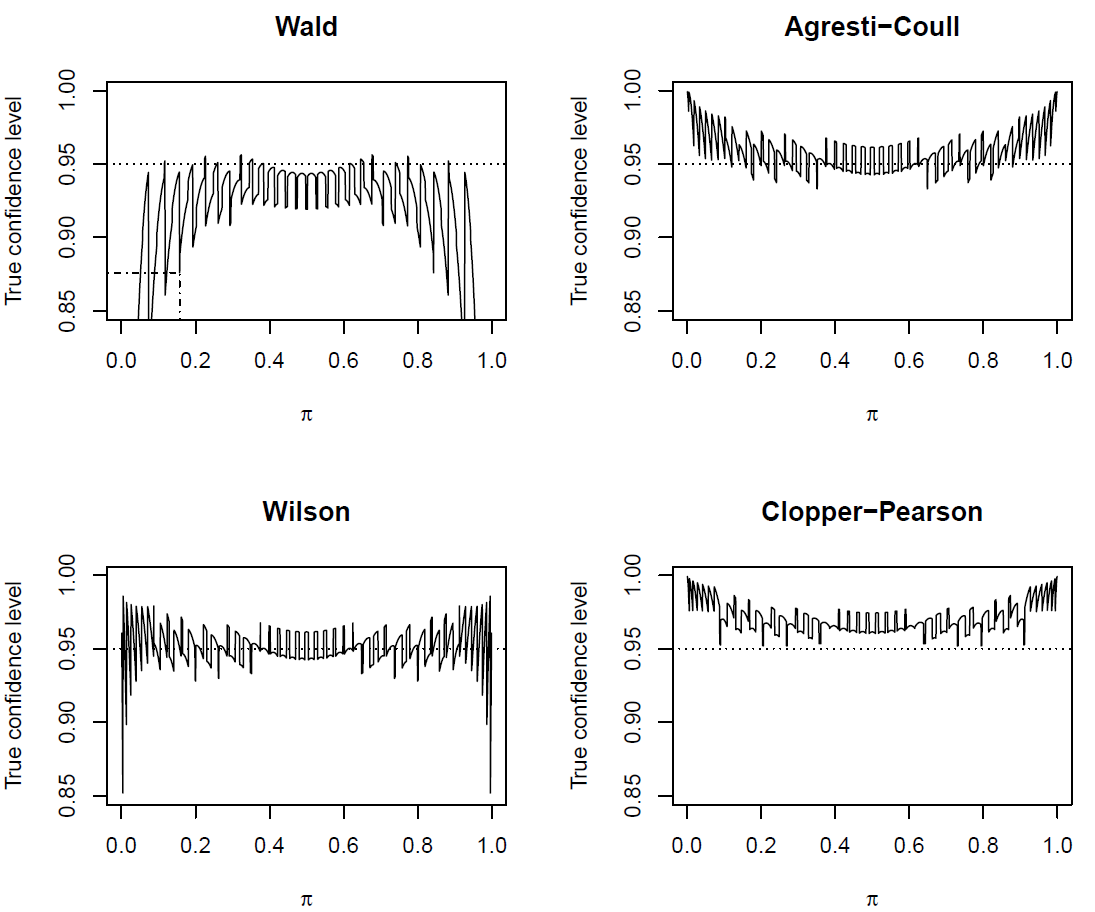
**Section 1.1.3 – True confidence levels for confidence intervals**

Below is a comparison of the performance of four confidence intervals for n = 40. The solid line represents the true confidence level for a particular value of π.



What does the true confidence level (a.k.a., coverage level) mean?

* Suppose a random sample of size n = 40 is taken from a population and a 95% Wald confidence interval is calculated.
* Suppose another random sample of size n = 40 is taken from the same population and a 95% Wald confidence interval is calculated.
* Suppose this process is repeated 10,000 times.
* We would expect 9,500 out of 10,000 (95%) confidence intervals to contain π.
* Unfortunately, this does not often happen. It only is guaranteed to happen when n = ∞ for the Wald interval.
* The true confidence level is the percent of times the confidence intervals contain or “cover” π.

This process described by above is called a Monte Carlo simulation.

The plots of the previous page actually perform these calculations a little different way, which we will soon discuss. For now, you can think of these plots as being constructed by repeating the Monte Carlo simulation for values of π from 0.0005 to 0.9995 by 0.0005. At each π, the true confidence level is calculate. For example, the true confidence level using the Wald interval is 0.8759 for π = 0.157 (see dot-dashed line). For the Wilson and Agresti-Coull intervals, it is 0.9507 at this same value of π.

Example: Calculate estimated true confidence level for Wald (ConfLevel.R)

Suppose 1,000 samples are taken (rather than 10,000 mentioned previously) to speed up the demonstration here.

> alpha <- 0.05

> pi <- 0.157

> n <- 40

> numb.bin.samples <- 1000 # Number of binomial samples of size n

> set.seed(4516)

> w <- rbinom(n = numb.bin.samples, size = n, prob = pi)

> counts <- table(x = w)

> counts

x

1 2 3 4 5 6 7 8 9 10 11 12 13

8 35 64 123 147 165 172 123 76 46 26 11 4

> pi.hat <- w/n

> pi.hat[1:10]

[1] 0.150 0.150 0.175 0.200 0.200 0.150 0.200 0.075 0.125 0.100

> var.wald <- pi.hat\*(1-pi.hat)/n

> lower <- pi.hat - qnorm(p = 1-alpha/2) \* sqrt(var.wald)

> upper <- pi.hat + qnorm(p = 1-alpha/2) \* sqrt(var.wald)

> data.frame(w, pi.hat, lower, upper)[1:10,]

w pi.hat lower upper

1 6 0.150 0.039344453 0.2606555

2 6 0.150 0.039344453 0.2606555

3 7 0.175 0.057249138 0.2927509

4 8 0.200 0.076040994 0.3239590

5 8 0.200 0.076040994 0.3239590

6 6 0.150 0.039344453 0.2606555

7 8 0.200 0.076040994 0.3239590

8 3 0.075 -0.006624323 0.1566243

9 5 0.125 0.022511030 0.2274890

10 4 0.100 0.007030745 0.1929693

> save <- pi>lower & pi<upper # Check if pi is within interval

> save[1:10]

[1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

> mean(save)

[1] 0.878

> true.conf <- mean(save)

> cat("An estimate of the true confidence level is:", round(true.conf,4), "\n")

An estimate of the true confidence level is: 0.878

Important: Notice how the calculations are performed using vectors.

We saw previously that for a Wald interval that



for a large sample when using a normal distribution approximation for the random variable version of . What we are doing in this example is trying to calculate the left side of this expression through estimating the probabilities for W = 0, 1, …, n with the use of Monte Carlo simulation.

We can sum up the estimated probabilities for Wwhen its corresponding confidence interval contains π. Through calculating each possible confidence interval for π when n = 40, one will notice that π = 0.157 is within the interval for w = 4 to 11 (this will be demonstrated shortly). Thus,

> sum(counts[4:11])/numb.bin.samples

[1] 0.878

> prop.w <- counts/numb.bin.samples # Proportion for each w

> obs.w <- as.integer(names(table(w))) # Obtain w number

> binom.prob <- round(dbinom(x = obs.w, size = n, prob = pi), digits = 4)

> data.frame(obs.prop = prop.w, binom.prob = binom.prob)

obs.prop.x obs.prop.Freq binom.prob

1 1 0.008 0.0080

2 2 0.035 0.0292

3 3 0.064 0.0689

4 4 0.123 0.1187

5 5 0.147 0.1591

6 6 0.165 0.1729

7 7 0.172 0.1564

8 8 0.123 0.1201

9 9 0.076 0.0795

10 10 0.046 0.0459

11 11 0.026 0.0233

12 12 0.011 0.0105

13 13 0.004 0.0042

> sum(prop.w)

[1] 1

> sum(binom.prob) # Note: not equal to 1 because some possible values of w were not observed

[1] 0.9967

> ck <- obs.w >= 4 & obs.w <= 11

> sum(prop.w\*ck)

[1] 0.878

We can find the actual true confidence level without Monte Carlo simulation! Below are the steps:

1. Find all possible intervals that one could have with w = 0, 1, …, n.
2. Form I(w) = 1 if the interval for a w contains π and 0 otherwise.
3. Calculate the true confidence level as



The key to using a non-simulation based approach is there are a finite number of possible values for the random variable of interest. In other settings beyond confidence intervals for π, this will usually not occur and simulation will be the only approach for a finite sample size n.

Example: Calculate actual true confidence level for Wald (ConfLevel.R)

> alpha <- 0.05

> pi <- 0.157

> n <- 40

> w <- 0:n

> pi.hat <- w/n

> pmf <- dbinom(x = w, size = n, prob = pi)

> var.wald <- pi.hat\*(1-pi.hat)/n

> lower <- pi.hat - qnorm(p = 1-alpha/2) \* sqrt(var.wald)

> upper <- pi.hat + qnorm(p = 1-alpha/2) \* sqrt(var.wald)

> save <- pi>lower & pi<upper

> sum(save\*pmf)

[1] 0.875905

> data.frame(w, pi.hat, round(data.frame(pmf, lower, upper),4), save)[1:13,]

w pi.hat pmf lower upper save

1 0 0.000 0.0011 0.0000 0.0000 FALSE

2 1 0.025 0.0080 -0.0234 0.0734 FALSE

3 2 0.050 0.0292 -0.0175 0.1175 FALSE

4 3 0.075 0.0689 -0.0066 0.1566 FALSE

5 4 0.100 0.1187 0.0070 0.1930 TRUE

6 5 0.125 0.1591 0.0225 0.2275 TRUE

7 6 0.150 0.1729 0.0393 0.2607 TRUE

8 7 0.175 0.1564 0.0572 0.2928 TRUE

9 8 0.200 0.1201 0.0760 0.3240 TRUE

10 9 0.225 0.0795 0.0956 0.3544 TRUE

11 10 0.250 0.0459 0.1158 0.3842 TRUE

12 11 0.275 0.0233 0.1366 0.4134 TRUE

13 12 0.300 0.0105 0.1580 0.4420 FALSE

> # For pi = 0.157

> sum(dbinom(x = 4:11, size = n, prob = pi))

[1] 0.875905

How can you perform this simulation for more than one π and then produce a coverage plot similar to what was shown at the beginning of these notes?

Repeat the process for π from 0.0005 to 0.9995 by 0.0005 and plot the actual true confidence levels. See ConfLevelsWaldOnly.R and ConfLevel4Intervals.R.

Why does the true confidence level (coverage) oscillate? The reason involves the discreteness of a binomial random variable. Examine the data frame in the previous example relative to if π = 0.156.

Clopper-Pearson interval

Perhaps you always want to a have true confidence level at or greater than the stated level. If so, you can use the Clopper-Pearson interval!

First, we need to discuss what a beta probability distribution represents. Let V be a random variable from a beta distribution. Then



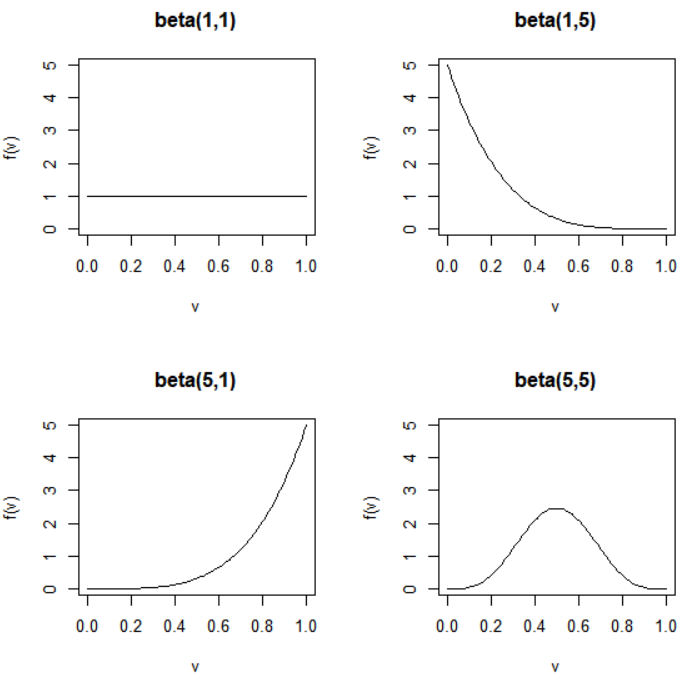
for 0 < v < 1, a > 0, and b > 0. Note that Γ() is the “gamma” function where Γ(c) = (c-1)! for an integer c. The gamma function is more generally defined as

 for c > 0.

The α quantile of a beta distribution, denoted by vα or beta(α; a, b), is

.

Below are plots of various Beta distributions from Beta.R



The Clopper-Pearson confidence interval is

beta(α/2; w, n-w+1) < π < beta(1-α/2; w+1, n-w)

This is derived using the relationship between the binomial and beta distributions (see problem #2.40 on p. 82 of Casella and Berger, 2002). If w = 0, the lower limit is taken to be 0. If w = n, the upper limit is taken to be 1. Because a beta distribution is used and because how lower and upper limits are found when w = 0 or n, the interval’s limits are always between 0 and 1!

Often, the interval is stated using an F-distribution instead. This comes about the relationship between beta and F-distributions. The two intervals are the same in the end. I use the beta distribution version becomes it is more compact and easy to calculate via a computer.

This Clopper-Pearson (CP) interval is GUARANTEED to have an actual true confidence (coverage) level ≥ 1-α! Examine the actual true confidence level plot at the beginning of these notes. Below are comments about its true confidence level:

* This is an example of a conservative interval.
* Conservative intervals often achieve this conservativeness due to being wider than other intervals. Why would a wide interval be bad?
* Brown et al. (2001, p. 113) say this interval is “wastefully conservative and is not a good choice for practical use”. I think this is a little harsh.
* The Blaker (2000, 2001) interval provides a little less conservative version of an exact interval. If you are interested in details about it, please see the paper and a short discussion in my book.

Intervals like the Clopper-Pearson are called “exact” confidence intervals because they use the exact distribution for W (binomial). This does not mean their true confidence level is exactly (1-α)100% as you can see from the actual true confidence level plot.

Example: Field goal kicking (CIpi.R)

Below is the code used to calculate the Clopper-Pearson confidence interval.

> lower <- qbeta(p = alpha/2, shape1 = w, shape2 = n-w+1)

> upper <- qbeta(p = 1-alpha/2, shape1 = w+1, shape2 = n-w)

> data.frame(lower, upper)

lower upper

1 0.1215523 0.7376219

> qbeta(p = c(alpha/2, 1-alpha/2), shape1 = c(w, w+1), shape2 = c(n-w+1, n-w))

[1] 0.1215523 0.7376219

The binom package calls the Clopper-Pearson interval the “exact” interval.

> library(binom)

> binom.confint(x = w, n = n, conf.level = 1 –

alpha, methods = "exact")

method x n mean lower upper

1 exact 4 10 0.4 0.1215523 0.7376219

A summary of all intervals:

|  |  |  |
| --- | --- | --- |
|  | Lower | Upper |
| Wald | 0.0964 | 0.7036 |
| Agresti-Coull | 0.1671 | 0.6884 |
| Wilson | 0.1682 | 0.6873 |
| Clopper-Pearson | 0.1216 | 0.7376 |