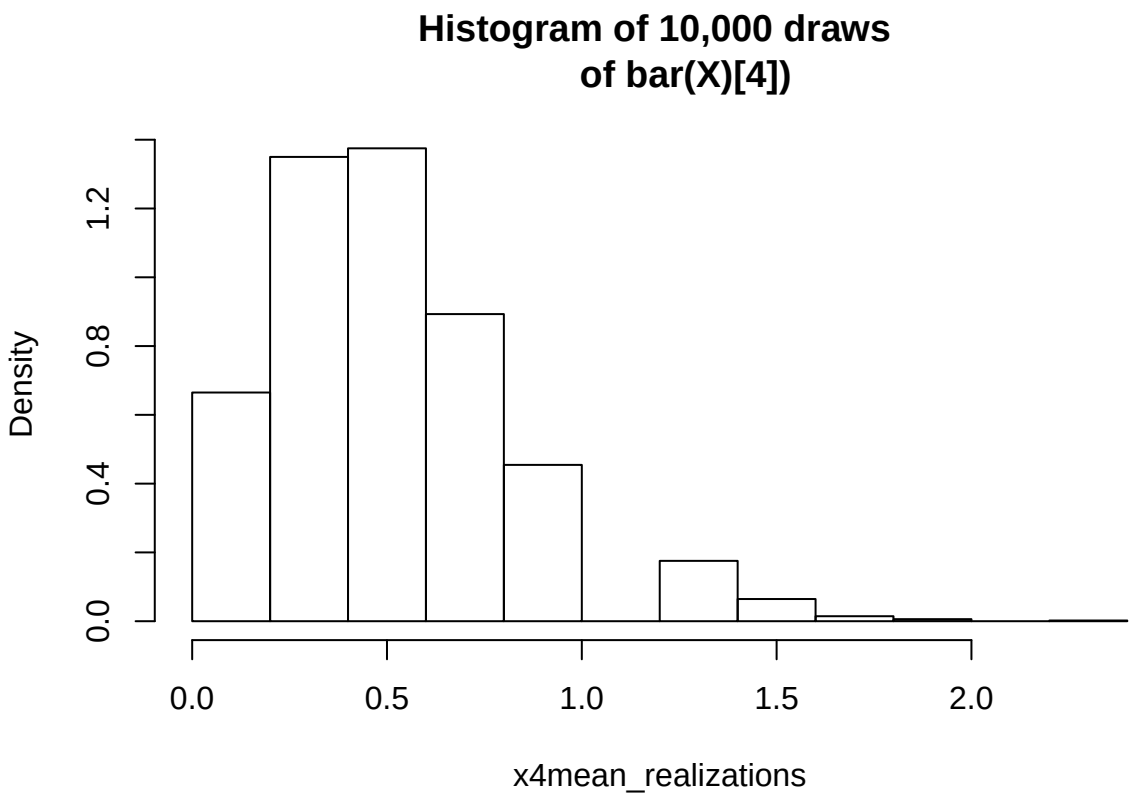
```
simulations=10000
x4mean_realizations=rep(NA,simulations)
```

ii) In this code chunk, write a `for` loop that repeats part (b) 10,000 times, storing the sample mean generated in iteration i as `xbar.realizations[i]`. `xbar.realizations` will now contain 10,000 draws from the sampling distribution of $\bar X_4$.

```
set.seed(6)
simulations=10000
x4mean_realizations=rep(NA,simulations)
for (i in 1:simulations) {
draws=rpois(4,.5)
x4mean_realizations[i]=mean(draws)
}
```

d) Create a probability histogram of `xbar.realizations`. Set `main=expression(paste('Histogram of 10,000 Draws of ',bar(X)[n]))`.

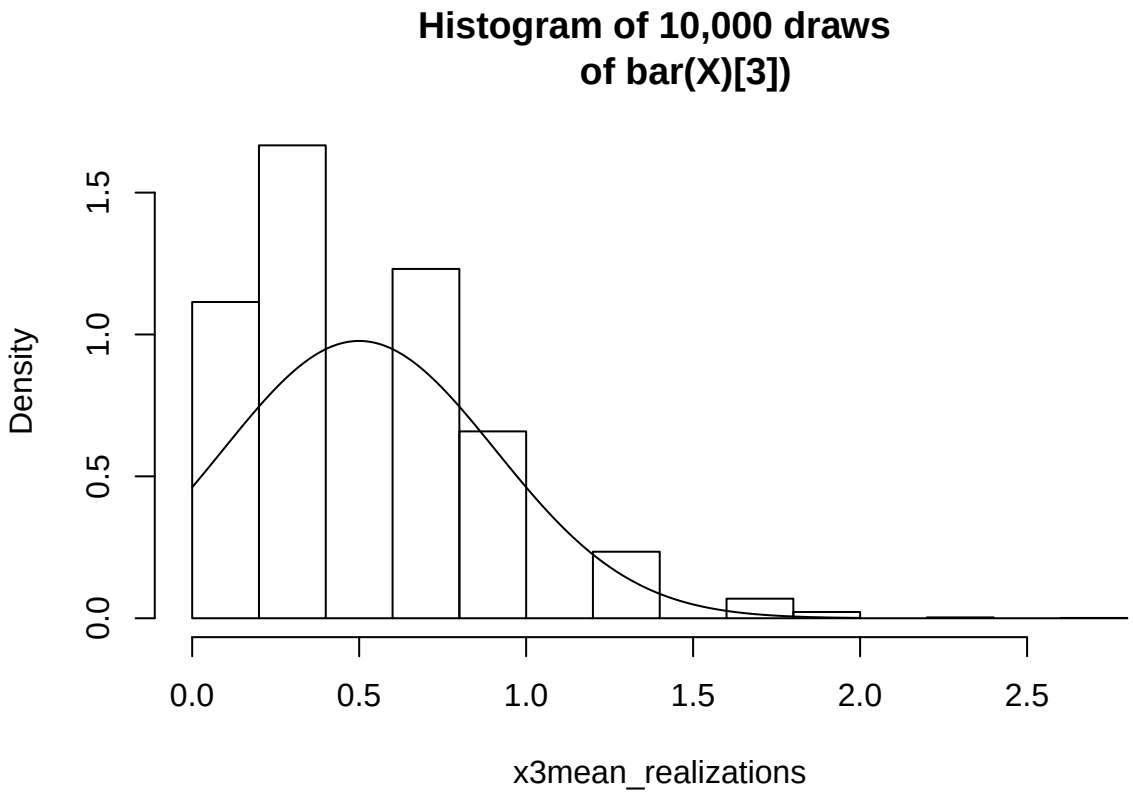
```
hist(x4mean_realizations, freq=FALSE, main="Histogram of 10,000 draws
     of bar(X)[4])" )
```



e) Include a code chunk here that defines, `n=3`. In this code chunk, include code that repeats parts (b) - (d) using a random sample of `n` observations instead of 4. Also, as in Problem 3iii), overlay the probability histogram that is created in the last step with the probability density function for a $N(0.5, \sqrt{0.5/n})$ distribution. To do so, use the following:

```
xvalues = seq(0,2,.01)
yvalues = dnorm(xvalues,.5, sqrt(0.5/n))
lines(xvalues,yvalues)
```

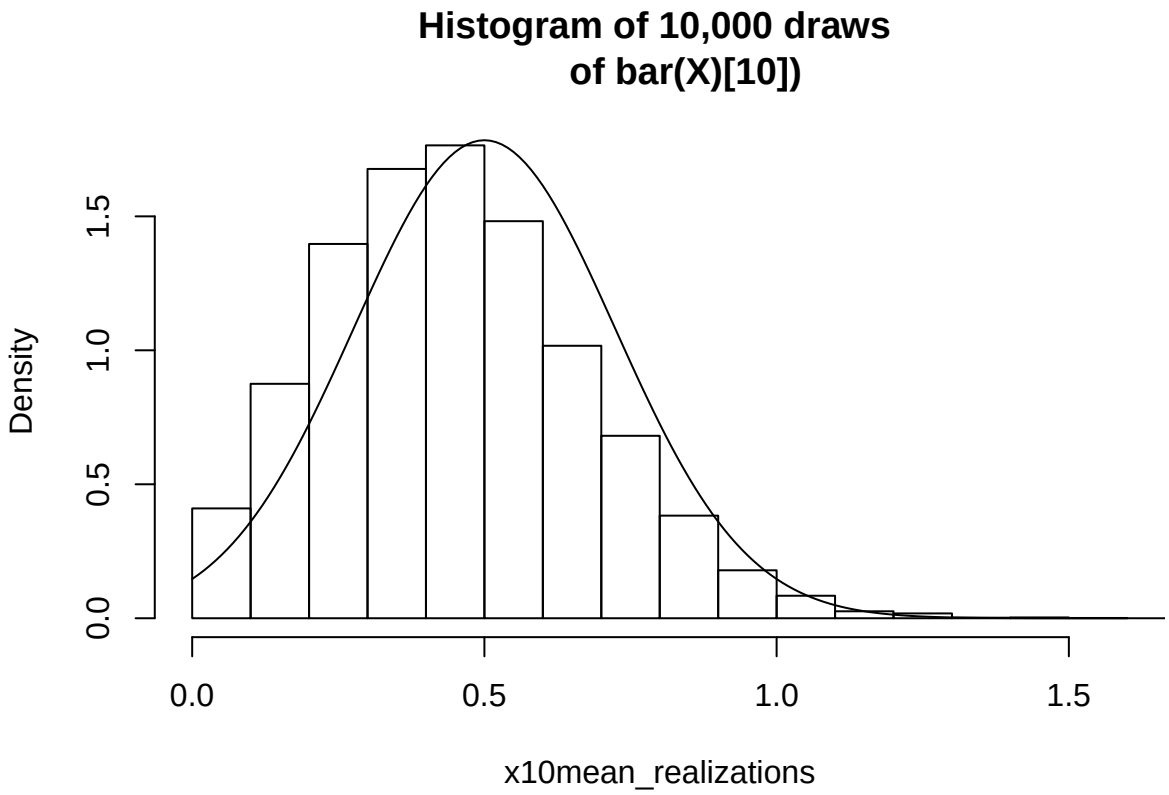
```
set.seed(7)
n=3
simulations=10000
x3mean_realizations=rep(NA,simulations)
for (i in 1:simulations) {
draws=rpois(n,.5)
x3mean_realizations[i]=mean(draws)
}
hist(x3mean_realizations, freq=FALSE, main="Histogram of 10,000 draws
     of bar(X)[3])")
xvalues = seq(0,2,.01)
yvalues = dnorm(xvalues,.5, sqrt(0.5/n))
lines(xvalues,yvalues)
```



f) Repeat part (e) three times. The first time, set `n=10`. The second time, set `n=30`. The third time, set `n=50`. Be sure you include the probability histogram overlaid with the pdf for $N(0.5, \sqrt{0.5/n})$ for every choice of `n`.

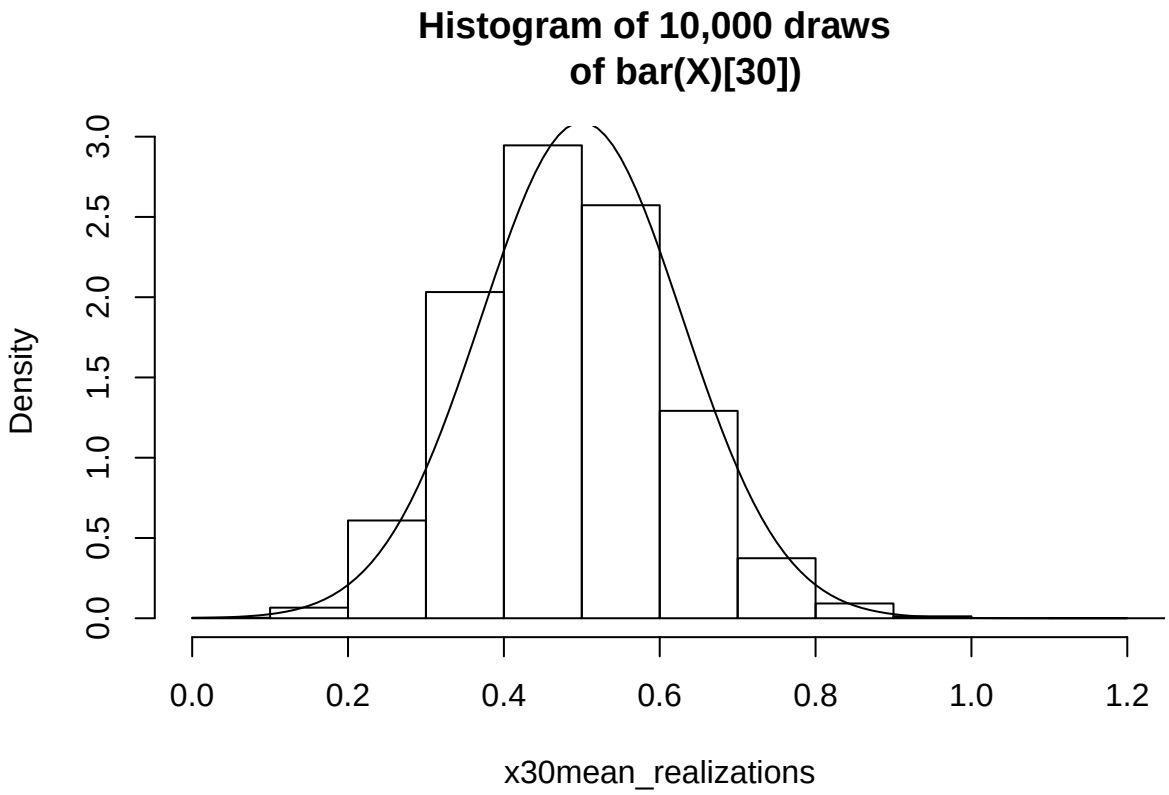
Case 1: n=10

```
set.seed(8)
n=10
simulations=10000
x10mean_realizations=rep(NA,simulations)
for (i in 1:simulations) {
draws=rpois(n,.5)
x10mean_realizations[i]=mean(draws)
}
hist(x10mean_realizations, freq=FALSE, main="Histogram of 10,000 draws
     of bar(X)[10])")
xvalues = seq(0,2,.01)
yvalues = dnorm(xvalues,.5, sqrt(0.5/n))
lines(xvalues,yvalues)
```



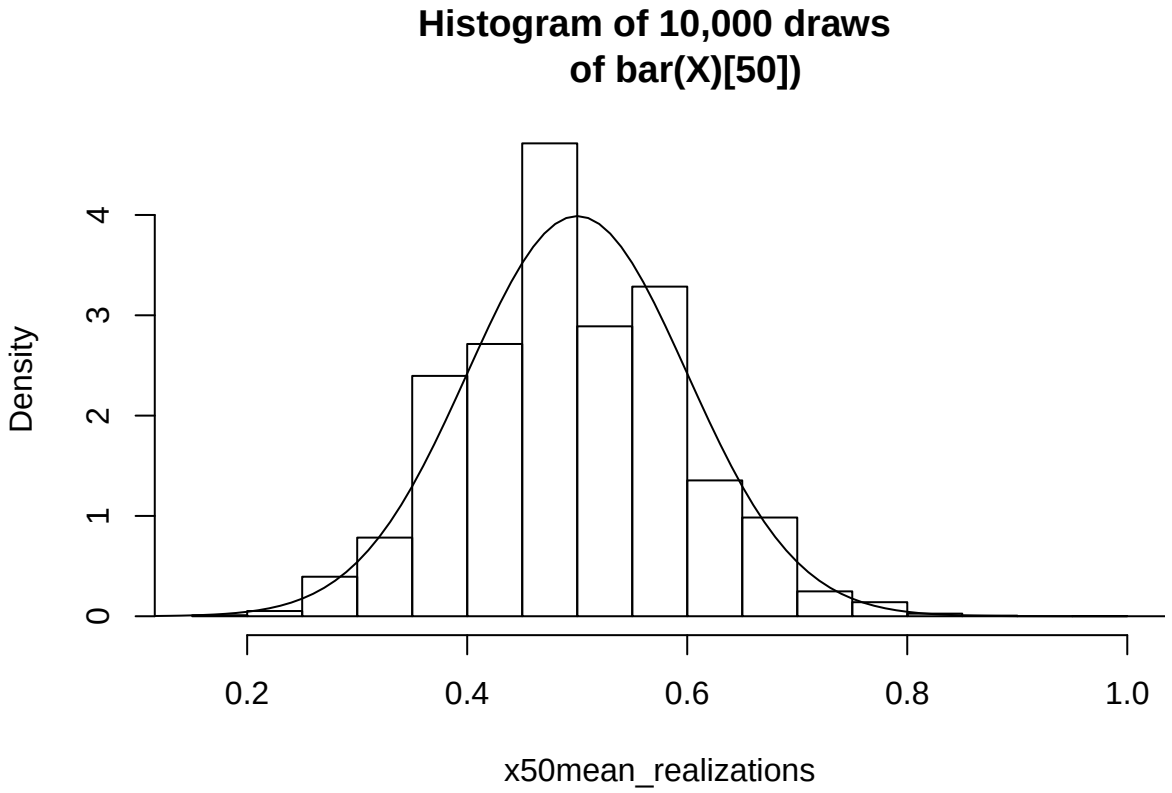
Case 2: n=30

```
set.seed(9)
n=30
simulations=10000
x30mean_realizations=rep(NA,simulations)
for (i in 1:simulations) {
draws=rpois(n,.5)
x30mean_realizations[i]=mean(draws)
}
hist(x30mean_realizations, freq=FALSE, main="Histogram of 10,000 draws
     of bar(X)[30])")
xvalues = seq(0,2,.01)
yvalues = dnorm(xvalues,.5, sqrt(0.5/n))
lines(xvalues,yvalues)
```



Case 3: n=50

```
set.seed(10)
n=50
simulations=10000
x50mean_realizations=rep(NA,simulations)
for (i in 1:simulations) {
draws=rpois(n,.5)
x50mean_realizations[i]=mean(draws)
}
hist(x50mean_realizations, freq=FALSE, main="Histogram of 10,000 draws
     of bar(X)[50])")
xvalues = seq(0,2,.01)
yvalues = dnorm(xvalues,.5, sqrt(0.5/n))
lines(xvalues,yvalues)
```



As we can see the above results suggest that if $X_1, ..., X_n$ are independent draws from a distribution with mean μ (in our case $\mu = \lambda = 0.5$ and standard deviation σ (in our case $\sigma = \sqrt{\lambda} = \sqrt{0.5}$), then for a large n, the sample mean $\bar{X}_n$ is approximately normal with mean μ and standard deviation $\frac{\sigma}{\sqrt{n}}$

$$\bar{X}_n \approx N(\mu, \frac{\sigma}{\sqrt{n}})$$

In our case this result is quite interesting since we were generating all the values of X from a poisson distribution, and results that for a n sufficient large, $\bar{X}_n$ of a Poisson distribution follow a normal distribution $N(\mu = \lambda, \frac{\sigma=\sqrt{\lambda}}{\sqrt{n}})$.