Chapter 2 and 3 Statistical Rethinking

Allyson Cameron

2022-09-20

Chapter 2

"Easy" Questions ***2E1. Which of the expressions below correspond to the statement: the probability of rain on Monday?***

(2) Pr(rain| Monday) and (4) Pr(rain,Monday)/Pr(Monday)

2E2. Which of the following statements corresponds to the expression: Pr(Monday| rain)?

(3) "The probability that it is Monday, given that it is raining."

2E3. Which of the expressions below correspond to the statement: the probability that it is Monday, given that it is raining?

(1) Pr(Monday| rain) and (4) Pr(rain| Monday)Pr(Monday)/Pr(rain)

2E4.The Bayesian statistician Bruno de Finetti (1906–1985) began his 1973 book on probability theory with the declaration: "PROBABILITY DOES NOT EXIST." The capitals appeared in the original, so I imagine de Finetti wanted us to shout this statement. What he meant is that probability is a device for describing uncertainty from the perspective of an observer with limited knowledge; it has no objective reality. Discuss the globe tossing example from the chapter, in light of this statement. What does it mean to say "the probability of water is 0.7"?

When we say the probability of water is 0.7 we are saying that as the observer, we estimate the probability will be 0.7. We cannot guarantee this (we're uncertain) but only choose an estimate parameter based on what we do know: that there is some water and some land.

Medium Questions ***2M1. Recall the globe tossing model from the chapter. Compute and plot the grid approximate posterior distribution for each of the following sets of observations. In each case, assume a uniform prior for p.***

(1) W, W, W
(2) W, W, W, L
(3) L, W, W, L, W, W, W

Let's do each one separately. I will use the code from the book/lecture and specify for each variation 1/2/3.

```r
# let's start by loading the packages we will need
library(rethinking)
```

```
## Loading required package: rstan
```

```
## Loading required package: StanHeaders
```

```
##
## rstan version 2.26.13 (Stan version 2.26.1)
```

```
## For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores()).
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)
## For within-chain threading using ‘reduce_sum()‘ or ‘map_rect()‘ Stan functions,
## change ‘threads_per_chain‘ option:
## rstan_options(threads_per_chain = 1)
```

```
## Loading required package: parallel
```

```
## Loading required package: dagitty
```

```
## rethinking (Version 2.01)
```

```
##
## Attaching package: 'rethinking'
```

```
## The following object is masked from 'package:stats':
##
##     rstudent
```

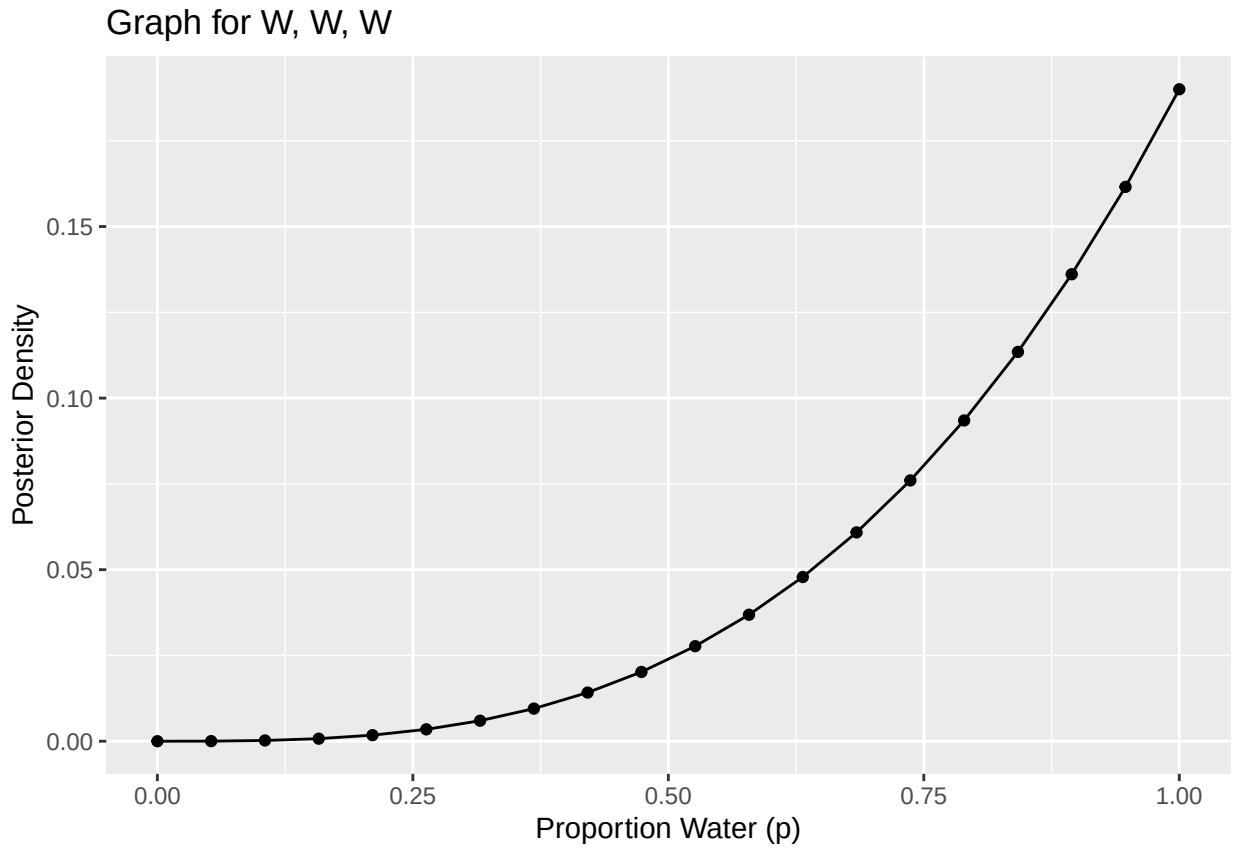
```r
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
```

```
## v ggplot2 3.3.6      v purrr   0.3.4
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.0      v stringr 1.4.0
## v readr   2.1.2      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x tidyr::extract() masks rstan::extract()
## x dplyr::filter()  masks stats::filter()
## x dplyr::lag()     masks stats::lag()
## x purrr::map()     masks rethinking::map()
```

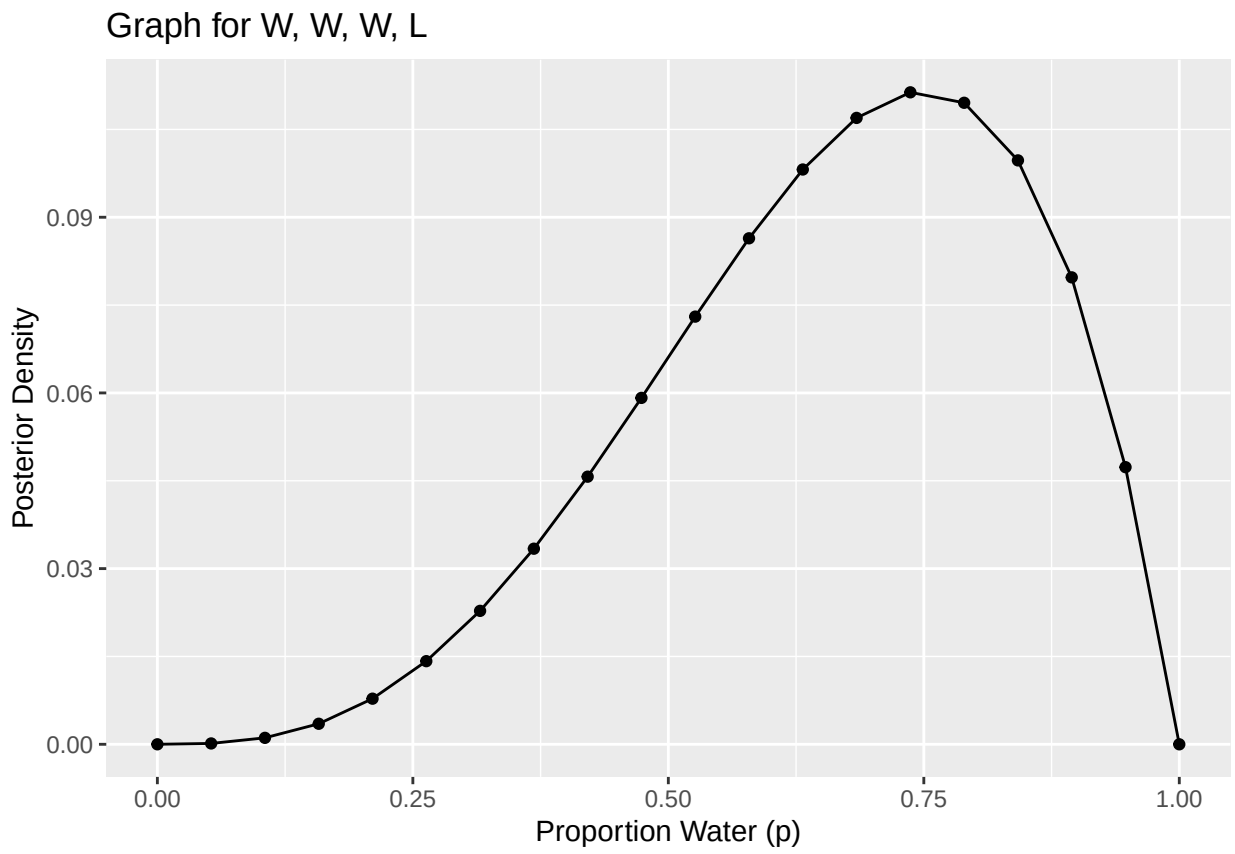
```r
# next let's work on #1

plot_1 <- tibble(p_grid =  seq(from = 0, to = 1, length.out = 20),
                 prior = rep(1,20)) %>%
```

```
          mutate(prob_data1 = dbinom(3, size = 3, prob = p_grid)) %>%
          mutate(posterior1 = prob_data1 * prior / sum(prob_data1 * prior))


ggplot(plot_1, aes(x = p_grid, y = posterior1)) +
  geom_point() + geom_line() + labs(x = "Proportion Water (p)",
                                    y = "Posterior Density",
                                    title = "Graph for W, W, W")
```



Now, let's make a graph for W, W, W, L.

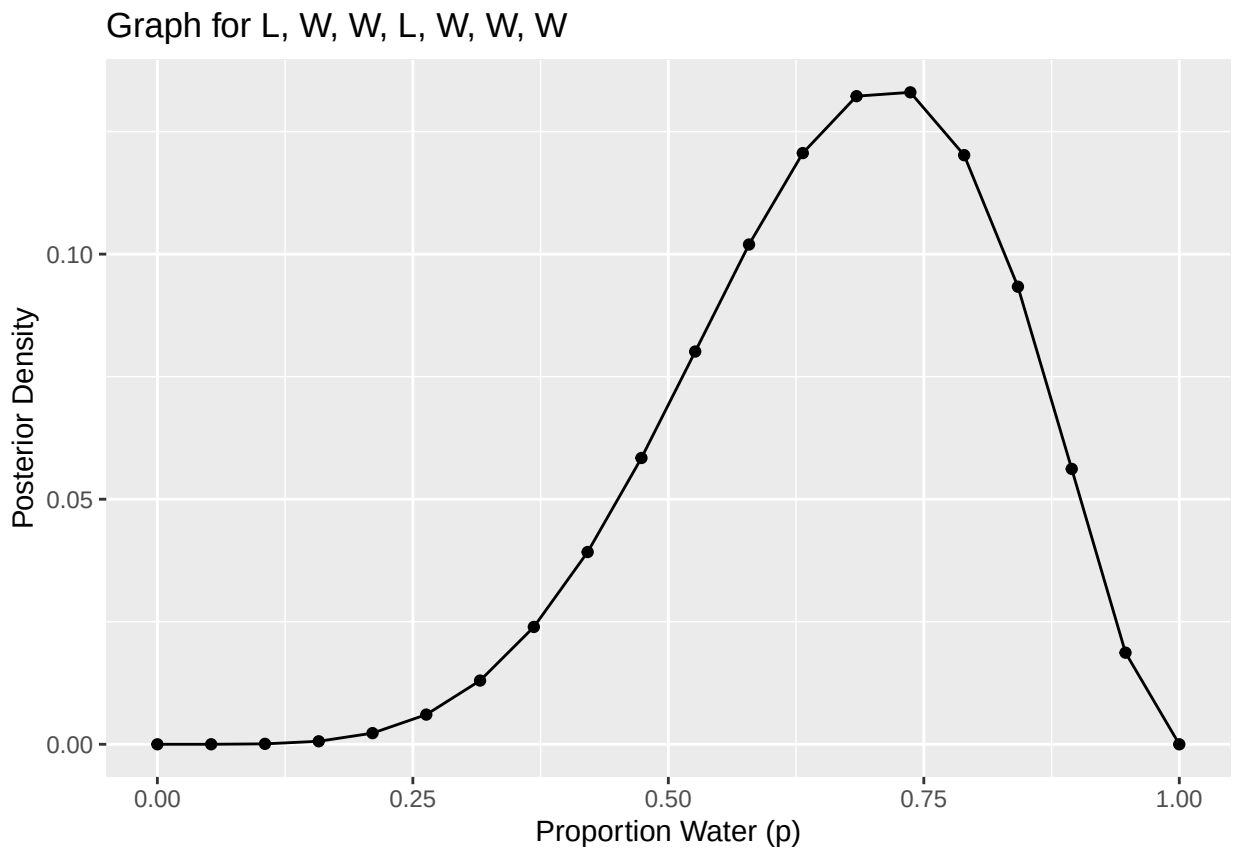
```
plot_2 <- tibble(p_grid =  seq(from = 0, to = 1, length.out = 20),
                 prior = rep(1,20)) %>%
          mutate(prob_data2 = dbinom(3, size = 4, prob = p_grid)) %>%
          mutate(posterior2 = prob_data2 * prior / sum(prob_data2 * prior))

ggplot(plot_2, aes(x = p_grid, y = posterior2)) +
  geom_point() + geom_line() + labs(x = "Proportion Water (p)",
                                    y = "Posterior Density",
                                    title = "Graph for W, W, W, L")
```



Notice how now the graph forms an upside down U because we know that there is some land and the proportion of water can no longer be 1.

Lastly, let's make our graph for L, W, W, L, W, W, W. We'll use similar code as before.

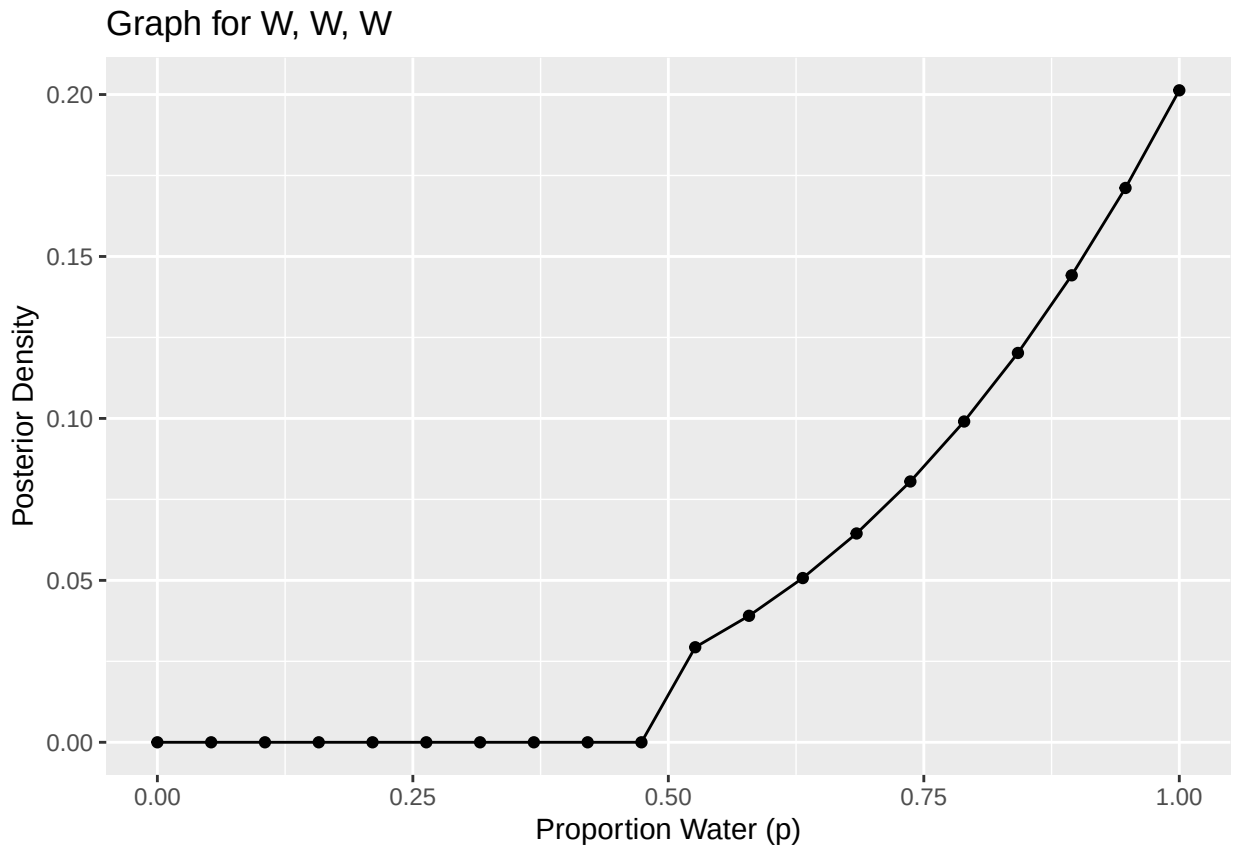
```
plot_3 <- tibble(p_grid =  seq(from = 0, to = 1, length.out = 20),
                 prior = rep(1,20)) %>%
          mutate(prob_data3 = dbinom(5, size = 7, prob = p_grid)) %>%
          mutate(posterior3 = prob_data3 * prior / sum(prob_data3 * prior))

ggplot(plot_3, aes(x = p_grid, y = posterior3)) +
  geom_point() + geom_line() + labs(x = "Proportion Water (p)",
                                    y = "Posterior Density",
                                    title = "Graph for L, W, W, L, W, W, W")
```



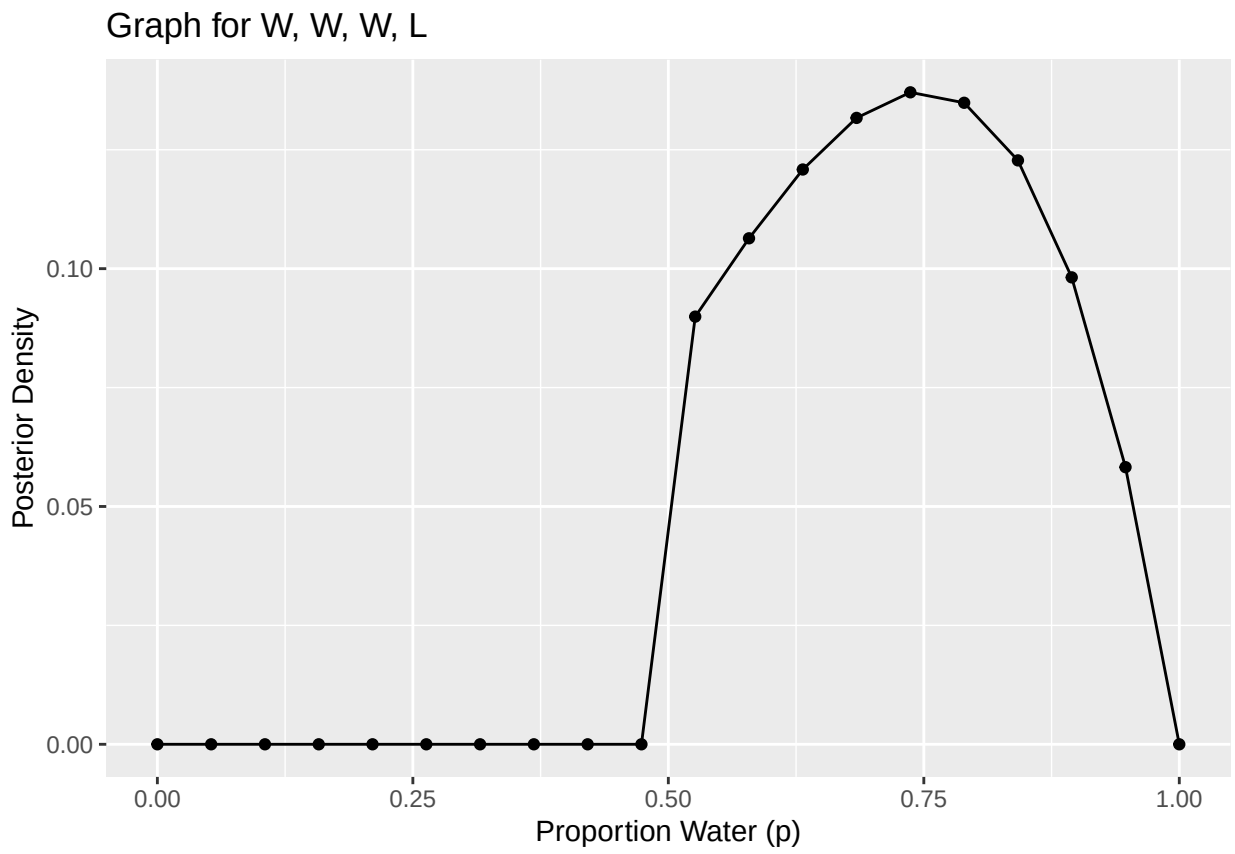
2M2. Now assume a prior for p that is equal to zero when p < 0.5 and is a positive constant when p >= 0.5. Again compute and plot the grid approximate posterior distribution for each of the sets of observations in the problem just above.

For this, we will just adjust how we define the prior and use the same code for everything else.

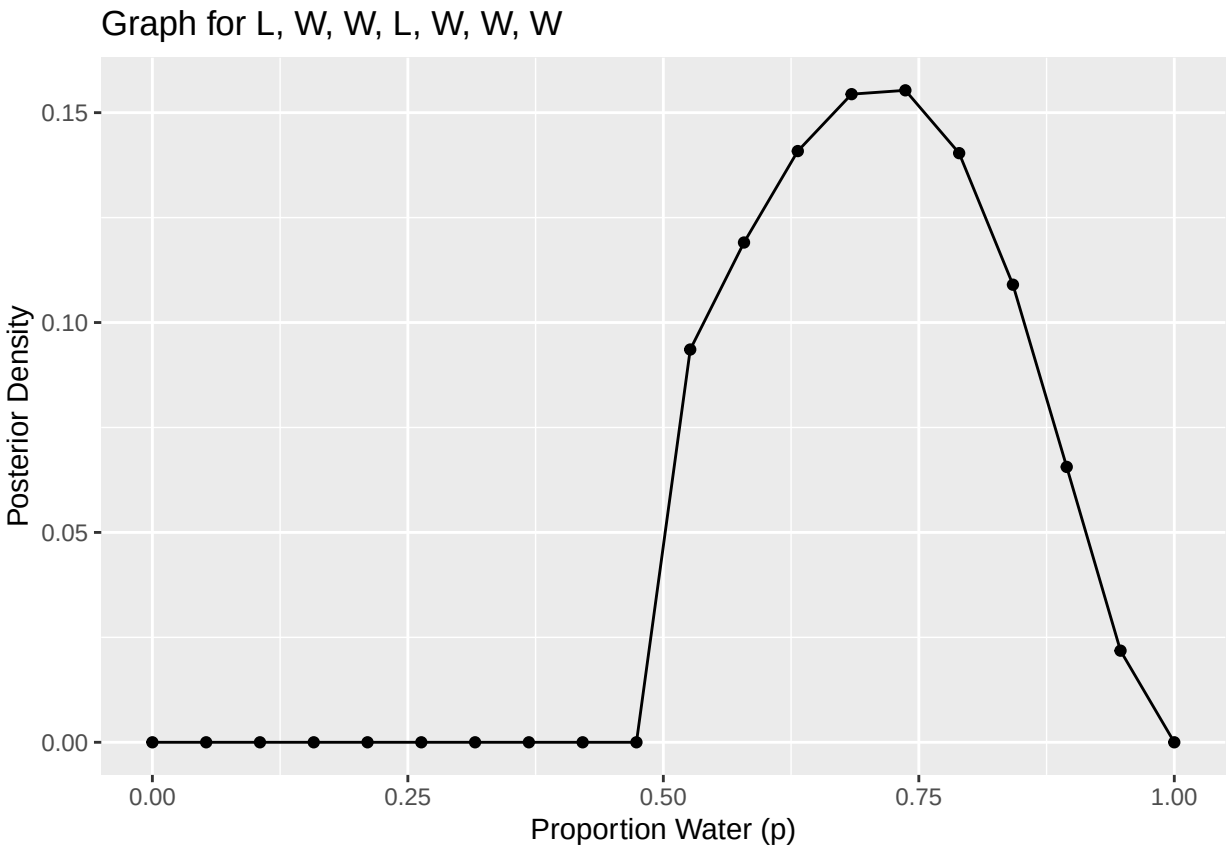
```
# number 1

plot_1a <- tibble(p_grid =  seq(from = 0, to = 1, length.out = 20),
                  prior = if_else(p_grid >= 0.5, 1, 0)) %>%
           mutate(prob_data1 = dbinom(3, size = 3, prob = p_grid)) %>%
           mutate(posterior1a = prob_data1 * prior / sum(prob_data1 * prior))

ggplot(plot_1a, aes(x = p_grid, y = posterior1a)) +
  geom_point() + geom_line() + labs(x = "Proportion Water (p)",
                                    y = "Posterior Density",
                                    title = "Graph for W, W, W")
```




```
# number 2
plot_2a <- tibble(p_grid =  seq(from = 0, to = 1, length.out = 20),
                  prior = if_else(p_grid >= 0.5, 1, 0)) %>%
           mutate(prob_data2 = dbinom(3, size = 4, prob = p_grid)) %>%
           mutate(posterior2a = prob_data2 * prior / sum(prob_data2 * prior))

ggplot(plot_2a, aes(x = p_grid, y = posterior2a)) +
  geom_point() + geom_line() + labs(x = "Proportion Water (p)",
                                    y = "Posterior Density",
                                    title = "Graph for W, W, W, L")
```




```
# number 3
# one more time... :)
plot_3a <- tibble(p_grid =  seq(from = 0, to = 1, length.out = 20),
                  prior = if_else(p_grid >= 0.5, 1, 0)) %>%
           mutate(prob_data3 = dbinom(5, size = 7, prob = p_grid)) %>%
           mutate(posterior3a = prob_data3 * prior / sum(prob_data3 * prior))

ggplot(plot_3a, aes(x = p_grid, y = posterior3a)) +
  geom_point() + geom_line() + labs(x = "Proportion Water (p)",
                                    y = "Posterior Density",
                                    title = "Graph for L, W, W, L, W, W, W")
```



2M3. Suppose there are two globes, one for Earth and one for Mars. The Earth globe is 70% covered in water. The Mars globe is 100% land. Further suppose that one of these globes—you don't know which—was tossed in the air and produced a "land" observation. Assume that each globe was equally likely to be tossed. Show that the posterior probability that the globe was the Earth, conditional on seeing "land" (Pr(Earth\|land)), is 0.23.

```
# let's start by inputting the things we know
# we'll talk in terms of land because this is the condition

# we know that 30% of the earth must be land
e_land <-  0.3

# we also know that 100% of Mars is covered in land
m_land <-  1.0

#they're equally likely to be picked so there's a 50% probability for either
p_either <- .50

# the formula we will use is (P(B|A)*P(A))/P(B)

# P(B|A) = e_land

# P(A) = p_either, this is just the likelihood we'll pick earth
pba <- e_land * p_either

# P(B) = land overall this will take a little bit more calculating
```

```
    # we have this for earth (e_land * p_either)
    # let's just do the same with mars (m_land * p_either) and add them
pb <- pba + (m_land*p_either)

# now lets complete the equation and round to two digits
round(pba / pb, 2)
```

```
## [1] 0.23
```

2M4. Suppose you have a deck with only three cards. Each card has two sides, and each side is either black or white. One card has two black sides. The second card has one black and one white side. The third card has two white sides. Now suppose all three cards are placed in a bag and shuffled. Someone reaches into the bag and pulls out a card and places it flat on a table. A black side is shown facing up, but you don't know the color of the side facing down. Show that the probability that the other side is also black is 2/3. Use the counting method (Section 2 of the chapter) to approach this problem. This means counting up the ways that each card could produce the observed data (a black side facing up on the table).

Using the logic from when we wanted to see what would happen for just one blue marble, during the updating section of the book, we will just count the "ways to produce" based on how many opportunities there are to see a black side of the card and then use this as our denominator. We will use the count of possibilities for our preferred outcome as the numerator.

thought process on ways to produce:
WHITE, WHITE = 0
[BLACK], [BLACK] = 2 *
WHITE, [BLACK] = 1

*this is the number we will use as our numerator as it what we are looking for, that the card has 2 black sides.

```
# lets be funny and put this in some code for consistency
library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select
```

```
ways <- c(0,2,1)
fractions(ways/sum(ways))
```

```
## [1]   0 2/3 1/3
```

For what we are looking at: a card with two black sides, we get a result of 2/3.

2M5. Now suppose there are four cards: B/B, B/W, W/W, and another B/B. Again suppose a card is drawn from the bag and a black side appears face up. Again calculate the probability

that the other side is black.

WHITE, WHITE = 0
[BLACK], [BLACK] = 2
[BLACK], [BLACK] = 2
WHITE, [BLACK] = 1

```
ways2 <- c(0,2,2,1)
fractions(ways2/sum(ways2))
```

```
## [1]   0 2/5 2/5 1/5
```

```
fractions(2/5 + 2/5)
```

```
## [1] 4/5
```

Now, the probability is 4/5.

2M6. Imagine that black ink is heavy, and so cards with black sides are heavier than cards with white sides. As a result, it's less likely that a card with black sides is pulled from the bag. So again assume there are three cards: B/B, B/W, and W/W. After experimenting a number of times, you conclude that for every way to pull the B/B card from the bag, there are 2 ways to pull the B/W card and 3 ways to pull the W/W card. Again suppose that a card is pulled and a black side appears face up. Show that the probability the other side is black is now 0.5. Use the counting method, as before.

We are going to use our same first ways to produce as the prior count, the second number will be the count after the experiment. we will end up multiplying these number together and then doing the same procedure as before.

WHITE, WHITE = 0 | 3
[BLACK], [BLACK] = 2 | 1 *
WHITE, [BLACK] = 1 | 2

```
prior_c <- c(0,2,1)
exp_c <- c(3,1,2)

sum_c <- sum(prior_c * exp_c)
(prior_c * exp_c)/sum_c
```

```
## [1] 0.0 0.5 0.5
```

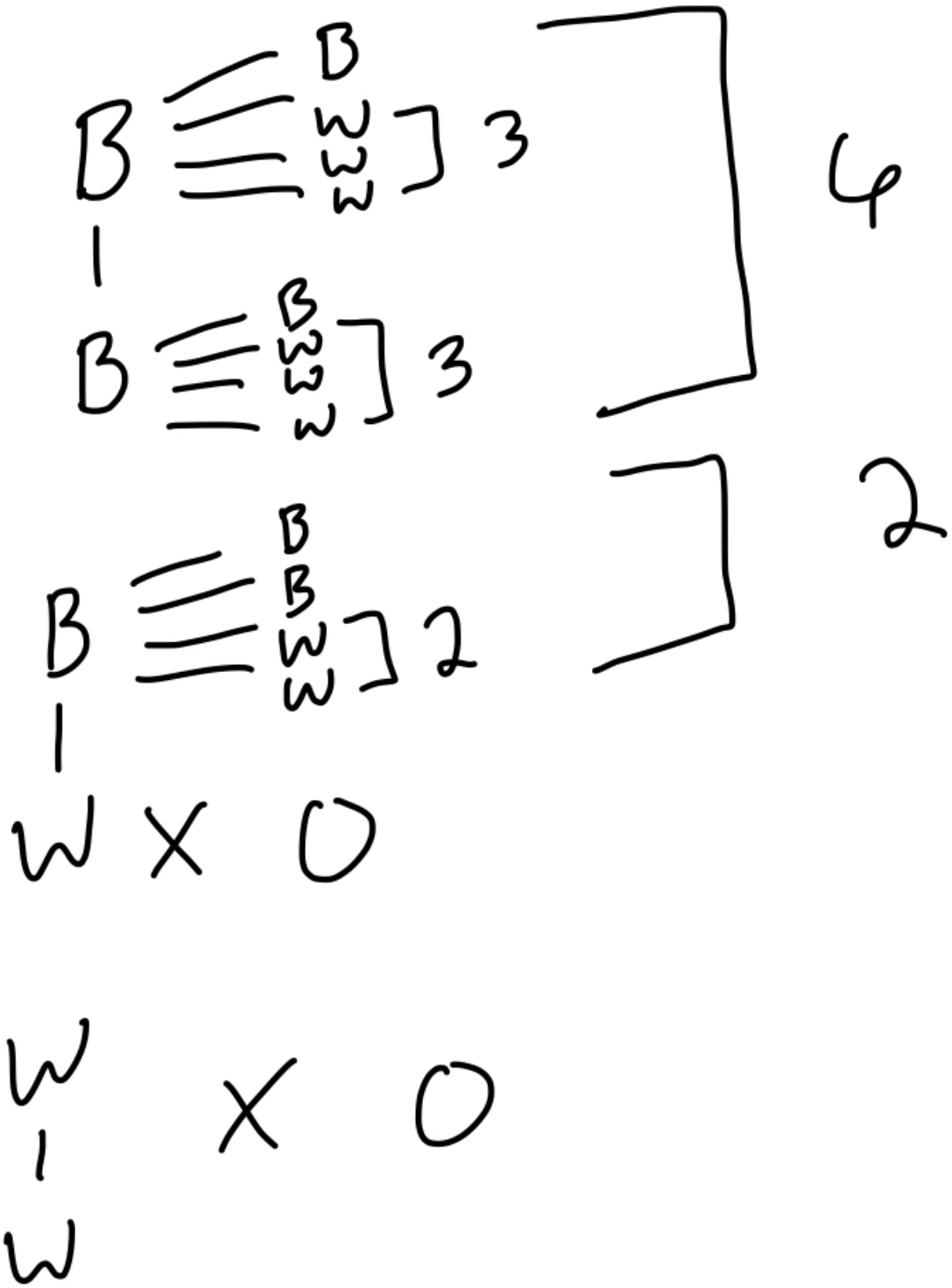
The probability the other side is black is now 0.5.

2M7. Assume again the original card problem, with a single card showing a black side face up. Before looking at the other side, we draw another card from the bag and lay it face up on the table. The face that is shown on the new card is white. Show that the probability that the first card, the one showing a black side, has black on its other side is now 0.75. Use the counting method, if you can. Hint: Treat this like the sequence of globe tosses, counting all the ways to see each observation, for each possible first card.

For this one you will start by making your paths through the garden. (See picture below).

As you can see, there are 6 possible ways from the B/B card to get to a white sided card, there are 2 possible ways from the B/W to get to the white sided card, and the W/W/ card can automatically be eliminated

B/B = 6
B/W = 2
W/W = 0




```
ways3 <- c(6,2,0)
ways3/sum(ways3)
```

```
## [1] 0.75 0.25 0.00
```

We get back that 0.75 is the probability for B/B.

I'll attempt a hard (I'm NOT crazy... so probably just the one). :)

after further reading of the syllabus I think this won't count for anything as it seems like I was supposed to do them all. Whoops. I hope you'll give me feedback on the one still at least!

2H1. Suppose there are two species of panda bear. Both are equally common in the wild and live in the same places. They look exactly alike and eat the same food, and there is yet no genetic assay capable of telling them apart. They differ however in their family sizes. Species A gives birth to twins 10% of the time, otherwise birthing a single infant. Species B births twins 20% of the time, otherwise birthing singleton infants. Assume these numbers are known with certainty, from many years of field research. Now suppose you are managing a captive panda breeding program. You have a new female panda of unknown species, and she has just given birth to twins. What is the probability that her next birth will also be twins?

```
# let's input what we know
# prob for species A
a_twins <- .10
a_single <- .90

# prob for species B

b_twins <- .20
b_single <- .80

# prob of seeing either species in the wild
see_ab <- 0.5

# now what's the probability of twins in general?
# P(twins|A) * P(A) + (P(twins|B) + P(B))
twins <- (a_twins*see_ab) + (b_twins*see_ab)

# again we will use the formula (P(B|A)*P(A))/P(B)
# B will be twins, A will be species A or B

# lets start with P(A|twins) = P(twins|A)*P(A)/P(twins)
a <- (a_twins*see_ab)/twins

# now # lets start with P(B|twins) = P(twins|B)*P(B)/P(twins)
b <- (b_twins*see_ab)/twins

# now for we have probabilities for each separate occurrence of twins
# lets multiply again so that we can see what it would be for the second time

fractions(a*a_twins + b*b_twins)
```

```
## [1] 1/6
```

The probability that her next birth will also be twins is 1/6.

Chapter 3

"Easy" Questions Apparently, we need some code first to help with these problems. Let's use it here.

```
# again let's load packages
library(rethinking)
library(tidyverse)

# we're going to rewrite this in a tibble
easy_data <-
  tibble(p_grid =  seq(from = 0, to = 1, length.out = 1000),
         prior = rep(1, 1000)) %>%
mutate(likelihood = dbinom(6, size = 9, prob = p_grid)) %>%
mutate(posterior = likelihood * prior / sum(likelihood * prior))

# I need more clarification on this set.seed function
?set.seed # it helps make your randomized samples constant
set.seed(100)

# now, from easy_data let's take a sample
# let's make this easier on ourselves later
# and create an object for our sample size

n_samples <- 1e4

# take the samples
samples <- easy_data %>%
  slice_sample(n = n_samples, weight_by = posterior, replace = TRUE)

glimpse(samples)
```

```
## Rows: 10,000
## Columns: 4
## $ p_grid     <dbl> 0.7137137, 0.3573574, 0.5985986, 0.7177177, 0.6296296, 0.46~
## $ prior      <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,~
## $ likelihood <dbl> 0.26051046, 0.04643060, 0.24993675, 0.25825703, 0.26588812,~
## $ posterior  <dbl> 0.0026077123, 0.0004647707, 0.0025018693, 0.0025851554, 0.0~
```

Now, let's begin.

3E1. How much posterior probability lies below $p = 0.2$?

```
# to do this, we find the frequency of the parameter values below 0.2
samples %>%
  filter(p_grid < 0.2) %>%
  summarize(sum = n()/n_samples)
```

```
## # A tibble: 1 x 1
##       sum
```

```
##    <dbl>
## 1 0.0004
```

0.04% of the posterior probability is below 0.2.

3E2. How much posterior probability lies above $p = 0.8$?

```
samples %>%
  filter(p_grid > 0.8) %>%
  summarize(sum = n()/n_samples)
```

```
## # A tibble: 1 x 1
##     sum
##   <dbl>
## 1 0.112
```

About 11% of the posterior probability is above 0.8.

3E3. How much posterior probability lies between $p = 0.2$ and $p = 0.8$?

```
samples %>%
  filter(p_grid > 0.2 & p_grid < 0.8) %>%
  summarize(sum = n()/n_samples)
```

```
## # A tibble: 1 x 1
##     sum
##   <dbl>
## 1 0.888
```

88.8% of the posterior probability is between 0.2 and 0.8.

3E4. 20% of the posterior probability lies below which value of p?

```
samples %>%
  summarize(`20th percentile` = quantile(p_grid, p = .2))
```

```
## # A tibble: 1 x 1
##   `20th percentile`
##               <dbl>
## 1             0.519
```

20% of the posterior probability lies below a parameter value of about 0.52.

3E5. 20% of the posterior probability lies above which value of p?

```
samples %>%
  summarize(`80th percentile` = quantile(p_grid, p = .8))
```

```
## # A tibble: 1 x 1
##   `80th percentile`
##               <dbl>
## 1             0.756
```

20% of the posterior probability lies above a parameter value of about 0.76.

3E6. Which values of p contain the narrowest interval equal to 66% of the posterior probability?

```
# to do this, we will use the HPDI
# this gives us the narrowest interval
# containing the specified probability mass

rethinking::HPDI(samples$p_grid, prob = .66)
```

```
##     |0.66     0.66|
## 0.5085085 0.7737738
```

The narrowest interval equal to 66% of the posterior probability lies between the parameters 0.51 and 0.77.

3E7. Which values of p contain 66% of the posterior probability, assuming equal posterior probability both below and above the interval?

```
rethinking::PI(samples$p_grid, prob = .66)
```

```
##       17%       83%
## 0.5025025 0.7697698
```

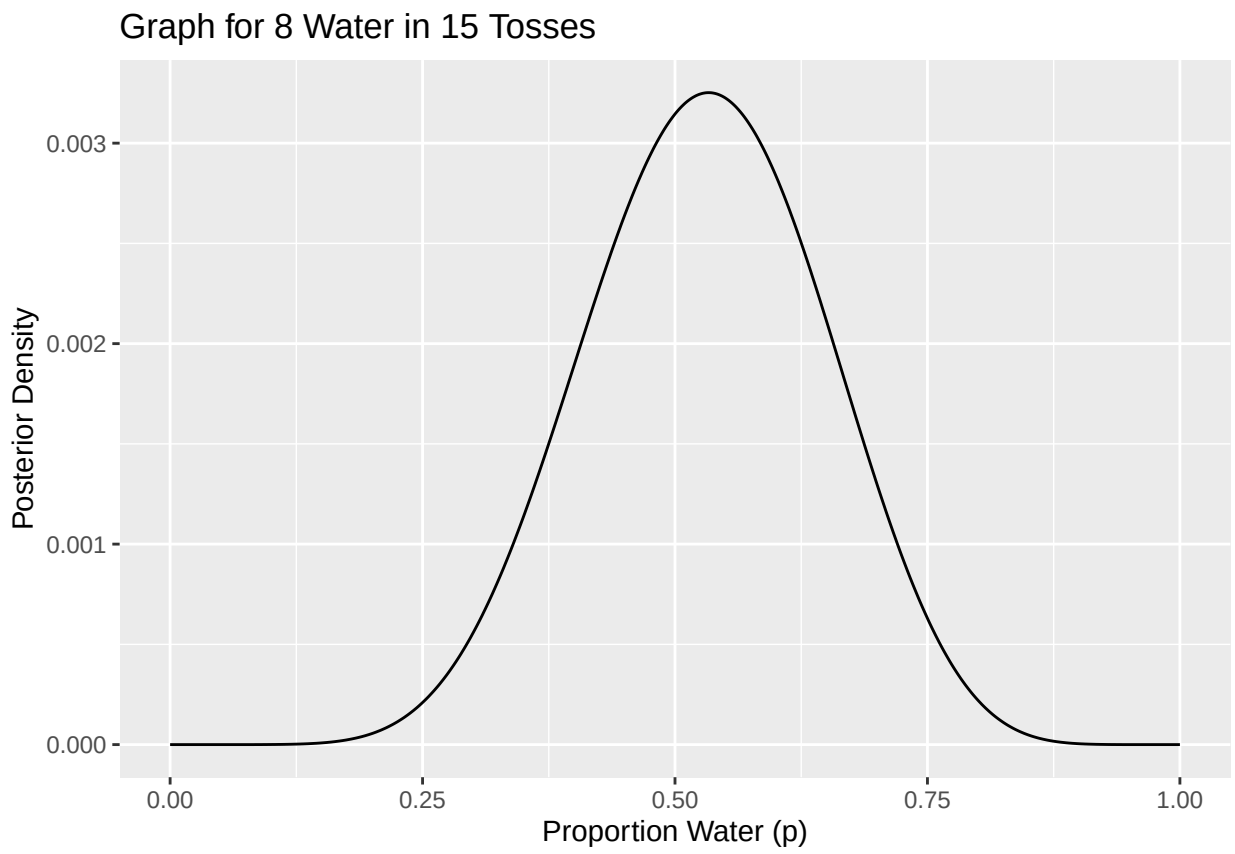
66% of the posterior probabilities is contained within the parameters 0.50 and 0.77.

Medium Questions ***3M1. Suppose the globe tossing data had turned out to be 8 water in 15 tosses. Construct the posterior distribution, using grid approximation. Use the same flat prior as before.***

```
new_data <-
  tibble(p_grid =  seq(from = 0, to = 1, length.out = 1000),
         prior = rep(1, 1000)) %>%
mutate(likelihood = dbinom(8, size = 15, prob = p_grid)) %>%
mutate(posterior = likelihood * prior / sum(likelihood * prior))

glimpse(new_data)
```

```
## Rows: 1,000
## Columns: 4
## $ p_grid     <dbl> 0.000000000, 0.001001001, 0.002002002, 0.003003003, 0.00400~
## $ prior      <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,~
## $ likelihood <dbl> 0.000000e+00, 6.441396e-21, 1.637466e-18, 4.167270e-17, 4.1~
## $ posterior  <dbl> 0.000000e+00, 1.031655e-22, 2.622568e-20, 6.674306e-19, 6.6~
```

```
# let's look at it
ggplot(new_data, aes(x = p_grid, y = posterior)) +
  geom_line() + labs(x = "Proportion Water (p)",
                                    y = "Posterior Density",
                                    title = "Graph for 8 Water in 15 Tosses")
```



3M2. Draw 10,000 samples from the grid approximation from above. Then use the samples to calculate the 90% HPDI for p.

```
# let's take 10,000 samples here
n_samp <- 10000
set.seed(100)
samples2 <- new_data %>%
  slice_sample(n = n_samp, weight_by = posterior, replace = TRUE)

# now let's calculate the 90% HDPI
rethinking::HPDI(samples2$p_grid, prob = .9)
```

```
##      |0.9      0.9|
## 0.3343343 0.7217217
```

The narrowest region with 90% of the posterior probability is within the parameters 0.33 and 0.72.

3M3. Construct a posterior predictive check for this model and data. This means simulate the distribution of samples, averaging over the posterior uncertainty in p. What is the probability

of observing 8 water in 15 tosses?

```
# first, let's simulate the distribution by averaging over the uncertainty
# do this within our samples
# let's just add this variable on to our sample data set.

w <- samples2 %>%
  mutate(prop_para =
                  rbinom(n = n_samp , size = 15, prob = samples2$p_grid))

#  now let's filter out for 8 tosses
# then see the frequency of the parameter values at that amount
w %>%
  filter(prop_para == 8) %>%
  summarize(sum = n()/n_samp )
```

```
## # A tibble: 1 x 1
##     sum
##   <dbl>
## 1 0.150
```

There's about a 15% probability of observing 8 water in 15 tosses.

3M4. Using the posterior distribution constructed from the new (8/15) data, now calculate the probability of observing 6 water in 9 tosses.

```
# we'll use the same set of samples and just change the size to 9 tosses
w2 <- samples2 %>%
  mutate(prop_para =
                  rbinom(n = 10000, size = 9, prob = samples2$p_grid))

#  now let's filter out for 6 tosses
# then see the frequency of the parameter values at that amount
w2 %>%
  filter(prop_para == 6) %>%
  summarize(sum = n()/10000)
```

```
## # A tibble: 1 x 1
##     sum
##   <dbl>
## 1 0.184
```

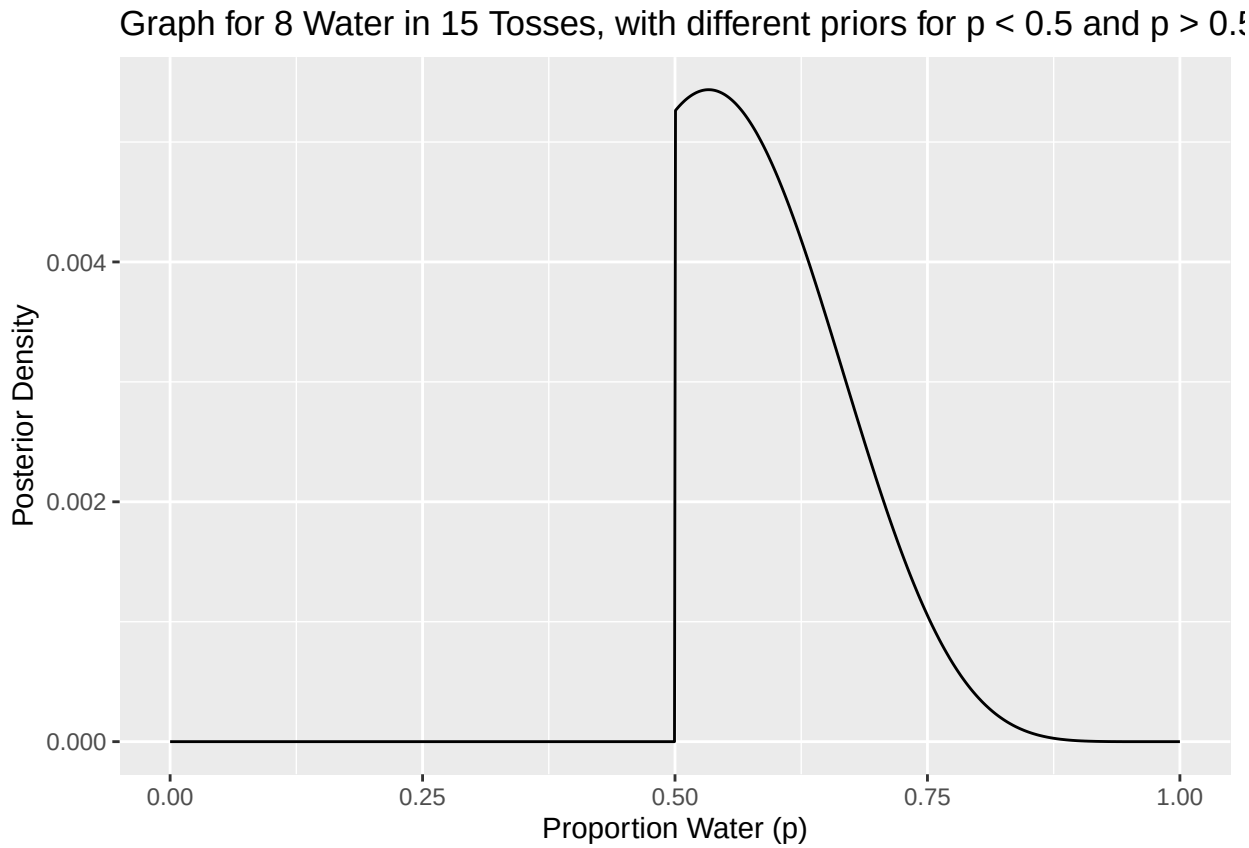
There's around an 18% probability of observing 6 water in 9 tosses.

3M5. Start over at 3M1, but now use a prior that is zero below p = 0.5 and a constant above p = 0.5. This corresponds to prior information that a majority of the Earth's surface is water. Repeat each problem above and compare the inferences. What difference does the better prior make? If it helps, compare inferences (using both priors) to the true value p = 0.7.
Part 1: creating the grid approximation

```
new_data5 <-
  tibble(p_grid =  seq(from = 0, to = 1, length.out = 1000),
         prior = if_else(p_grid >= 0.5, 1, 0)) %>%
mutate(likelihood = dbinom(8, size = 15, prob = p_grid)) %>%
mutate(posterior = likelihood * prior / sum(likelihood * prior))

glimpse(new_data)
```

```
## Rows: 1,000
## Columns: 4
## $ p_grid     <dbl> 0.000000000, 0.001001001, 0.002002002, 0.003003003, 0.00400~
## $ prior      <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,~
## $ likelihood <dbl> 0.000000e+00, 6.441396e-21, 1.637466e-18, 4.167270e-17, 4.1~
## $ posterior  <dbl> 0.000000e+00, 1.031655e-22, 2.622568e-20, 6.674306e-19, 6.6~
```

```
# let's look at it
ggplot(new_data5, aes(x = p_grid, y = posterior)) +
  geom_line() +
  labs(x = "Proportion Water (p)",
       y = "Posterior Density",
       title = paste0("Graph for 8 Water in 15 Tosses, with different priors",
       " for p < 0.5 and p > 0.5"))
```



While the other graph looks more like a normal distribution this graph sharply increases at 0.5. While its a little extreme, it makes more sense that most of the possible values be after 0.5 with our true value being at 0.7. #### Part 2 : taking the samples and HDPI

```
# let's take 10,000 samples here
set.seed(100)
samples3 <- new_data5 %>%
  slice_sample(n = n_samp, weight_by = posterior, replace = TRUE)

# now let's calculate the 90% HDPI
rethinking::HPDI(samples3$p_grid, prob = .9)
```

```
##      |0.9      0.9|
## 0.5005005 0.7097097
```

Now, we see that the narrowest region with 90% of the posterior probability is within the parameters 0.50 and 0.71. The width is a lot smaller, and a lot more realistic in comparison to the true p value of 0.7.

```
w5a <- samples3 %>%
  mutate(prop_para =
                  rbinom(n = 10000, size = 15, prob = samples3$p_grid))

#  now let's filter out for 8 tosses
# then see the frequency of the parameter values at that amount
w5a  %>%
  filter(prop_para == 8) %>%
  summarize(sum = n()/10000)
```

Part 3: creating the posterior predictive check

```
## # A tibble: 1 x 1
##     sum
##   <dbl>
## 1 0.163
```

There is now about a 16% probability of observing 8 water in 15 tosses. This is a small increase from 14%.

```
w5b <- samples3 %>%
  mutate(prop_para =
                  rbinom(n = 10000, size = 9, prob = samples3$p_grid))

#  now let's filter out for 6 tosses
# then see the frequency of the parameter values at that amount
w5b %>%
  filter(prop_para == 6) %>%
  summarize(sum = n()/10000)
```

Part 4: using 6 water and 9 tosses

```
## # A tibble: 1 x 1
##     sum
##   <dbl>
## 1 0.235
```

Now, there's around an 24% probability of observing 6 water in 9 tosses. This is a greater increase now from 18% (in relation to the change for the part before).

3M6. Suppose you want to estimate the Earth's proportion of water very precisely. Specifically, you want the 99% percentile interval of the posterior distribution of p to be only 0.05 wide.This means the distance between the upper and lower bound of the interval should be 0.05. How many times will you have to toss the globe to do this?

```
# let's try for ten
 w <- rbinom(1, size = 10, prob = 0.7)

data_ten <-
  tibble(p_grid =  seq(from = 0, to = 1, length.out = 1000),
         prior = rep(1, 1000)) %>%
mutate(likelihood = dbinom(w, size = 10, prob = p_grid)) %>%
mutate(posterior = likelihood * prior / sum(likelihood * prior))


sample_ten <- data_ten %>%
  slice_sample(n = n_samples, weight_by = posterior, replace = TRUE)

rethinking::PI(sample_ten$p_grid, prob = .99)
```

```
##        1%      100%
## 0.3003003 0.9289289
```

```
 1.0000000 - 0.6146096 #= 0.385
```

```
## [1] 0.3853904
```

```
# let's try for 100
 w100 <- rbinom(1, size = 100, prob = 0.7)

data_100 <-
  tibble(p_grid =  seq(from = 0, to = 1, length.out = 1000),
         prior = rep(1, 1000)) %>%
mutate(likelihood = dbinom(w100, size = 100, prob = p_grid)) %>%
mutate(posterior = likelihood * prior / sum(likelihood * prior))


sample_100 <- data_100 %>%
  slice_sample(n = n_samples, weight_by = posterior, replace = TRUE)

rethinking::PI(sample_100$p_grid, prob = .99)
```

```
##        1%      100%
## 0.5205205 0.7607608
```

```
0.8028078 - 0.5715716  # = 0.231
```

```
## [1] 0.2312362
```

```
# let's try for 1000

 w1000 <- rbinom(1, size = 1000, prob = 0.7)

data_1000 <-
  tibble(p_grid =  seq(from = 0, to = 1, length.out = 1000),
         prior = rep(1, 1000)) %>%
mutate(likelihood = dbinom(w1000, size = 1000, prob = p_grid)) %>%
mutate(posterior = likelihood * prior / sum(likelihood * prior))


sample_1000 <- data_1000 %>%
  slice_sample(n = n_samples, weight_by = posterior, replace = TRUE)

rethinking::PI(sample_1000$p_grid, prob = .99)
```

```
##        1%      100%
## 0.6236236 0.7007007
```

```
 0.7367417 - 0.6636587 # = 0.073
```

```
## [1] 0.073083
```

```
 # let's try for 2000

 w2000 <- rbinom(1, size = 2000, prob = 0.7)

data_2000 <-
  tibble(p_grid =  seq(from = 0, to = 1, length.out = 1000),
         prior = rep(1, 1000)) %>%
mutate(likelihood = dbinom(w2000, size = 2000, prob = p_grid)) %>%
mutate(posterior = likelihood * prior / sum(likelihood * prior))


sample_2000 <- data_2000 %>%
  slice_sample(n = n_samples, weight_by = posterior, replace = TRUE)

rethinking::PI(sample_2000$p_grid, prob = .99)
```

```
##        1%      100%
## 0.6806807 0.7317317
```

```
0.7417417  - 0.6896897
```

```
## [1] 0.052052
```

```
# = 0.052, closer but not quite...
# so maybe the answer is a little more than 2000?

 # let's try for 2050

 w2050 <- rbinom(1, size = 2050, prob = 0.7)

data_2050 <-
  tibble(p_grid =  seq(from = 0, to = 1, length.out = 1000),
         prior = rep(1, 1000)) %>%
mutate(likelihood = dbinom(w2050, size = 2050, prob = p_grid)) %>%
mutate(posterior = likelihood * prior / sum(likelihood * prior))


sample_2050 <- data_2050 %>%
  slice_sample(n = n_samples, weight_by = posterior, replace = TRUE)

rethinking::PI(sample_2050$p_grid, prob = .99)
```

```
##        1%      100%
## 0.6806807 0.7327327
```

```
0.7177177 - 0.6666667 # 0.051, CLOSER
```

```
## [1] 0.051051
```

```
 # let's try for 2200

 w2200 <- rbinom(1, size = 2200, prob = 0.7)

data_2200 <-
  tibble(p_grid =  seq(from = 0, to = 1, length.out = 1000),
         prior = rep(1, 1000)) %>%
mutate(likelihood = dbinom(w2200, size = 2200, prob = p_grid)) %>%
mutate(posterior = likelihood * prior / sum(likelihood * prior))


sample_2200 <- data_2200 %>%
  slice_sample(n = n_samples, weight_by = posterior, replace = TRUE)

rethinking::PI(sample_2200$p_grid, prob = .99)
```

```
##        1%      100%
## 0.6676677 0.7187187
```

```
0.7337337 - 0.6846847 # 0.049, too far let's go back a little bit
```

```
## [1] 0.049049
```

```
# lets try 2100
 w2100 <- rbinom(1, size = 2100, prob = 0.7)

data_2100 <-
  tibble(p_grid =  seq(from = 0, to = 1, length.out = 1000),
         prior = rep(1, 1000)) %>%
mutate(likelihood = dbinom(w2100, size = 2100, prob = p_grid)) %>%
mutate(posterior = likelihood * prior / sum(likelihood * prior))


sample_2100 <- data_2100 %>%
  slice_sample(n = n_samples, weight_by = posterior, replace = TRUE)

rethinking::PI(sample_2100$p_grid, prob = .99)
```

```
##        1%      100%
## 0.6736687 0.7237287
```

```
0.7257257 - 0.6746747 # 0.051, close again jeez
```

```
## [1] 0.051051
```

```
# lets try 2150
 w2150 <- rbinom(1, size = 2150, prob = 0.7)

data_2150 <-
  tibble(p_grid =  seq(from = 0, to = 1, length.out = 1000),
         prior = rep(1, 1000)) %>%
mutate(likelihood = dbinom(w2150, size = 2150, prob = p_grid)) %>%
mutate(posterior = likelihood * prior / sum(likelihood * prior))


sample_2150 <- data_2150 %>%
  slice_sample(n = n_samples, weight_by = posterior, replace = TRUE)

rethinking::PI(sample_2150$p_grid, prob = .99)
```

```
##        1%      100%
## 0.6726727 0.7247247
```

```
0.7127127 - 0.6626627 # 0.05005
```

```
## [1] 0.05005
```

It would take somewhere around 2150 tosses.