Section 4

Foundations of Statistical Inference (PLSC 503)

2/13/2019

Contents

Code Review

Let's practice writing better code. We have gone over loops in previous sections. That's fine, but for simulations it's better than you learn how to write functions and make use of the `replicate()` function in `R`. Let's practice using Exercise 2 from the last pset.

Pset 2, Exercise 2

```r
load("pset2-data.RData")

set.seed(503) # Everyone use this for your seed going forward please!

# Number of simulations to run & number of draws
nsims <- 5000
ndraw <- 4

# Initialize matrix for storage
Xdraws <- matrix(data = NA, nrow = nsims, ncol = ndraw)

# For loop approach:
for (s in 1:nsims) {
  Xdraws[s,] <- sample(age_pop, ndraw, replace = TRUE)
}

# Using replicate:
Xdraws <- t(replicate(2, sample(age_pop, ndraw, replace = TRUE)))

# Using a function plus replicate:
sim_function <- function(nsims = 5000, replace = TRUE){
  sample(age_pop, nsims, replace = replace)
}
```

```r
# Cool, we don't need to transpose it now!
Xdraws <- replicate(4, sim_function())

# Great Expectations:
colMeans(Xdraws)
```

```
## [1] 56.4594 56.4406 56.4412 56.4290
```

```r
# Apply Yourself:
apply(Xdraws, 2, mean)
```

```
## [1] 56.4594 56.4406 56.4412 56.4290
```

The Monty Hall Problem

The Monty Hall Problem[1] is a very famous puzzle that you may have learned about in a probability class, or seen in a movie. The puzzle was originally stated as follows:

> *Suppose you're on a game show, and you're given the choice of three doors: Behind one door is a car; behind the others, goats. You pick a door, say No. 1, and the host, who knows what's behind the doors, opens another door, say No. 3, which has a goat. He then says to you, "Do you want to pick door No. 2?" Is it to your advantage to switch your choice?*

As you may know from the movie "21"[2], the correct answer is to switch. This puzzle generated a bunch of controversy. When Marilyn vos Savant presented the correct solution in her column for *Parade* magazine in 1990, thousands of people wrote in to tell her she was wrong[3].

To avoid confusion for the problem, make the following standard assumptions:

1. The host (Monty) must always open a door that was not picked by the contestant.
2. Monty must always open a door that has a goat behind it and never the car! If he has a choice to make between two goat doors, he chooses at random.
3. Monty must always offer the chance to switch between the originally chosen door and the remaining closed door.

Show that contestants who switch have a better chance of winning the car than contestants who stick with their initial choice. First, provide an analytic solution. Second, convice yourself of this using simulation. For the simulation solution, you should write a function called `monty_hall` that takes two arguments: `sims` for the number of simulations to run and `strategy` for one of three strategy choices: "stay", "switch" or "random". The "random" strategy should just pick one of the three doors at random. The input to `sims` should be an integer (≥ 1000) and the `strategy` argument should be a character. In the function, assume that your initial choice of door is random.

There is no "right" way to do the simulation. You don't need to use any random number generators. The only function you need to use is `sample()`. I would recommend using a for loop and taking advantage of nested if statements. This may not be the most efficient way, but I think it is the easiest. There are actually two methods (that I know of) for the analytic solution. I will show both in section.

[1] https://en.wikipedia.org/wiki/Monty_Hall_problem
[2] clip: https://www.youtube.com/watch?v=Zr_xWfThjJ0
[3] Have a look at some of the comments she received here: http://marilynvossavant.com/game-show-problem/

Code Sketch

```r
monty_hall <- function(strategy = "switch", sims = 1000, ...) {
  # Initialize door vector.

  # Initialize vector to keep track of wins.

  # Inner loop.
  for(i in 1:sims) {
  # Randomly determine winning door.

  # Randomly determine your first choice.

  # The host reveals one of the doors you did not pick (which contains a
  #   goat). [Caution: what happens if you get lucky and pick the winning
  #   door first?]

  # Result if you decide to stay.

  # Or if you decide to switch.

  # Or if you pick a random strategy.

  # Tally the outcome for this sim. End loop.
  }

  # Return the win percentage, averaged across all sims.
  return(...)
}
```

Analytic solution:

Label the doors 1 through 3. Assume you pick door 1 (it doesn't matter what door we use here). The host opens another door, with a goat behind it. Let C_i be the event that the car is behind door i. Then,

$$\Pr(\text{win car}) = \Pr(\text{win car}|C_1) \cdot \frac{1}{3} + \Pr(\text{win car}|C_2) \cdot \frac{1}{3} + \Pr(\text{win car}|C_3) \cdot \frac{1}{3}$$

Suppose we switch. If the car is in fact behind door 1 (our first choice), then we get a goat, so $\Pr(\text{win car}|C_1) = 0$. If, however, the car is **not** behind door 1, then the revelation of the goat behind door 2 or 3 gives us some valuable information: the car is behind the unopened door. Therefore, switching is a good idea since

$$\Pr(\text{win car}) = 0 \cdot \frac{1}{3} + 1 \cdot \frac{1}{3} + 1 \cdot \frac{1}{3} = \frac{2}{3}$$

Another solution to this uses Bayes' rule,

$$\Pr(A|B) = \frac{\Pr(B|A)\Pr(A)}{\Pr(B)}$$

Suppose Monty opens door 2. We can use this to find the conditional probability of success by switching, given the evidence from the door opening. Let M_j be the event that he opens door $j \in \{2, 3\}$. Suppose he opens door 2. Then,

$$\Pr(C_1|M_2) = \frac{\Pr(M_2|C_1)\Pr(C_1)}{\Pr(M_2)}$$

We know that $\Pr(M_2) = \Pr(M_3) = 1/2$ and $\Pr(C_1) = 1/3$. But what is $\Pr(M_2|C_1)$? Well we assume Monty picks at random so his decision is independent of where the car is. Thus $\Pr(M_2|C_1) = 1/2$. Putting this all together,

$$\Pr(C_1|M_2) = \frac{(1/2)(1/3)}{(1/2)} = \frac{1}{3}$$

So, given that he opens door 2, the probability that the car is behind the door you originally chose is 1/3. The probability that the car is behind door number 3 is $1 - 1/3 = 2/3$.

Simulation solution

```r
monty_hall <- function(strategy = "switch", sims = 1000, first_choice = NULL) {

  # Initialize door vector.
  doors <- 1:3

  # Initialize vector to keep track of wins.
  prize <- NULL

  # Inner loop.
  for(i in 1:sims) {

    # Randomly determine winning door.
    win_door <- sample(doors, 1)

    # Randomly determine your first choice.
    if(is.null(first_choice))
      first_choice <- sample(doors, 1)
    else
      first_choice <- first_choice

    # The host reveals one of the doors you did not pick (which contains
    # a goat).
    if(win_door != first_choice){
      opened_door <- doors[-c(win_door, first_choice)]
    } else {
      opened_door <- sample(doors[-c(win_door, first_choice)], 1)
    }

    # Result if you decide to stay.
    if(strategy == "stay"){
      result <- first_choice
    } else if(strategy == "switch") {
      # Or if you decide to switch.
      result <- doors[-c(opened_door, first_choice)]
    } else {
      # Or if you pick a random strategy.
      result <- sample(doors[-opened_door], 1)
    }

    # Tally the outcome for this sim.
    if(result == win_door) {
      prize[i] <- "Car!"
      } else
        prize[i] <- "Goat!"
    }

  # Return the win percentage, averaged across all sims.
  return(mean(prize == "Car!"))
}
```

Now let's compare the strategies,

```r
set.seed(123)

## Suppose I always pick door number 1 first. What happens if I stay?
monty_hall(strategy = "stay", sims = 10^4, first_choice = 1)
```

```
## [1] 0.3302
```

```r
## What if I switch?
monty_hall(strategy = "switch", sims = 10^4, first_choice = 1)
```

```
## [1] 0.6611
```

```r
## What if I pick my second choice at random?
monty_hall(strategy = "random", sims = 10^4, first_choice = 1)
```

```
## [1] 0.5041
```

```r
## Suppose instead I pick the first choice at random. Does this change
## anything? (Spoiler: No!)
monty_hall(strategy = "stay", sims = 10^4)
```

```
## [1] 0.3348
```

```r
monty_hall(strategy = "switch", sims = 10^4)
```

```
## [1] 0.6652
```

```r
monty_hall(strategy = "random", sims = 10^4)
```

```
## [1] 0.5001
```

As expected, I'm better off switching than staying. I could actually do better than staying by flipping a coin to decide. Note that this win probability is just the average of the other two win probabilities $1/2 \cdot (1/3 + 2/3)$, or just the expected value of a coin flip over two choices :-).

Interactive solution

A more fun interactive function that you can play with (stolen from Blitzstein and Hwang, *Introduction to Probability* pp. 73-74) appears below. Store this function in memory and then type `monty()` into the console to play the game.

```r
monty <- function() {
  doors <- 1:3

  # randomly pick where the car is
  cardoor <- sample(doors, 1)

  # prompt player
  print("Monty Hall says 'Pick a door, any door!'")

  # receive the player's choice of door (should be 1, 2, or 3)
  chosen <- scan(what = integer(), nlines = 1, quiet = TRUE)

  # pick Monty's door (can't be the player's door or the car door)
  if (chosen != cardoor)
    montydoor <- doors[-c(chosen, cardoor)]
  else
    montydoor <- sample(doors[-chosen], 1)

  # find out whether the player wants to switch doors
  print(paste("Monty opens door ", montydoor, "!", sep = ""))
  print("Would you like to switch (y/n)?")
  reply <- scan(what = character(), nlines = 1, quiet = TRUE)

  # interpret what player wrote as "yes" if it starts with "y"
  if (substr(reply, 1, 1) == "y")
    chosen <- doors[-c(chosen, montydoor)]

  # announce the result of the game!
  if (chosen == cardoor)
    print("You won!")
  else
    print ("You lost!")

}
```

Concept Review

Let θ denote some quantity of interest, e.g. the "population mean" for some random variable X. In frequentist statistics (this course) θ is some **fixed** unknown quantity that we want to learn about; for example, a parameter from some probability density function, a (constant) treatment effect, etc.

We are often interested in obtaining a point estimate of θ, using an **estimator**, which is typically written as $\hat{\theta}$. Some writers will occasionally use the notation $\hat{\theta}_n$ to refer to an estimator. This is nice, because the n subscript reminds us that it depends on the data, i.e. $\hat{\theta}_n = g(X_1, \ldots, X_n)$, which is why $\hat{\theta}_n$ is a **random variable**.

Let's take an example. We will continute to use this simple example throughout the section notes today. Suppose $X_1, \ldots, X_n \sim \text{Bernoulli}(\theta)$ and let $\hat{\theta}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$ denote an estimator for θ, i.e. the sample mean.

Variance Estimation

Note that an estimator $\hat{\theta}_n$ has a **sampling distribution**, and the standard deviation of the sampling distribution is called the **standard error**:

$$\text{se}(\hat{\theta}_n) = \sqrt{\text{Var}(\hat{\theta}_n)}$$

We also have the **estimated standard error**: $\hat{\text{se}}(\hat{\theta}_n)$. The standard error for θ in our running example is $\text{se}[\hat{\theta}_n] = \sqrt{\text{Var}(\hat{\theta}_n)} = \sqrt{\theta(1-\theta)/n}$, and the estimated standard error is $\hat{\text{se}}[\hat{\theta}_n] = \sqrt{\hat{\text{Var}}(\hat{\theta}_n)} = \sqrt{\hat{\theta}_n(1-\hat{\theta}_n)/n}$. Estimators always wear hats.

Question: Is the value of $\text{Var}(\hat{\theta})$ known?.

No! We are using the observed data to learn about θ. Typically we will use the observed data to construct an estimator, e.g. $\hat{\text{Var}}(\hat{\theta}) = \frac{\hat{\theta}(1-\hat{\theta})}{n}$.

Question: What is the variance of a Bernoulli r.v.?

Well, these are all IID draws from a Bernoulli,

$$\text{Var}[X_1] = \mathbb{E}[X_1^2] - \mathbb{E}[X_1]^2 = \sum_x x^2 f_{X_1}(X_1 = x) - \left(\sum_x x f_{X_1}(X_1 = x)\right)^2$$

We already know that $\mathbb{E}[X_1] = \theta$. Therefore, $\mathbb{E}[X_1]^2 = \theta^2$. Let's solve for the other bit,

$$\mathbb{E}[X_1^2] = \sum_x x^2 f_{X_1}(X_1 = x) = 1^2 \cdot \text{Pr}(X_1 = 1) + 0^2 \cdot \text{Pr}(X_1 = 0) = 1 \cdot \theta + 0 \cdot (1-\theta) = \theta$$

Now we can conclude that $\text{Var}[X_1] = \theta - \theta^2 = \theta(1-\theta)$

Bias

$\hat{\theta}_n$ is an **unbiased** estimator for θ if $\mathbb{E}[\hat{\theta}_n] = \theta$. Alternatively, we can define the bias of an estimator as:

$$\text{bias}(\hat{\theta}_n) = \mathbb{E}[\hat{\theta}_n] - \theta$$

and note that $\text{bias}(\hat{\theta}_n) = 0$ when the estimator is unbiased.

Caution: bias is a property of an estimator, not an estimate

Question: is our estimator for θ unbiased?

$$\mathbb{E}[\hat{\theta}_n] = \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} X_i\right] = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[X_i] = \theta$$

Question: Is $\mathbb{E}[X_1]$ an unbiased estimator for θ?

Yes! Since $\mathbb{E}[X_1] = \theta$ because it's an IID draw from Bernoulli(θ).

Question: what about the estimator $\tilde{\theta}_n = \frac{s+1}{n+2}$ where $s = \sum_{i=1}^{n} X_i$?

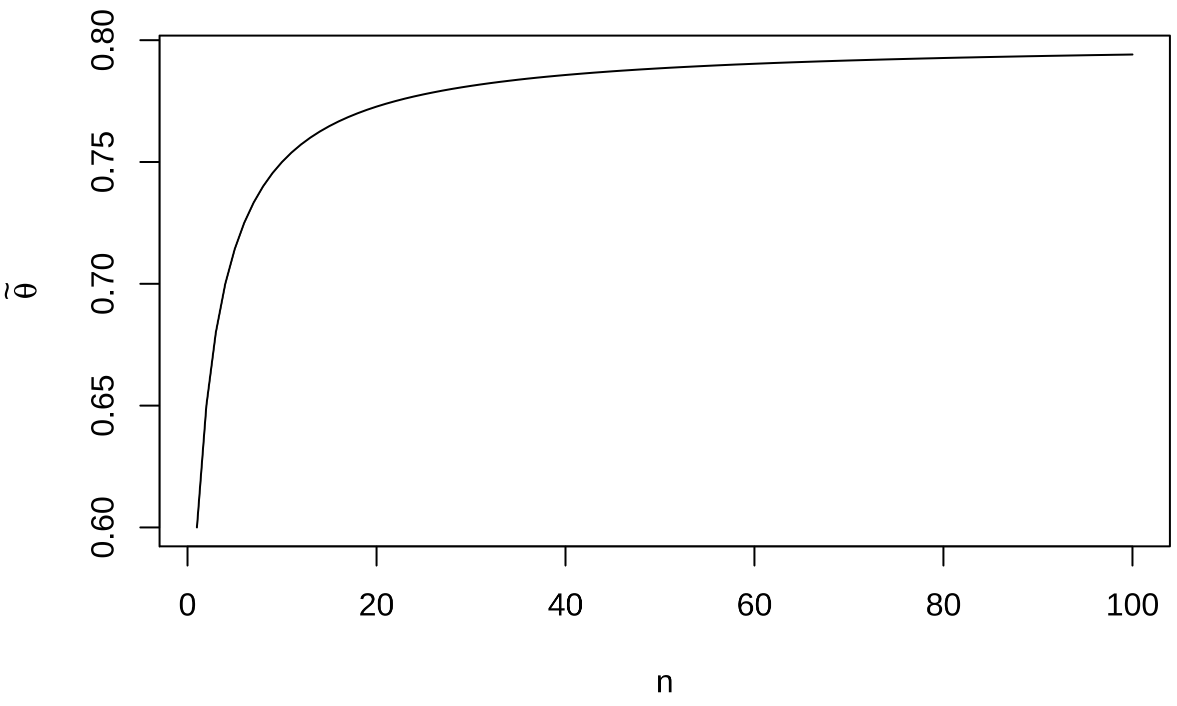
Note that this can be re-written as $\tilde{\theta}_n = \lambda_n\hat{\theta}_n + \frac{1}{2}(1-\lambda_n)$, where $\lambda_n = \frac{n}{n+2}$, which is approximately 1 when n gets "large" (set $n = 100$ to see this)[4].

$$\mathbb{E}[\tilde{\theta}_n] = \mathbb{E}\left[\lambda_n\hat{\theta}_n + (1-\lambda_n)\tilde{\theta}_n\right] = \lambda_n\mathbb{E}[\hat{\theta}_n] + \frac{1}{2}(1-\lambda_n)]$$
$$= \lambda_n\theta + \frac{1}{2}(1-\lambda_n)$$

which is clearly biased. Let's examine this numerically, letting $\theta = 0.8$ and n vary from 1 to 100.

```r
theta_squiggle <- function(n, theta) {
  lambda <- n/(n+2)
  return(lambda*theta + 0.5*(1-lambda))
  }

plot(1:100, theta_squiggle(n = 1:100, theta = 0.8), type = "l", xlab = "n",
     ylab = expression(tilde(theta)))
```



[4]This is a degenerate example of a Bayesian estimator where $\tilde{\theta}$ is the posterior mean and the prior is Beta($\alpha = 1, \beta = 1$) with mean 1/2. It's "degenerate" because the prior is not informative or "flat" when $\alpha = \beta = 1$

Consistency

$\hat{\theta}_n$ is a **consistent** estimator for θ if $\hat{\theta}_n \xrightarrow{P} \theta$, where the notation $\xrightarrow{P}$ reads "converges in probability". As Fredrik explained in class, this means that $\Pr(|\hat{\theta}_n - \theta| > \epsilon) \to 0$ as $n \to \infty$ for every $\epsilon > 0$. Another way of stating this: If $\text{bias}[\hat{\theta}_n] \to 0$ and $\text{se}[\hat{\theta}_n]$ as $n \to \infty$, then $\hat{\theta}_n$ is a consistent estimator for θ.

The key insight here is that the difference between the estimator and the truth is trivial in "large" samples. The study of how "large" a sample needs to be falls within the domain of folks who study the convergence of random variables. This is an important area of study in statistics, that falls outside the scope of this course[5].

Question: Is $\hat{\theta}_n$ a consistent estimator for θ?

Yes. We know that $\mathbb{E}[\hat{\theta}_n] = \theta$, because it's an unbiased estimator. Further, $\text{se}[\hat{\theta}_n] = \sqrt{\theta(1-\theta)/n}$, which clearly goes to zero as n approaches infinity. Thus, $\hat{\theta}_n \xrightarrow{P} \theta$. This estimator is **unbiased** and **consistent**.

Question: Is $\mathbb{E}[X_1]$ a consistent estimator for θ?

No! $\text{Var}(X_1) = \theta(1-\theta)$. This does not shrink to zero as $n \to \infty$

Question: Is $\tilde{\theta}_n$ a consistent estimator for θ?

Yes. We informally showed this with the plot in the previous section. A more formal argument is to note that $\text{se}(\tilde{\theta}_n) \to 0$ and $\text{bias}[\hat{\theta}_n] \to 0$ as $n \to \infty$. We need to get the expression for the variance in order to do this.

$$\text{Var}(\tilde{\theta}_n) = \text{Var}\left[\lambda_n \hat{\theta}_n + \frac{1}{2}(1-\lambda_n)\right] = \lambda_n^2 \text{Var}[\hat{\theta}_n] = \lambda_n^2 \frac{\theta(1-\theta)}{n} = \frac{n^2}{(n+2)^2}\frac{\theta(1-\theta)}{n} = \theta(1-\theta)\frac{n}{(n+2)^2}$$

which clearly approaches 0 as $n \to \infty$. An easy way to see this is to note that the degree of the numerator n is lower than the degree of the demoninator $(n+2)^2$. This implies that the numerator grows slower than the denominator as $n \to \infty$.

[5]see http://www.stat.cmu.edu/~larry/=stat700/Lecture7.pdf for an overview

Chebyshev's Inequality

Let $\mu = \mathbb{E}[X]$ and $\sigma^2 = \text{Var}[X]$. Then,

$$\Pr\left(|X - \mu| \geq \epsilon\right) \leq \frac{\sigma^2}{\epsilon^2}$$

for all $\epsilon > 0$.

Question: How "close" is $\hat{\theta}_n$ to θ?

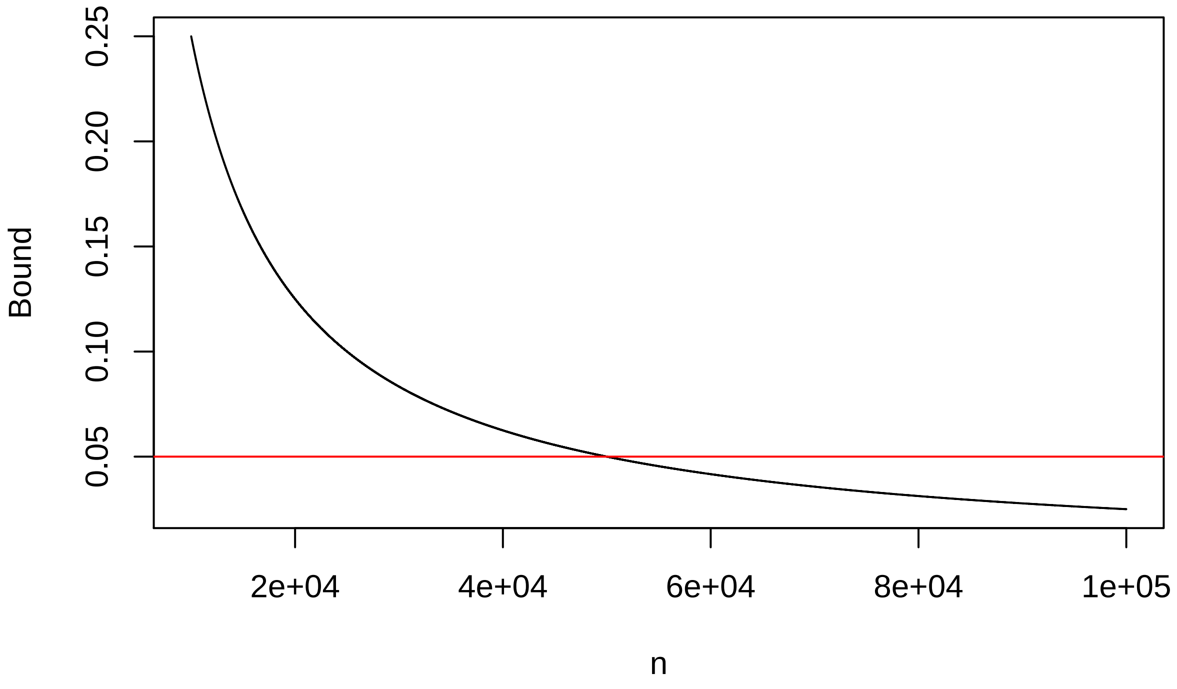
$$\Pr\left(|\hat{\theta}_n - \theta| \geq \epsilon\right) \leq \frac{\text{Var}[\hat{\theta}_n]}{\epsilon^2} = \frac{\theta(1-\theta)}{n\epsilon^2} \leq \frac{1}{4n\epsilon^2}$$

Wait what happened? $\theta(1-\theta) \leq \frac{1}{4}$ for any $\theta \in [0,1]$. In other words, the variance of a Bernoulli r.v. is maximized at $\theta = 0.50$.

Suppose we decide that $\epsilon = 0.01$ is "small". Then $\Pr\left(|\hat{\theta}_n - \theta| \geq \epsilon\right) \leq \frac{1}{n \cdot 0.0004}$. So when $n = 100,000$ the bound is 0.025. In words, the probability that the estimator differs from the target parameter by some small amount is no greater than 0.025 if we take 100,000 draws. Let's plot this for fixed ϵ and n from 10,000 to 100,000. Note that when $n = 50,001$ the bound is below the Holy Threshold of 0.05 (red horizontal line). We therefore might need 50,001 IID draws before we could say that our estimator was "close enough" to the target parameter.

```r
cheby <- function(n, e = 0.01){
  1/(4*n*e^2)
}

plot(10000:100000, cheby(n = 10000:100000), type = "l", xlab = "n", ylab = "Bound")
abline(h = 0.05, col = "red")
```



Mean Squared Error (MSE)

The quality of an estimator is often assessed using Mean Squared Error (MSE). What is this?

$$
\begin{aligned}
\mathbb{E}[\hat{\theta}_n - \theta]^2 &= \mathbb{E}[\hat{\theta}_n - \bar{\theta}_n + \bar{\theta}_n - \theta]^2 \\
&= \mathbb{E}[\hat{\theta}_n - \bar{\theta}_n]^2 + 2(\bar{\theta}_n - \theta)\mathbb{E}[\hat{\theta}_n - \bar{\theta}_n] + \mathbb{E}[\bar{\theta}_n - \theta]^2 \\
&= \mathbb{E}[\hat{\theta}_n - \bar{\theta}_n]^2 + \mathbb{E}[\bar{\theta}_n - \theta]^2 \\
&= \mathrm{Var}[\hat{\theta}_n] + \mathrm{bias}^2(\hat{\theta}_n)
\end{aligned}
$$

Note that $2(\bar{\theta}_n - \theta)\mathbb{E}[\hat{\theta}_n - \bar{\theta}_n]$ in the second line dissapears. Why?

$$
\begin{aligned}
2(\bar{\theta}_n - \theta)\mathbb{E}[\hat{\theta}_n - \bar{\theta}_n] &= 2(\bar{\theta}_n - \theta)(\mathbb{E}[\hat{\theta}_n] - \mathbb{E}[\bar{\theta}_n]) \\
&= 2(\bar{\theta}_n - \theta)(\bar{\theta}_n - \bar{\theta}_n) \\
&= 0
\end{aligned}
$$

We will demonstrate a useful application of this in a future section after we cover linear regression. As a preview, note that we can replace $\hat{\theta}_n$ with some generic function of the observed data, e.g. $\hat{f}(X)$, and the logic carries through.

Question: What is the MSE of $\hat{\theta}_n$?

It's just $\mathrm{Var}[\hat{\theta}_n] = \frac{\theta(1-\theta)}{n}$ because $\mathrm{bias}^2(\hat{\theta}_n) = 0$.

Question: What is the MSE of $\tilde{\theta}_n$?

It's very ugly:

$$
\begin{aligned}
\mathrm{MSE}(\tilde{\theta}_n) &= \mathrm{Var}[\tilde{\theta}_n] + \mathrm{bias}^2(\tilde{\theta}_n) \\
&= \frac{\theta(1-\theta)n}{(n+2)^2} + \left(\frac{1-2\theta}{n+2}\right)^2 \\
&= \frac{(1-2\theta)^2 + \theta(1-\theta)n}{(n+2)^2}
\end{aligned}
$$

But of course it does approach zero as $n \to \infty$! Why would we ever use something like this? Well for small n the MSE can be much, much lower. Suppose $\theta = 1/2$ and note that $\mathrm{Var}[\tilde{\theta}_n] < \mathrm{Var}[\hat{\theta}_n]$ in small samples. So even though $\tilde{\theta}_n$ is biased we can improve MSE in these settings. This illustrates the bias/variance tradeoff that often exists in statistics:

```r
mse_tilde <- function(n, theta  = 1/2){
  ((1-2*theta)^2 + theta*(1-theta)*n)/(n+2)^2
}

mse_hat <- function(n, theta = 1/2){
  (theta*(1-theta))/n
}

# What's the MSE when we can only flip the coin twice?
mse_tilde(n = 2)
```

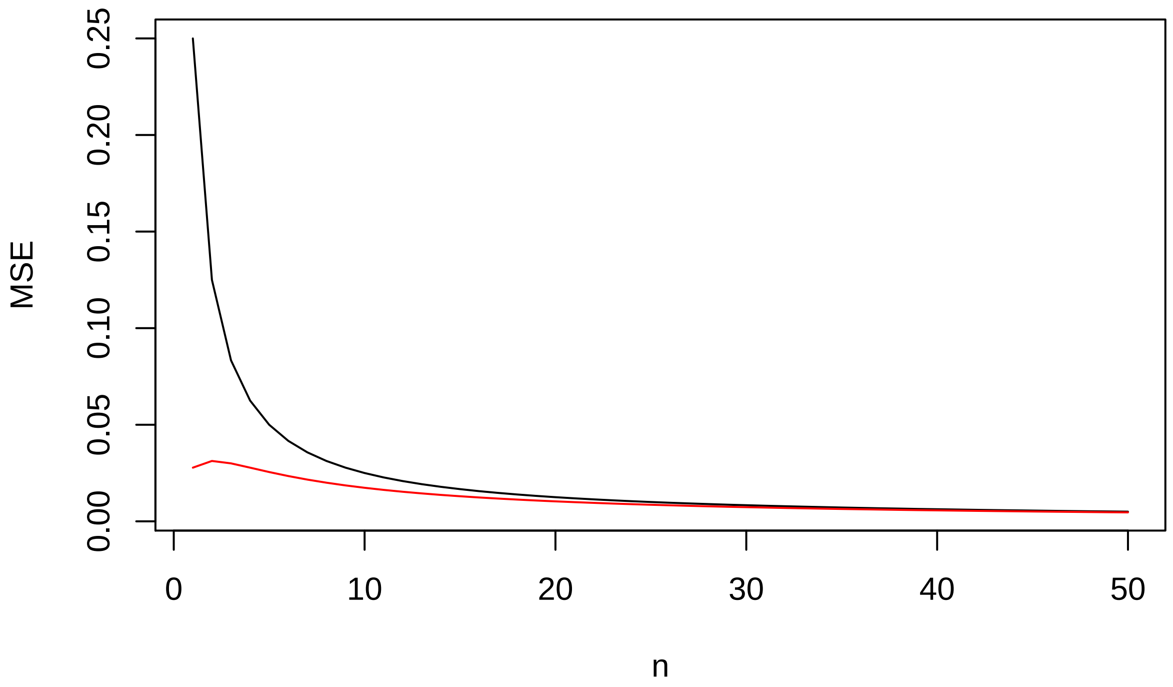
```
## [1] 0.03125
```

```r
mse_hat(n = 2)
```

```
## [1] 0.125
```

This relative advantage is going to dissapear pretty quickly as n increases. Let's visualize this by plotting the functions (red line is the biased estimator):

```r
plot(1:50, mse_hat(n = 1:50), type = "l", xlab = "n",
     ylab = "MSE")
lines(1:50, mse_tilde(n = 1:50), type = "l",  col = "red")
```



Note that the MSE difference is going to be largest in small samples when $\theta = 1/2$. [Exercise: why?][6]

[6]You might want to play around on this website and use the coin flipping widget: http://marcharper.codes/2015-08-24/bayesian.html