```
#PO 841: Quantitative Research Methods
#Claire Leavitt
# 10/1/2014
#Lab 4


# -----------------
# VARIANCE OF X-BAR
# -----------------

# Let's illustrate one of the theoretical results from lecture--what happens to the
 variance of X-bar as n increases--while practicing graphing and doing simulation
 using the apply() function we went over in Lab #3.

#First, we will tell R that we want to put four smaller graphs on the same screen, 2
 by 2, so we can compare them side by side.

par(mfrow=c(2,2)) #To fill by columns, use mfcol=

#Let's examine the sampling distribution of X-bar, the sample mean, as n (the sample
 size) increases. To do this we are going to do four simulations. The first draws
 1000 samples of size 2; the second, 1000 of size 10; the third, size 50; and the
 fourth, size 100. We generate independent standard normal random variables  using
 the function rnorm(), and organize them in matrices that each have 1000 columns and
 n rows. Then we use apply() to calculate the mean of each column and store that in a
 new vector. Finally, we draw a histogram that will show us the sampling distribution
 of X-bar in each case.

sim1<-matrix(rnorm(2000),nrow=2) #In Lab #3, we specified the parameters of the
 normal distribution (mean and standard deviation) when using rnorm. The default in R
 is to standardize the normal distribution so that mean = 0 and standard deviation =
 1. Unless we specify other parameters, R will assume we are standardizing the
 distribution.
sim1[,1:4] #Inspect the first four columns to a,ke sure everything looks good

xbar.for.n2<-apply(sim1,2,mean) #Each column represents one simulation of 2 draws
 from a standard normal distribution, so specify margin=2
length(xbar.for.n2) #We have 1000 x-bars that we want to show the distribution of.
 We call n=2 here because for each sample, you're randomly drawing 2 numbers; in
 total, we're pulling 1000 samples.

hist(xbar.for.n2,breaks=10,xlim=c(-2.5,2.5)) #The "breaks" argument states that we
 want a histogram with approximately 10 bars. The "xlim" argument specifies the scale
 of the x-axis; we'll specify this scale for all our histograms so that they can be
 easily compared.

#Now let's draw ten random numbers from a standard normal distribution and simulate
 this 1000 times:
sim2<-matrix(rnorm(10000),nrow=10)
```

```r
sim2[,1:4] #Inspect first four columns

xbar.for.n10<-apply(sim2,2,mean)
length(xbar.for.n10)
hist(xbar.for.n10,breaks=10,xlim=c(-2.5,2.5))

#Now let's draw 50 random numbers from a standard normal distribution and do this
 1000 times:
sim3<-matrix(rnorm(50000),nrow=50)
sim3[,1:4] #Ispect the first four columns
xbar.for.n50<-apply(sim3,2,mean)
hist(xbar.for.n50,breaks=10,xlim=c(-2.5,2.5))

#Finally, let's draw 100 random numbers from a standard normal distribution and do
 this 1000 times:
sim4<-matrix(rnorm(100000),nrow=100)
xbar.for.n100<-apply(sim4,2,mean)
hist(xbar.for.n100,breaks=10,xlim=c(-2.5,2.5))

#As you can see, as n increases, the variance of x-bar decreases -- the data becomes
 more closely grouped around the mean.

#Remember, what we varied in these simulations was the sample size--the number of
 X's that we used to create our x-bars. We did NOT vary the number of samples/
 simulations. In each case we did 1,000 simulations of a random sample (n=2,10,50 or
 100), so in each case we're distributing the same number of x-bars.

length(xbar.for.n2)
length(xbar.for.n10)
length(xbar.for.n50)
length(xbar.for.n100)


# -------------------------------------
# CENTRAL LIMIT THEOREM: DEMONSTRATIONS
# -------------------------------------

#In the previous example, we sampled normally-distributed random variables (using
 rnorm), so we should not be surprised that their sum or average is also normal. But
 the central limit theorem tells us that the sum and/or the mean of a large number of
 independent random variables ("large" means approximately  n>30) will be
 approximately normally distributed regardless of their individual distributions. To
 wit:

#Let's create 9 different vectors, named u1...u9, each with 1000 uniformly
 distributed random variables:

u1<-runif(1000)
u2<-runif(1000)
```

```
u3<-runif(1000)
u4<-runif(1000)
u5<-runif(1000)
u6<-runif(1000)
u7<-runif(1000)
u8<-runif(1000)
u9<-runif(1000)

#Now let's create 9 new variables, each of which is the sum of the first n variables
 we created above:

sum1<-u1
mean(sum1)
sum2<-u1+u2
mean(sum2)
sum3<-u1+u2+u3
sum4<-u1+u2+u3+u4
sum5<-u1+u2+u3+u4+u5
sum6<-u1+u2+u3+u4+u5+u6
sum7<-u1+u2+u3+u4+u5+u6+u7
sum8<-u1+u2+u3+u4+u5+u6+u7+u8
sum9<-u1+u2+u3+u4+u5+u6+u7+u8+u9

#Since we're going to draw a total of 9 histograms, let's set up the graphics window
 for 3 rows, 3 columns, filling by row:

par(mfrow=c(3,3))

#Let's look at the histograms of sum1, sum3, sum7, and sum9. We will also draw the
 corresponding normal PDF on each graph, using the command curve():

hist(sum1,breaks=20,freq=F) #The "freq" argument states that we want a histogram
 with probability density on the Y axis, not the frequency of each Xi. Compare:

hist(sum1,breaks=20)


#Now we want to plot the normal probability distribution curve:
curve(dnorm(x,mean(sum1),sd(sum1)),add=T,col='red') #Specify the probability
 distribution (we're plotting the pdf, so specify dnorm) and its parameters within
 the "curve" command. The "add" argument specifies that we want to add this curve to
 an existing plot, and the "col" argument specifies the color of the curve.

hist(sum3,breaks=20,freq=F)
curve(dnorm(x,mean(sum3),sd(sum3)),add=T,col='red')

hist(sum7,breaks=20,freq=F)
curve(dnorm(x,mean(sum7),sd(sum7)),add=T,col='red')
```

```
hist(sum9,breaks=20,freq=F)
curve(dnorm(x,mean(sum9),sd(sum9)),add=T,col='red')

#As you can see, the central limit theorem kicks in pretty quickly.


#The central limit theorem says that the sum and/or the mean of a large number of
 independent random variables will be approximately normally distributed regardless
 of their individual distributions. Let's demonstrate that the means of a non-normal
 distribution are approximately normally distributed as n gets larger. We'll use the
 Poisson distribution for this example, and will use the apply() function to
 demonstrate the CLT more quickly and efficiently.

pois<-sapply(1:10, function(x) rpois(1000,7.2)) #Create a vector of 1000 randomly-
 selected numbers from the Poisson distribution where lambda=7.2, and do this ten
 times. The sapply() command will create a matrix for you.
dim(pois) #Each column represents one experiment (of 1000 random draws)
head(pois)

pois1<-apply(pois, 2, mean) #Create a vector of the means of all 10 experiments
 (since each column represents one experiment, we specify margin=2)
pois1

#Let's plot the distribution and impose a normal PDF curve:
par(mfrow=c(2,2))
hist(pois1,breaks=10,freq=F)
curve(dnorm(x,mean(pois1),sd(pois1)),add=T,col='red')

#Now lets' try the same experiment, but do it 100 times instead of just 10 times:
pois2<-sapply(1:100, function(x) rpois(1000,7.2))
dim(pois2)
pois2<-apply(pois2, 2, mean)
pois2
hist(pois2,breaks=10,freq=F)
curve(dnorm(x,mean(pois2),sd(pois2)),add=T,col='red')

#And 1000 times:
pois3<-sapply(1:1000, function(x) rpois(1000,7.2))
dim(pois3)
pois3<-apply(pois3, 2, mean)
hist(pois3,breaks=10,freq=F)
curve(dnorm(x,mean(pois3),sd(pois3)),add=T,col='red')

#And 10,000 times:
pois4<-sapply(1:10000, function(x) rpois(1000,7.2))
dim(pois4)
pois4<-apply(pois4, 2, mean)
hist(pois4,breaks=10,freq=F)
curve(dnorm(x,mean(pois4),sd(pois4)),add=T,col='red')
```

```
#As you can see, as n increases (and remember, n here is the number of MEANS of the
 1000 random draws, not the random draws themselves), the distribution of the means
 approaches normal.

#We can also use the "for" loop to demonstrate the same thing:

#Let's draw 1000 random numbers from the Poisson distribution ten times:
pois1<-c()
for(i in 1:10) {pois<-rpois(1000,7.2)
    pois1[i]<-mean(pois)}
pois1

#And 100 times:
pois2<-c()
for(i in 1:100) {pois<-rpois(1000,7.2)
    pois2[i]<-mean(pois)}
pois2

#And 1,000 times:
pois3<-c()
for(i in 1:1000) {pois<-rpois(1000,7.2)
    pois3[i]<-mean(pois)}
pois3

#And 10,000 times:
pois4<-c()
for(i in 1:10000) {pois<-rpois(1000,7.2)
    pois4[i]<-mean(pois)}
pois4

#Now let's compare the histograms:
par(mfrow=c(2,2))

hist(pois1,breaks=10,freq=F)
curve(dnorm(x,mean(pois1),sd(pois1)),add=T,col='red')

hist(pois2,breaks=10,freq=F)
curve(dnorm(x,mean(pois2),sd(pois2)),add=T,col='red')

hist(pois3,breaks=10,freq=F)
curve(dnorm(x,mean(pois3),sd(pois3)),add=T,col='red')

hist(pois4,breaks=10,freq=F)
curve(dnorm(x,mean(pois4),sd(pois4)),add=T,col='red')
```