Environmental Health Hazard Analysis

2021

Part 1: Quantile estimation

Population quantiles indicate the values above or below which a specified proportion of the population falls. This can be useful for a wide range of applications, for example:

- What level of nitrate (contaminant from fertilizer) is higher than 90% of tapwater in California?
  - *i.e.*, what is the 0.9-quantile for nitrate concentration?
  - Could be used as the basis for a diagnostic test for water quality.
- How many months do 70% of advanced-stage lymphoma patients live?
  - *i.e.*, what is the 0.7-quantile of survival times?
  - Useful information in counseling patients.

Formally, if $F(x)$ is the CDF of the population distribution of interest, the p-quantile is defined as

$$\theta_p = F^{-1}(p)$$

Sample order statistics are the basis for estimating population quantiles. The simplest estimator of θ_p from an *iid* sample $X_1, \ldots, X_n$ is

$$\hat{\theta}_p = X_{(k)} \quad \text{where} \quad k = \text{floor}(np)$$

This can be viewed as the inverse empirical CDF function, since

$$X_{(k)} = \min \left\{ x : \underbrace{\frac{1}{n} \sum_{i=1}^{n} I(X_i \leq x)}_{\hat{F}(x)} \geq p \right\}$$

where $\hat{F}(x)$ is just the proportion of X_i's smaller than or equal to x.

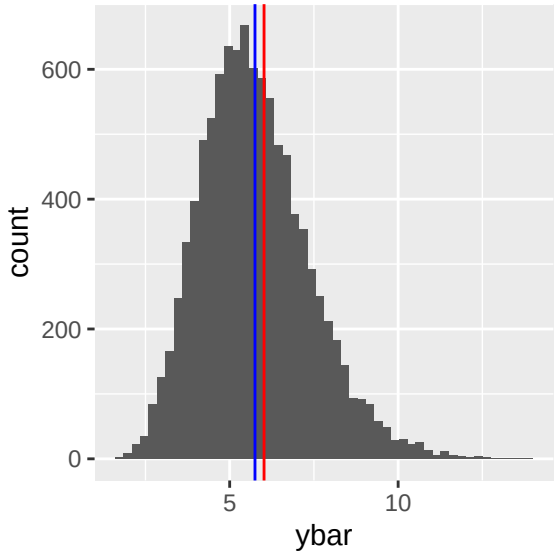
Here sampling distribution of $\hat{\theta}_p$ through simulation will be explored.

Bias

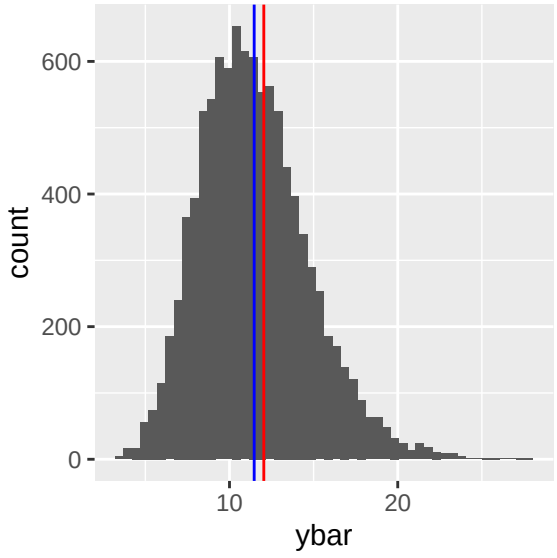
First we'll look at the apparent bias in a few cases (where 'cases' means population distributions). This exploration won't be conclusive, but it should give a rough idea of whether the estimator seems obviously biased.

1. Simulating the sampling distribtuion of $\hat{\theta}_{0.7}$ based on $10,000$ samples of size $n = 20$ from an exponential(5) distribution, and plotting the histogram. We'll find the true value of $\theta_{0.7}$ using `qexp()`, and add a vertical line in red on the plot at that value. Calculate the average value of $\hat{\theta}_{0.7}$ from the $10,000$ simulated values; this is a numerical approximation of $\mathbb{E}\hat{\theta}_{0.7}$, which will be added as a vertical line in blue to the histogram.

```r
# function to calculate the estimator
thetahat <- function(x, p){
  quantile(x, probs = p, type = 1)
}

# function to draw random samples and compute the estimator
sim_fn <- function(n, p, mu){
  y <- rexp(n, rate = 1/mu)
  out <- thetahat(y, p)
  return(out)
}

# set the sample size and number of simulations
n <- 20
nsim <- 10000

# run simulation
x <- sapply(1:nsim, function(i){sim_fn(n, p=0.7, mu=5)})
y <- qexp(0.7, 1/5)
z <- mean(x)

# construct plot
x_data <- data.frame(ybar = x)
plot1 <- ggplot(aes(x = ybar), data = x_data) +
  geom_histogram(bins = 50)
plot1 + geom_vline(xintercept = y, color = "red") + geom_vline(xintercept = z, color = "blue")
```



2. Repeat the previous part for an exponential(10) distribution; increasing the population parameter increases the *heaviness* of the tail, thereby increasing every quantile.



3. Bias of the estimator based on the previous two experiments

The estimator is biased because there is not a difference of zero between the expected value of the estimator and the true value of the parameter being estimated.

Variance

The variance will be explored, and more specifically, how the variance of the quantile estimator $\hat{\theta}_p$ is affected by sample size and the value of p. We'll consider what we would expect to happen as these two values change.

4. Do you think that the variance of $\hat{\theta}_p$ will increase or decrease as n increases? How do we think that the variance of $\hat{\theta}_p$ will depend on p? Are there other factors we expect that might affect the sampling variance of $\hat{\theta}_p$?

A: I think that the variance of $\hat{\theta}_p$ will decrease as n increases. The variance of $\hat{\theta}_p$ will increase as p increases as well. The number of simulations is another factor that might affect the sampling variance of $\hat{\theta}_p$ because the greater the number of simulations, the more accurate and higher the variance is. Another factor is the mu because as the value of mu increases so does the variance.

5. Simulate the sampling distribtuion of $\hat{\theta}_p$ based on $10,000$ samples of size $n = 20$ from an exponential(5) distribution for $p = 0.6, 0.8, 0.9, 0.99$. Plot the histogram for each case. What seems to happen to the variance when estimating higher quantiles?

```
# set the sample size and number of simulations
n <- 20
nsim <- 10000

# first case: p = 0.6
thetahat6_sim <- sapply(1:nsim, function(i){sim_fn(n, p=0.6, mu=5)})
var(thetahat6_sim)
```

```
## [1] 1.712497
```

```
# second case: p = 0.8
thetahat8_sim <- sapply(1:nsim, function(i){sim_fn(n, p=0.8, mu=5)})
var(thetahat8_sim)
```

```
## [1] 4.439745
```

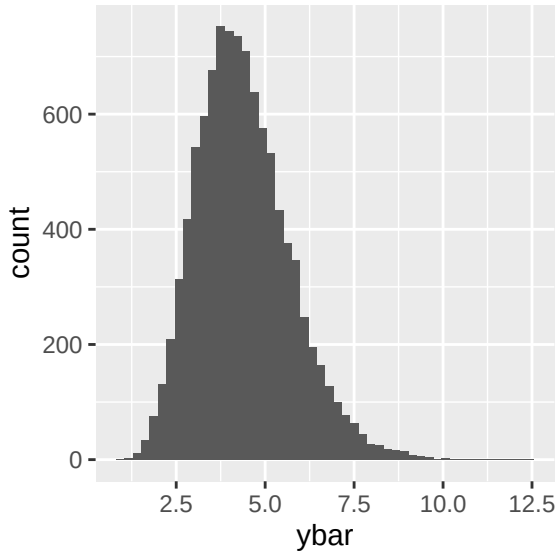
```
# third case: p = 0.9
thetahat9_sim <- sapply(1:nsim, function(i){sim_fn(n, p=0.9, mu=5)})
var(thetahat9_sim)
```

```
## [1] 8.583824
```

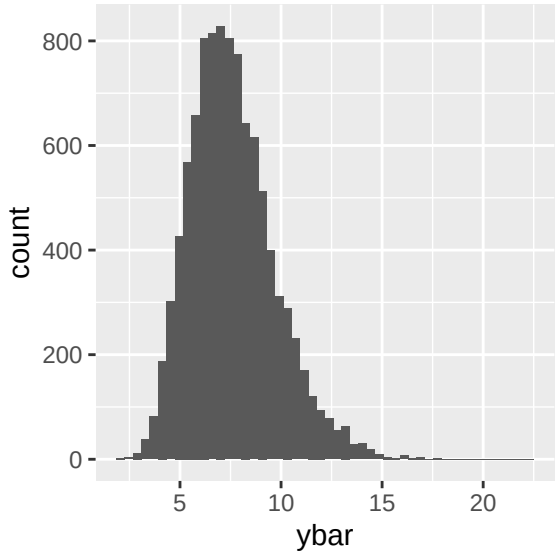
```
# fourth case: p = 0.99
thetahat99_sim <- sapply(1:nsim, function(i){sim_fn(n, p=0.99, mu=5)})
var(thetahat99_sim)
```

```
## [1] 41.13891
```

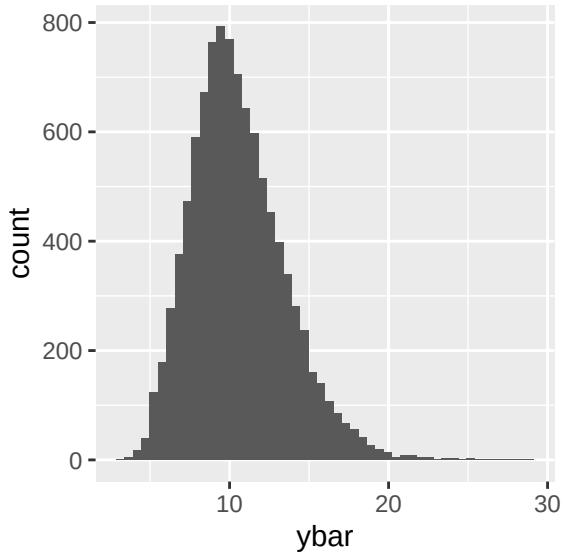
```
# construct plots:
# hist(thetahat6_sim, breaks = 50, freq = F)
x_data <- data.frame(ybar = thetahat6_sim)
ggplot(aes(x = ybar), data = x_data) +
  geom_histogram(bins = 50)
```



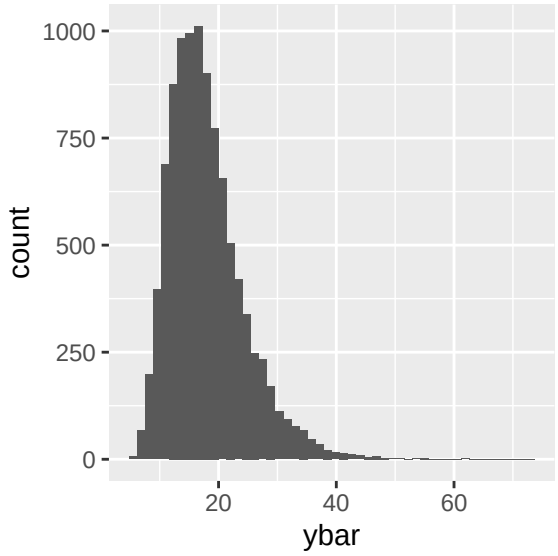

```
# hist(thetahat8_sim, breaks = 50, freq = F)
x_data <- data.frame(ybar = thetahat8_sim)
ggplot(aes(x = ybar), data = x_data) +
  geom_histogram(bins = 50)
```




```
# hist(thetahat9_sim, breaks = 50, freq = F)
x_data <- data.frame(ybar = thetahat9_sim)
ggplot(aes(x = ybar), data = x_data) +
  geom_histogram(bins = 50)
```

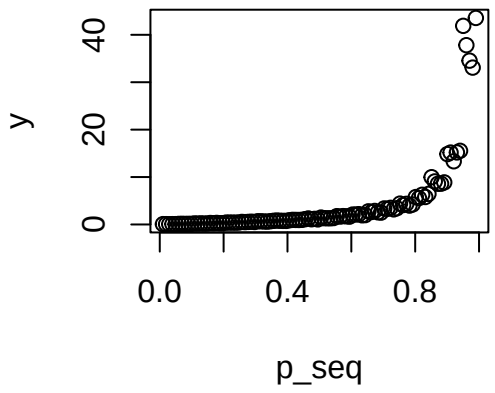



```
# hist(thetahat99_sim, breaks = 50, freq = F)
x_data <- data.frame(ybar = thetahat99_sim)
ggplot(aes(x = ybar), data = x_data) +
  geom_histogram(bins = 50)
```



A: The variance increases when estimating higher quantiles.

6. (*) Adapt your codes from the previous part to write a function that, given p, calculates the variance of 1000 simulated values of $\hat{\theta}_p$ computed from samples of size $n = 20$ from an exponential(5) distribution. Use the function to calculate the variance of $\hat{\theta}_p$ for a fine sequence of p from 0.01 to 0.99 (see `?seq`), and plot the variance against p.



7. Now simulate the sampling distribution of $\hat{\theta}_{0.99}$ based on $10,000$ samples of size $n = 20,000$ from an exponential(5) distribution. Plot the histogram. What is the effect of increasing the sample size?

```
# repeat the fourth case from step 5 with larger n
n <- 20000
nsim <- 10000
```

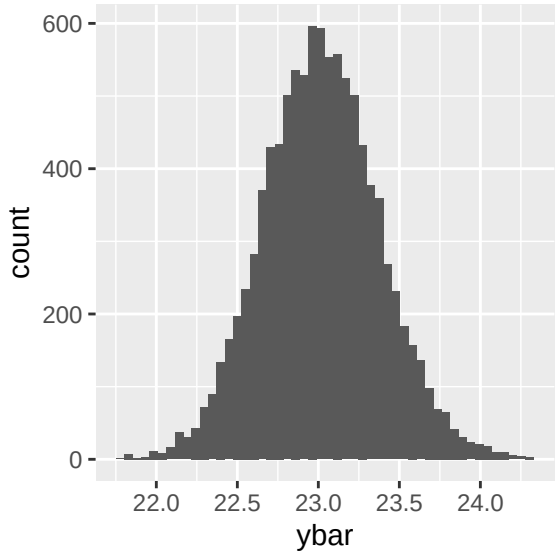
```
thetahat99n_sim <- sapply(1:nsim, function(i){sim_fn(n, p=0.99, mu=5)})
mean(thetahat99n_sim)
```

```
## [1] 23.01896
```

```
var(thetahat99n_sim)
```

```
## [1] 0.1284315
```

```
# plot and compare
#hist(thetahat99n_sim, breaks = 50, freq = F)
x_data <- data.frame(ybar = thetahat99n_sim)
ggplot(aes(x = ybar), data = x_data) +
  geom_histogram(bins = 50)
```



A:The effect on increasing the sample size is that the distribution seems Gaussian and the variance has decreased.

8. Based on the previous parts, when do you think it is possible to estimate a quantile with relatively good accuracy, and when do you think it is more difficult?

A; It is possible to estimate a quantile with relatively good accuracy the larger the number of sample size and number of simulations as the value is more precise; however, when these values are small, they are less precise.

Normal approximation

9. (*) Theoretically, there is a normal approximation for the sampling distribution of $\hat{\theta}_p$, which is

$$\sqrt{n}\left(\hat{\theta}_p - \theta_p\right) \xrightarrow{d} N\left(0, V\right) \quad \text{where} \quad V = \frac{p(1-p)}{\left(f(\theta_p)\right)^2}$$

Choose any settings for n, p and simulate $10,000$ values of

$$\frac{\hat{\theta}_p - \theta_p}{\left[\frac{p(1-p)}{n(f(\theta_p))^2}\right]^{\frac{1}{2}}}$$

based on an exponential(10) population distribution. Plot the histogram and overlay a standard normal density. How good is the approximation for the setting you chose?

Part 2: Application

Now that you have some idea of the properties of the quantile estimator, you can leverage that background to inform a simple data analysis. The following data come from the CA Office of Environmental Health Hazard Assessment and comprise measurements of drinking water contaminants for a $n = 203$ census tracts in California.

```
# import and preview data
load('data/water.RData')
water %>% as_tibble()
```

```
## # A tibble: 203 x 20
##    `Boundary ID` Syste~1 Bound~2 Syste~3 Popul~4 Conne~5 Censu~6 Arsen~7 Hexav~8
##            <dbl> <chr>   <chr>   <chr>     <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1        410002 Cal-Wa~ System  C        100435   28249   91949    0      2.10
##  2        710004 Brentw~ System  C         54700   17939   41911    1.5    3.95
##  3        710007 Diablo~ System  C         36014   10654   35525    0      1.52
##  4        710008 City O~ System  C         62500   17429   63000    1.25   0
##  5       1000244 Shady ~ System  C           160      60      19    0      0
##  6       1000324 Mannin~ System  C           144       1      34    0.7    2.6
##  7       1010024 Cws - ~ System  C         24587    6230   24011    1.12   1.68
##  8       1510004 Casa L~ System  C           900     248    3576    2.43   0.375
##  9       1510022 West K~ System  C         21181    7955   21227    3.53   0.609
## 10       1510031 Bakers~ System  C        138309   42857   99159    2.66   1.15
## # ... with 193 more rows, 11 more variables: `Cadmium (ppb)` <chr>,
## #   `DBCP (ppb)` <chr>, `Lead (ppb)` <dbl>, `Nitrate (ppm)` <dbl>,
## #   `Perchlorate (ppb)` <dbl>, `PCE (ppb)` <chr>, `TCE (ppb)` <chr>,
## #   `TCP (ppb)` <chr>, `TTHMs (ppb)` <chr>,
## #   `Combination Radium 226, 228 (PCI/L)` <dbl>, `Uraniuim (PCI/L)` <dbl>, and
## #   abbreviated variable names 1: `System Name`, 2: `Boundary Type`,
## #   3: `System Type`, 4: `Population Served`, 5: Connections, ...
```

We'll work with the data on arsenic concentration (in parts per billion).

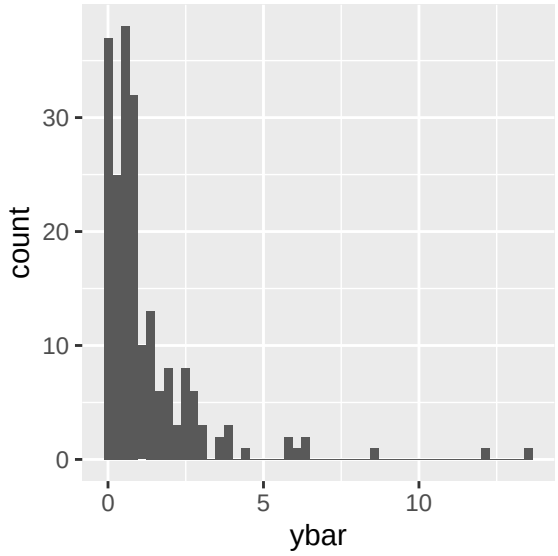
```
# grab the arsenic column
arsenic <- water$`Arsenic (ppb)`
```

The sample quantile can be computed by the `quantile` function in R. For example, the 95th percentile is:

```
# compute thetahat_{0.95}
quantile(arsenic, probs = 0.95, type = 1)
```

```
##   95%
## 3.796
```

10. Plot a histogram of the data. Informed by your simulations in Part 1, the sample size, and the shape of the histogram, how accurate do you think the sample 0.95-quantile is for the population 0.95-quantile?



A: The sample quantile is not accurate for the population quantile because there's a large variability in the upper regions of the histogram as indicated.

11. Here you'll construct what is known as a *bootstrap* confidence interval for $\theta_{0.95}$. A *resample* of the data can be drawn using `sample`:

```
sample(arsenic, size = 200, replace = T)
```

```
##   [1]  0.000  0.287  0.857  0.853  0.160  0.521  0.000  0.051  0.000  0.159
##  [11]  1.426  0.735  0.000  2.072  5.739  0.709  0.553  0.664  0.832  3.025
##  [21]  0.051  3.140  3.140  0.183  0.013  0.000  0.553  3.050 13.528  0.183
##  [31]  0.000  0.239  0.160  0.425  0.440 12.067  0.051  0.000  2.015  0.226
##  [41]  0.896  0.225  0.239 12.067  2.090  0.780  2.090  0.287  3.483  3.050
##  [51]  0.367  0.025  1.967  0.735  0.515  0.000  0.689  0.458  0.025  3.050
##  [61]  0.553  0.594  0.051  0.226  0.025  0.420  0.117  0.389  2.066  6.372
##  [71]  3.800  0.000  0.051  1.614  0.000  0.051  0.412  0.664  0.428  0.803
##  [81]  2.066  0.450  1.769  0.906  0.000  2.305  0.521  6.423  0.051  0.389
##  [91]  0.661  0.661  0.725  0.594  1.810  0.997  0.541  2.305  2.675  0.226
## [101]  0.392  0.428  0.000  0.025  1.769  0.000  0.515  0.553  0.594  0.000
## [111]  0.427  0.803  0.000  0.605  0.791  0.000  0.821  0.924  0.817  0.483
## [121]  0.367  0.000  1.641  0.829  0.496  2.664  1.263  1.810  2.369  0.051
## [131]  2.015  0.226  0.817  0.952  1.500  0.279  0.780  0.911  1.500  3.796
## [141]  1.614  0.000  8.446  2.305  0.738  0.181  4.299  0.051  0.491  0.420
## [151]  1.614  1.641  0.698  0.553  1.063  0.491  1.250  0.000  0.000  0.499
## [161]  0.491  1.488  1.264  5.739  6.372  0.523  1.967  0.183  6.423  0.665
## [171]  0.000  0.000  0.389  0.972  0.744  0.000  0.853  0.700  0.698  2.072
## [181]  2.047  0.676  1.063  0.664  0.594  1.264  0.389  2.015  0.523  2.434
## [191]  0.853  0.999  2.613  1.769  0.000  2.811  0.000  1.500  1.120  0.463
```

Write a function that draws a resample and computes the sample 0.95-quantile. Use this function to simulate 10,000 sample quantiles (in other words, simulating sample quantiles by treating the observed data as if it

were the population), and compute the standard deviation $S_{\hat{\theta}}$ of those $10,000$ values. Construct a bootstrap confidence interval as

$$\left(\hat{\theta}_{0.95} - 2S_{\hat{\theta}}, \hat{\theta}_{0.95} + 2S_{\hat{\theta}}\right)$$

12. Say an 'extremely high' arsenic concentration is defined as any concentration equal to or exceeding the 95th percentile. If you tested your tap water and found an arsenic concentration of 2.3 ppb, would you consider this is an extremely high concentration?

Yes, I would consider this an extremely high concentration since the value of 2.3 ppb lies within the bootstrap confidence interval, which is approximately from 2.0 to 5.6 ppb.

Codes

```
knitr::opts_chunk$set(echo = F, # don't show codes in document
                      results = 'markup', # format output
                      fig.align = 'center', # figure option
                      fig.width = 3,
                      fig.height = 3)
library(tidyverse)
# function to calculate the estimator
thetahat <- function(x, p){
  quantile(x, probs = p, type = 1)
}

# function to draw random samples and compute the estimator
sim_fn <- function(n, p, mu){
  y <- rexp(n, rate = 1/mu)
  out <- thetahat(y, p)
  return(out)
}

# set the sample size and number of simulations
n <- 20
nsim <- 10000

# run simulation
x <- sapply(1:nsim, function(i){sim_fn(n, p=0.7, mu=5)})
y <- qexp(0.7, 1/5)
z <- mean(x)

# construct plot
x_data <- data.frame(ybar = x)
plot1 <- ggplot(aes(x = ybar), data = x_data) +
  geom_histogram(bins = 50)
plot1 + geom_vline(xintercept = y, color = "red") + geom_vline(xintercept = z, color = "blue")

# change mu parameter and repeat
y <- sapply(1:nsim, function(i){sim_fn(n, p=0.7, mu=10)})

# construct plot
z <- qexp(0.7, 1/10)
x <- mean(y)

x_data <- data.frame(ybar = y)

plot2 <- ggplot(aes(x = ybar), data = x_data) +
  geom_histogram(bins = 50)

plot2 + geom_vline(xintercept = z, color = "red") + geom_vline(xintercept = x, color = "blue")
# set the sample size and number of simulations
n <- 20
nsim <- 10000

# first case: p = 0.6
```

```r
thetahat6_sim <- sapply(1:nsim, function(i){sim_fn(n, p=0.6, mu=5)})
var(thetahat6_sim)

# second case: p = 0.8
thetahat8_sim <- sapply(1:nsim, function(i){sim_fn(n, p=0.8, mu=5)})
var(thetahat8_sim)

# third case: p = 0.9
thetahat9_sim <- sapply(1:nsim, function(i){sim_fn(n, p=0.9, mu=5)})
var(thetahat9_sim)

# fourth case: p = 0.99
thetahat99_sim <- sapply(1:nsim, function(i){sim_fn(n, p=0.99, mu=5)})
var(thetahat99_sim)

# construct plots:
# hist(thetahat6_sim, breaks = 50, freq = F)
x_data <- data.frame(ybar = thetahat6_sim)
ggplot(aes(x = ybar), data = x_data) +
  geom_histogram(bins = 50)

# hist(thetahat8_sim, breaks = 50, freq = F)
x_data <- data.frame(ybar = thetahat8_sim)
ggplot(aes(x = ybar), data = x_data) +
  geom_histogram(bins = 50)

# hist(thetahat9_sim, breaks = 50, freq = F)
x_data <- data.frame(ybar = thetahat9_sim)
ggplot(aes(x = ybar), data = x_data) +
  geom_histogram(bins = 50)

# hist(thetahat99_sim, breaks = 50, freq = F)
x_data <- data.frame(ybar = thetahat99_sim)
ggplot(aes(x = ybar), data = x_data) +
  geom_histogram(bins = 50)

# wrap the whole simulation in an outer function of p only

# set the sample size and number of simulations
n <- 20
x <- 1000

sim_fn_var <- function(p){
  thetahat_sim <- sapply(1:x, function(i){sim_fn(n, p, 5)})
  var(thetahat_sim)
}

# generate sequence of p
p_seq <- seq(from = 0.01, to = 0.99, length = 100)

# use sapply or for loop to iterate your outer function over the values of `p_seq`
y <- sapply(1:length(p_seq), function(i){sim_fn_var(p_seq[i])})
```

```r
# Variance against p plot
plot(p_seq, y)



# repeat the fourth case from step 5 with larger n
n <- 20000
nsim <- 10000

thetahat99n_sim <- sapply(1:nsim, function(i){sim_fn(n, p=0.99, mu=5)})
mean(thetahat99n_sim)
var(thetahat99n_sim)

# plot and compare
#hist(thetahat99n_sim, breaks = 50, freq = F)
x_data <- data.frame(ybar = thetahat99n_sim)
ggplot(aes(x = ybar), data = x_data) +
  geom_histogram(bins = 50)

# function to compute standardized estimator
thetahat_std <- function(x, n, p){
  theta_p <- qexp(p, rate = 1/10)
  V <- p*(1 - p)/(dexp(theta_p, rate = 1/10))^2
  out <- sqrt(n)*(thetahat(x, p) - theta_p)/sqrt(V)
  return(out)
}

# set sample size and number of simulations

# run simulation

# construct histogram with N(0, 1) density overlaid

# import and preview data
load('data/water.RData')
water %>% as_tibble()
# grab the arsenic column
arsenic <- water$`Arsenic (ppb)`
# compute thetahat_{0.95}
quantile(arsenic, probs = 0.95, type = 1)


# construct histogram

n <- 203

q10 <- c()

for (i in 1:n) {
  q10[i] <- arsenic[i]
}

# hist(y, breaks = 50, freq = F)
```

```r
x_data <- data.frame(ybar = q10)
ggplot(aes(x = ybar), data = x_data) +
  geom_histogram(bins = 50)


sample(arsenic, size = 200, replace = T)
# write a function to compute a bootstrap draw of thetahat_{0.95}
thetahat_boot <- function(df, p){
  resample <- sample(df, size = 200, replace = T) # fill in here
  out <- thetahat(resample, p)
}

# set p and the number of simulations
p <- 0.95
nsim <- 10000

# iterate to produce the 'bootstrap sampling distribution'
thetahat_sim <- sapply(1:nsim, function(i){thetahat_boot(arsenic, p)})


# construct histogram

x_data <- data.frame(ybar = thetahat_sim)
ggplot(aes(x = ybar), data = x_data) +
  geom_histogram(bins = 50)

# compute standard deviation
thetahat_sd <- sd(thetahat_sim)



# More variation -- width and center change
ci1 <- function(i){
  thetahat_sd <- sd(sapply(1:x, function(i){thetahat_boot(arsenic, p)}))
  lwr <- thetahat_boot(arsenic, 0.95) - 2*(thetahat_sd)  # lower-bound
  upr <- thetahat_boot(arsenic, 0.95) + 2*(thetahat_sd)  # upper-bound
  return(c(lwr = lwr, upr = upr))
}
sapply(1:5, ci1)

# less variation -- width changes, only slightly
ci2 <- function(i){
  thetahat_sd <- sd(sapply(1:x, function(i){thetahat_boot(arsenic, p)}))
  lwr <- thetahat(arsenic, 0.95) - 2*(thetahat_sd)  # lower-bound
  upr <- thetahat(arsenic, 0.95) + 2*(thetahat_sd)  # upper-bound
  return(c(lwr = lwr, upr = upr))
}
sapply(1:5, ci2)

# generate code appendix
```