STAT 415 Lab 1

TYPE YOUR NAME HERE

Introduction

This lab will walk you through some of the code for performing a simulation like the one in Handout 1, and a spreadsheet analysis like the one in Handout 2. You will also start to explore the influence of changing priors.

Important note regarding code: We will cover code for performing and summarizing Bayesian analyses in much more detail throughout the course. For this introductory lab I have tried to keep the code as simple and bare bones as possible, especially with respect to plotting. There are certainly better ways to code, and better plots that could be made. You are welcome to jazz it up (or tidyveryse it up) if you want, but it's not required. The main goal for this lab is to understand the basic process.

Important note regarding the simulation process: The particular simulation process we use here (and in Handout 1) is used mainly to illustrate ideas in a fairly concrete way. While the logic of the simulation process is correct, in practice we would run into some technical difficulties (that we'll discuss later). Simulation does play an essential role in Bayesian statistics, but in practice much more efficient and sophisticated simulation methods are required. We'll study simulation and the role it plays in much more detail throughout the course. Again, the main goal for this lab is to understand the basic process.

Instructions

This RMarkdown file provides a template for you to fill in. Read the file from start to finish, completing the parts as indicated. Some code is provided for you. Be sure to run this code, and make sure you understand what it's doing. Some blank "code chunks" are provided; you are welcome to add more (CTRL-ALT-I) as needed. There are also a few places where you should type text responses. Feel free to add additional text responses as necessary.

You can run individual code chunks using the play button. You can use objects defined in one chunk in others. Just keep in mind that chunks are evaluated in order. So if you call something x in one chunk, but redefine x as something else in another chunk, it's essential that you evaluate the chunks in the proper order.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. When you are finished

- click **Knit** and check to make sure that you have no errors and everything runs properly. (Fix any problems and redo this step until it works.)
- Make sure your type your name(s) at the top of the notebook where it says "Type your name(s) here". If you worked in a team, you will submit a single notebook with both names; make sure both names are included
- Submit your completed files in Canvas.

Estimating a population proportion - coarse grid simulation

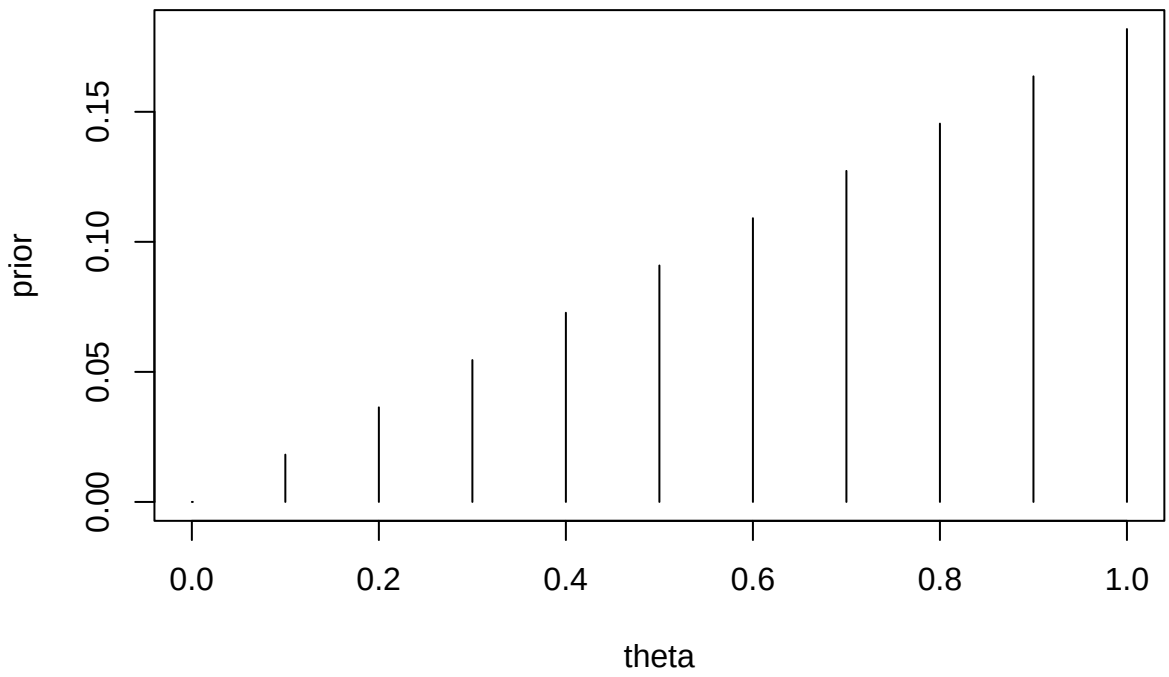
Suppose you're interested in θ, the population proportion of Cal Poly students whose dominant eye (right or left) is the same same as their dominant hand (right or left). We will use data from a sample of 36 Cal Poly students; you can assume this represents a randomly selected sample.

We'll start by assuming the only possible values of θ are 0, 0.1, ..., 0.9, 1.

```
theta = seq(0, 1, 0.1)
```

Suppose you initially think that most people have the same dominant eye and hand. So you put greater plausibility on values of θ near 1 and less plausibility of values of θ near 0. In particular, suppose you place 0 "units" of plausibility on 0, 1 unit on 0.1, 2 units on 0.2, and so on, with 10 units on 1. The following defines the corresponding prior distribution, rescaling the plausibility values so that they add up to 1 while maintaining the proper ratios.

```
units = 0:10

prior = units / sum(units)

plot(theta, prior, type = "h")
```



Now we'll simulate values of θ according to the prior distribution, using the `sample` function with the `prob` option. (Note: the `prob` option will automatically rescale values so they add up to 1, so we could have used `prob = units`.)

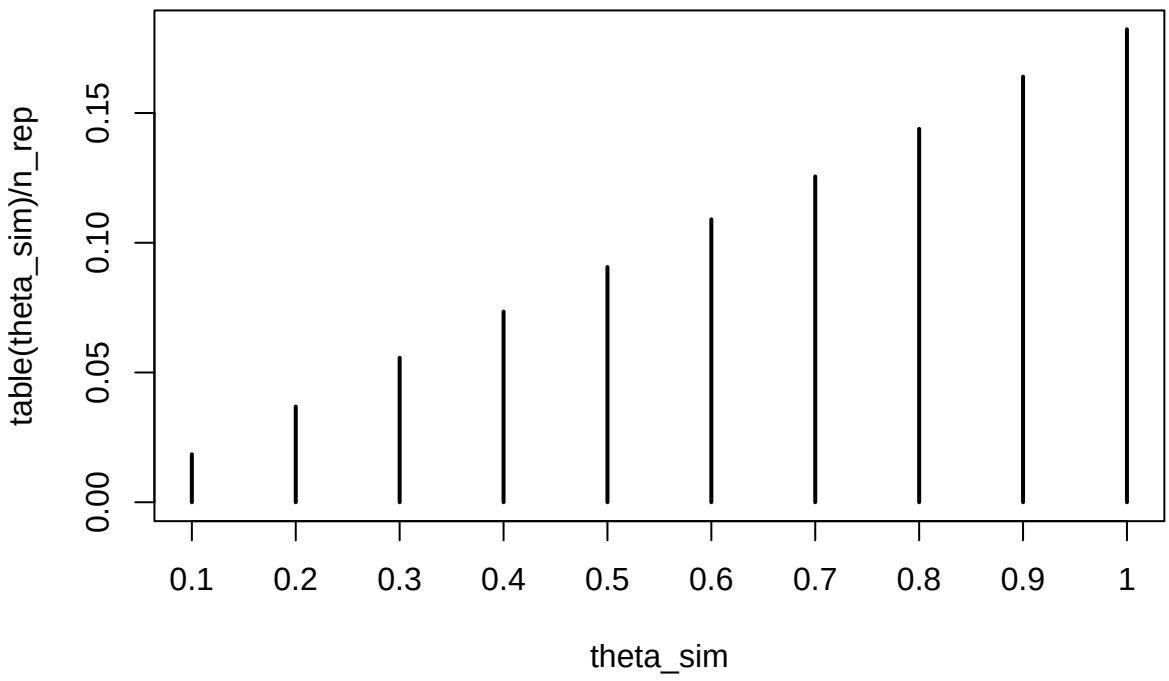
```r
n_rep = 100000

theta_sim = sample(theta, n_rep, replace = TRUE, prob = prior)
```

We table and plot the simulated values of θ just to check that they follow (approximately) the prior distribution. (The `table` function gives a quick way of counting values.)

```r
kable(table(theta_sim) / n_rep,
      digits = 4) # kable just adds nicer formatting to the table
```

theta_sim	Freq
0.1	0.0185
0.2	0.0369
0.3	0.0557
0.4	0.0735
0.5	0.0907
0.6	0.1090
0.7	0.1256
0.8	0.1439
0.9	0.1640
1	0.1822

```r
plot(table(theta_sim) / n_rep, type = "h")
```



We will use data from a sample of 36 Cal Poly students. Before looking at the sample data, we consider what might happen, keeping in mind our initial assessment of plausibility (so values of θ near 1 get more "weight" than values of θ near 0). For each simulated value of θ, we simulate a sample of size 36 and count the number of successes (Y). For example, if $\theta = 0.8$ we simulate a value Y from a Binomial(36, 0.8) distribution; if $\theta = 0.9$ we simulate a value Y from a Binomial(36, 0.9) distribution.

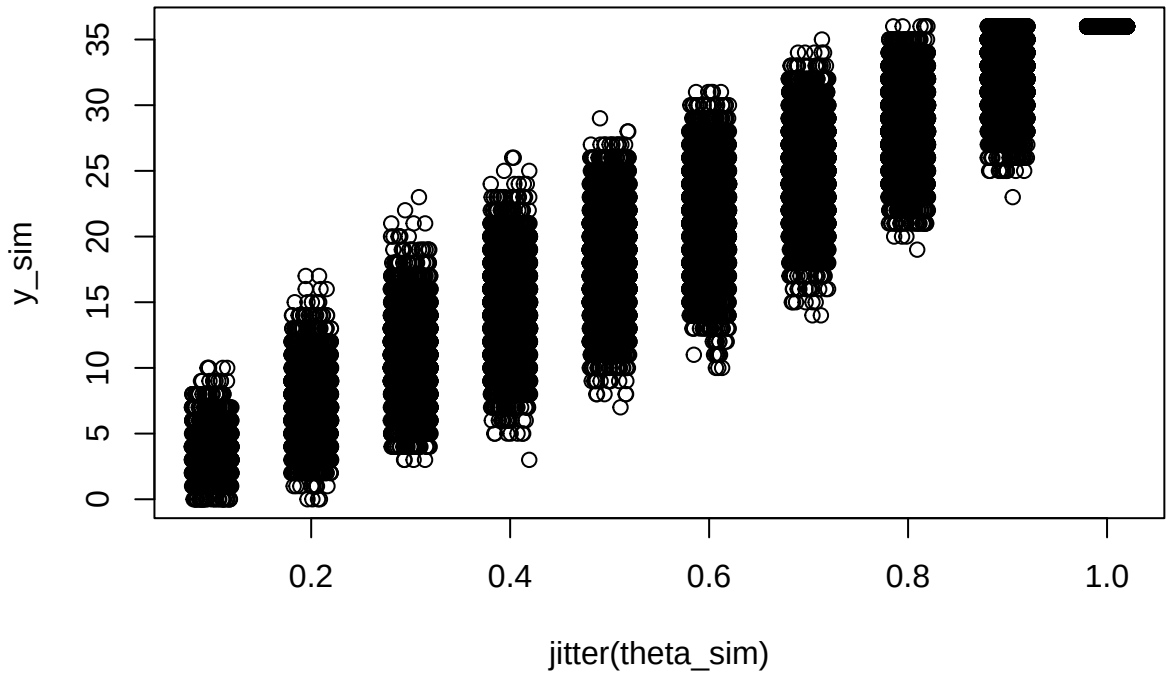
Note: `rbinom(1, 36, 0.8)` simulates a 1 value from a Binomial(36, 0.8) distribution; `rbinom(2, 36, 0.8)` simulates 2 values from a Binomial(36, 0.8) distribution; etc. But we want to simulate values from different Binomial distributions, depending on the value of θ. Fortunately, R is "vectorized". For example, if `theta` has two elements, (0.8, 0.9), then `rbinom(2, 36, theta)` will output two elements: the first element will be simulated from a Binomial(36, 0.8) distribution, and the second element will be simulated from a Binomial(36, 0.9) distribution.

```
n = 36

y_sim = rbinom(n_rep, n, theta_sim)
```

(The marginal distribution of the Y values is called the "prior predictive distribution". We will see uses for it soon, but we'll skip it for now.)

Now we plot the simulated (θ, Y) pairs. (The θ values have been "jittered" or "wiggled" a little in the plot so they don't coincide.)

```
plot(jitter(theta_sim), y_sim)
```



Remember, all of the simulated samples are hypothetical.

Now we observe sample data. In the sample of 36 students, 29 students have the same dominant right eye and right hand. (This is based on my STAT 217 class in Fall 2020, but you can assume it's a random sample.)

Consider only the simulated samples for which the sample count Y is equal to the observed count 29.

```
y_obs = 29

# put the simulated (theta, Y) pairs together in a data frame
sim = data.frame(theta_sim, y_sim)

# select only the rows where simulated Y equals observed y
# we could also get the theta values using `theta_sim[y_sim == y_obs]`
sim_given_y_obs = sim %>%
  filter(y_sim == y_obs)

# display a few rows
head(sim_given_y_obs)
```

```
##   theta_sim y_sim
## 1       0.8    29
## 2       0.8    29
## 3       0.7    29
## 4       0.8    29
## 5       0.8    29
## 6       0.9    29
```

Now we summarize the θ values for only the samples with a count equal to the observed count.

```
table(sim_given_y_obs$theta_sim)
```

```
##
##  0.5  0.6  0.7  0.8  0.9
##    1   50  713 2364  656
```

To approximate the distribution, we want to divide only by the number of repetitions that resulted in a count equal to the observed count.

```
n_rep_given_y_obs = sum(y_sim == y_obs)

n_rep_given_y_obs
```

```
## [1] 3784
```

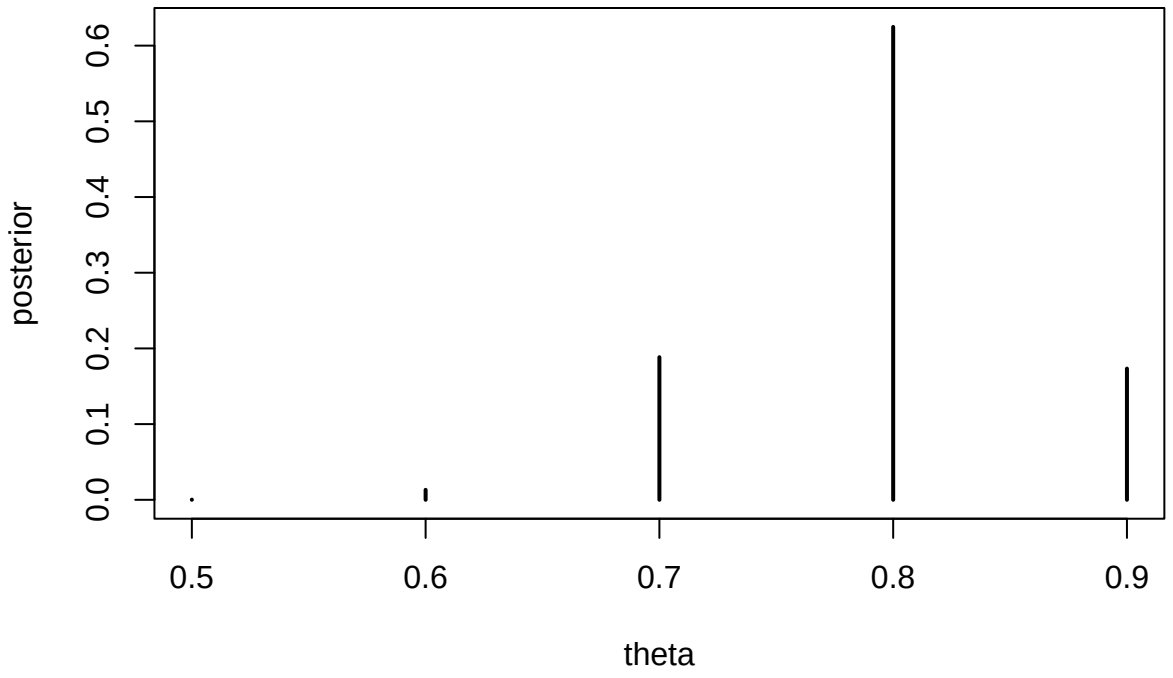
The table displays the approximate posterior distribution

```
kable(table(sim_given_y_obs$theta_sim) / n_rep_given_y_obs,
      digits = 4)
```

Var1	Freq
0.5	0.0003
0.6	0.0132
0.7	0.1884
0.8	0.6247
0.9	0.1734

The following plot displays the (approximate) posterior distribution.

```
plot(table(sim_given_y_obs$theta_sim) / n_rep_given_y_obs,
     type = "h",
     xlab = "theta",
     ylab = "posterior")
```



We often plot the prior and posterior distributions on the same plot. We won't do that yet. But do take a minute to compare the prior and posterior distributions.

Exercise. Write a sentence or two comparing the prior and posterior distributions. How has our assessment of plausibility changed after observing the sample data? Does this make sense?

TYPE YOUR RESPONSE HERE.

Estimating a population proportion - coarse grid spreadsheet

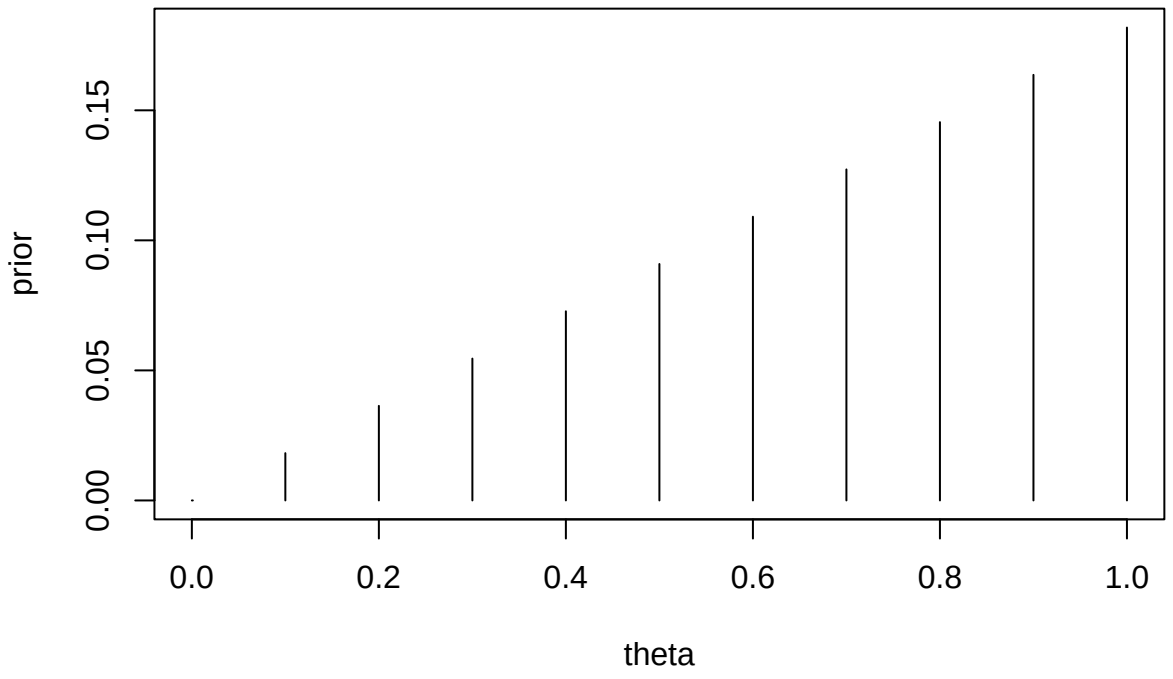
Continuing the previous section, we'll now use a spreadsheet/table approach to find the posterior distribution, as we did in Handout 2 (see Example 2.1 and solutions.)

Again, we'll assume that θ only takes values 0, 1, ..., 0.9, 1.

```
theta = seq(0, 1, 0.1)
```

We'll use the same prior as before (but we define it again here so that this section is self contained).

```
units = 0:10

prior = units / sum(units)

plot(theta, prior, type = "h")
```



The possible values of θ and the prior make up the first two columns of our spreadsheet.

Now we observe sample data (same as before). In the sample of 36 students, 29 students have the same dominant right eye and right hand.

Remember, we want to find the likelihood of observing a sample count of 29 in a sample of size 36 for each possible value of θ. If Y is the number of successes in the sample, then for a given θ the sample count Y follows a Binomial(36, θ) distribution, so we can find the likelihood of 29 successes using `dbinom(29, 36, theta)`. For example, we can find the likelihood of a sample count of 29 when $\theta = 0.8$ using `dbinom(29, 36, 0.8)`.

```
dbinom(29, 36, 0.8)
```

```
## [1] 0.1653428
```

We want to evaluate `dbinom(29, 36, theta)` for each possible value of θ. Again, fortunately, R is "vectorized". For example, if `theta` has two elements, (0.8, 0.9), then `dbinom(29, 36, theta)` will output two elements: the first element will be the value of `dbinom(29, 36, 0.8)`, and the second element will be the value of `dbinom(29, 36, 0.9)`.

```
n = 36

y_obs = 29

likelihood = dbinom(y_obs, n, theta)
```

Let's see what our spreadsheet looks like so far. We put the columns together, add a total row, and display.

```
data.frame(theta,
           prior,
           likelihood) %>%
  adorn_totals("row") %>%
  kable(digits = 6)
```

theta	prior	likelihood
0	0.000000	0.000000
0.1	0.018182	0.000000
0.2	0.036364	0.000000
0.3	0.054545	0.000000
0.4	0.072727	0.000001
0.5	0.090909	0.000121
0.6	0.109091	0.005039
0.7	0.127273	0.058784
0.8	0.145455	0.165343
0.9	0.163636	0.039319
1	0.181818	0.000000
Total	1.000000	0.268607

Remember, the likelihood column doesn't need to add up to anything in particular.

Note we want to compute the posterior distribution which revises our assessment of plausibility after observing the sample data. The key is: **posterior is proportional to the product of prior and likelihood.** For each value of θ, we compute the product of prior and likelihood. In the spreadsheet we go row-by-row multiplying the values in the prior and likelihood columns.

```
product = prior * likelihood
```

Let's see what the table looks like with the product column.

```
data.frame(theta,
           prior,
           likelihood,
           product) %>%
  adorn_totals("row") %>%
  kable(digits = 6)
```

theta	prior	likelihood	product
0	0.000000	0.000000	0.000000
0.1	0.018182	0.000000	0.000000
0.2	0.036364	0.000000	0.000000
0.3	0.054545	0.000000	0.000000
0.4	0.072727	0.000001	0.000000
0.5	0.090909	0.000121	0.000011
0.6	0.109091	0.005039	0.000550
0.7	0.127273	0.058784	0.007482
0.8	0.145455	0.165343	0.024050
0.9	0.163636	0.039319	0.006434
1	0.181818	0.000000	0.000000
Total	1.000000	0.268607	0.038526

The product column gives us the relative ratios. For example, the product column tells us that the posterior plausibility of 0.8 is 3.75 times greater than the posterior plausibility of 0.9 (3.75 = 0.024 / 0.064. Now we simply need to rescale these values — by dividing by the sum of the product column — to obtain posterior plausibilities in the proper ratios that add up to one. For example, the posterior probability of 0.624 for 0.8 is 0.024 (the value in the 0.8 row of the product column) divided by 0.0385 (the sum of the product column).

```
posterior = product / sum(product)
```

Here is the completed table.

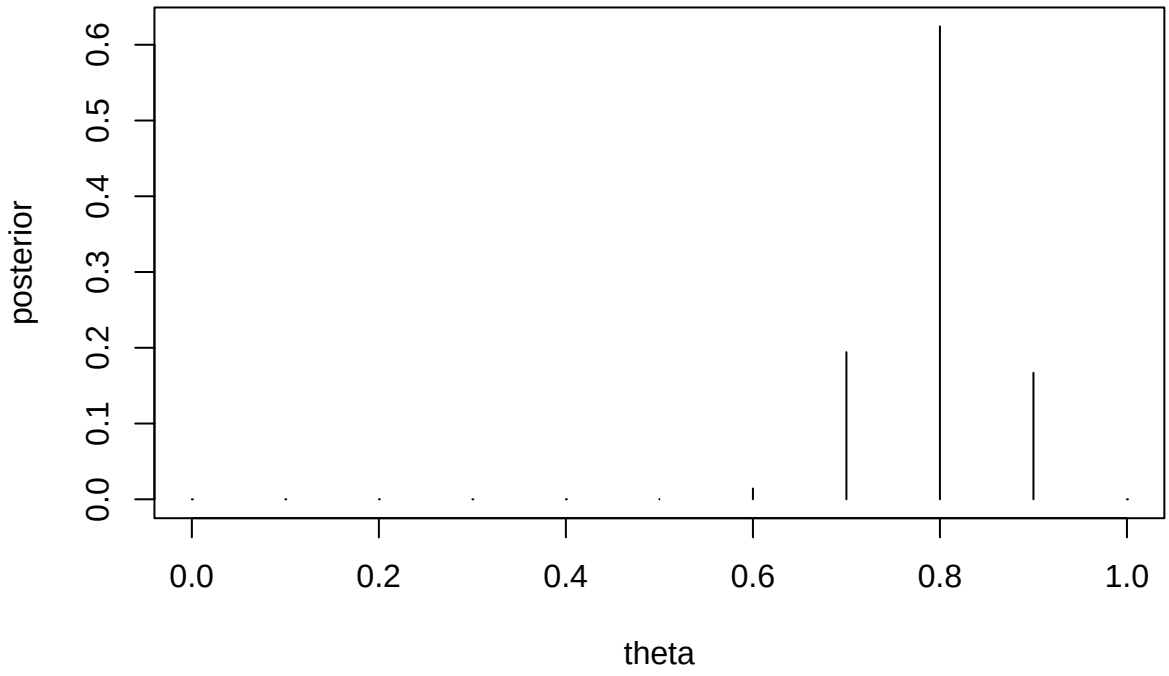
```
data.frame(theta,
           prior,
           likelihood,
           product,
           posterior) %>%
  adorn_totals("row") %>%
  kable(digits = 6)
```

theta	prior	likelihood	product	posterior
0	0.000000	0.000000	0.000000	0.000000
0.1	0.018182	0.000000	0.000000	0.000000
0.2	0.036364	0.000000	0.000000	0.000000
0.3	0.054545	0.000000	0.000000	0.000000
0.4	0.072727	0.000001	0.000000	0.000001
0.5	0.090909	0.000121	0.000011	0.000287
0.6	0.109091	0.005039	0.000550	0.014269
0.7	0.127273	0.058784	0.007482	0.194194
0.8	0.145455	0.165343	0.024050	0.624246

theta	prior	likelihood	product	posterior
0.9	0.163636	0.039319	0.006434	0.167002
1	0.181818	0.000000	0.000000	0.000000
Total	1.000000	0.268607	0.038526	1.000000

The plot displays the posterior distribution. Compare to the simulation-based approximation from the first section.

```
plot(theta, posterior, type = "h")
```



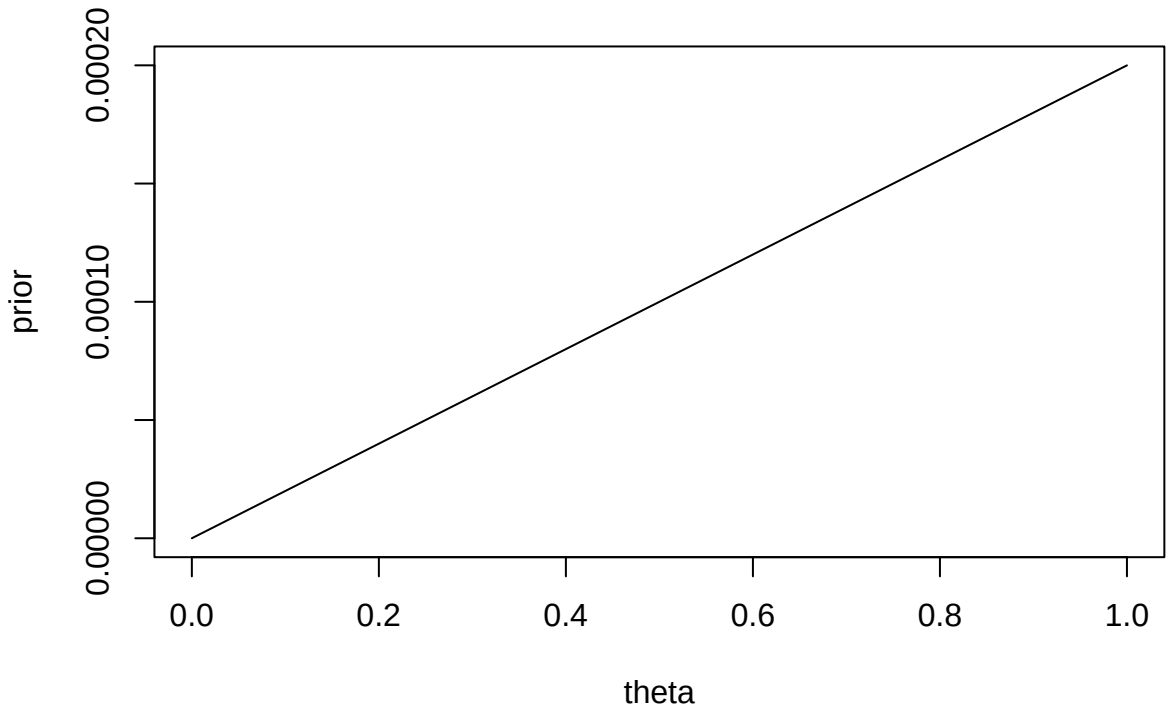
EXERCISE - Fine grid simulation

Now you will repeat the analysis from the first two sections, but assuming θ can take values 0, 0.0001, 0.0002, etc.

```
theta = seq(0, 1, 0.0001)
```

We'll assume a prior similar similar to the previous sections, with plausibility increasing linearly to a maximum value at $\theta = 1$.

```
units = theta

prior = units / sum(units)

plot(theta, prior, type = "l")
```



Assume the same sample data (29 out of 36).

Use simulation to approximate the posterior distribution of θ. Your end result should be a *histogram* of the posterior distribution.

Then write a few sentences describing the posterior distribution. What does it say about the plausibility of values of θ after observing sample data? How has the plausibility changed from the prior? Does this make sense?

Important note: There are a lot more possible values of θ now, so you will want to boost the number of repetitions in the simulation to see more of the possibilities. Also, since each possible value of θ will only occur at most a few times in the simulation, a spike plot is not reasonable. Instead, create *histograms* of θ values (e.g., using `hist`) You will NOT want to try to create or display a table in this case; just focus on the plots.

Important note: You can obviously just cut and paste earlier code. But I strongly encourage you to try to write the code on your own in this section.

```
# Type your code here.
# Add as many code cells as you want.
```

TYPE YOUR RESPONSE HERE.

EXERCISE - Fine grid spreadsheet

```
knitr::knit_exit()
```