# Binomial Proportion Confidence Interval

*Art Eschenlauer*

*January 23, 2019*

## Contents



### Introductory remarks

Many ways have been developed to estimate the confidence interval for an observed propoortion. See https://en.wikipedia.org/wiki/Binomial_proportion_confidence_interval for a list of them. Agresti and Coull (1998) recommend that elementary statistics instruction prefer the "Score" method of Wilson (1927) for estimating the confidence interval over the simpler, but insufficiently conservative, normal approximation. I prefer the Wilson Score method because it performs well for proportions that approach 0 or 1, where the normal approximation breaks down. It is a bit more complicated to calculate.

### A sanity check

Before computing the confidence interval, I first will take some wild guesses at the probabilities near the boundaries. For my example, I will assume that 73 successes were observed for 76 trials, and that I want a 95% confidence interval.

I manually adjusted my guesses to obtain a 2.5% area under probability density function (PDF) curve for each tail. This would require integration, but the cumulative density function (CDF) is the integration of the PDF. For R, the `stats::pbinom` function implements CDF for the binomial distribution.

So, what value do I expecting to be near the lower limit of the CI?

```
successes <- 73
trials <- 76
proportion <- as.double(successes) / as.double(trials)
pbinom(q = successes, size = trials, prob = proportion)
```

```
## [1] 0.5813266
```

```
low.guess <- 0.90816
high.guess <- 0.99178
pbinom(q = successes, size = trials, prob = low.guess)
```

```
## [1] 0.974985
```

And, what value do I expecting to be near the upper limit of the CI?

```
pbinom(q = successes, size = trials, prob = high.guess, lower.tail = FALSE)
```

```
## [1] 0.9749656
```

### The Wilson Score Interval - from Wilson 1927

The 1927 paper by Edwin Bidwell Wilson presented the following formula as a practical way to estimate the confidence interval for a proportion:

$$p = \frac{p_0 + \frac{t}{2} \pm \sqrt{p_0 q_0 t + \frac{t^2}{4}}}{1+t} \quad \left| \quad \begin{array}{l} \lambda = 1.96 \text{ (for 95 percent confidence)}, \\ t = \dfrac{\lambda^2}{n}, \quad p_0 = \dfrac{x}{n}, \quad q_0 = \dfrac{n-x}{n} \end{array} \right.$$

The CRAN R package `epitools` provides the `binom.wilson` function which can compute this:

```
epi_wilson_ci <-
  epitools::binom.wilson(x = successes, n = trials, conf.level = 0.95)
print(epi_wilson_ci)
```

```
##    x  n proportion     lower     upper conf.level
## 1 73 76  0.9605263 0.8902521 0.9864854       0.95
```

To demystify the `epitools` implemntation, I coded it directly from the equation above:

```
# computational equivalent of binom.wilson from CRAN 'epitools' package
binom.wilson <-
  function(x, n, conf.level = 0.95) {
    lambda  <- qnorm(0.5*(1 + conf.level))
    # Adapted from Wilson 1927, mostly by multipling numerator
    #   and denominator by n to reduce roundoff error
    n_t     <- ( lambda^2 )
    n_p_0   <- x
    n_q_0   <- n - x
    denom   <- n + n_t
    center  <- ( n_p_0 + n_t / 2 )
    radical <- ( n_p_0 * n_q_0 + n * (n_t / 4) ) * n_t / n
    # continuing
    delta   <- sqrt(radical)
    R.lower <- ( center - delta ) / denom
    R.upper <- ( center + delta ) / denom
    data.frame(
      x = x,
      n = n,
      proportion = as.double(n_p_0) / as.double(n),
      lower = R.lower,
      upper = R.upper,
      conf.level = conf.level
    )
  }
wilson_ci <- binom.wilson(x = successes, n = trials, conf.level = 0.95)
print(wilson_ci)
```

```
##    x  n proportion     lower     upper conf.level
## 1 73 76  0.9605263 0.8902521 0.9864854       0.95
```

There is negligible difference between the `epitools` implementation and my own:

```r
print(wilson_ci$lower - epi_wilson_ci$lower)
```

```
## [1] 1.110223e-16
```

```r
print(wilson_ci$upper - epi_wilson_ci$upper)
```

```
## [1] 1.110223e-16
```

I have not contrasted the two implementations to see which suffers more from round-off error, but it seems clear that they are consistent with one another.

## Conclusions

```r
print(
  sprintf("For %d successes over %d trials:"
  , wilson_ci$x
  , wilson_ci$n
  )
)
```

```
## [1] "For 73 successes over 76 trials:"
```

```r
print(
  sprintf("Proportion: %f"
  , wilson_ci$proportion
  )
)
```

```
## [1] "Proportion: 0.960526"
```

```r
print(
  sprintf("Wilson estimate of confidence interval for proportion: [%f,%f]"
  , wilson_ci$lower
  , wilson_ci$upper
  )
)
```

```
## [1] "Wilson estimate of confidence interval for proportion: [0.890252,0.986485]"
```

```r
print(
  sprintf("My low and high guesses: %f,%f"
  , low.guess
  , high.guess
  )
)
```

```
## [1] "My low and high guesses: 0.908160,0.991780"
```

```
print(
  sprintf("Differences from my guesses: %f, %f"
  , wilson_ci$lower - low.guess
  , wilson_ci$upper - high.guess
  )
)
```

```
## [1] "Differences from my guesses: -0.017908, -0.005295"
```

So, this seems to me to pass the sanity checks. I think that the results are not the same at least in part because the sanity checks are based on different actual proportions.

## About this document

This document was produced from an RMarkdown file, the source of which should be at:

https://eschen42.github.io/propci/propci.Rmd

## References

Agresti, Alan; Coull, Brent A. (1998). "Approximate is better than 'exact' for interval estimation of binomial proportions". The American Statistician. 52: 119-126. doi:10.2307/2685469.

Wilson, E. B. (1927). "Probable inference, the law of succession, and statistical inference". Journal of the American Statistical Association. 22: 209-212. doi:10.1080/01621459.1927.10502953.