Carson Slater

6376 Homework 14 Solutions

```r
source(here::here("helpers.R"))
```

Question 1

I have read the assigned reading, Rizzo's *Statistical Computing with R (2e)* Chapter 7 on Monte Carlo Methods in Inference.

Question 2

```r
nhanes <- NHANES |> janitor::clean_names()
```

(a)

```r
# get quantiles
nhanes |> select(age, height) |> summary()
```

```
      age            height     
 Min.   : 0.00   Min.   : 83.6  
 1st Qu.:17.00   1st Qu.:156.8  
 Median :36.00   Median :166.0  
 Mean   :36.74   Mean   :161.9  
 3rd Qu.:54.00   3rd Qu.:174.5  
 Max.   :80.00   Max.   :200.4  
                 NA's   :353    
```

```r
# count data
nhanes |>
  mutate(age_binned = case_when(
    age <= 17 ~ "child",
    age > 17 & age <= 36 ~ "youngadult",
    age > 36 & age <= 54 ~ "adult",
    age >= 54 ~ "elder"
  ), height_binned = case_when(
```

```r
    height <= 156.8 ~ "short",
    height > 156.8 & height <= 166 ~ "moderate",
    height > 166 & height <= 174.5 ~ "aboveaverage",
    height >= 174.5 ~ "tall"
  )) |> count(age_binned, height_binned) |>
  pivot_wider(id_cols = age_binned, names_from = height_binned, values_from = n)
```

```
# A tibble: 4 x 6
  age_binned aboveaverage moderate short  tall  `NA`
  <chr>             <int>    <int> <int> <int> <int>
1 adult               805      661   255   795    13
2 child               235      340  1499   149   296
3 elder               619      760   434   598    30
4 youngadult          741      658   237   861    14
```

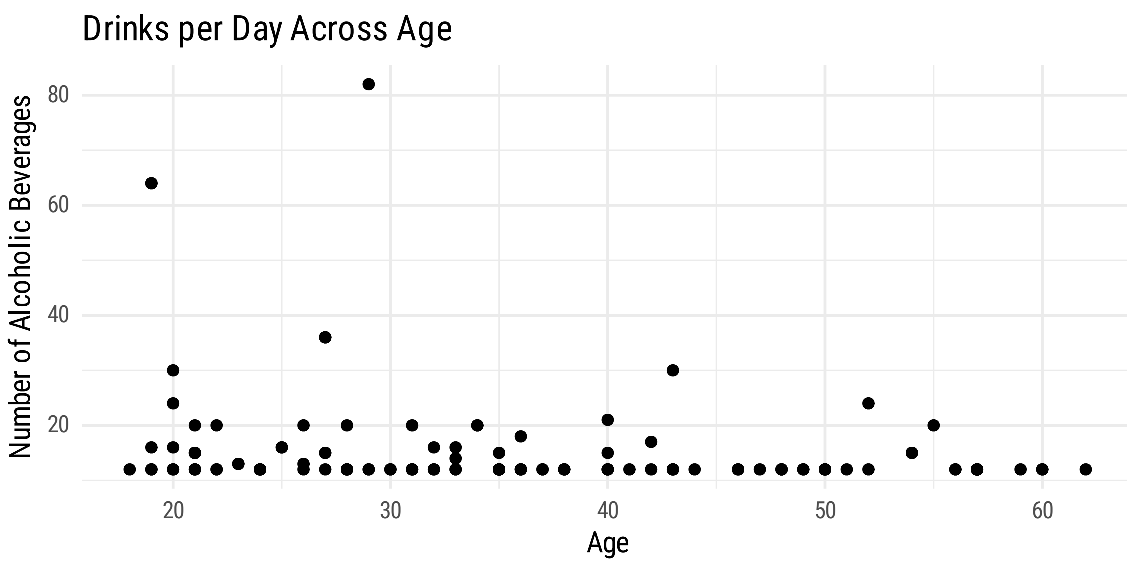
(b)

```r
nhanes |>
  dplyr::filter(age >= 18 & age <= 54) |>
  mutate(age_binned = case_when(
    age <= 17 ~ "child",
    age > 17 & age <= 36 ~ "youngadult",
    age > 36 & age <= 54 ~ "adult",
    age >= 54 ~ "elder"
  ), height_binned = case_when(
    height <= 156.8 ~ "short",
    height > 156.8 & height <= 166 ~ "moderate",
    height > 166 & height <= 174.5 ~ "aboveaverage",
    height >= 174.5 ~ "tall"
  )) |> count(age_binned, height_binned) |>
  pivot_wider(id_cols = age_binned, names_from = height_binned, values_from = n)
```

```
# A tibble: 2 x 6
  age_binned aboveaverage moderate short  tall  `NA`
  <chr>             <int>    <int> <int> <int> <int>
1 adult               805      661   255   795    13
2 youngadult          741      658   237   861    14
```

(c)

```
nhanes |>
  slice_max(alcohol_day, prop = 0.01) |>
  ggplot(aes(age, alcohol_day)) +
  geom_point() +
  labs(title = "Drinks per Day Across Age",
       y = "Number of Alcoholic Beverages",
       x = "Age")
```



(d)

```
nhanes |>
  group_by(gender) |>
  slice_max(alcohol_day, n = 5) |>
  ungroup()
```

```
# A tibble: 11 x 76
      id survey_yr gender   age age_decade age_months race1     race3    education
   <int> <fct>     <fct>  <int> <fct>           <int> <fct>     <fct>    <fct>
 1 57633 2009_10   female    27 " 20-29"          327 White     <NA>     High Sch~
 2 57633 2009_10   female    27 " 20-29"          327 White     <NA>     High Sch~
 3 57633 2009_10   female    27 " 20-29"          327 White     <NA>     High Sch~
```

```
 4 71366 2011_12   female    20 " 20-29"           NA White    White   Some Col~
 5 69389 2011_12   female    22 " 20-29"           NA Hispanic Hispan~ High Sch~
 6 62632 2011_12   male      29 " 20-29"           NA Other    Asian   9 - 11th~
 7 63150 2011_12   male      19 " 10-19"           NA White    White   <NA>
 8 63150 2011_12   male      19 " 10-19"           NA White    White   <NA>
 9 63734 2011_12   male      43 " 40-49"           NA Other    Other   Some Col~
10 58645 2009_10   male      52 " 50-59"          629 Mexican  <NA>    8th Grade
11 65619 2011_12   male      20 " 20-29"           NA Mexican  Mexican High Sch~
# i 67 more variables: marital_status <fct>, hh_income <fct>,
#   hh_income_mid <int>, poverty <dbl>, home_rooms <int>, home_own <fct>,
#   work <fct>, weight <dbl>, length <dbl>, head_circ <dbl>, height <dbl>,
#   bmi <dbl>, bmi_cat_under20yrs <fct>, bmi_who <fct>, pulse <int>,
#   bp_sys_ave <int>, bp_dia_ave <int>, bp_sys1 <int>, bp_dia1 <int>,
#   bp_sys2 <int>, bp_dia2 <int>, bp_sys3 <int>, bp_dia3 <int>,
#   testosterone <dbl>, direct_chol <dbl>, tot_chol <dbl>, ...
```

(e)

```
nhanes |>
  slice_max(alcohol_day, n = 5, by = gender)
```

```
# A tibble: 11 x 76
      id survey_yr gender   age age_decade age_months race1    race3   education
   <int> <fct>     <fct>  <int> <fct>           <int> <fct>    <fct>   <fct>
 1 62632 2011_12   male      29 " 20-29"           NA Other    Asian   9 - 11th~
 2 63150 2011_12   male      19 " 10-19"           NA White    White   <NA>
 3 63150 2011_12   male      19 " 10-19"           NA White    White   <NA>
 4 63734 2011_12   male      43 " 40-49"           NA Other    Other   Some Col~
 5 58645 2009_10   male      52 " 50-59"          629 Mexican  <NA>    8th Grade
 6 65619 2011_12   male      20 " 20-29"           NA Mexican  Mexican High Sch~
 7 57633 2009_10   female    27 " 20-29"          327 White    <NA>    High Sch~
 8 57633 2009_10   female    27 " 20-29"          327 White    <NA>    High Sch~
 9 57633 2009_10   female    27 " 20-29"          327 White    <NA>    High Sch~
10 71366 2011_12   female    20 " 20-29"           NA White    White   Some Col~
11 69389 2011_12   female    22 " 20-29"           NA Hispanic Hispan~ High Sch~
# i 67 more variables: marital_status <fct>, hh_income <fct>,
#   hh_income_mid <int>, poverty <dbl>, home_rooms <int>, home_own <fct>,
#   work <fct>, weight <dbl>, length <dbl>, head_circ <dbl>, height <dbl>,
#   bmi <dbl>, bmi_cat_under20yrs <fct>, bmi_who <fct>, pulse <int>,
#   bp_sys_ave <int>, bp_dia_ave <int>, bp_sys1 <int>, bp_dia1 <int>,
```

```
#   bp_sys2 <int>, bp_dia2 <int>, bp_sys3 <int>, bp_dia3 <int>,
#   testosterone <dbl>, direct_chol <dbl>, tot_chol <dbl>, ...
```

(f)

No response was necessary.

(g)

```r
nhanes |> select(bmi, age, height, weight)
```

```
# A tibble: 10,000 x 4
     bmi   age height weight
   <dbl> <int>  <dbl>  <dbl>
 1  32.2    34   165.   87.4
 2  32.2    34   165.   87.4
 3  32.2    34   165.   87.4
 4  15.3     4   105.   17
 5  30.6    49   168.   86.7
 6  16.8     9   133.   29.8
 7  20.6     8   131.   35.2
 8  27.2    45   167.   75.7
 9  27.2    45   167.   75.7
10  27.2    45   167.   75.7
# i 9,990 more rows
```

(h)

No response was necessary.

(i)

```r
nhanes |>
  mutate(bp_sys_avg = (bp_sys1 + bp_sys2 + bp_sys3) / 3,
         bp_dia_avg = (bp_dia1 + bp_dia2 + bp_dia3) / 3) |>
  count(bp_sys = near(bp_sys_avg, bp_sys_ave),
        bp_dia = near(bp_dia_avg, bp_dia_ave))
```

```
# A tibble: 5 x 3
  bp_sys bp_dia     n
  <lgl>  <lgl>  <int>
1 FALSE  FALSE   6578
2 FALSE  TRUE     706
3 TRUE   FALSE    628
4 TRUE   TRUE      59
5 NA     NA      2029
```

(j)

No response was necessary.

(k)

No response was necessary.

(m)

```r
nhanes |>
  summarise(avg_height = mean(height, na.rm = TRUE), .by = c(gender, age_decade))
```

```
# A tibble: 18 x 3
   gender age_decade avg_height
   <fct>  <fct>           <dbl>
 1 male   " 30-39"         177.
 2 male   " 0-9"           116.
 3 female " 40-49"         163.
 4 male   " 60-69"         175.
 5 male   " 50-59"         176.
 6 female " 10-19"         158.
 7 female " 50-59"         162.
 8 female " 0-9"           116.
 9 male   " 10-19"         167.
10 male   " 40-49"         176.
11 female " 60-69"         161.
12 female " 20-29"         163.
13 male   " 20-29"         176.
14 female " 30-39"         164.
```

```
15 female " 70+"           159.
16 female  <NA>            156.
17 male    <NA>            172.
18 male   " 70+"           173.
```

(n)

No response was necessary.

Question 3

(a)

If $X_1, \ldots, X_n \sim Bernoulli(p)$, then the frequentist estimator for p given $\mathbf{x}$ is $\hat{p} = \frac{\sum_{i=1}^{n} x_i}{n}$.

(b)

Let $Y = \sum_{i=1}^{n} X_i$. We know that $Y \sim \text{Binomial}(n, p)$. Since $\hat{p} = \frac{Y}{n}$, the exact distribution of $\hat{p}$ is a binomial random variable scaled by the n parameter (can be shown with moment-generating functions).

(c)

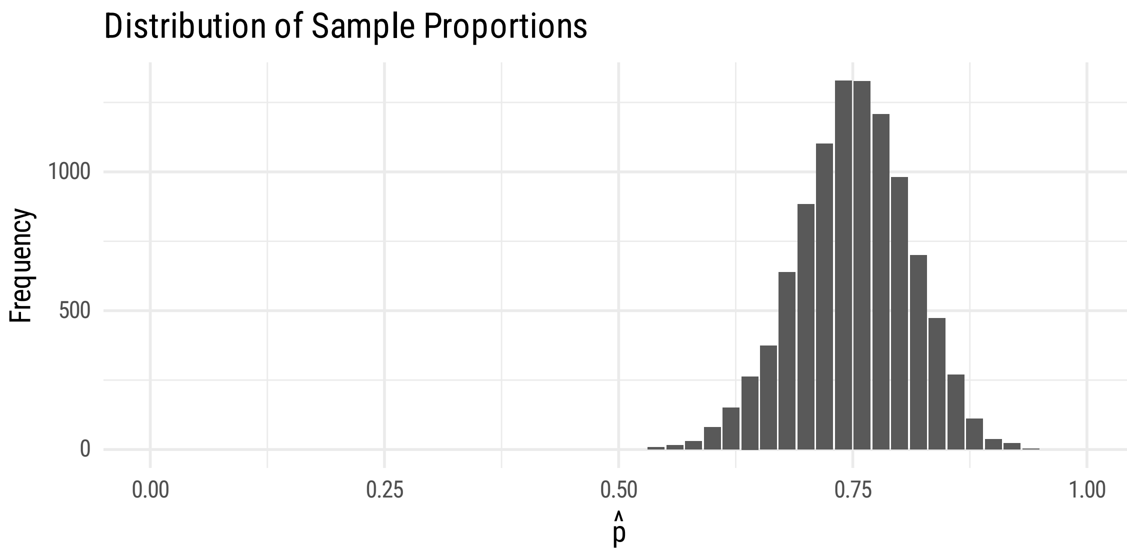
```r
rbern <- function(n, p) +(runif(n) <= p)
```

```r
set.seed(1)
rbern(10, 0.9)
```

```
 [1] 1 1 1 0 1 1 0 1 1 1
```

(d)

```r
phats <- replicate(1e4, mean(rbern(50, 0.75)), simplify = TRUE)

phats |>
  tibble() |>
  ggplot(aes(phats)) +
    geom_bar() +
    labs(x = expression(hat(p)), y = "Frequency") +
    ggtitle("Distribution of Sample Proportions") +
    xlim(c(0,1))
```



(e)

```r
g <- function(x) sapply(x, \(x) ifelse(x <= 0 || x >= 1, NaN, x/(1-x)))
```

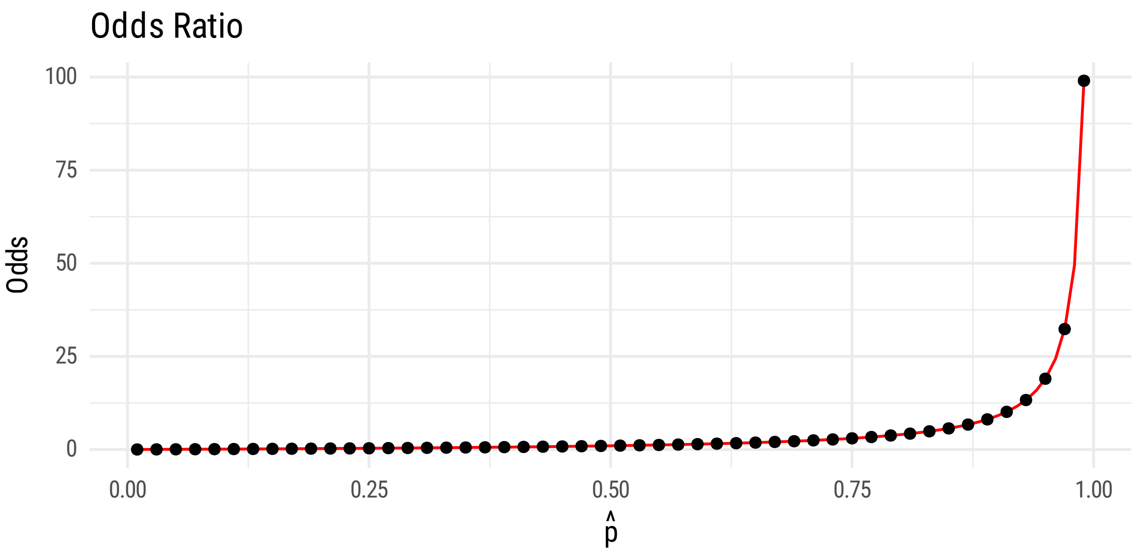
```r
g(c(-.1, 0, .25, .50, .75, 1, 1.1))
```

```
[1]       NaN       NaN 0.3333333 1.0000000 3.0000000       NaN       NaN
```

(f)

The support of the random variable $g(\hat{p})$ is not quite $\mathbb{R}_{(0,\infty)}$, as for small values of n, there finite values $g(\hat{p})$ can take on, but as $n \to \infty$, then the support of $g(\hat{p})$ goes to $\mathbb{R}_{(0,\infty)}$.

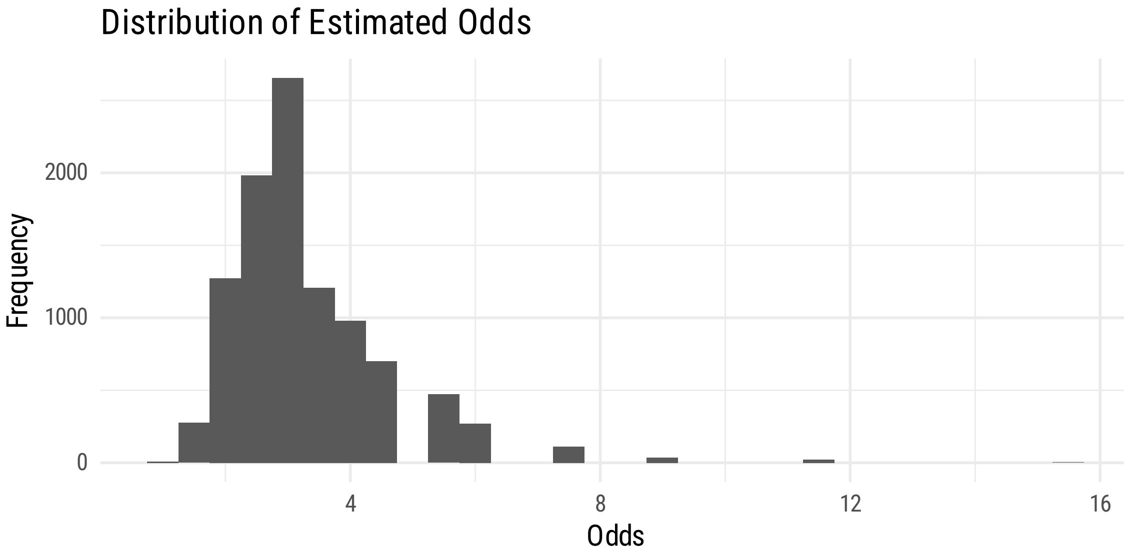
```r
tibble(phat = seq(0.01, 0.99, length.out = 50),
       g = g(phat)) |>
  ggplot(aes(x = phat, y = g)) +
    geom_function(fun = g, color = "red") +
    geom_point() +
    labs(title = "Odds Ratio",
         x = expression(hat(p)),
         y = "Odds")
```



(g)

```r
or_estim <- phats / (1 - phats)

or_estim |>
  tibble() |>
  ggplot(aes(or_estim)) +
    geom_histogram() +
    labs(x = "Odds",
         y = "Frequency",
         title = "Distribution of Estimated Odds")
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



(h)

According to the central limit theorem, $\hat{p} \sim \mathcal{N}(p, \frac{p(1-p)}{n})$.

(i)

If $X_1, \ldots, X_n \sim \text{Bernoulli}(p)$, then an estimator for p given $\mathbf{x}$ is $\hat{p}_n = \frac{\sum_{i=1}^n x_i}{n}$. Define $g(\hat{p}_n) = \frac{\hat{p}_n}{1-\hat{p}_n}$. We know that

$$\sqrt{n}\{\hat{p}_n - p\} \xrightarrow{\mathcal{D}} \mathcal{N}(0, p(1-p)).$$

By the first order delta method, we have that

$$\sqrt{n}\{g(\hat{p}_n) - g(p)\} \xrightarrow{\mathcal{D}} \mathcal{N}(0, p(1-p)\{g'(p)\}^2).$$

Since $g'(p)^2 = \frac{1}{(1-p)^4}$,

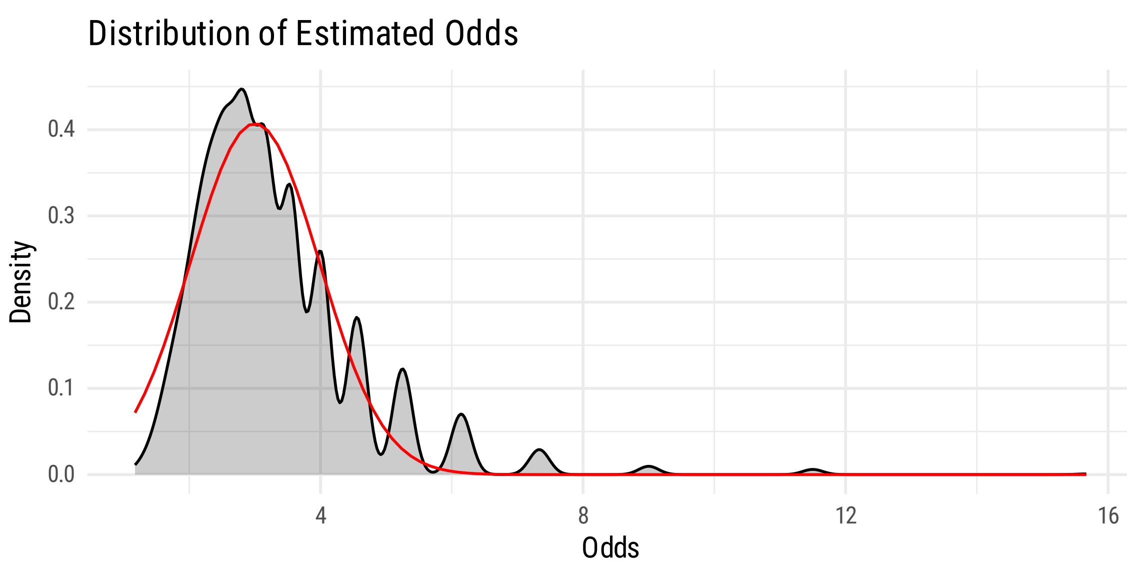
$$\sqrt{n}\{g(\hat{p}_n) - g(p)\} \xrightarrow{\mathcal{D}} \mathcal{N}(0, \frac{p}{(1-p)^3}).$$

So by the first order delta method, we have that

$$g(\hat{p}) \approx \mathcal{N}\left(\frac{p}{1-p}, \frac{p}{n(1-p)^3}\right).$$

(j)

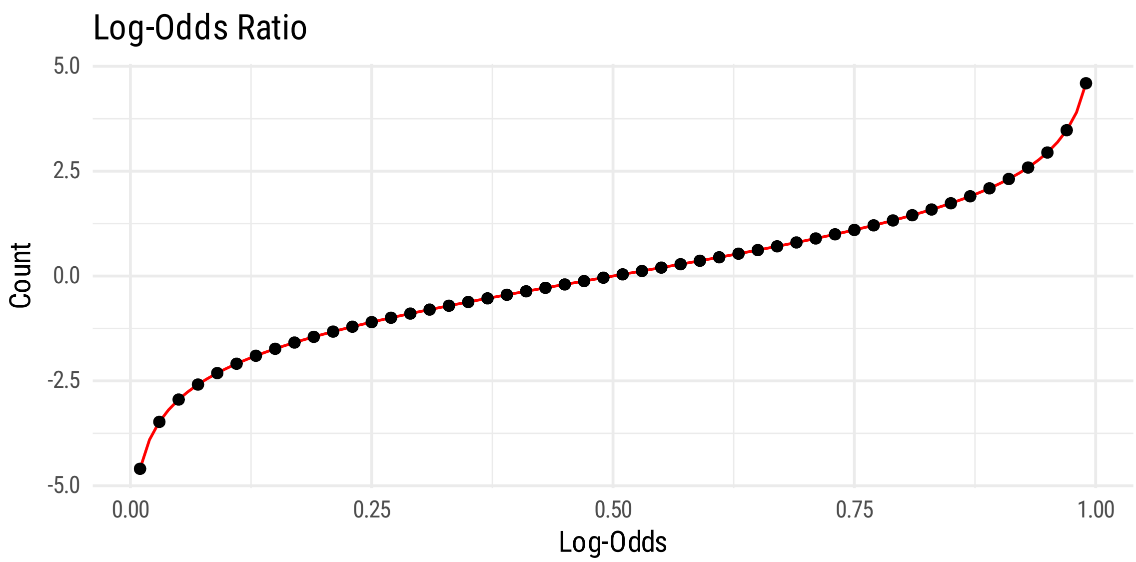
```r
p <- 0.75
n <- 50
odds <- p / (1 - p)
delta_var <- p / (n*(1-p)^3)

or_estim |>
  tibble() |>
  ggplot(aes(or_estim)) +
    geom_density(fill = "black", alpha = 0.2) +
    geom_function(fun = \(x) dnorm(x, mean = odds, sd = sqrt(delta_var)), color = "red") +
    labs(x = "Odds",
         y = "Density",
         title = "Distribution of Estimated Odds")
```



(k)

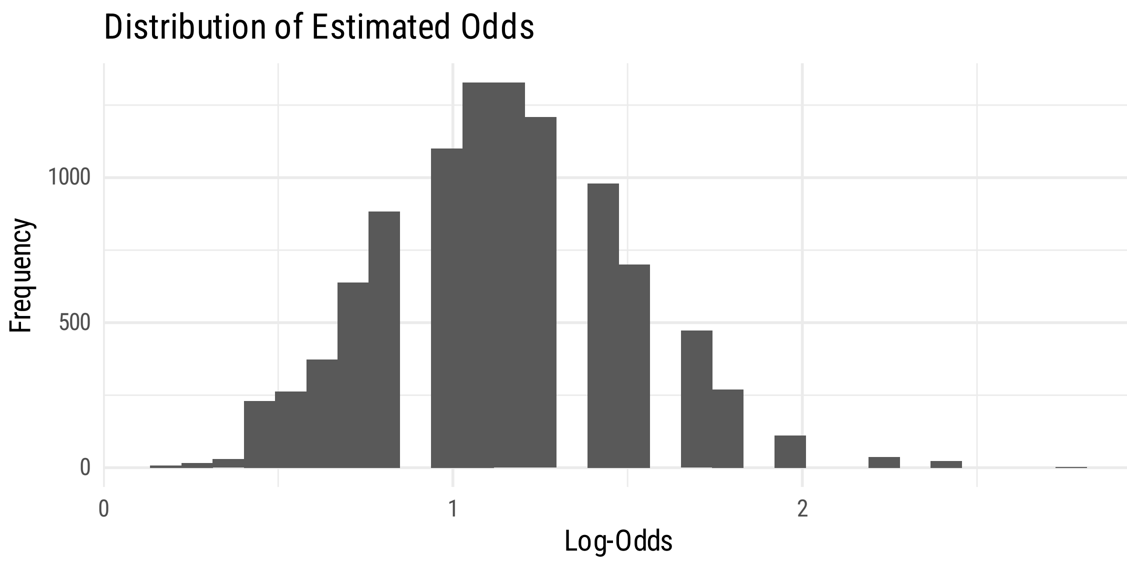
```r
h <- function(x) g(x) |> log()

tibble(phat = seq(0.01, 0.99, length.out = 50),
       h = h(phat)) |>
  ggplot(aes(x = phat, y = h)) +
```

```
    geom_function(fun = h, color = "red") +
    geom_point() +
    labs(title = "Log-Odds Ratio",
         x = "Log-Odds",
         y = "Count")
```



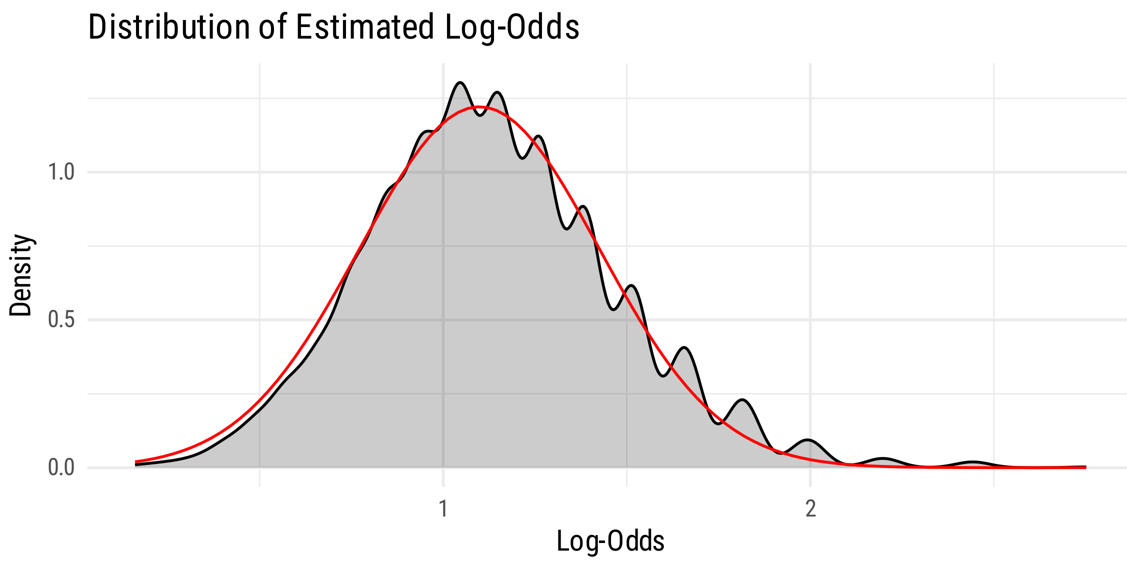
(l)

```
tibble(log_odds = log(or_estim)) |>
  ggplot(aes(log_odds)) +
    geom_histogram() +
    labs(x = "Log-Odds",
         y = "Frequency",
         title = "Distribution of Estimated Odds")
```



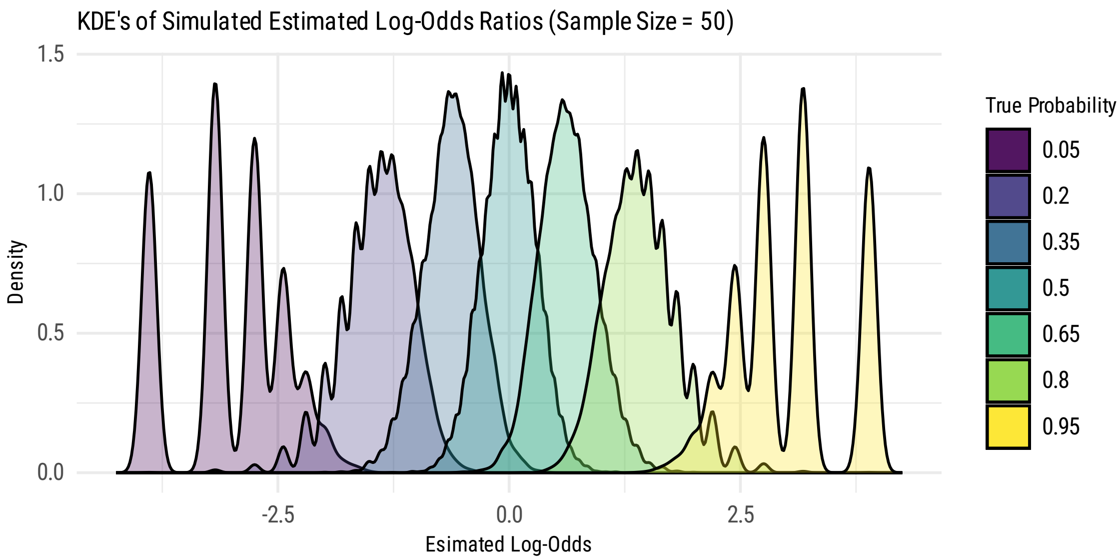
(m)

```
log_odds <- log(p / (1 - p))
log_odds_delta_var <- 1/(p * (1 - p)*n)

tibble(log_odds = log(or_estim)) |>
  ggplot(aes(log_odds)) +
    geom_density(fill = "black", alpha = 0.2) +
    geom_function(fun = \(x) dnorm(x, mean = log_odds, sd = sqrt(log_odds_delta_var)), color
    labs(x = "Log-Odds",
         y = "Density",
         title = "Distribution of Estimated Log-Odds")
```



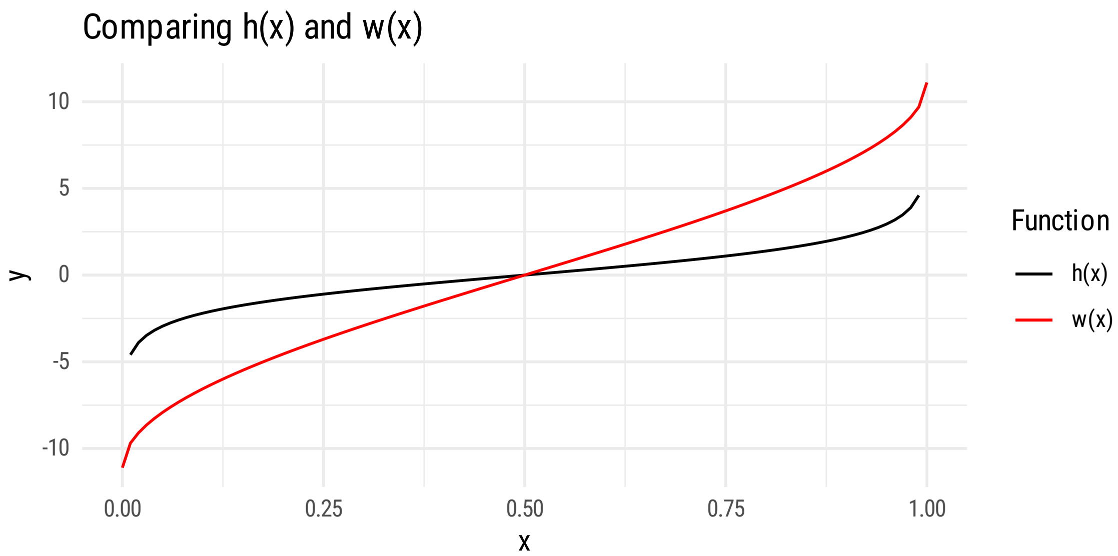
(n)

```
true_p <- seq(0.05, 0.95, 0.15)

hxhats <- lapply(true_p, \(x) h(replicate(1e4, mean(rbern(50, x)), simplify = TRUE))) |>
  as.data.frame()

colnames(hxhats) <- sapply(1:7, \(x) paste0("p", x))

cols <- viridis(7)

hxhats |> ggplot() +
  geom_density(aes(x = p1, fill = "p1"), alpha = 0.3) +
  geom_density(aes(x = p2, fill = "p2"), alpha = 0.3) +
  geom_density(aes(x = p3, fill = "p3"), alpha = 0.3) +
  geom_density(aes(x = p4, fill = "p4"), alpha = 0.3) +
  geom_density(aes(x = p5, fill = "p5"), alpha = 0.3) +
  geom_density(aes(x = p6, fill = "p6"), alpha = 0.3) +
  geom_density(aes(x = p7, fill = "p7"), alpha = 0.3) +
  labs(x = "Esimated Log-Odds", y = "Density",
       title = "KDE's of Simulated Estimated Log-Odds Ratios (Sample Size = 50)") +
  xlim(c(-4.25, 4.25)) +
  scale_fill_manual(values = cols, labels = true_p, name = "True Probability") +
  guides(fill = guide_legend(title = "True Probability")) +
  theme(title = element_text(size = 8))
```



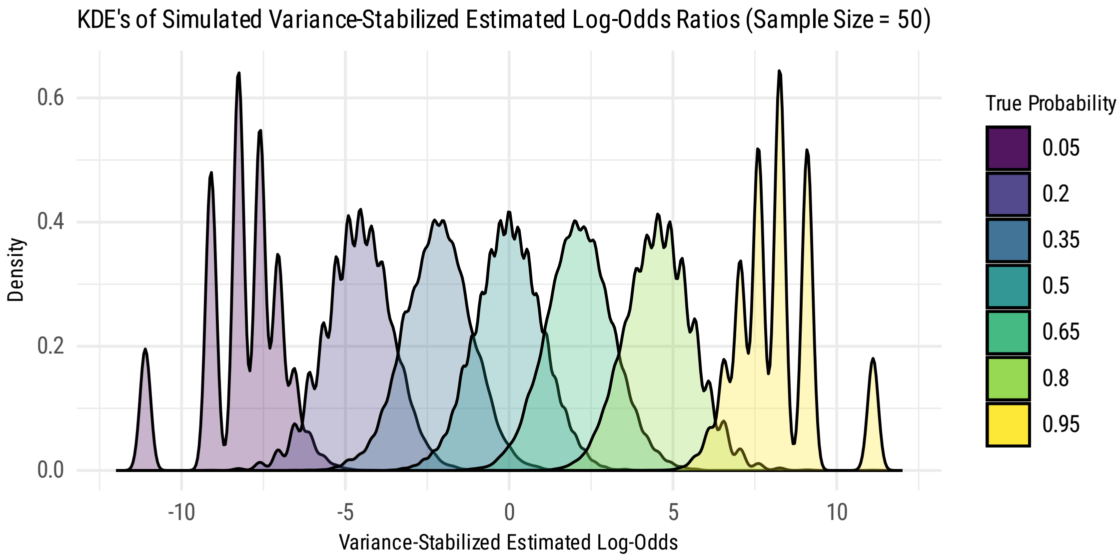
(o)

```
w <- function(x) (sqrt(n)/2)*(pi - 4*acos(sqrt(x)))

ggplot() +
  geom_function(fun = h, aes(color = "h(x)")) +
  geom_function(fun = w, aes(color = "w(x)")) +
  labs(title = "Comparing h(x) and w(x)",
       x = "x",
       y = "y",
       color = "Function") +
  scale_color_manual(values = c("black", "red"),
                     labels = c("h(x)", "w(x)")) +
  guides(color = guide_legend(title = "Function"))
```



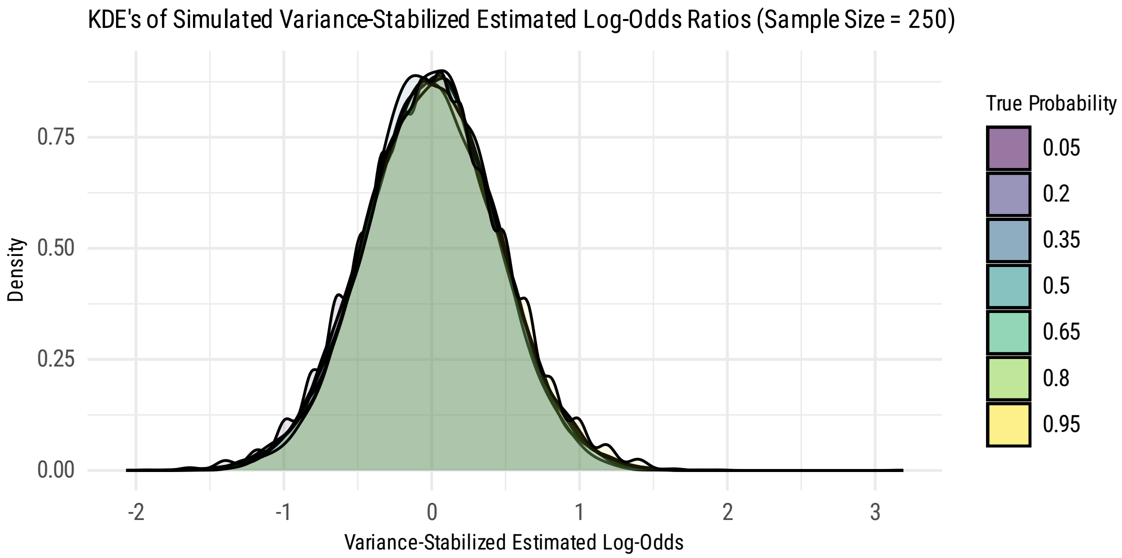
(p)

```
wxhats <- lapply(true_p, \(x) w(replicate(1e4, mean(rbern(50, x)), simplify = TRUE))) |>
  as.data.frame()

colnames(wxhats) <- sapply(1:7, \(x) paste0("p", x))

wxhats |> ggplot() +
  geom_density(aes(x = p1, fill = "p1"), alpha = 0.3) +
  geom_density(aes(x = p2, fill = "p2"), alpha = 0.3) +
  geom_density(aes(x = p3, fill = "p3"), alpha = 0.3) +
  geom_density(aes(x = p4, fill = "p4"), alpha = 0.3) +
  geom_density(aes(x = p5, fill = "p5"), alpha = 0.3) +
  geom_density(aes(x = p6, fill = "p6"), alpha = 0.3) +
  geom_density(aes(x = p7, fill = "p7"), alpha = 0.3) +
  labs(x = "Variance-Stabilized Estimated Log-Odds", y = "Density",
       title = "KDE's of Simulated Variance-Stabilized Estimated Log-Odds Ratios (Sample Siz
  xlim(c(-12, 12)) +
  scale_fill_manual(values = cols, labels = true_p, name = "True Probability") +
  guides(fill = guide_legend(title = "True Probability")) +
  theme(title = element_text(size = 8))
```

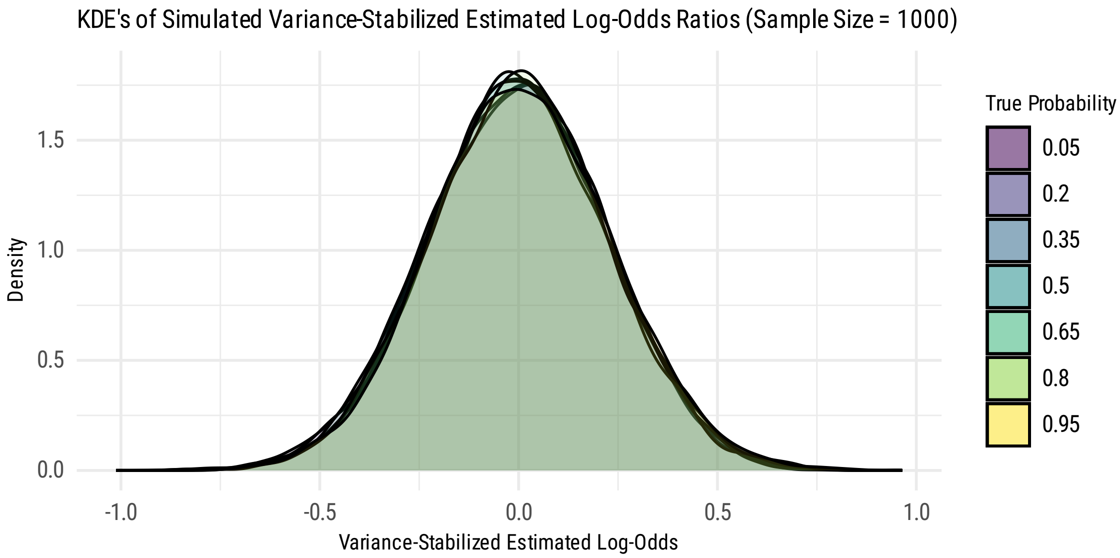


(q)

```
# for n = 250
wxhats_centered250 <- lapply(true_p, \(x) w(replicate(1e4, mean(rbern(250, x)), simplify = TI

colnames(wxhats_centered250) <- sapply(1:7, \(x) paste0("p", x))

wxhats_centered250 |> ggplot() +
  geom_density(aes(x = p1, fill = "p1"), alpha = 0.1) +
  geom_density(aes(x = p2, fill = "p2"), alpha = 0.1) +
  geom_density(aes(x = p3, fill = "p3"), alpha = 0.1) +
  geom_density(aes(x = p4, fill = "p4"), alpha = 0.1) +
  geom_density(aes(x = p5, fill = "p5"), alpha = 0.1) +
  geom_density(aes(x = p6, fill = "p6"), alpha = 0.1) +
  geom_density(aes(x = p7, fill = "p7"), alpha = 0.1) +
  labs(x = "Variance-Stabilized Estimated Log-Odds", y = "Density",
       title = "KDE's of Simulated Variance-Stabilized Estimated Log-Odds Ratios (Sample Size
  scale_fill_manual(values = cols, labels = true_p, name = "True Probability") +
  guides(fill = guide_legend(title = "True Probability")) +
  theme(title = element_text(size = 8))
```




```
# for n = 1000
wxhats_centered1k <- lapply(true_p, \(x) w(replicate(1e4, mean(rbern(1000, x)), simplify = TR

colnames(wxhats_centered1k) <- sapply(1:7, \(x) paste0("p", x))

wxhats_centered1k |> ggplot() +
  geom_density(aes(x = p1, fill = "p1"), alpha = 0.1) +
  geom_density(aes(x = p2, fill = "p2"), alpha = 0.1) +
  geom_density(aes(x = p3, fill = "p3"), alpha = 0.1) +
  geom_density(aes(x = p4, fill = "p4"), alpha = 0.1) +
  geom_density(aes(x = p5, fill = "p5"), alpha = 0.1) +
  geom_density(aes(x = p6, fill = "p6"), alpha = 0.1) +
  geom_density(aes(x = p7, fill = "p7"), alpha = 0.1) +
  labs(x = "Variance-Stabilized Estimated Log-Odds", y = "Density",
       title = "KDE's of Simulated Variance-Stabilized Estimated Log-Odds Ratios (Sample Size
  scale_fill_manual(values = cols, labels = true_p, name = "True Probability") +
  guides(fill = guide_legend(title = "True Probability")) +
  theme(title = element_text(size = 8))
```



Question 4

(a)

```
z_int <- function(x, conf = 0.95) {
  if(all(x %in% c(0, 1))) {
    if (conf < 0 | conf > 1) stop("`conf` must be a value between 0 and 1.")
    alpha <- 1 - conf
    phat <- mean(x)
    int <- qnorm(conf + alpha/2)*sqrt(phat*(1 - phat)/n)
    c(
      "lower" = phat - int,
      "upper" = phat + int
    )
  } else stop("Input vector must contain only 0s and 1s.")
}
```

```
set.seed(1)
n <- 50
(x <- rbinom(n, 1, .75))
```

```
 [1] 1 1 1 0 1 0 0 1 1 1 1 1 1 1 0 1 1 0 1 0 0 1 1 1 1 1 1 1 0 1 1 1 1 1 0 1 0 1
[39] 1 1 0 1 0 1 1 0 1 1 1 1
```

```r
z_int(x)
```

```
   lower    upper 
0.618419 0.861581 
```

(b)

```r
`%is_in%` <- function(x, int) {
  stopifnot(length(int) == 2)
  if (x >= int[1] & x <= int[2]) TRUE
  else FALSE
}
```

```r
1 %is_in% c(0, 2)
```

```
[1] TRUE
```

```r
1 %is_in% -1:0
```

```
[1] FALSE
```

(c)

```r
estimate_coverage <- function(p, n = 50, iter = 1e4, ci_fun = z_int) {
  samps <- future_map(1:iter, ~ rbinom(n, 1, p), .options = furrr_options(seed = TRUE))
  future_map(samps, \(samp) p %is_in% ci_fun(samp)) |> unlist() |> mean()
}
```

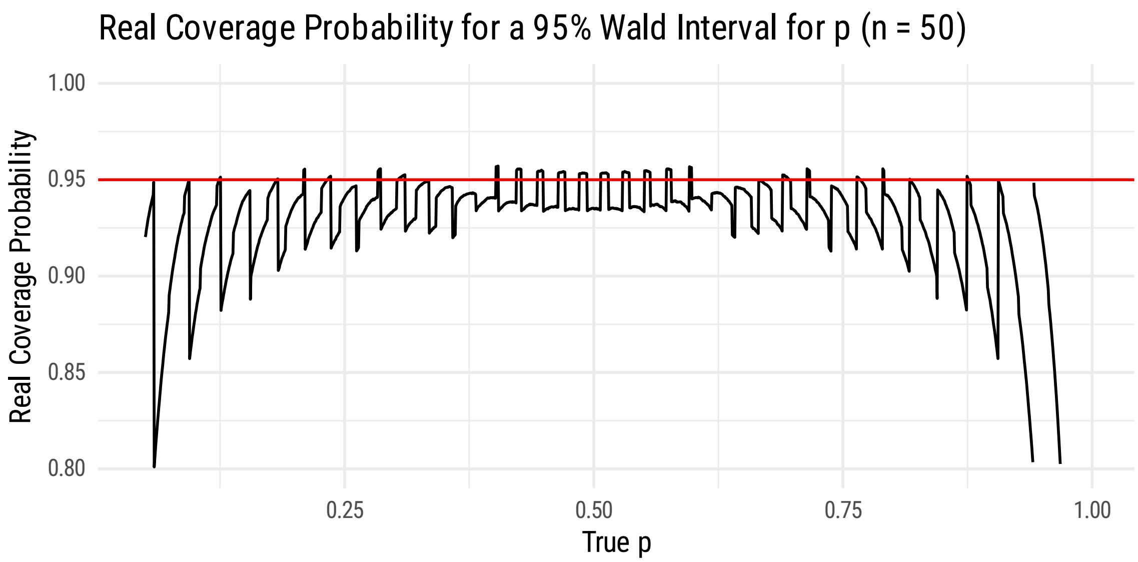
```r
estimate_coverage(0.8, 50, iter = 1e6)
```

```
[1] 0.93731
```

(d)

```r
cov_probs <- tibble(
  "p"        = seq(0.05, 0.995, by = 0.0005),
  "real_cov" = future_map(p, \(p) estimate_coverage(p, 50, iter = 1e6)) |> unlist()
  )
```

```r
cov_probs |>
  ggplot(aes(x = p, y = real_cov)) +
  geom_line() +
  geom_hline(yintercept = 0.95, color = "red") +
  labs(title = "Real Coverage Probability for a 95% Wald Interval for p (n = 50)",
       x = "True p",
       y = "Real Coverage Probability") +
  ylim(c(0.8, 1))
```



(e)

```r
agresti_coull_int <- function(x, conf = 0.95) {
  if(all(x %in% c(0, 1))) {
    if (conf < 0 | conf > 1) stop("`conf` must be a value between 0 and 1.")
    alpha <- 1 - conf
    kappa <- qnorm(conf + alpha/2)
```

```
    n_tilde <- n + kappa^2
    p_tilde <- (sum(x) + kappa^2/2) / n_tilde
    int <- kappa*sqrt(p_tilde*(1 - p_tilde)/n_tilde)
    c(
      "lower" = p_tilde - int,
      "upper" = p_tilde + int
    )
  } else stop("Input vector must contain only 0s and 1s.")
}

# testing this interval
agresti_coull_int(x)
```

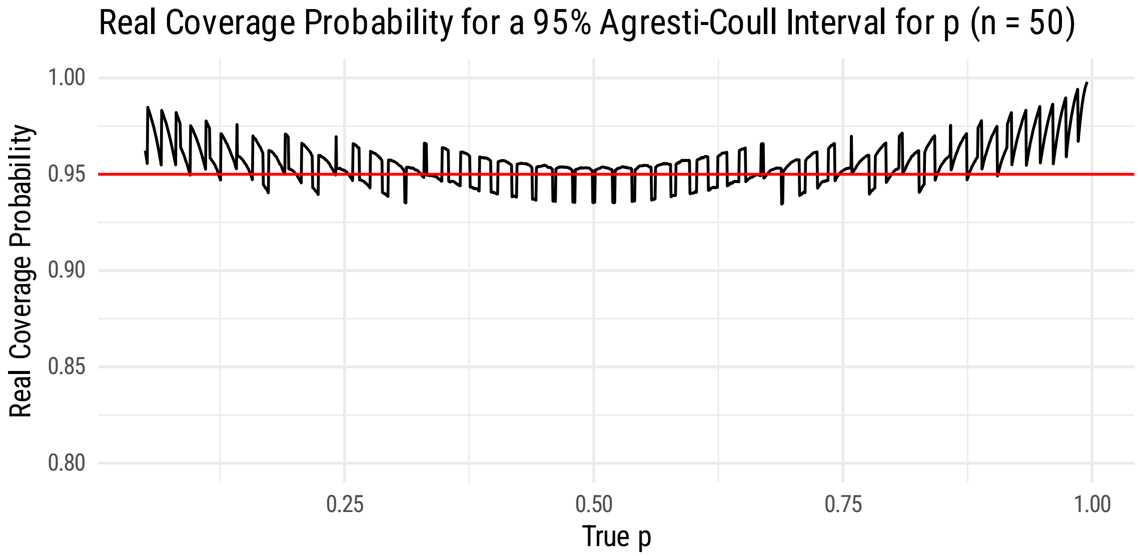
```
    lower     upper
0.6033242 0.8424290
```

```
p %is_in% agresti_coull_int(x)
```

```
[1] TRUE
```

```
cov_probs_ac <- tibble(
  "p"        = seq(0.05, 0.995, by = 0.0005),
  "real_cov" = future_map(p, \(p) estimate_coverage(p, 50, iter = 1e6, ci_fun = agresti_coul
  )
```

```
cov_probs_ac |>
  ggplot(aes(x = p, y = real_cov)) +
  geom_line() +
  geom_hline(yintercept = 0.95, color = "red") +
  labs(title = "Real Coverage Probability for a 95% Agresti-Coull Interval for p (n = 50)",
       x = "True p",
       y = "Real Coverage Probability") +
  ylim(c(0.8, 1))
```


Real Coverage Probability for a 95% Agresti-Coull Interval for p (n = 50)
1.00
0.95
0.90
0.85
0.80
Real Coverage Probability
0.25
0.50
0.75
1.00
True p