Bryan Oliande

Project 2

PROBLEM 1:

Population Average: Mu = E(x) = 1(0.05) + 2(0.05) + 3(0.05) + 4(0.1) + 5(0.15) + 6(0.15) + 7(0.4) + 8(0.05) = **5.55**

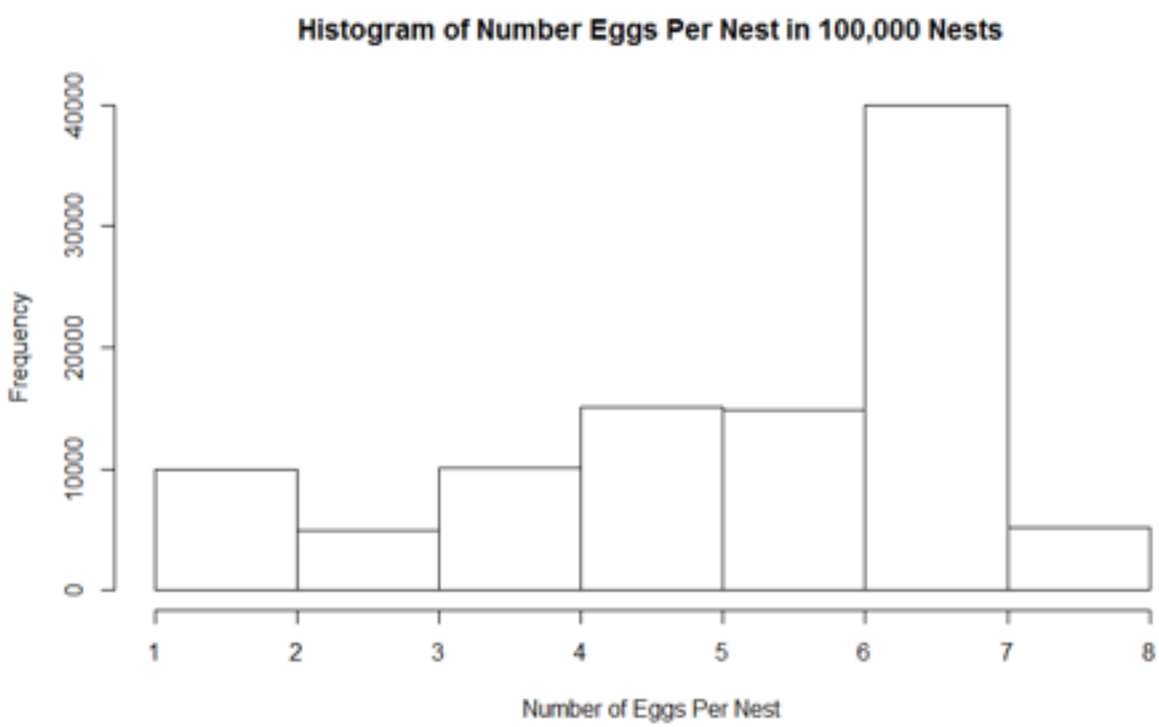
E(x)^2 = 30.8025

E(x^2) = 1(0.05) + 4(0.05) + 9(0.05) + 16(0.1) +25(0.15) +36(0.15) + 49(0.40) + 64(0.05) = 34.25

Var(x) = E(x^2) – E(x)^2 = 34.25 – 30.8025 = 3.448

Population Standard Deviation: Sigma = sqrt(Var(x)) = sqrt(3.448) = **1.857**

PROBLEM 2:

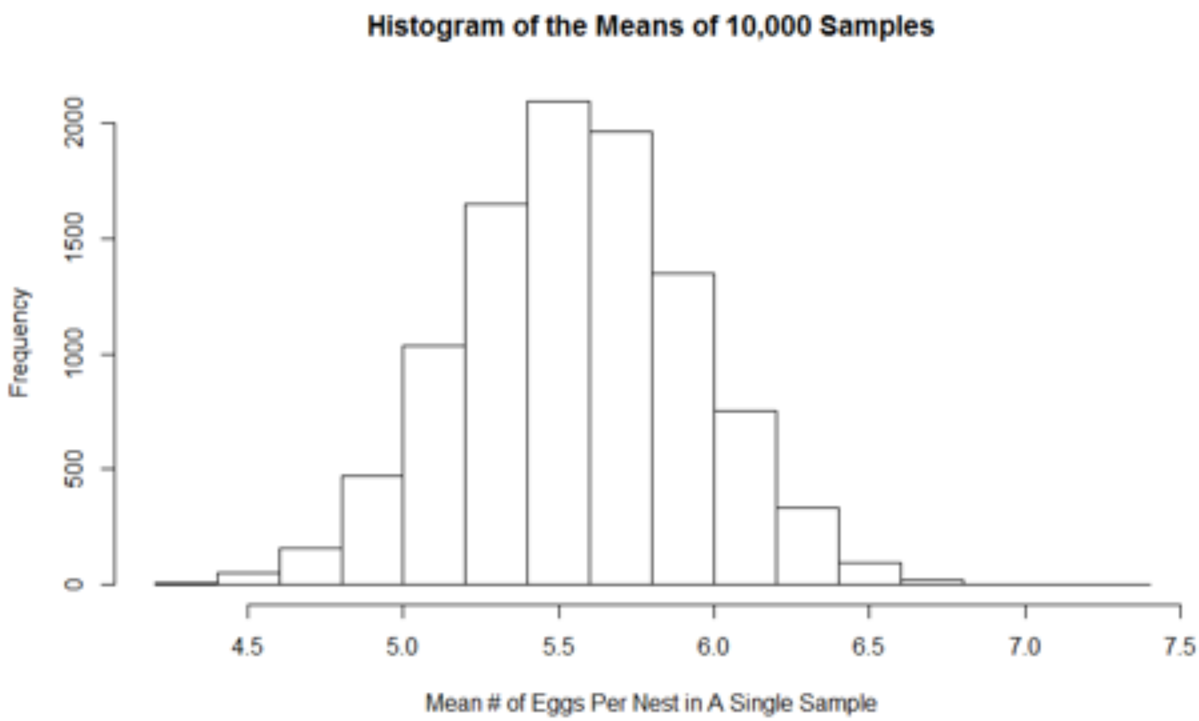


PROBLEM 3:

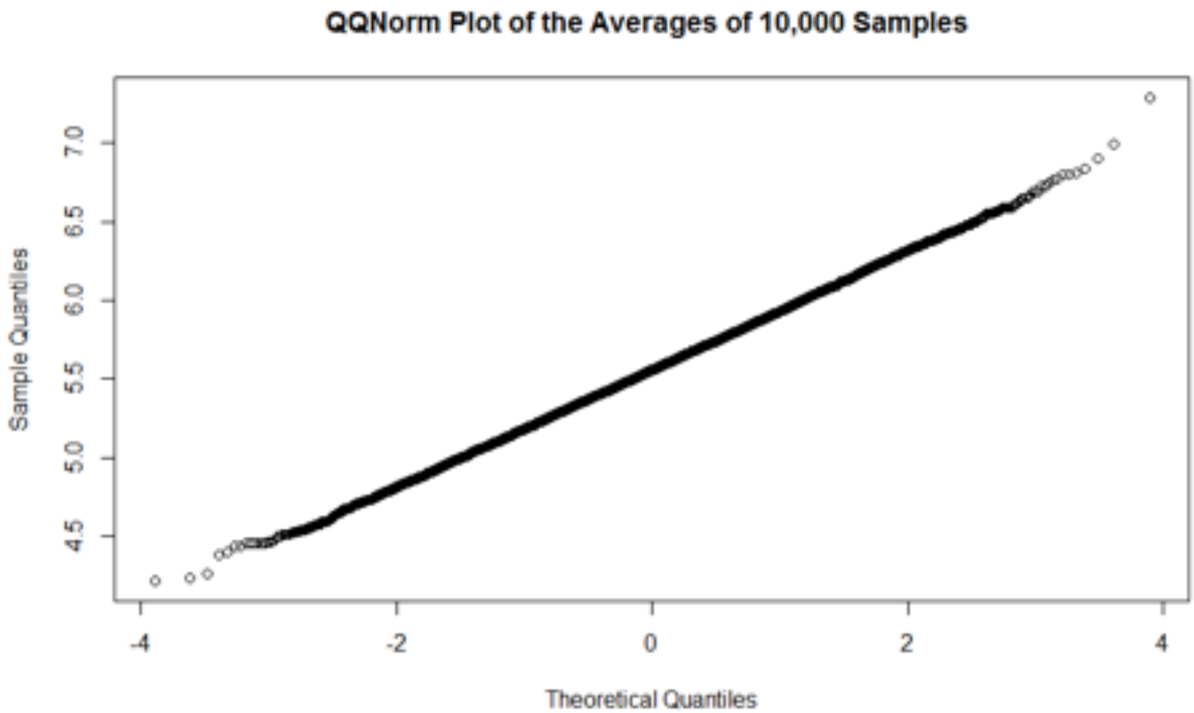
A: In order to use a T-test we must know that the population is distributed approximately normal with no outliers and no prominent skew. The distribution does not completely satisfy these conditions as the distribution has a skew to the left. Therefore, we cannot use a T-test.

B: We cannot use a Z-test because the sample size is 5 nests. With n=5, we cannot apply the Central Limit Theorem to this distribution and we are unable to use a Z-test.

C:



D:



Based on the previous 2 charts, we can say that the distribution looks approximately normal but light tailed.

E:

The distribution of xbar is distributed approximately normally with mean 5.55 (the population mean) and standard deviation 1.857/sqrt(5) = 0.8305. 7.2 eggs is almost (right on the border of) 2 standard deviations away from 5.55, so it would be unusual. In fact, when testing (see appendix for R code) the 10,000 samples of 5 nests each, only 1 of them had an average of 7.2 eggs or greater.

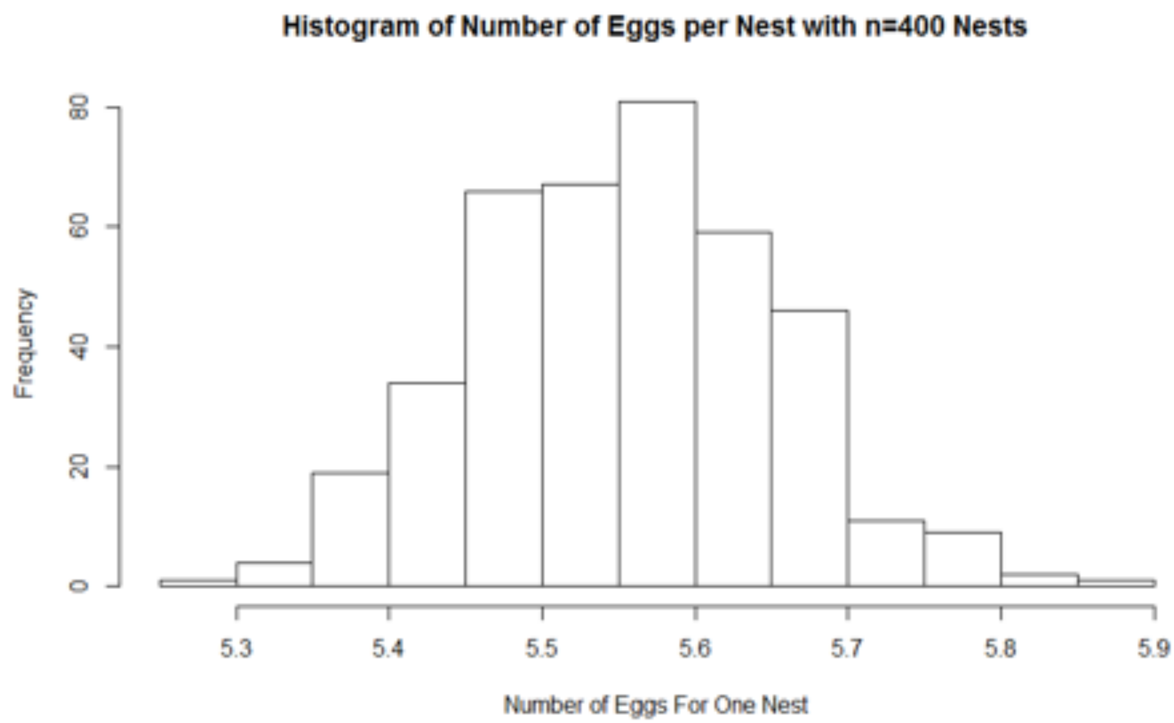
F:

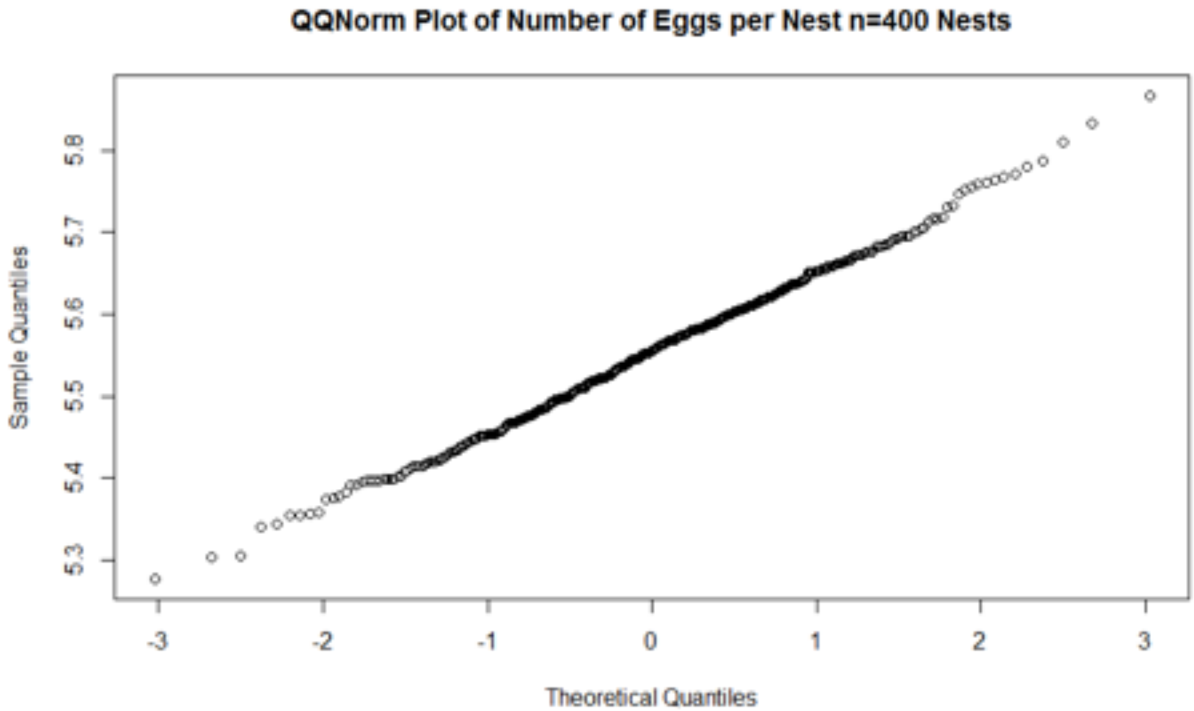
Only 8,751 of the 10,000 confidence intervals produced (see Appendix) contained the true mean. Theoretically, 9,500 of them should have. Because we used n = 5 < 30, our distribution isn't normal enough and we cannot apply the Central Limit Theorem. The ecologists' observations are highly unusual. She should have sampled more nests.

PROBLEM 4:

Number of Nests: n = 400

Although my chosen number of nests is a bit high, I was not satisfied with the "normality" of my graphs when testing out n's of a lower magnitude. I still am not completely satisfied with their "normality" but when doing real statistics it would be very difficult to obtain a sample of 400 nests, so I am not going to increase the n anymore.





A: Because my n is large enough and the population standard deviation is known, I am choosing to use the Z-distribution.

B: **(i)** Out of 10,000 95% Confidence Intervals, 9,500 should contain the true mean.

(ii) When I made 10,000 Confidence Intervals in R (see Appendix), 9479 contained the true mean 5.55. This is MUCH better than the number of confidence intervals obtained in Problem 3 because we used an n greater than 30. So we can apply the Central Limit Theorem.

C:

- **(i)** I expect to reject the null hypothesis (alpha)10000 = (0.05)10000 = 250 times.
- **(ii)** I rejected the null hypothesis 283 times. (See Appendix)
- **(iii)** A Type 1 Error would be rejecting the null hypothesis (mu = 5.55) when it is actually true. We know that mu = 5.55, all the times we reject the Ho are Type 1 Errors. In this simulation 283 Type 1 Errors occurred.
- **(iv)** A Type 2 Error would be failing to reject Ho (mu = 5.55) when Ha (mu != 5.55) is actually true. We now that mu= 5.55, so a Type 2 Error is not possible.

D: Central Limit Theorem. For problem 3, n = 5 produced a very low number of confidence intervals that contained the actual mean. For problem 4, n = 400 produced almost exactly the theoretical amount of confidence intervals that should contain the true mean for a 95% confidence interval.

APPENDIX

```
set.seed(124)
library(ggplot2)

#PROBLEM2
#-----------------------------------------------
problem2 <- sample(8, 100000, prob = c(0.05, 0.05, 0.05, 0.1, 0.15, 0.15, 0.40, 0.05),
replace = T)
hist(problem2, main = "Histogram of Number Eggs Per Nest in 100,000 Nests", xlab =
"Number of Eggs Per Nest", breaks = 6)
#-------------------------------------

#PROBLEM3
#------------------------------------------------

#(C)
means <- rep(NA, 10000)
sds <- rep(NA, 10000)
nests <- rep(NA, 5)

for(i in 1:10000) #each outer loop is 1 sample of size n=5 nests
{
  nests <- rnorm(n=5, mean = 5.55, sd = 1.857/sqrt(5))
  means[i] <- mean(nests)
  sds[i] <- sd(nests)
}

hist(means, main = "Histogram of the Means of 10,000 Samples", xlab = "Mean # of Eggs
Per Nest in A Single Sample")
```

```
#(D) They look approximately normally distributed with a few outliers.
qqnorm(means, main = "QQNorm Plot of the Averages of 10,000 Samples ")

count3e <- 0
#(E)
for(i in 1:10000)
{
  if(means[i] >= 7.2)
  {
    count3e <- (count3e + 1)
  }
}
#(F):
trueCount <- 0
CIlowerbound <- rep(NA, 10000)
CIupperbound <- rep(NA, 10000)
for(i in 1:10000)
{
  CIlowerbound[i] <- means[i] - qnorm(0.975)*sds[i]/sqrt(5)
  CIupperbound[i] <- means[i] + qnorm(0.975)*sds[i]/sqrt(5)


  if( (CIlowerbound[i] <= 5.55) & (CIupperbound[i] >= 5.55) )
  {
    trueCount <- (trueCount + 1) #8751
  }
}

#-------------------------------------------------------
```

```
#PROBLEM 4
#-----------------------------------------------

#a
n <- 400
sigma <- 1.857 #population SD
mu0 <- 5.55 #population mean

prob4 <- rnorm(n, mean = 5.55, sd = 1.857/sqrt(400))
hist(prob4, main = "Histogram of Number of Eggs per Nest with n=400 Nests", xlab =
"Number of Eggs For One Nest")
qqnorm(prob4, main = "QQNorm Plot of Number of Eggs per Nest n=400 Nests")


#b


means4 <- rep(NA, 10000)
sds4 <- rep(NA, 10000)


for(i in 1:10000)
{
  sample4 <- rnorm(n, mean = 5.55, sd = 1.857/sqrt(400))
  means4[i] <- mean(sample4)
  sds4[i] <- sd(sample4)
}

trueCount4 <- 0
CIlowerbound <- rep(NA, 10000)
CIupperbound <- rep(NA, 10000)
```

```
for(i in 1:10000)
{
  CIlowerbound[i] <- means4[i] - qnorm(0.975)*sds4[i]/sqrt(n)
  CIupperbound[i] <- means4[i] + qnorm(0.975)*sds4[i]/sqrt(n)

  if( (CIlowerbound[i] <= 5.55) & (CIupperbound[i] >= 5.55) )
  {
    trueCount4 <- (trueCount4 + 1) #9479
  }
}



#c

sigma <- 1.857 #population SD
mu0 <- 5.55 #population mean
rejectCount <- 0



for(i in 1:10000)
{
  sample4 <- rnorm(n, mean = 5.55, sd = 1.857)
  means[i] <- mean(sample4)
  z <- ( (means[i] - mu0) / (sigma/sqrt(n)) )
  pval <- 2 *pnorm(z)

  if (pval < 0.05)
  {
    rejectCount<- (rejectCount + 1)

  }
```

```
}
```