ST 502 Project 1

Taylor Cesarski and Jie Chen

Goals of Simulation Study

Methods Used & Calculations

The first method we will examine is the Wald Interval. The Wald interval is often presented in introductory statistics courses as the method used to calculate a confidence interval for a sample proportion. It relies on the Central Limit Theorem and estimates the standard error with use of the sample proportion instead of the true proportion (since this in theory would be unknown).

The Wald Interval can be found using the following function:

```
wald_int <- function(y, n, alpha = 0.05){
  c(y/n -qnorm(1-alpha/2)*sqrt(((y/n)*(1-y/n))/n),
    y/n +qnorm(1-alpha/2)*sqrt(((y/n)*(1-y/n))/n))
}
```

The third method we will examine is the Clopper-Pearson Interval. This is referred to as an exact interval because it doesn't rely on the asymptotic normality seen in the Wald Interval. The Clopper-Pearson Interval is based on the idea of "inverting equal-tailed binomial tests" and it ensures that the coverage probability is at least (1-α) (INSERT CITATION). The endpoints for this interval come from the $F_{a,b,c}$ distribution where a represents the numerator degrees of freedom, b represents the denominator degrees of freedom, and c represents the probability/area to the right.

The Clopper-Pearson Interval can be found using the following function:

```
CP_int <- function(y, n, alpha = 0.05){
  if(y==0){
    return(c(0,0))
  }
```

```
  else if(y==n){
    return(c(1,1))
  }
  else{
  c((1 + (n-y+1)/(y*qf(alpha/2,2*y, 2*(n-y+1))))^-1,
    (1 + (n-y)/((y+1)*qf(1-alpha/2, 2*(y+1), 2*(n-y))))^-1)
  }
}
```

```
bootstrap_int <- function(y, n, alpha =0.05){
  phat <- y/n

  boot_estimates <- replicate(100,{
    boot_sample <- rbinom(1, size = n, prob = phat)
    boot_phat <- boot_sample/n
  })
return(c(quantile(boot_estimates, 0.025), quantile(boot_estimates, 0.975)))
}
```

The fifth method we will examine is a raw percentile interval using a parametric bootstrap. The idea of a parametric bootstrap is to resample from a fitted distribution with replacement to get many samples of the same size. Then find the bootstrap estimate and generate the bootstrap distribution. Finally, we will use the (1-α/2) and α/2 quantiles from this bootstrap distribution as the confidence interval.

Creation of Data Process and Code

We want to generate 1000 random samples from a binomial where n varies across n=15, 30, and 100 and p values from 0.01 to 0.99 for 15 total values of p for a total of 45 combinations. We will then generate 1000 95% confidence intervals for each of our combinations and find properties of these confidence intervals. In order for a confidence interval to be "good", we want the observed coverage probability to be equal to or close to (1-α). Some intervals will inherently miss where the true probability falls below or above the interval. Ideally, we would want these proportions that miss on both sides to be equal to α/2. We also want a relatively narrow interval so that it gives us helpful information. For example, it is not helpful to say the true probability is between 0 and 1 or even 0.05 and 0.95 because the interval is too wide to even be beneficial. Therefore, we want to compare the average lengths of our intervals.

We will start by considering the Wald Interval.

```r
set.seed(200)
sample_sizes <- c(15, 30, 100)
p <- c(0.01, 0.05, 0.10, 0.20, 0.25, 0.35, 0.45, 0.5, 0.6, 0.65, 0.75, 0.8, 0.9, 0.95, 0.9

wald_confidence_int <- list()
cp_confidence_int <- list()
boot_percentile_confidence_int <- list()

for(size in sample_sizes){
  for(prob in p){
   wald_CIs <- data.frame(lower=numeric(1000), upper=numeric(1000))
   cp_CIs <- data.frame(lower=numeric(1000), upper=numeric(1000))
   boot_CIs <- data.frame(lower=numeric(1000), upper=numeric(1000))

   data <- rbinom(n=1000, size, prob)
for(i in 1:1000){
  y<-data[i]

  wald_interval <- wald_int(y, size)

  wald_CIs$lower[i] <- wald_interval[1]
  wald_CIs$upper[i] <- wald_interval[2]

  cp_interval <- CP_int(y, size)

  cp_CIs$lower[i] <- cp_interval[1]
  cp_CIs$upper[i] <- cp_interval[2]

  boot_interval <- bootstrap_int(y, size)

  boot_CIs$lower[i] <- boot_interval[1]
  boot_CIs$upper[i] <- boot_interval[2]

}
  wald_confidence_int[[paste0("Wald", "size_", size, "prob_", prob)]] <- wald_CIs
  cp_confidence_int[[paste0("Clopper-Pearson", "size_", size, "prob_", prob)]] <- cp_CIs
  boot_percentile_confidence_int[[paste0("Bootstrap Percentile", "size_", size, "prob_", p

  }
}
```

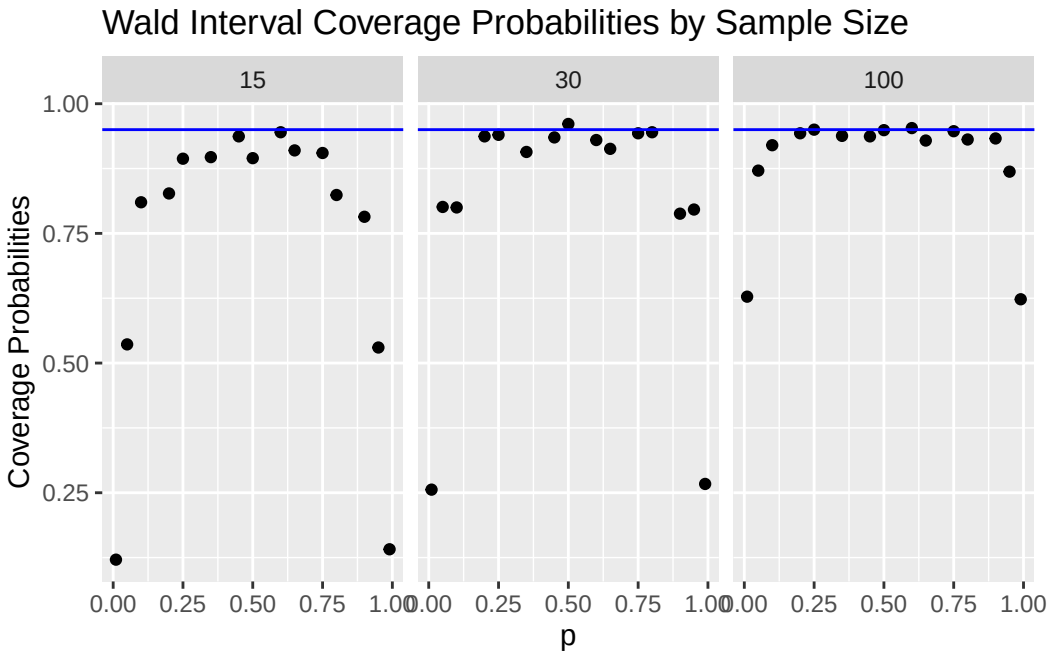
Code to Execute Monte Carlo Study (maybe join with above)

We will create a data frame for the Wald Interval that contains the sample size (n), the probability of success (p), the average length of the intervals, the coverage probability, the proportion of intervals that miss above, and the proportion of intervals that miss below.

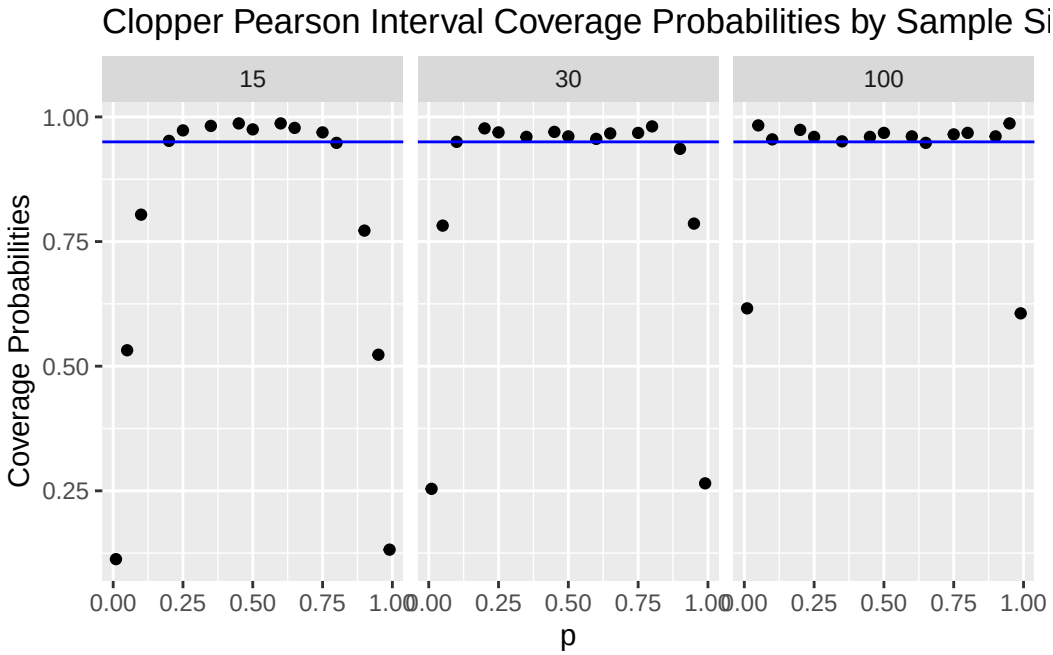
```
#Initialize a "lengths" numeric vector to hold average lengths of size 45.
wald_lengths <- numeric(45)
cp_lengths <- numeric(45)
boot_lengths <- numeric(45)

#Use a for loop to calculate the average lengths for each set of 1000 intervals.
for(i in 1:45){
  wald_lengths[i] <- mean(wald_confidence_int[[i]]$upper - wald_confidence_int[[i]]$lower)
  cp_lengths[i] <- mean(cp_confidence_int[[i]]$upper - cp_confidence_int[[i]]$lower)
  boot_lengths[i] <- mean(boot_percentile_confidence_int[[i]]$upper - boot_percentile_conf

}

#Create a vector that contains the probabilities 3 times.
repeated_p <- rep(p, 3)
#Initialize a numeric vector of length 45 to hold the coverage probabilities.
wald_coverage_probabilities <- numeric(45)
cp_coverage_probabilities <- numeric(45)
boot_coverage_probabilities <- numeric(45)


#Use a for loop to calculate the coverage probability (% containing true p) for each set o
for (i in 1:45) {
  wald_coverage_probabilities[i] <- mean(wald_confidence_int[[i]]$upper > repeated_p[i] &
  cp_coverage_probabilities[i] <- mean(cp_confidence_int[[i]]$upper > repeated_p[i] & cp_c
  boot_coverage_probabilities[i] <- mean(boot_percentile_confidence_int[[i]]$upper > repea
}



#Find the percentage where the true p is above the interval.
wald_miss_low <- numeric(45)
cp_miss_low <- numeric(45)
boot_miss_low <- numeric(45)

for (i in 1:45) {
```

```r
  wald_miss_low[i] <- mean(wald_confidence_int[[i]]$upper < repeated_p[i])
  cp_miss_low[i] <- mean(cp_confidence_int[[i]]$upper < repeated_p[i])
  boot_miss_low[i] <- mean(boot_percentile_confidence_int[[i]]$upper < repeated_p[i])
}

#Find the percentage where the true p is below the interval.

wald_miss_high <- numeric(45)
cp_miss_high <- numeric(45)
boot_miss_high <- numeric(45)

for (i in 1:45) {
  wald_miss_high[i] <- mean(wald_confidence_int[[i]]$lower > repeated_p[i])
  cp_miss_high[i] <- mean(cp_confidence_int[[i]]$lower > repeated_p[i])
  boot_miss_high[i] <- mean(boot_percentile_confidence_int[[i]]$lower > repeated_p[i])
}


#Create a data frame to hold information about the Wald Interval.

#Create a vector of the values of n.
n_values <- c(rep(15,15), rep(30, 15), rep(100,15))
#Use the repeated p values created above.
repeated_p <- rep(p, 3)

#Store the n, p, lengths, coverage probabilities, and % missing above and below.
wald_summary <- data.frame(n = n_values, p = repeated_p, avg_int_lengths = wald_lengths, c

cp_summary <- data.frame(n = n_values, p = repeated_p, avg_int_lengths = cp_lengths, cover

boot_summary <- data.frame(n = n_values, p = repeated_p, avg_int_lengths = boot_lengths, c


#Summary with just n, p, and coverage probabilities.
simplified_wald_summary <- data.frame(sample_size = n_values, p = repeated_p, coverage_pro

simplified_cp_summary <- data.frame(sample_size = n_values, p = repeated_p, coverage_probs

simplified_boot_summary <- data.frame(sample_size = n_values, p = repeated_p, coverage_pro
```

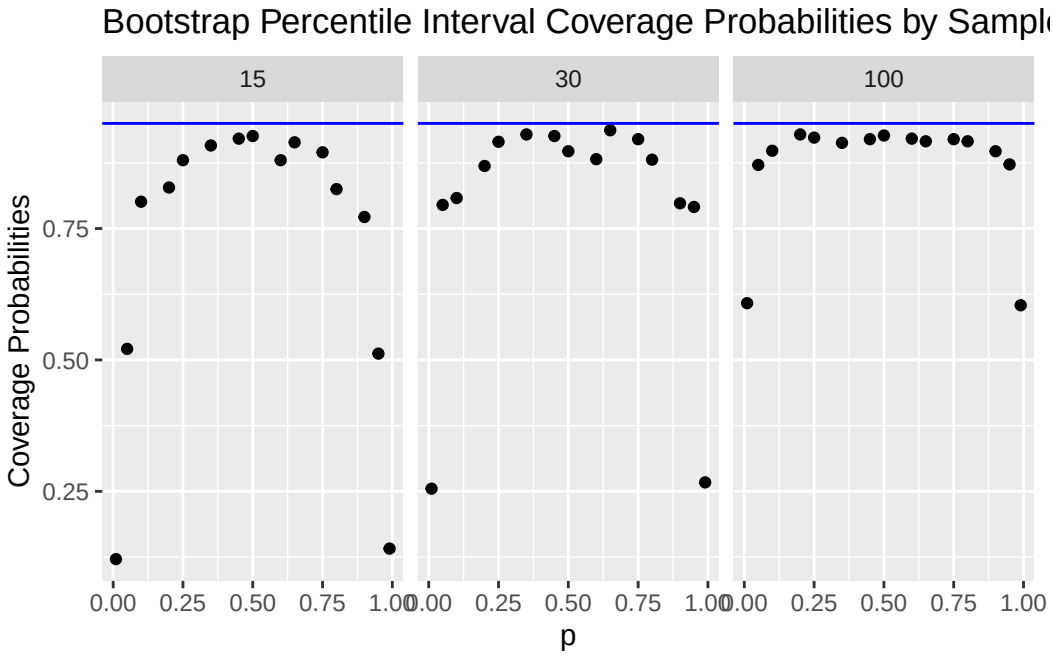
```r
library(ggplot2)
ggplot(simplified_wald_summary, aes(x=p, y=wald_coverage_probabilities)) +
  geom_point()+
  facet_wrap(~sample_size) +
  labs(title = "Wald Interval Coverage Probabilities by Sample Size",
       y = "Coverage Probabilities") +
  geom_hline(yintercept = 0.95, color = "blue")
```




```r
ggplot(simplified_cp_summary, aes(x=p, y=cp_coverage_probabilities)) +
  geom_point()+
  facet_wrap(~sample_size) +
  labs(title = "Clopper Pearson Interval Coverage Probabilities by Sample Size",
       y = "Coverage Probabilities") +
  geom_hline(yintercept = 0.95, color = "blue")
```




```
ggplot(simplified_boot_summary, aes(x=p, y=boot_coverage_probabilities)) +
  geom_point()+
  facet_wrap(~sample_size) +
  labs(title = "Bootstrap Percentile Interval Coverage Probabilities by Sample Size",
       y = "Coverage Probabilities") +
  geom_hline(yintercept = 0.95, color = "blue")
```


Bootstrap Percentile Interval Coverage Probabilities by Sampl
15
30
100
Coverage Probabilities
0.75
0.50
0.25
0.00
0.25
0.50
0.75
1.00
p