STAT 426 Project Writeup

Question 1

Cursory Experiments

Right when I opened the app, I simulated many samples of size 100 with a population proportion of .7 using 95% confidence intervals. Over many simulations, 95% of my confidence intervals contain the true population proportion when using the Wald interval, just as I would hope. This is not surprising though, because .7 is not very close to 1 and 100 is a fairly large population.

Using a 90% confidence interval with the same settings as in part a, surprisingly, the proportion of confidence intervals containing the true population proportion is below the 90% that I would expect using Wald, with an observed proportion of confidence intervals containing the true parameter at about 87.2. This problem goes away when I switch to using the score which converges to 90% coverage. The adjusted Wald converges to around 89.7% which is below the theoretical 90% coverage.

Now, what would happen if I lowered the sample size? Would this make the coverages vary farther from their theoretical levels. Using 95% confidence intervals, a population proportion of .7 and a sample size of 10, I notice that the Wald confidence interval does not meet its theoretical guarantees, and contains the population proportion only 84% of the time. Here, the score converges to around 92.4%. The adjusted Wald converges to around 95.3% which is better than the other methods in this particular case.

A natural question to now ask is what happens when we use a very large sample size, but a population proportion near 1. It is unclear if the large sample size will result in $\hat{p}$ having a normal distribution or if p being near 1 will cause it to not have a normal distribution. Using a 95% confidence intervals, a sample size of 1000, and a population proportion of .99, the Wald interval does not fare very well in that it converges to around 92.6%. The score converges to around 96.3% while the adjusted Wald converges to around 96.4%.

All procedures do well we creating a 99% confidence interval, with a sample size of 100, and a population proportion of .7. The Wald interval and the score interval converge to 98.8%. The adjusted Wald converges to 98.9%.

From these cursory experiments, I have found that when the situation is less than ideal, the Wald interval tends to perform worse than both the score and the adjusted

Wald.

Systematic Investigation

I will now simulate 500,000 confidence intervals with varying parameters and compare the results across the different interval generating techniques.

Figure 1

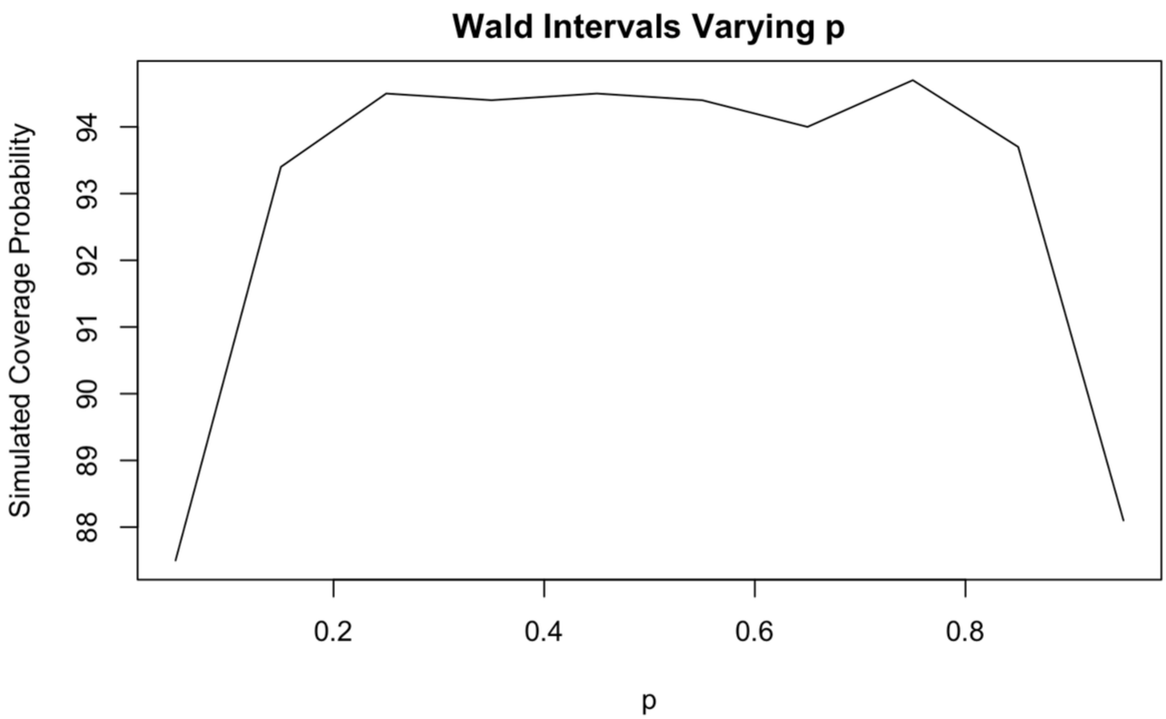
Note: cl = confidence level

Parameters	Wald	Score	Adjusted Wald
p=.5, n=100, cl=95%	94.3%	94.3%	94.3%
p=.99, n=100, cl=95%	63.2%	92.1%	98.2%
p=.5, n=10, cl=95%	89.1%	97.8%	97.9%
p=.5, n=500, cl=95%	94.6%	94.6%	94.6%
p=.5, n=100, cl=99%	98.8%	98.8%	98.8%
p=.5, n=100, cl=90%	91.1%	91.1%	91.1%
p=.01, n=10, cl=95%	9.5%	90.4%	99.6%

As was hypothesized in the cursory experiments, I found that for relatively ideal (p far enough from 0 or 1, and n sufficiently large) conditions, all three methods perform roughly the same as is evidenced on rows 1, 4, 5 and 6 of Figure 1. Under less than ideal conditions, I found large differences between estimated coverage probabilities and the theoretical levels. Most notably evidenced by the final row of the Figure 1 where the Wald interval only achieves 9.5% coverage.

The following graph illustrates the coverage probabilities as a function of the unknown parameter p using Walds method with n = 100 and cl = 95%. For each observed data value, I perform 10,000 simulations.

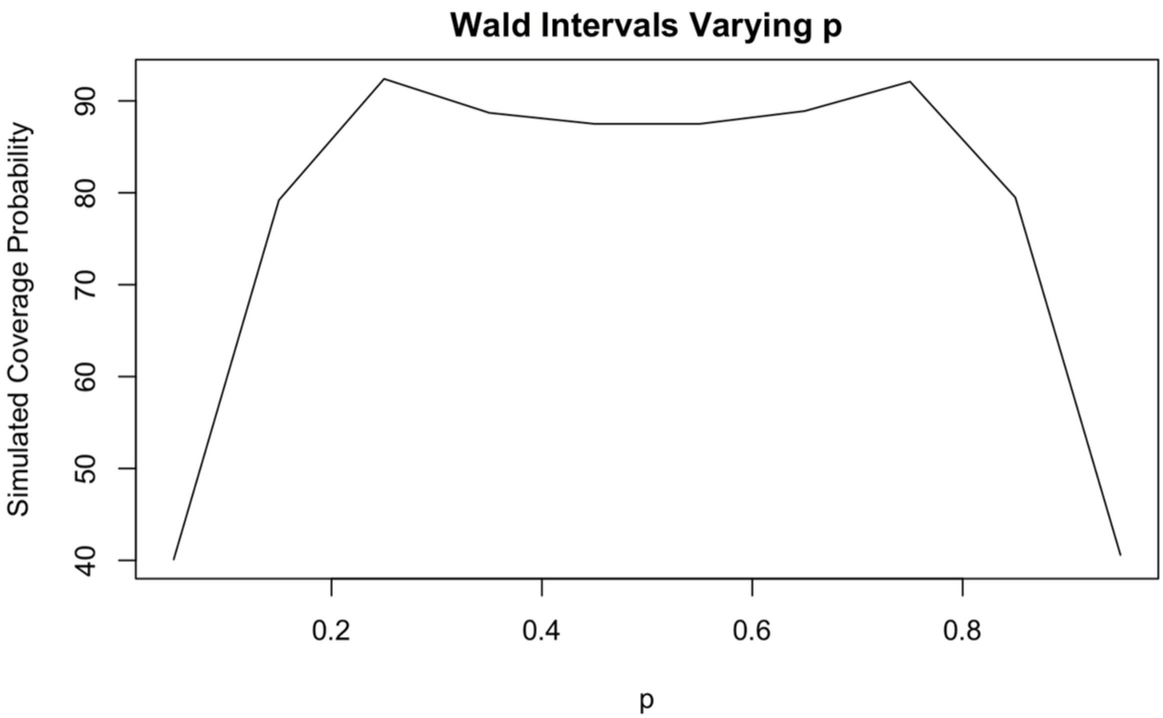
Figure 2



As expected, the Wald intervals tend to do the best when p is between .25 and .75. Other than some simulation margin of error it seems to be fairly constant over this range. The interval performs slightly worse at p = .15 and p = .85 and significantly worse at p = .05 and p = .95.

Now, I will run this same experiment, however I will make the conditions much more difficult for the Wald interval by decreasing the sample size to 10.
Now n = 10 and cl = 95%. I will up the number of intervals to 30,000 as the sample size is now smaller and so it is more computationally feabible.

Figure 3



This result is unexpected and very interesting. I was assuming that the highest simulated coverage probabilities would occur near p = .5 as they do in Figure 2, barring some simulation margin of error. However there appears to be a valley around p =.5. The highest values occurred at p=.25 and p =.75 which I would not have guessed. I am glad that I ran this experiment and found this, however it is unclear whether this was caused merely by chance given that I only simulated 30,000 intervals for each data point.

To summarize, it seems that when p is moderate (roughly between .25 and .75), when the sample size is large enough (large enough depends on the other parameters), then regardless of the confidence level, all three methods: Wald, Score, and Adjusted Wald perform well and similar to each other. I hypothesize that if all data for population proportion estimation problems were this well behaved, there would be no need for three different methods, and the most simple and intuitive, Wald, would be sufficient.

Now consider an unideal situation in which p=.01, and n=10 with cl=95%. The Wald interval depends on the central limit theorem and the normality of $\hat{p}$ (note the z multiple in the formula). We have seen in class that when p is close to 0 or 1, the sample size required for a Binomial sample proportion to converge to normal is higher than when p is closer to .5. In this case, n=10, is much too small for our sample proportion to be roughly normal. So it makes sense that the Wald interval would perform poorly here. More specifically, under these circumstances, Wald has an abysmal 9.5% coverage probability. This is roughly 85% worse than its theoretical value. In this situation other methods such as score and adjusted Wald prove their worth. In this situation, score has a simulated coverage probability of 90.4% and adjusted Wald has a coverage probability of 99.6%. Both do much better than Wald, and impressively the adjusted Wald achieves almost 5% more coverage than its theoretical value. While I do not know the derivations of the score and adjusted Wald intervals, these results make me think that they were developed to handle situations when the Wald interval performs poorly.

To close out this section, I will give some recommendations based on a few hours of experimenting with these intervals. If the sample data is relatively well behaved, meaning that p is not too close to 0 or 1 and the sample size is large enough (possibly 30 is large enough but larger sample sizes are needed for p closer to 0 or 1), then I would recommend the Wald interval. The reason being that it is the simplest to compute, has the simplest theory for deriving and justifying it, and it gives roughly the exact same performance as the other two methods in these situations.

On the other hand, if you find yourself in a less than ideal situation with a p close to 0 or 1 and/or a small sample size (again small is relative to what p is) then I would not recommend the Wald interval as it may give coverage probabilities far below its theoretical guarantees. Where I would advise you now depends on how willing you are for your confidence interval to not contain p. If you accept the fact that this is a difficult situation and still want fairly tight confidence intervals and coverage probabilities close to your specified confidence level, I would go with score intervals, but be warned the coverage probabilities are likely going to fall short of your specified confidence level. If you are unwilling to have coverage probabilities below your specified threshold I would recommend an adjusted Wald interval, but be warned that you will likely get wide confidence intervals and as a result will likely have coverage probabilities above your specified threshold.

Question 2

For this question, I decided to choose the Exponential distribution problem. The problem is as follows: estimating λ based on $X_1, \ldots, X_n$ i.i.d. Exponential random variables.

For this problem, there are two different confidence interval generating procedures that I will explore. First, I will take advantage of the Exponential distributions simplicity and will create the exact procedure that we derived in Exercise 27.3. Next, I will use the fact that $\frac{1}{\bar{X}}$ is relatively normal when n is large to create an approximate confidence interval based on the $\pm z * SE$ approach where z is an appropriate multiple of a Normal(0,1) distribution.

Below, Figure 5 contains the bulk of my code to run these simulations. I hope the function names are self-explanatory. At the very bottom, I include one line of example code simulating coverage probabilities.

Figure 4

```
simulate_data <- function(n, lambda){
  val <- rexp(n, lambda)
  return (val)
}

get_q1_q2_exact <- function(alpha, n){
  q1 <- qgamma(alpha/2, n, 1)
  q2 <- qgamma(1 - alpha/2, n, 1)
  return (c(q1, q2))
}

get_confidence_interval_exact <- function(alpha, n, lambda){
  data <- simulate_data(n, lambda)
  q1_q2 <- get_q1_q2_exact(alpha, n)
```

```
  q1 <- q1_q2[1]
  q2 <- q1_q2[2]
  X_bar <- mean(data)
  lower_bound <- (q1/n)*(1/X_bar)
  upper_bound <- (q2/n)*(1/X_bar)
  return (c(lower_bound, upper_bound))
}

get_z_score <- function(alpha){
  val <- qnorm(1 - alpha/2)
  return (val)
}

get_margin_of_error <- function(alpha, data, n){
  z_score <- get_z_score(alpha)
  X_bar <- mean(data)
  margin_of_error <- z_score * (1 / (X_bar * sqrt(n)))
  return (margin_of_error)
}



get_confidence_interval_approx <- function(alpha, n, lambda){
  data <- simulate_data(n, lambda)
  margin_of_error <- get_margin_of_error(alpha, data, n)
  X_bar <- mean(data)
  point_estimate <- 1 / X_bar
  lower_bound <- point_estimate - margin_of_error
  upper_bound <- point_estimate + margin_of_error
  return (c(lower_bound, upper_bound))
```

```
}


contains_lambda <- function(alpha, n, lambda, exact){
  if (exact){
    interval <- get_confidence_interval_exact(alpha, n, lambd
a)
  } else if (!exact){
    interval <- get_confidence_interval_approx(alpha, n, lambd
a)
  }
  if (between(lambda, interval[1], interval[2])){
    return (1)
  }
  else {
    return (0)
  }
}


find_coverage_probability <- function(alpha, n, lambda, reps,
exact){
  total_covered <- 0
  for (i in 1:reps){
    total_covered = total_covered + contains_lambda(alpha, n,
lambda, exact)
  }
  return (total_covered / reps)
}
# this will return the simulated coverage probability, with th
e following parameters
# cl=95%, n=3, lambda=10000, number of intervals=10000, approx
imate (normal)
```

```
# confidence interval
find_coverage_probability(.05, 3, 10000, 100000, F)
```

Exploration

I believe that putting the results into a table is likely more readable than when it is just reported in paragraph form, so I will begin with a table. For each of these simulations, I performed 100,000 iterations. I tried to experiment with a variety of different parameter combinations, including ones that I thought the procedure would easily handle and ones I thought would be more difficult for it.

Figure 5

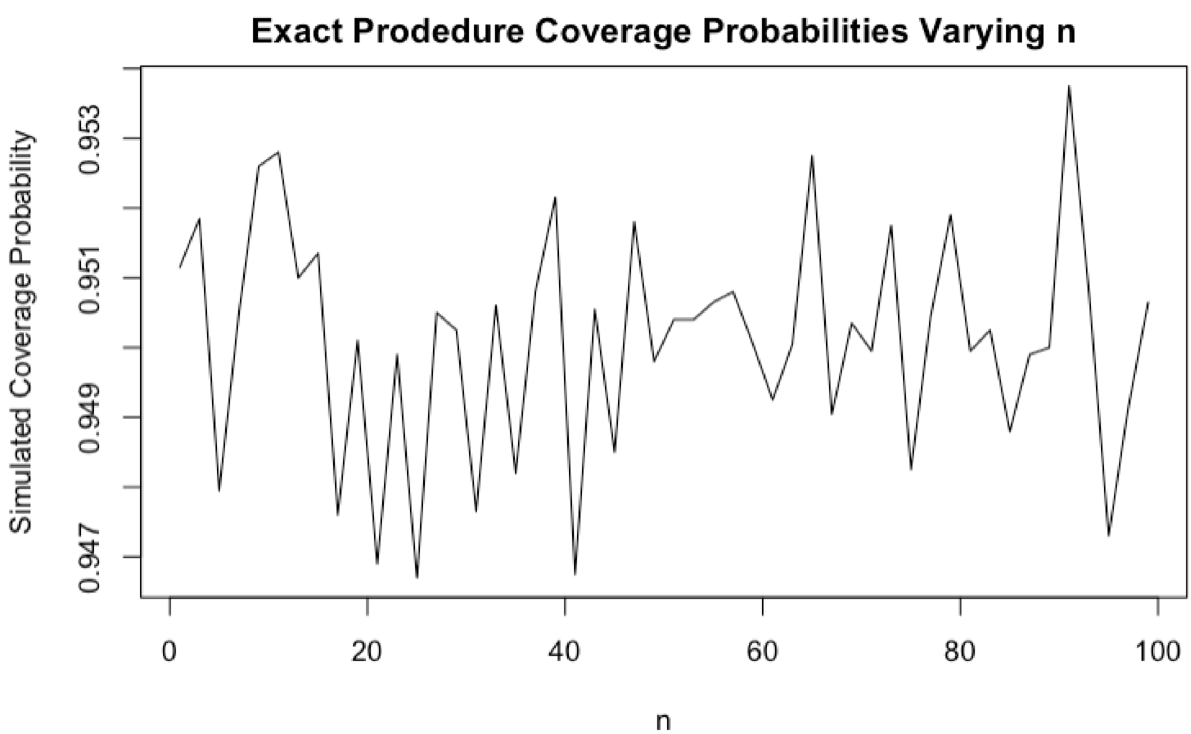
Parameters	Exact	Approximate
$\lambda = 1$, n=100, cl=95%	94.9%	95.1%
$\lambda = 1$, n=10, cl=95%	95.0%	95.5%
$\lambda = 1$, n=3, cl=95%	95.1%	95.3%
$\lambda = 10$, n=100, cl=95%	95.0%	95.0%
$\lambda = .01$, n=100, cl=95%	95.0%	95.0%
$\lambda = .01$, n=3, cl=95%	95.0%	95.3%
$\lambda = .0001$, n=3, cl=95%	95.0%	95.3%
$\lambda = .0001$, n=3, cl=99%	99.0%	98.0%
$\lambda = .0001$, n=3, cl=90%	89.9%	93.1%
$\lambda = 10000$, n=3, cl=95%	94.9%	95.3%
$\lambda = 10000$, n=3, cl=99%	99.0%	97.9%

Looking at Figure 5, it appears that the confidence intervals for Exponential λ are more stable and resistant to difficult parameters than the Binomial confidence intervals were. Even with less than idea parameters, namely $\lambda = .01$, n=3, cl=95%, both methods were within .5% of the theoretical coverage probability which was surprising. Each time I tried parameters that I thought would make the normal approximation less justified and thus lessen the simulated coverage, I was only able to get the simulated coverage probabilities to diverge slightly from the theoretical probabilities.

Now I will create some plots that illustrate how the coverage probabilities vary when I change n, the sample size. These plots will give further evidence to the points I made in the previous paragraph, namely the fact that these intervals are very resistant to difficult parameters.

For these experiments, I used $\lambda = .0001$, cl=95% with 20,000 reps per value of n.

Figure 6



In this figure, we see no meaningful trend in the data. It appears that there is a lot of simulation margin of error, however if you look at the y-axis the scale only varies by .006 which is not much at all. As a result, I can conclude that with the fixed parameters I am using here, the efficacy of the exact Exponential λ confidence interval is relatively independent of the sample size.

Figure 7

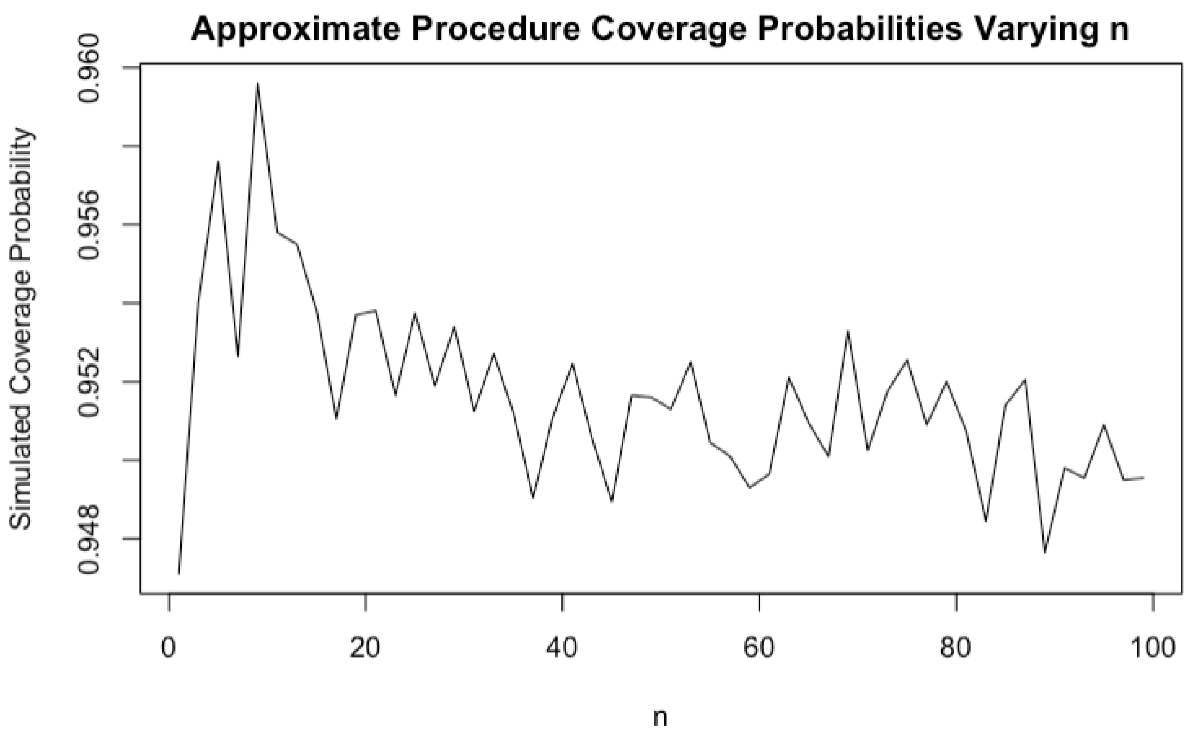


Figure 7 provides somewhat of a contrast to Figure 6. In Figure 7, when n is close to 0, I get below 95% coverage, which is somewhat expected. However, when n increases, the coverage quickly shoots up to above 95% with the peak coverage probability coming somewhere around n=12. Once n gets to around 20 the coverage probability is just above 95% and seems to slowly trend down to 95% as n gets closer to 100. I did not expect to see the highest coverage probabilities of the graph occur when n is between 0 and 20, but the larger margin of error for these small n values is likely contributing to this. Also, the y-axis only spans about 1.2% coverage, so even while we see some trend in the data, in the scheme of things, this is a fairly negligible trend.

Summary

Overall, I was impressed by how much better these two estimation procedures perform compared to the Wald interval when it comes to stability in cases where the parameter is an extreme value. If you compare Figure 5 to Figure 1 you can see how much closer the coverage probabilities are to their theoretical values under less than ideal circumstances in this Exponential estimation situation as compared to the Binomial situation.

One possible explanation for this is the fact that Binomial is a discrete distribution so it could potentially need larger sample sizes before it approaches a normal distribution compared to Exponential which is continuous. Also, I learned in office hours that the gamma distribution approaches normal even when the parameter n is fairly small. Since the sample mean of Exponential is related to a sum of Exponentials and a sum of i.i.d. Exponential distributions follows a gamma distribution, this could be contributing to the good performance of these Exponential confidence intervals.

There was a slight discrepancy of performance between the exact λ confidence interval and the approximate λ confidence interval, however this discrepancy was arguably negligible. While on first glance you might notice a significant difference when comparing the shapes of Figure 6 and Figure 7, upon looking at the scale on the y-axis you will realize that the differences are small. Looking at Figure 5, you can see that the exact procedure performs the same or better than the approximate procedure in all trials I ran, however I don’t believe this discrepancy was ever significant.

In conclusion, for this problem you should use the exact λ confidence interval if you care a lot about your theoretical coverage probabilities being close to your confidence level. If for some reason you want to use the approximate interval, you can rest assured that it will likely perform fairly well even in the most difficult estimation situations.

Question 3

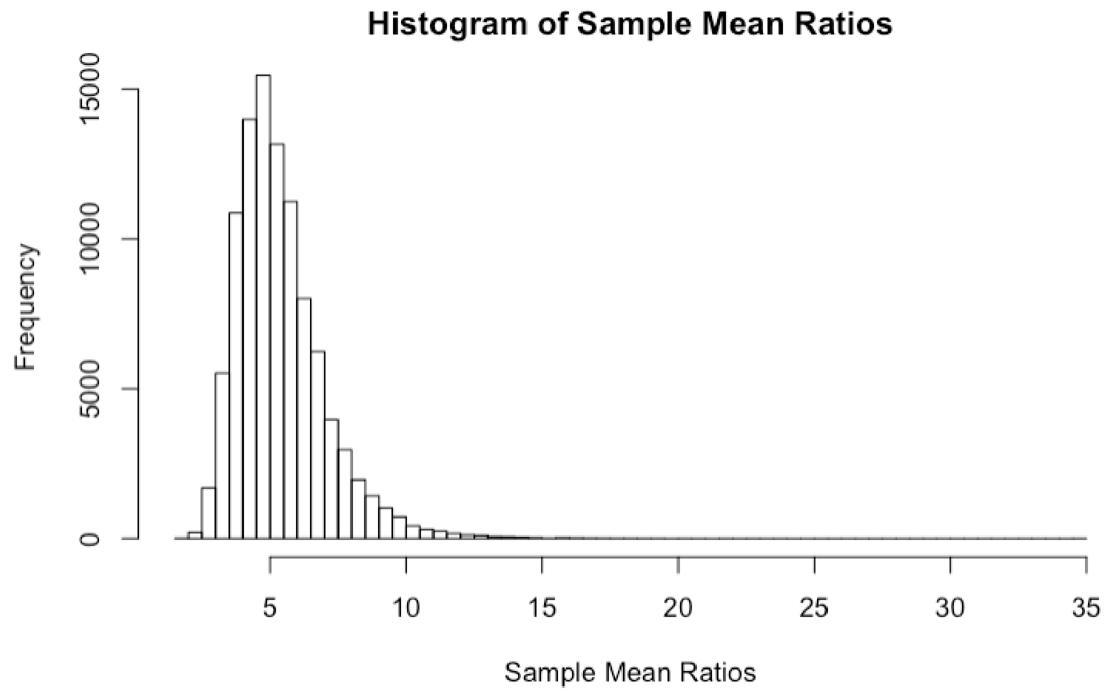
Motivation

For this question, I am looking at the problem of estimating a parameter $\theta = \frac{\mu_X}{\mu_Y}$ where μ_X and μ_Y are the population means of two Poisson distributions. I will be using data from two random samples to estimate this parameter. I will have $X_1, X_2, \ldots, X_n$ i.i.d. random variables from a $Poisson(\mu_X)$ distribution and $Y_1, Y_2, \ldots, Y_m$ i.i.d. random variables from a $Poisson(\mu_Y)$ distribution. The random samples from each respective Poisson distribution are independent of each other. Also, note that n does not have to equal m necessarily.

To estimate the parameter θ I will use two confidence interval procedures. First, I will use a bootstrap procedure. Then, I will utilize the approximate formulas from investigation 4 and use normal approximations to construct an approximate confidence interval. I will then compare the simulated coverage probabilities from each procedure and make some conclusions.

Bootstrap Interval

Figure 8



Before I begin discussing my results, I want to briefly mention Figure 8. As you can see in Figure 8, the histogram of sample mean ratios is very far from Normal with a long right tail. Thus, Normal intervals do not seem to be appropriate here. Now, I must consider whether 95% bootstrap intervals are appropriate or if there is enough bias such that a 95% bootstrap pivotal interval is more appropriate.

I also used bootstrapping to estimate the bias of $\hat{\theta} = \frac{\bar{X}}{\bar{Y}}$ in estimating $\theta = \frac{\mu_X}{\mu_Y}$ using the procedure discussed in Handout 31. In order to estimate this bias, defined as $E[\hat{\theta}] - \theta$, I had to use a concrete population. I chose $n_X = 30, \mu_X = 3, n_Y = 30, \mu_Y = 1$ where n_X, n_Y are the number of observations in the sample from the X population and the Y population respectively and μ_X, μ_Y are the population means of X and Y respectively. I ran a simulation with a confidence level of 95% and 10,000 approximations of bias where each approximation of bias consisted of 10,000 bootstrap sample mean ratios. I found that the average bias was 0.1188 while the average standard error was 0.7406. This means that the bias was approximately 16% of the standard error, which I believe is a high enough percentage to justify using the bootstrap pivotal confidence interval. However, note that I made this decision based on a two arbitrarily chosen populations.

The next figure contains my code for the bootstrap intervals.

Figure 9

```
# note that WBI is: Want Bias Information. Set WBI: True for a
lternative output useful # for determining bias


get_bootstrap_ci <- function(n_x, mu_x, n_y, mu_y, nreps, alph
a, WBI){
  sample_mean_ratios <- numeric(nreps)
  X <- rpois(n_x, mu_x)
  Y <- rpois(n_y, mu_y)
  fake_parameter <- mean(X) / mean(Y)
  real_parameter <- mu_x / mu_y
  for (i in 1:nreps){
    X_bootstrap <- sample(X, n_x, replace = TRUE)
    Y_bootstrap <- sample(Y, n_y, replace = TRUE)
    sample_mean_ratios[i] <- mean(X_bootstrap) / mean(Y_bootst
rap)


  }
  confidence_interval <- 2 * fake_parameter - quantile(sample_
mean_ratios, c(1 - alpha/2, alpha/2))
  bias <- mean(sample_mean_ratios) - fake_parameter
  if (WBI){
    return (list(bias, sd(sample_mean_ratios)))
  }
  return (list(confidence_interval, real_parameter))
}


contains_parameter <- function(n_x, mu_x, n_y, mu_y, nreps, al
pha, WBI){
```

```
  CI_and_parameter <- get_bootstrap_ci(n_x, mu_x, n_y, mu_y, n
reps, alpha, WBI)
  CI <- CI_and_parameter[[1]]
  real_parameter <- CI_and_parameter[[2]]
  if (between(real_parameter, CI[1], CI[2])){
    return (1)
  } else {
    return (0)
  }
}



estimate_coverage_probabilities <- function(n_x, mu_x, n_y, mu
_y, nreps_perCI, num_CIs, alpha, WBI){
  contains_parameter_counter <- 0
  for (i in 1:num_CIs){
    contains_parameter_counter = contains_parameter_counter +
contains_parameter(n_x, mu_x, n_y, mu_y, nreps_perCI, alpha, W
BI)
  }
  return (contains_parameter_counter / num_CIs)
}


approximate_bias <- function(n_x, mu_x, n_y, mu_y, nreps, alph
a, bias_reps){
  bias_values <- numeric(bias_reps)
  SEs <- numeric(bias_reps)
  for (i in 1:bias_reps){
      bias_info <- get_bootstrap_ci(n_x, mu_x, n_y, mu_y, nrep
s, alpha, T)
      bias <- bias_info[[1]]
```

```
      se <- bias_info[[2]]
      bias_values[i] <- bias
      SEs[i] <- se
  }
  return (list(mean(bias_values), mean(SEs)))
}



"this is an example of how to check coverage probabilities for my bootstrap 95% pivotal confidence intervall. n_x, n_y are the number of observations in the sample from their respective populations. mu_x, mu_y are the means of their respective populations. nreps_perCI specifies how many bootstrap samples to use to make one confidence interval and num_CIs is the number of intervals to create. alpha corresponds to the confidence level and WBI determines if we want information on the bias and standard error"
estimate_coverage_probabilities(n_x=30, mu_x=3, n_y=30, mu_y=3, nreps_perCI=1000, num_CIs=1000, alpha=.05, WBI=F)


"This is an example of how to call a function that checks for bias and SE. nreps is the total number of repititions and bias_reps is the number of bootstrap samples used to compute one value of the bias. The other parameters are the same"


approximate_bias(n_x=30, mu_x=3, n_y=30, mu_y=1, nreps=1000, alpha=.05, bias_reps=1000)
```

Normal Interval using Approximate Formulas

Dr. Ross told me in office hours that the MLE of $\theta = \frac{\mu_X}{\mu_Y}$ is $\hat{\theta} = \frac{\bar{X}}{\bar{Y}}$ which makes a lot of intuitive sense. I am assuming that $\hat{\theta}$ is Normal which seems like a reasonable assumption given $\bar{X}$ and $\bar{Y}$ will be relatively Normal for large sample sizes. My confidence interval will be of the form $\hat{\theta} \pm zSE(\hat{\theta})$ where z is an appropriate multiple from a Normal(0,1) distribution. Therefore, I just have to find $SE(\hat{\theta})$. From investigation four, and some logic that I did not Latex, I find:

$SE(\hat{\theta}) = SE(\frac{\bar{X}}{\bar{Y}}) = \sqrt{(\frac{\bar{X}}{\bar{Y}})^2((\frac{\frac{\sqrt{\bar{X}}}{\sqrt{n_x}}}{\bar{X}})^2 + (\frac{\frac{\sqrt{\bar{Y}}}{\sqrt{n_y}}}{\bar{Y}})^2)}$. (sorry that the formula is messy).

If $\hat{\theta}$ in fact has an approximate Normal distribution, I imagine that this confidence interval should work well.

Figure 10 contains the code implementing this.

Figure 10

```
 get_normal_confidence_interval <- function(n_x, mu_x, n_y, mu_y, alpha){
    X <- rpois(n_x, mu_x)
    Y <- rpois(n_y, mu_y)
    X_bar <- mean(X)
    Y_bar <- mean(Y)
    estimate <- X_bar / Y_bar
    SE <- sqrt((X_bar/Y_bar)^2 * (( (sqrt(X_bar)/sqrt(n_x))/X_bar)^2 + ((sqrt(Y_bar)/sqrt(n_y))/Y_bar)^2))
    z_score <- qnorm(1 - alpha/2)
    lower_bound <- estimate - z_score * SE
    upper_bound <- estimate + z_score * SE
    return (c(lower_bound, upper_bound))
}


get_normal_coverage <- function(n_x, mu_x, n_y, mu_y, alpha, nreps){
  cover_count <- 0
  real_parameter <- mu_x / mu_y
```

```
  for (i in 1:nreps){
    confidence_interval <- get_normal_confidence_interval(n_x,
mu_x, n_y, mu_y, alpha)
    if (between(real_parameter, confidence_interval[1], confid
ence_interval[2])){
      cover_count = cover_count + 1
    }
  }
  return (cover_count / nreps)
}


# this function just prints a histogram of the ratio of sample
means to determine when theta hat is relatively Normal and whe
n it is not
simulate_Xbar_over_Ybar <- function(n_x, mu_x, n_y, mu_y, nrep
s){
  Xbar_over_Ybar <- numeric(nreps)
  for (i in 1:nreps){
    X <- rpois(n_x, mu_x)
    Y <- rpois(n_y, mu_y)
    Xbar_over_Ybar[i] <- mean(X) / mean(Y)
    sd[i] <- sd(Xbar_over_Ybar)
  }
  hist(Xbar_over_Ybar, breaks = 100)
}


# here is an example of using the function that simulates cove
rage percent. n_x, n_y are the number of observations in popul
ations X and Y respectively. mu_x, mu_y are the population mea
ns of their respective populations. alpha corresponds to the c
onfidence level and nreps specifies the number of iterations
get_normal_coverage(30, 3, 30, 3, .05, 10000)
```

Before closing out this section, I will include some plots checking the normality of $\hat{\theta}$.

Figure 11 shows the sampling distribution of $\hat{\theta}$ when n_x=n_y=30 and mu_x=mu_y=3 and nreps = 10,000. This appears to be somewhat normal. The main deviations from normality are coming from the pointedness towards the center and slight skewness.

Figure 11

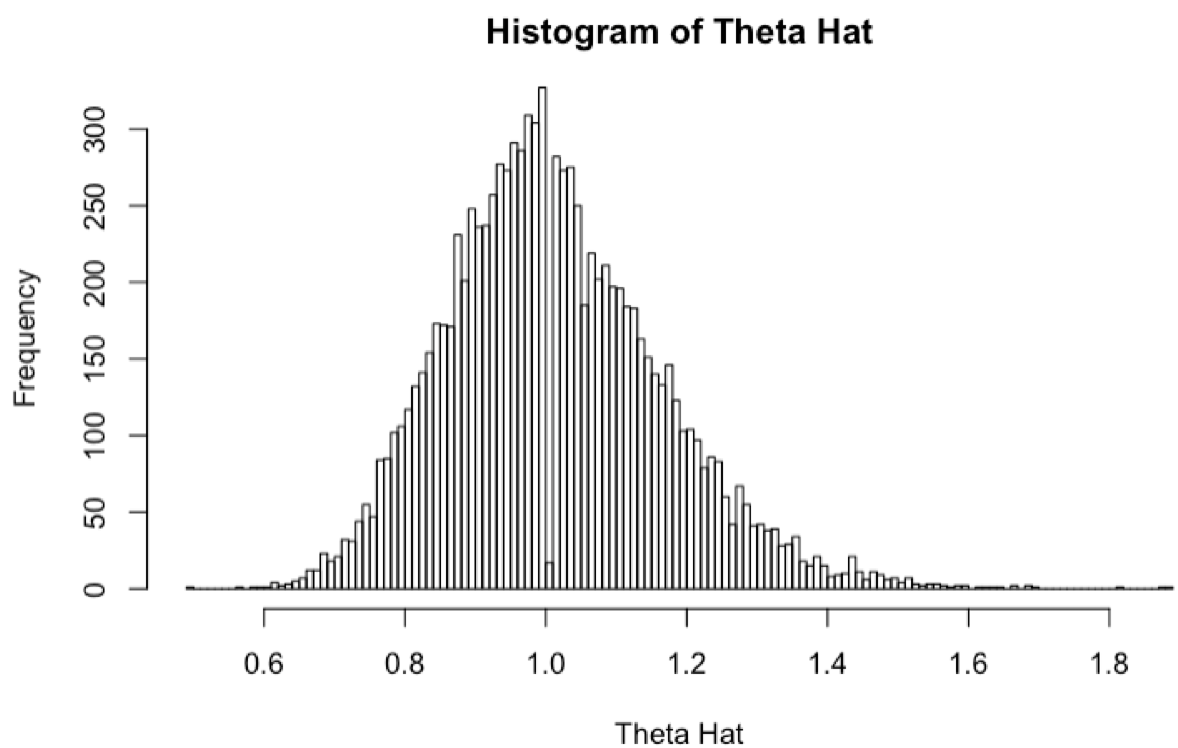
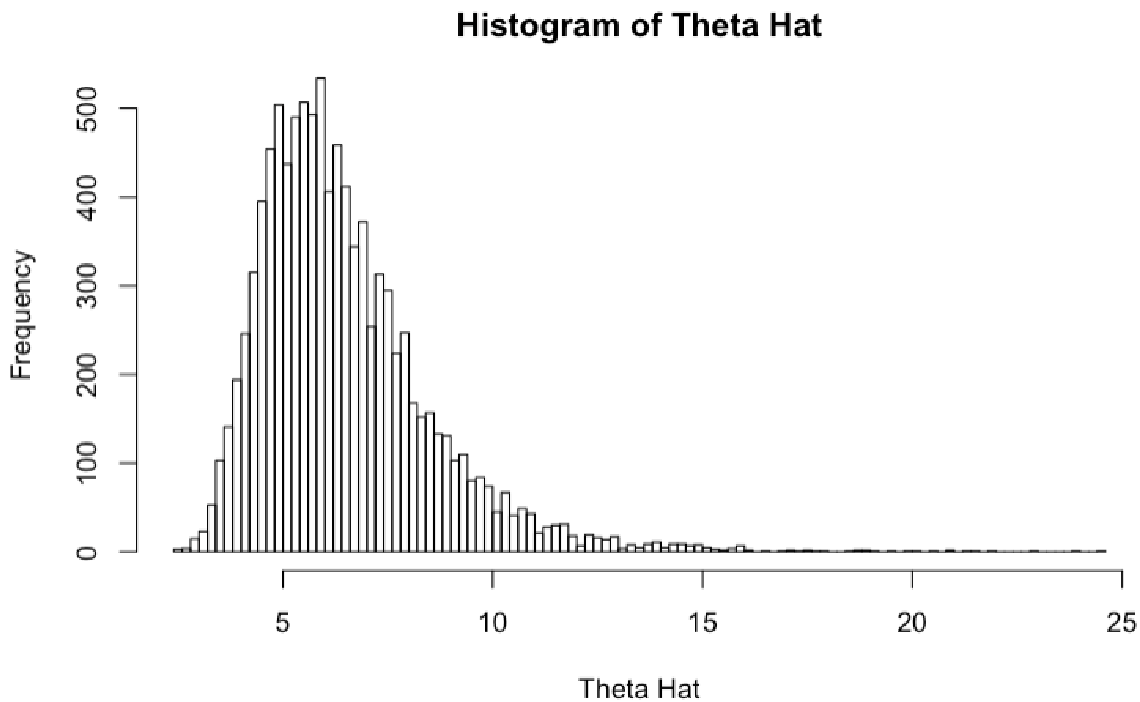


Figure 12 shows the sampling distribution of $\hat{\theta}$ when n_x=n_y=30 and mu_x=3 and mu_y=.5 with nreps = 10,000. This histogram is not normal due to its long right tail.

Figure 12



Based on Figure 11 and Figure 12 it appears that the mean parameters affect the sampling distribution of $\hat{\theta}$. I will now see if this is the case with some concrete results.

Results

Figure 13 compares and contrasts the two confidence interval procedures. Throughout, I used nreps_perCI=1,000 and num_CIs=1,000 for the bootstrap intervals (explained above) and nreps=10,000 for the Normal intervals.

Figure 13

Parameters	Bootstrap	Normal
n_x=30,mu_x=3,n_y=30,mu_y=1,cl=95%	90.7%	94.8%
n_x=30,mu_x=3,n_y=30,mu_y=3,cl=95%	93.6%	95.3%
n_x=5,mu_x=3,n_y=5,mu_y=3,cl=95%	84.9%	92.6%

n_x=30,mu_x=10,n_y=30,mu_y=1,cl=95%	91.5%	94.9%
n_x=30,mu_x=1,n_y=30,mu_y=10,cl=95%	92.2%	94.5%
n_x=10,mu_x=1,n_y=300,mu_y=10,cl=95%	88.1%	92.3%
n_x=30,mu_x=3,n_y=30,mu_y=3,cl=90%	88.8%	89.6%
n_x=30,mu_x=3,n_y=30,mu_y=3,cl=99%	96.8%	98.4%

It is very difficult to test all combinations of parameters. However, based on Figure 13, I can conclude that both interval procedures work fairly well in a wide variety of situations. Overall, it appears that the Normal intervals outperformed the bootstrap pivotal confidence intervals even in situations where the ratio of the means was large. For example, in row 4 of Figure 13, we see that when μ_x was 10 times larger than μ_y we are only .1% off in coverage from the theoretical value using the Normal interval. This is surprising given Figure 12 where μ_x was 6 times larger than μ_y and the sampling distribution of $\hat{\theta}$ was very skewed and far from Normal. For this same situation the bootstrap coverage was 3.5% off of its theoretical value.

If you take into consideration run-time, the Normal procedure was far superior. To generate 10,000 confidence intervals we need to go through 10,000 iterations of a for loop. For the bootstrap procedure on the other hand, to generate 10,000 confidence intervals we must go through 10,000 * number of bootstrap samples per confidence interval, iterations of for loops.

To summarize, I think that bootstrapping is a very interesting and powerful technique. However, in situations where we have procedures that can utilize Normal approximations, bootstrapping likely will fall short. However, the beauty of bootstrapping is that it can be applied to estimate the standard error of any statistic!