Stat 505 Assignment 11 Solutions

18 points

1. Exercise 1 p 152-3

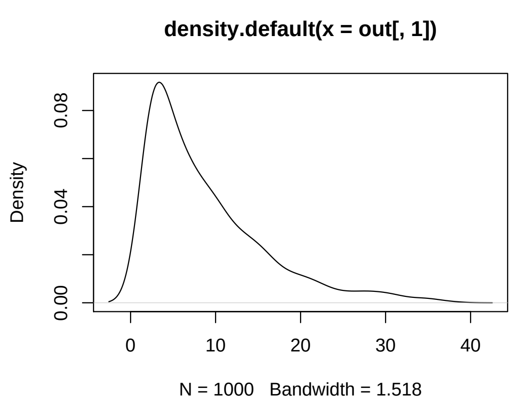
   (a) Here's my function. 3

```
shotfn <- function(p = 0.6) {
    ## function to record shots and score till 2 in a row are missed.
    if (p > 1 | p < 0)
        stop("p must be between 0 and 1")
    n <- score <- 0  ## initialize counters
    keep.going <- TRUE
    while (keep.going) {
        new.shot <- rbinom(1, 1, p)
        if (new.shot == 1) {
            n <- n + 1  ## one more shot
            score <- score + 1  ## add 0 or 1 to score
        } else {
            second.shot <- rbinom(1, 1, p)
            n <- n + 2  ## another shot
            score <- score + second.shot  ## add 1 or 0
            if (second.shot == 0) {
                keep.going <- FALSE  ## time to retire
            }
        }
    }  ## return number of shots and (total shots made) = score
    unlist(list(n.shots = n, score = score))
}
```

   (b) Running 1000 times, I get this distribution of shot numbers.

	Min	Q1	Median	Q3	Max	Mean	StdDev
1	2.00	3.00	7.00	12.00	38.00	8.80	6.99

2



   (c) Scatterplot of number of shots versus proportion of shots made.

1

2. The methods we teach for building confidence intervals for a proportion are based on asymptotic normality of the sample proportion. For small sample sizes, it is not surprising that they don't provide the coverage we'd like. More surprising to me is the fact that coverage rates also vary depending on the true proportion.
   This article suggests that we should use what some others call the "Wilson" estimator, which adds two success and two failures to the counts before computing the sample proportion.

   (a) How do we estimate the long run coverage rate of a method for building confidence intervals? 1
   We need to generate random data from a known proportion, build a confidence interval using the method of interest, and check to see if the interval contains the proportion we started with. By repeating the process many times, we can estimate the coverage rate: the proportion of times that the method captures the true parameter, and compare that to the 'nominal' or 'target' coverage rate.

   (b) Build a function (to take arguments n, p, Nreps, and confidence = 1-) which will generate Nreps random Binomials from Bin(n,p) distribution, create the standard "Wald" CI with the specified confidence level, and check to see what proportion of the intervals contain the true p. 1

```
Wald.cover1 <- function(p, n, Nreps, alpha) {
    ## Compute mean coverage of the usual confidence interval
    phat <- rbinom(Nreps, n, p)/n
    sum(abs(p - phat)/sqrt(phat * (1 - phat)/n) < qnorm(1 - alpha/2))/Nreps
}
```

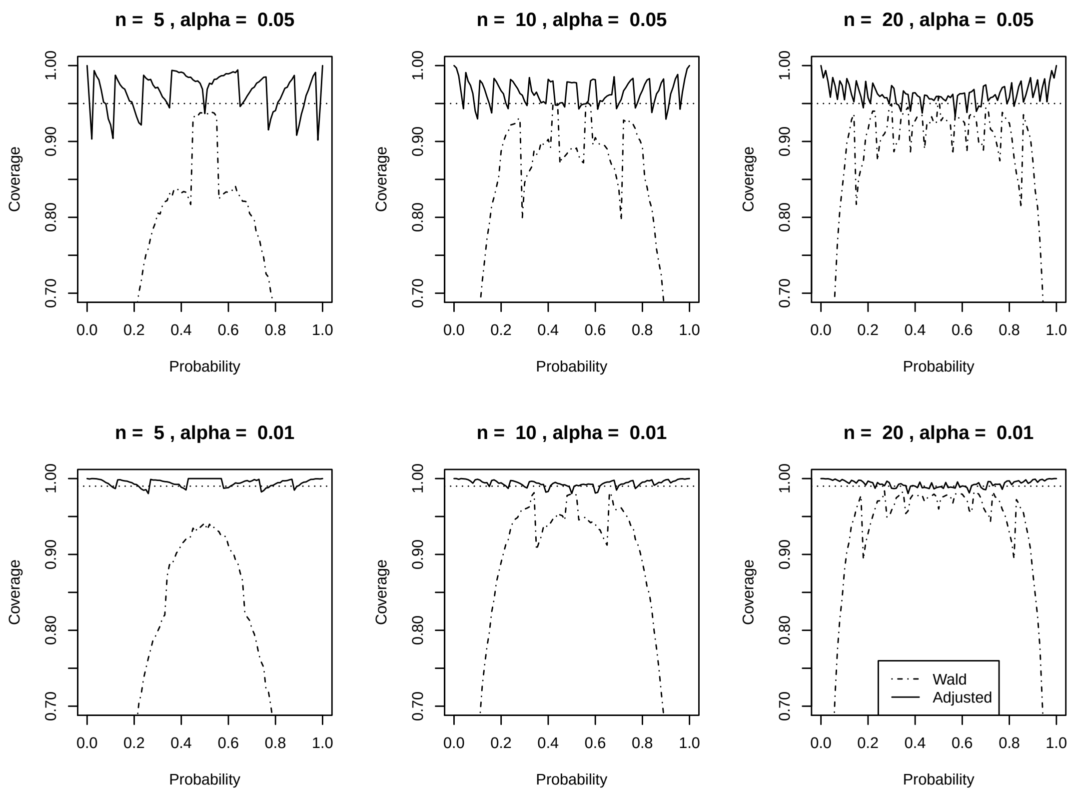
   No loop is needed. This computes a vector of binomials and checks overall coverage rate by simply summing TRUE's when $\widehat{p}$ is within our margin of error of the initial p.

   (c) Similarly, build a function to compute coverage for the "Wilson" confidence interval method. 1

```
Wilson.cover1 <- function(p, n, Nreps, alpha) {
    ## Compute mean coverage of a Wilson confidence interval
```

```
    ptilde <- (2 + rbinom(Nreps, n, p))/(n + 4)
    sum(abs(p - ptilde)/sqrt(ptilde * (1 - ptilde)/(n + 4)) < qnorm(1 -
        alpha/2))/Nreps
}
```

(d) Recreate Figure 1 in the article, comparing the two methods at confidence levels 95 and 99% for sample sizes 5, 10, 20, and for a sequence of at least 50 true proportions (p) from 0 to 1.



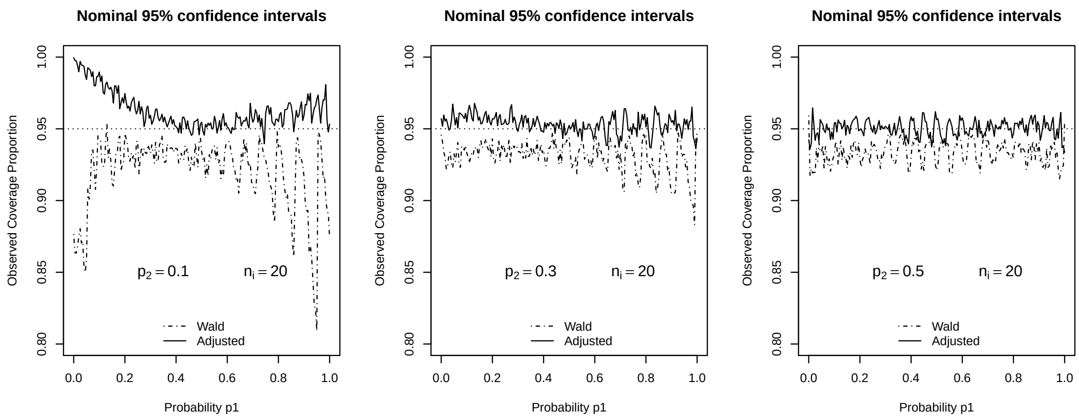
4

I use `sapply` *to "apply" the* `Wald.cover1` *function to each probability in turn.*

(e) Recreate Figure 4 for a 95% CI for a difference in two proportions using the two methods.

2

```
Wald.cover2 <- function(p1, p2 = 0.3, n1 = 20, n2 = 20, nsim = 5000,
    alpha = 0.05) {
    phat1 <- rbinom(nsim, n1, p1)/n1
    phat2 <- rbinom(nsim, n2, p2)/n2
    sum(abs(phat1 - phat2 - (p1 - p2))/sqrt(phat1 * (1 - phat1)/n1 +
        phat2 * (1 - phat2)/n2) < qnorm(1 - alpha/2))/nsim
}
Wilson.cover2 <- function(p1, p2 = 0.3, n1 = 20, n2 = 20, nsim = 5000,
    alpha = 0.05) {
    ptilde1 <- (1 + rbinom(nsim, n1, p1))/(n1 + 2)
    ptilde2 <- (1 + rbinom(nsim, n2, p2))/(n2 + 2)
    sum(abs(ptilde1 - ptilde2 - (p1 - p2))/sqrt(ptilde1 * (1 - ptilde1)/(n1 +
        2) + ptilde2 * (1 - ptilde2)/(n2 + 2)) < qnorm(1 - alpha/2))/nsim
}
```



3

R Code

1. Shots

```
shotfn <- function(p = 0.6) {
    ## function to record shots and score till 2 in a row are missed.
    if (p > 1 | p < 0)
        stop("p must be between 0 and 1")
    n <- score <- 0  ## initialize counters
    keep.going <- TRUE
    while (keep.going) {
        new.shot <- rbinom(1, 1, p)
        if (new.shot == 1) {
            n <- n + 1  ## one more shot
            score <- score + 1  ## add 0 or 1 to score
        } else {
            second.shot <- rbinom(1, 1, p)
            n <- n + 2  ## another shot
            score <- score + second.shot  ## add 1 or 0
            if (second.shot == 0) {
                keep.going <- FALSE  ## time to retire
            }
        }
    }  ## return number of shots and (total shots made) = score
    unlist(list(n.shots = n, score = score))
}
```

```
out <- t(sapply(1:1000, function(x) shotfn()))
xtable(matrix(c(fivenum(out[, 1]), mean(out[, 1]), sd(out[, 1])), 1,
    7, dimnames = list(NULL, c("Min", "Q1", "Median", "Q3", "Max", "Mean",
        "StdDev"))))
plot(density(out[, 1]))
```

```
out <- cbind(out[, 1:2], out[, 2]/out[, 1])
plot(jitter(out[, 1], 1), jitter(out[, 3], 5), xlab = "Number of Shots",
    ylab = "Proportion Made")
```

2. Confidence Interval Coverage

```
Wald.cover1 <- function(p, n, Nreps, alpha) {
    ## Compute mean coverage of the usual confidence interval
    phat <- rbinom(Nreps, n, p)/n
    sum(abs(p - phat)/sqrt(phat * (1 - phat)/n) < qnorm(1 - alpha/2))/Nreps
}
```

```
Wilson.cover1 <- function(p, n, Nreps, alpha) {
    ## Compute mean coverage of a Wilson confidence interval
    ptilde <- (2 + rbinom(Nreps, n, p))/(n + 4)
    sum(abs(p - ptilde)/sqrt(ptilde * (1 - ptilde)/(n + 4)) < qnorm(1 -
        alpha/2))/Nreps
}
```

```
probs <- seq(0, 1, 0.01)
par(mfrow = c(2, 3))
for (alpha in c(0.05, 0.01)) {
    for (sample.size in c(5, 10, 20)) {
        plot(probs, sapply(probs, Wald.cover1, sample.size, 10000, alpha),
            type = "l", lty = 4, ylim = c(0.7, 1), xlab = "Probability",
            ylab = "Coverage", main = paste("n = ", sample.size, ", alpha = ",
                alpha))
        abline(h = 1 - alpha, lty = 3)
        lines(probs, sapply(probs, Wilson.cover1, sample.size, 10000,
            alpha))
    }
}
legend("bottom", lty = c(4, 1), c("Wald", "Adjusted"))
```

```
Wald.cover2 <- function(p1, p2 = 0.3, n1 = 20, n2 = 20, nsim = 5000,
    alpha = 0.05) {
    phat1 <- rbinom(nsim, n1, p1)/n1
    phat2 <- rbinom(nsim, n2, p2)/n2
    sum(abs(phat1 - phat2 - (p1 - p2))/sqrt(phat1 * (1 - phat1)/n1 +
        phat2 * (1 - phat2)/n2) < qnorm(1 - alpha/2))/nsim
}
Wilson.cover2 <- function(p1, p2 = 0.3, n1 = 20, n2 = 20, nsim = 5000,
    alpha = 0.05) {
    ptilde1 <- (1 + rbinom(nsim, n1, p1))/(n1 + 2)
    ptilde2 <- (1 + rbinom(nsim, n2, p2))/(n2 + 2)
    sum(abs(ptilde1 - ptilde2 - (p1 - p2))/sqrt(ptilde1 * (1 - ptilde1)/(n1 +
        2) + ptilde2 * (1 - ptilde2)/(n2 + 2)) < qnorm(1 - alpha/2))/nsim
}
```

```
prob1 <- seq(0, 1, 0.005)
n1 <- n2 <- 20
prob2 <- c(0.1, 0.3, 0.5)
```

```r
par(mfrow = c(1, 3))
for (p2 in prob2) {
    plot(prob1, sapply(prob1, Wald.cover2, p2), type = "l", lty = 4,
        ylim = c(0.8, 1), xlab = "Probability p1", ylab = "Observed Coverage Proportion",
        main = "Nominal 95% confidence intervals")
    abline(h = 0.95, lty = 3)
    lines(prob1, sapply(prob1, Wilson.cover2, p2))
    text(x = 0.35, y = 0.85, bquote(p[2] == .(p2)), cex = 1.2)
    text(x = 0.75, y = 0.85, expression(n[i] == 20), cex = 1.2)
    legend("bottom", lty = c(4, 1), c("Wald", "Adjusted"), bty = "n")
}
```