Week 1 Homework

Randy Boyes

09/01/2022

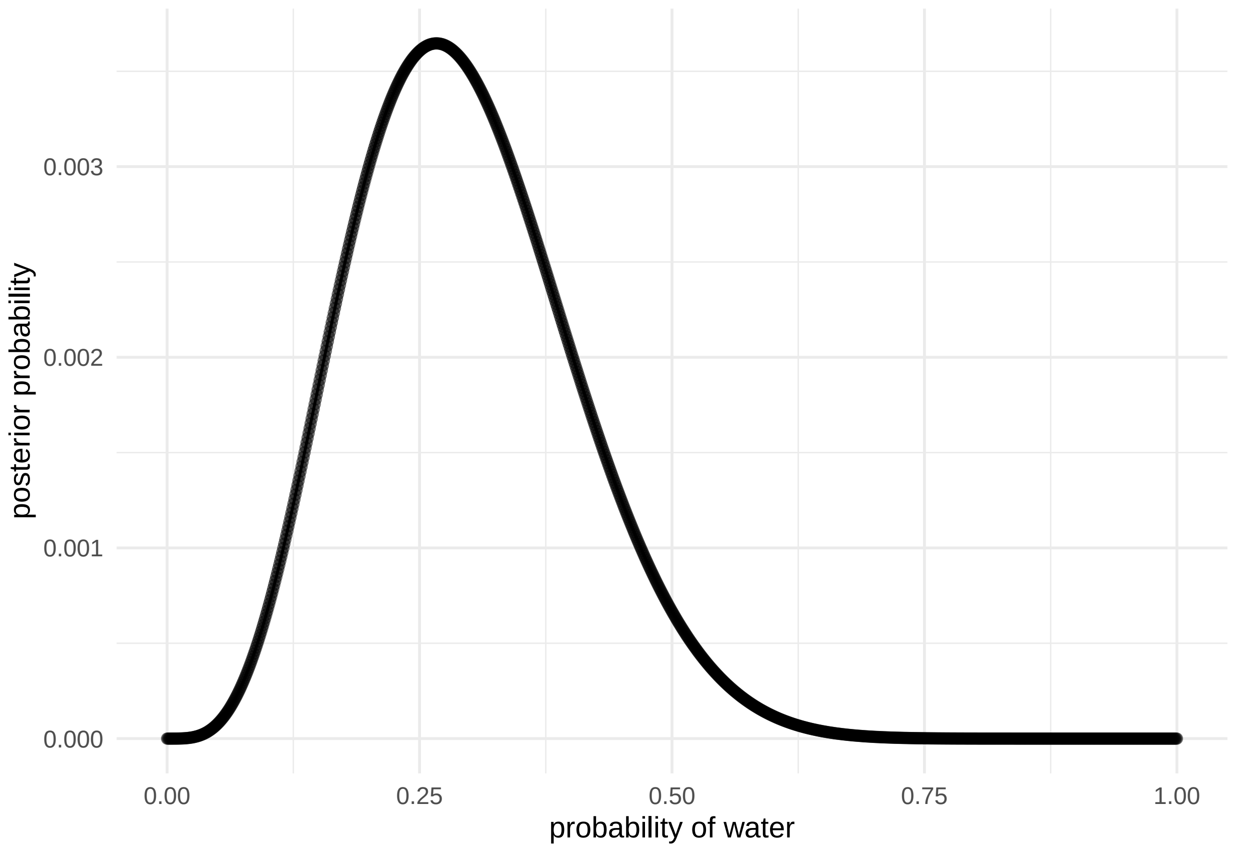
Problem 1: Suppose the globe tossing data (Chapter 2) had turned out to be 4 water and 11 land. Construct the posterior distribution, using grid approximation. Use the same flat prior as in the book.

Start by defining a couple of functions. First, one to generalize the code from R 2.3, with a default to uniform prior. Second function takes the result of the first and plots it using ggplot.

```r
create_posterior <- function(water, land,
                             grid_resolution,
                             prior = rep(1, grid_resolution)){

  p_grid <- seq(from=0, to=1, length.out = grid_resolution)
  likelihood <- dbinom(water, size = (water + land), prob = p_grid)
  unstd.posterior <- likelihood * prior

  return(data.frame(x = p_grid, y = unstd.posterior / sum(unstd.posterior)))
}

water_plot <- function(df){
  ggplot(data = df, aes(x = x, y = y)) +
  geom_line() +
  geom_point(aes(alpha = 0.25)) +
  theme_minimal() +
  theme(legend.position = "none") +
  ylab("posterior probability") +
  xlab("probability of water")
}
```

Use the defined functions to create a posterior distribution from the data in Problem 1 (4 water, 11 land). Grid resolution is set to 1000 and appears to give sufficient detail.

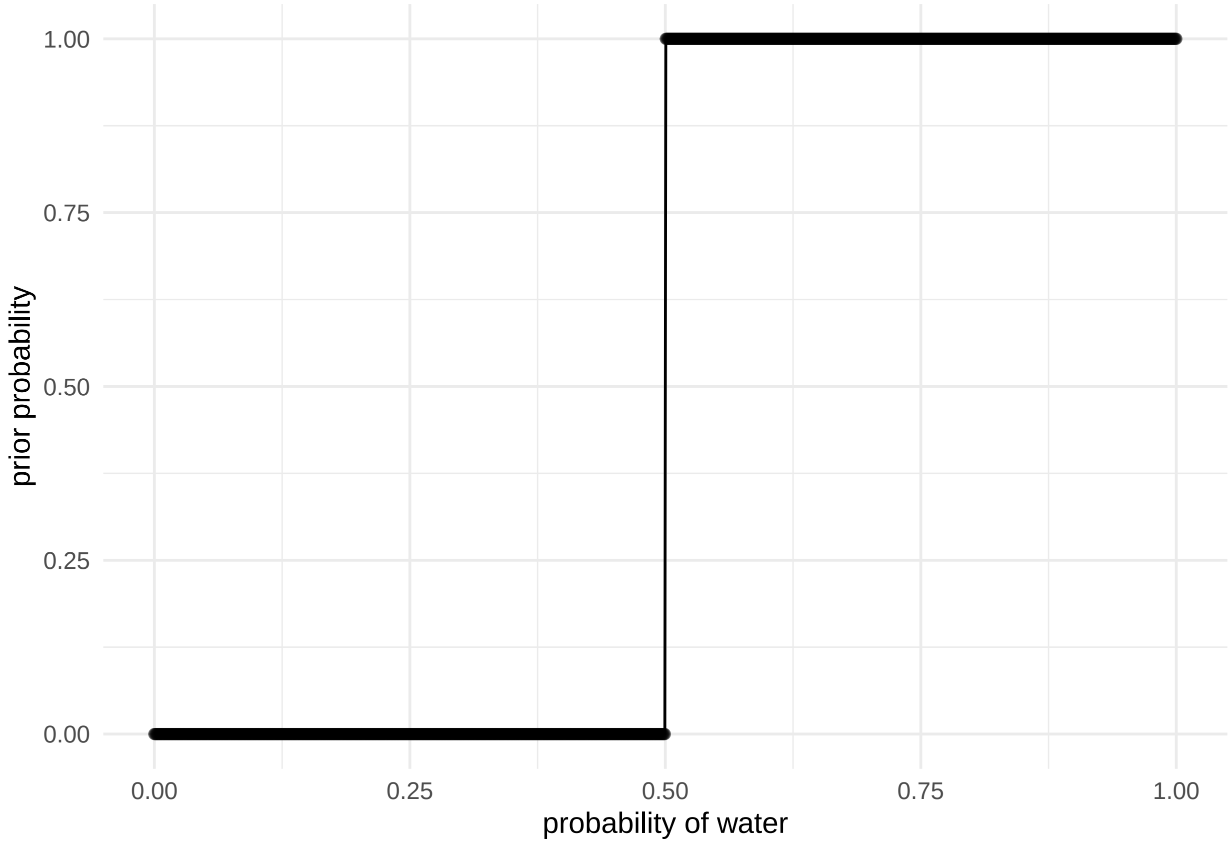
```r
create_posterior(water = 4, land = 11, grid_resolution = 1000) %>%
  water_plot()
```


posterior probability
0.003
0.002
0.001
0.000
0.00
0.25
0.50
0.75
1.00
probability of water

Problem 2: Now suppose the data are 4 water and 2 land. Compute the posterior again, but this time 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.

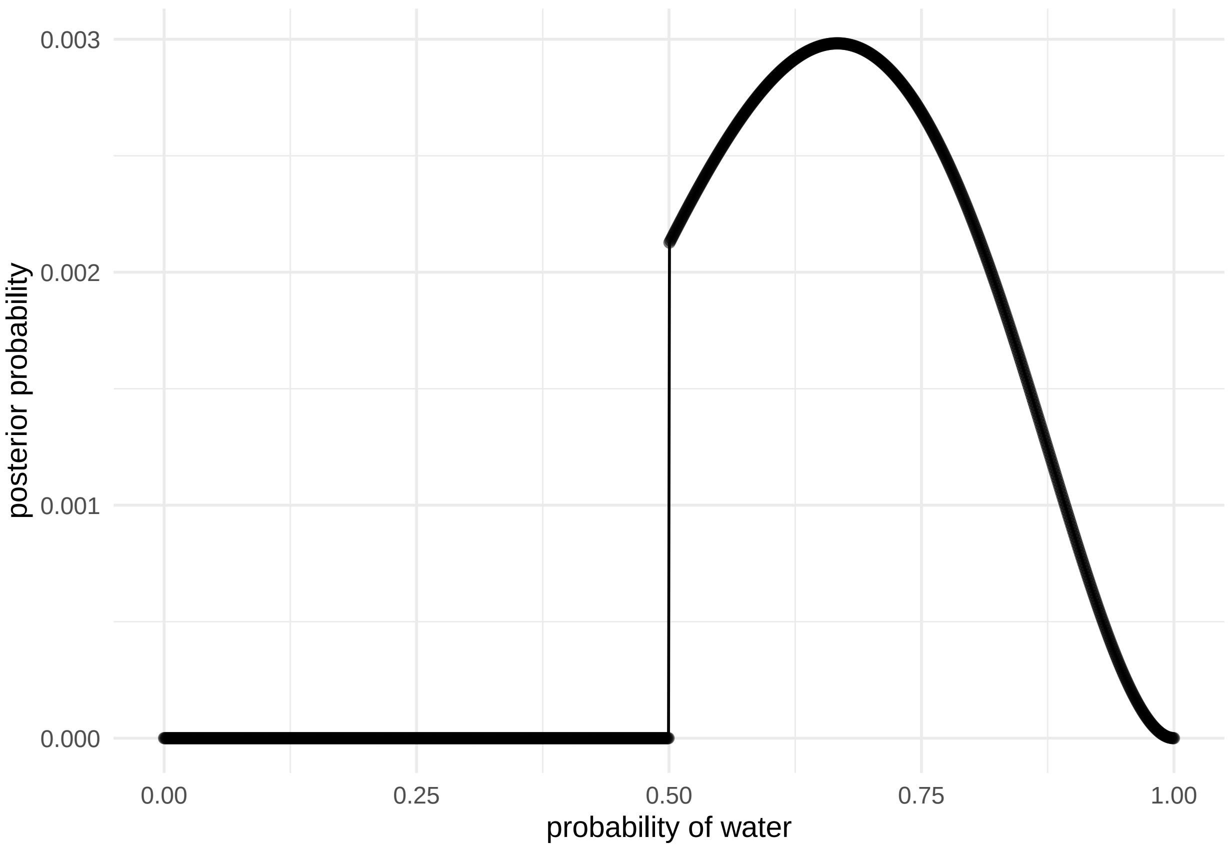
We can easily define such a prior, but plot to make sure:

```
data.frame(x = seq(from=0, to=1, length.out = 1000),
           y = rep(c(0, 1), each = 500)) %>%
  water_plot() + ylab("prior probability")
```



Pass this prior to the original function:

```
create_posterior(4, 2, 1000, prior = rep(c(0, 1), each = 500)) %>%
  water_plot()
```


0.003
0.002
0.001
0.000
posterior probability
0.00
0.25
0.50
0.75
1.00
probability of water

Problem 3: For the posterior distribution from 2, compute 89% percentile and HPDI intervals. Compare the widths of these intervals. Which is wider? Why? If you had only the information in the interval, what might you misunderstand about the shape of the posterior distribution?

Use `sample` to extract samples from the posterior distribution, then use `quantile` to get the percentile interval and `rethinking::HDPI()` to get the HDPI interval.

```r
q2 <- create_posterior(4, 2, 1000, prior = rep(c(0, 1), each = 500))

samples <- sample(q2$x,
                  prob = q2$y,
                  size = 1e6,
                  replace = TRUE)

percentile <- quantile(samples, c(0.055, 0.945))
hdpi <- HPDI(samples)
```

```r
gt::gt(data.frame(Method = c("HPDI", "Percentile"),
                  Low = c(hdpi[1], percentile[1]),
                  High = c(hdpi[2], percentile[2]),
                  Width = c(hdpi[2] - hdpi[1], percentile[2] - percentile[1])))
```

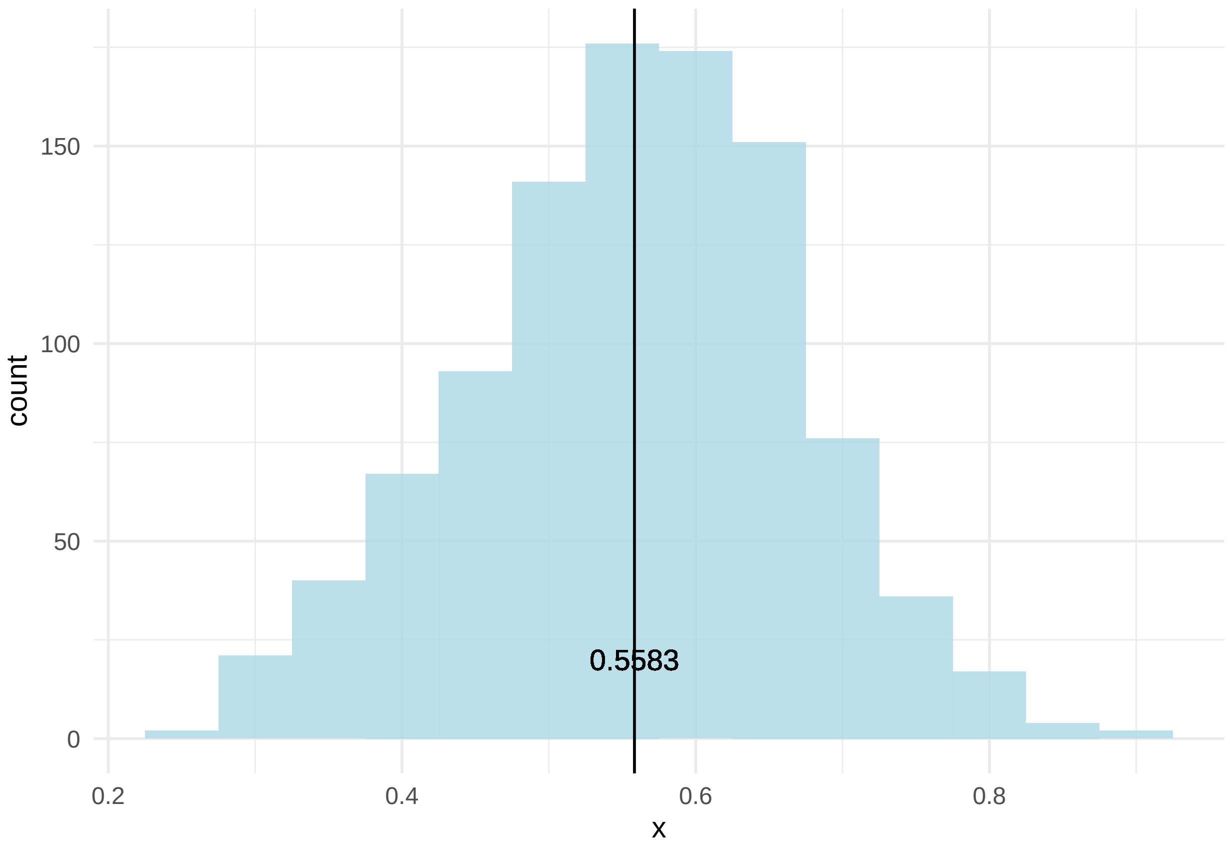
Method	Low	High	Width
HPDI	0.5005005	0.8408408	0.3403403
Percentile	0.5245245	0.8788789	0.3543544

As would be expected, the HDPI interval is narrower. This should be true in general due to the definition of a HPDI (The HPDI is the narrowest interval containing the specified probability mass). With only the information of either interval, we miss the "cliff" present in the distribution. The interval does not convey that values below 0.50 are not actually possible (from the perspective of the model).

4. OPTIONAL CHALLENGE. Suppose there is bias in sampling so that Land is more likely than Water to be recorded. Specifically, assume that 1-in-5 (20%) of Water samples are accidentally recorded instead as "Land". First, write a generative simulation of this sampling process. Assuming the true proportion of Water is 0.70, what proportion does your simulation tend to produce instead? Second, using a simulated sample of 20 tosses, compute the unbiased posterior distribution of the true proportion of water.

```r
biased_sample <- function(true_proportion = 0.7, land_bias = 0.2, samples = 20){
  n_water <- rbinom(1, samples, true_proportion)
  n_bias <- rbinom(1, n_water, land_bias)
  return((n_water - n_bias)/samples)
}
```

```r
draws <- purrr::map(1:1000, ~biased_sample(true_proportion = 0.7, land_bias = 0.2, samples = 20)) %>%
  reduce(rbind.data.frame) %>%
  magrittr::set_colnames("x")

ggplot(draws) +
  geom_histogram(aes(x = x), binwidth = 0.05, alpha = 0.8, fill = "light blue") +
  geom_vline(xintercept = mean(draws$x)) +
  geom_text(x = mean(draws$x), y = 20, label = mean(draws$x), nudge_x = 5) +
  theme_minimal()
```



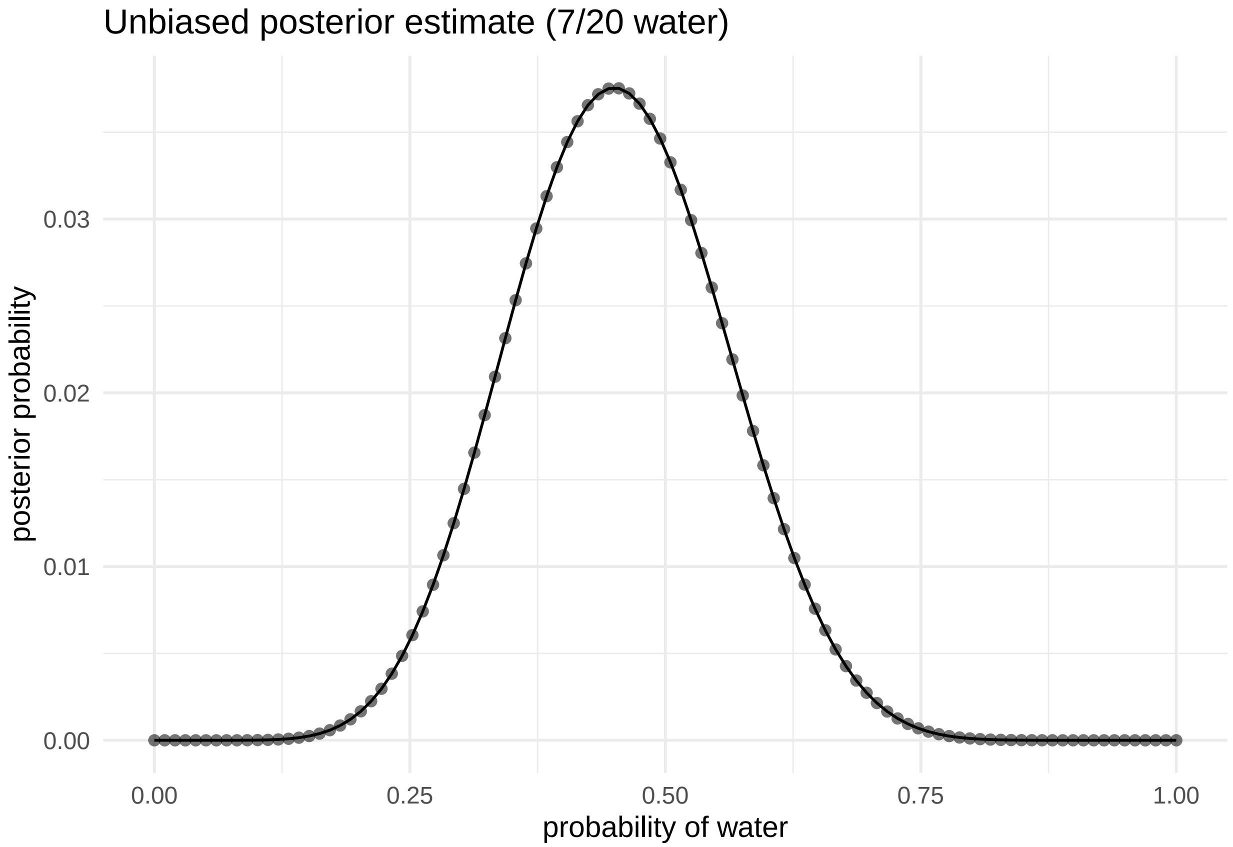
The biased sampling process tends to produce estimates of approximately 0.56 when the true proportion is 0.70.

```
water = biased_sample() * 20
land = 20 - water
```

First attempt: adjust the observed number of water observations using the known bias. This method requires rounding the adjusted number to the nearest int to get a result from dbinom, introducing some inaccuracy. Need some way to combine the first and second random processes.

```
create_posterior_biased <- function(water, land,
                                    grid_resolution,
                                    land_bias = 0.2,
                                    prior = rep(1, grid_resolution)){

  p_grid <- seq(from=0, to=1, length.out = grid_resolution)

  likelihood <- dbinom(round(water/(1-land_bias), 0),
                       size = (water + land), prob = p_grid)

  unstd.posterior <- likelihood * prior

  return(data.frame(x = p_grid, y = unstd.posterior / sum(unstd.posterior)))
}
```

```
create_posterior_biased(water, land, 100) %>%
  water_plot() + labs(title = paste0("Unbiased posterior estimate (", water, "/20 water)"))
```


Unbiased posterior estimate (7/20 water)
posterior probability
0.03
0.02
0.01
0.00
0.00
0.25
0.50
0.75
1.00
probability of water