Bayesian Analysis - Assignment 1

Chris Hayduk

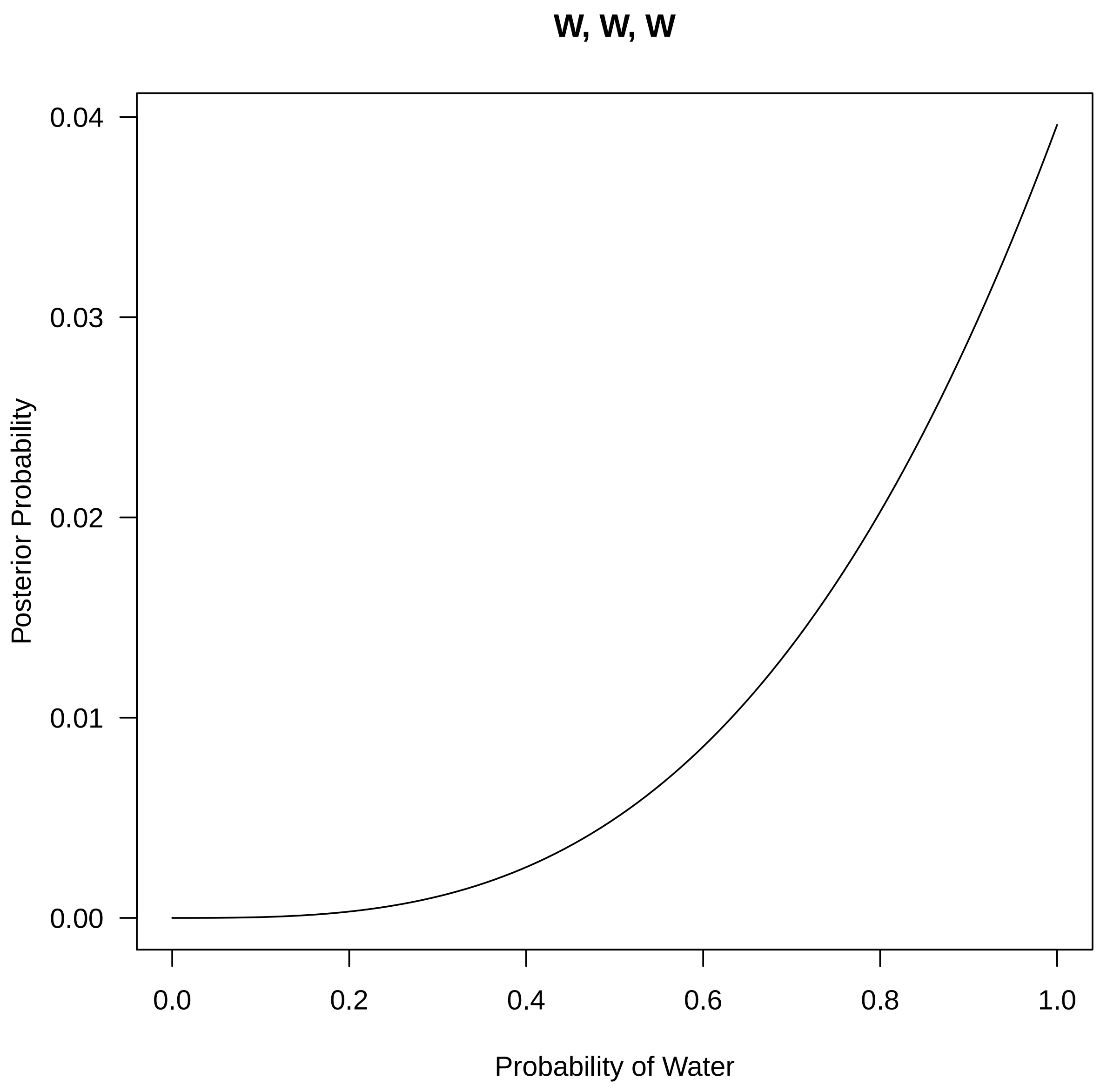
February 9, 2019

1 Ch. 2

Problem 1.
Write function to take in the p_grid, a character vector of the data, and a prior. Compute the grid approximate posterior distribution and return a list of data frames containing p_grid, the computed posterior, and the data.

```
compute_grid_approximate_posterior <- function(p_grid, data, prior){
  num_successes <- sum(data == "W")
  num_trials <- length(data)

  likelihood <- dbinom(num_successes, size = num_trials, prob = p_grid)

  unstd.posterior <- likelihood*prior

  posterior <- unstd.posterior/sum(unstd.posterior)

  grid_approximation <- data.frame(p_grid = p_grid, posterior = posterior)

  data1 <- data.frame(data)

  answer <- list(grid_approximation, data1)

  return(answer)
}

#Example run of function

data <- c("W", "W", "W")

p_grid <- seq(from = 0, to = 1, length.out = 100)

prior <- rep(1, length(p_grid))
```

```r
grid_approximation <- compute_grid_approximate_posterior(p_grid, data, prior)

plot(grid_approximation[[1]]$p_grid, grid_approximation[[1]]$posterior,
     type = "l",
     xlab = "Probability of Water", ylab = "Posterior Probability",
     main = paste(grid_approximation[[2]]$data, collapse = ", "),
     las = 1)
```



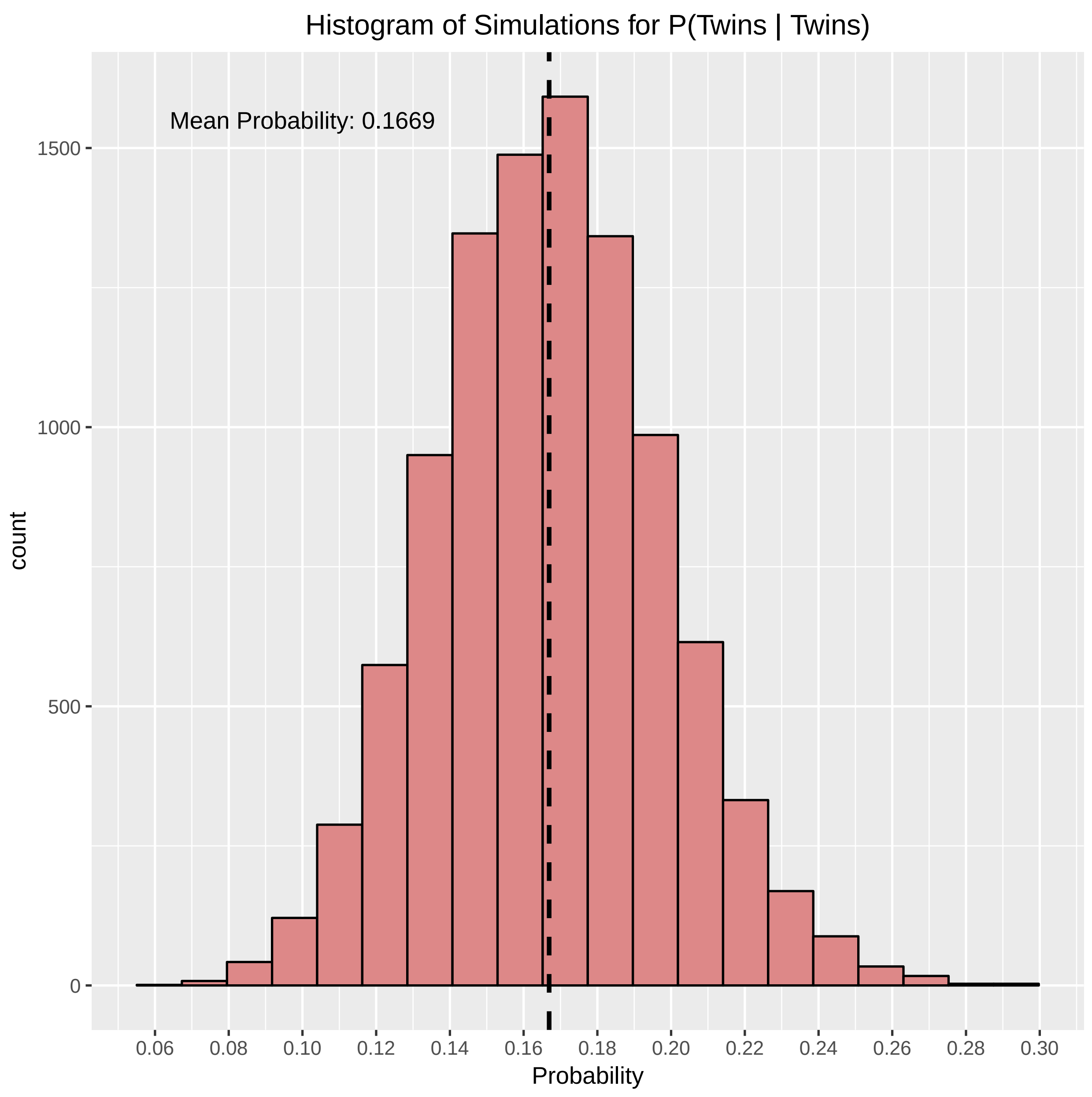
Problem 2.
Use simulation to approximate the result of 2M4. The target probability is $\frac{2}{3}$.

```
data <- c("card1_black", "card1_black",
          "card2_black", "card2_white",
          "card3_white", "card3_white")

sampled_data <- sample(data, size = 100000, replace = TRUE)

ans <- sum((sampled_data == "card1_black"))

ans <- ans/sum((sampled_data == "card1_black" | sampled_data == "card2_black"))

print(ans)

## [1] 0.6637323
```

Problem 3.
Use simulation to approximate the result of 2H1 The target probability is 0.1666667.

```
#Target probability is about 0.167
probability_of_consecutive_twins <- function(total_number_of_moms){
  simulated_moms <- rbinom(total_number_of_moms, 1, 0.5)

  speciesA <- sum(simulated_moms)

  speciesB <- total_number_of_moms - speciesA

  speciesA_twins1 <- rbinom(speciesA, 1, 0.1)

  speciesA_twins1 <- sum(speciesA_twins1)

  speciesB_twins1 <- rbinom(speciesB, 1, 0.2)

  speciesB_twins1 <- sum(speciesB_twins1)

  speciesA_twins2 <- rbinom(speciesA_twins1, 1, 0.1)

  speciesB_twins2 <- rbinom(speciesB_twins1, 1, 0.2)

  prob <- (sum(speciesA_twins2) + sum(speciesB_twins2))

  prob <- prob/(length(speciesA_twins2) + length(speciesB_twins2))

  return(prob)
}
```

```r
#Now let's simulate 10000 runs and plot the results
#Set total_number_of_moms at 1000
df <- data.frame(results = rep(0, 10000))
for(i in 1:10000){
  df$results[i] <- probability_of_consecutive_twins(1000)
}
mean_prob <- mean(df$results)
mean_prob_text <- paste0("Mean Probability: ", round(mean(df$results), 4))

ggplot(df, aes(x = results)) +
  geom_histogram(bins = 20, colour="black", fill="#DD8888") +
  scale_x_continuous(breaks = pretty(df$results, n = 10)) +
  geom_vline(aes(xintercept=mean_prob),
            color="black", linetype="dashed", size=1) +
  annotate("text", x = 0.1, y = 1550, label = mean_prob_text) +
  xlab("Probability") +
  ggtitle("Histogram of Simulations for P(Twins | Twins)") +
  theme(plot.title = element_text(hjust = 0.5))
```



As can be seen by the simulation, our function approximates the true probability reasonably well after 10,000 iterations.

2 Ch. 3

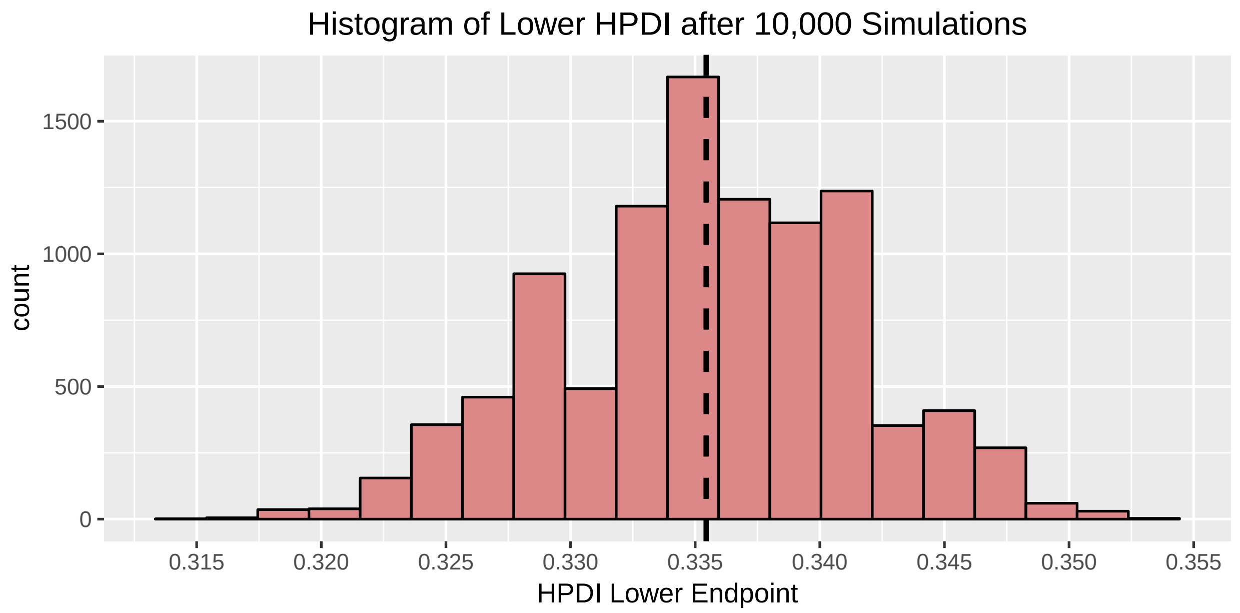
Problem 1.
Run the code from 3M2 10,000 times. How much variation do you see in the endpoints? What does it mean? Also, see how your answers change if you increase the length of p_grid and/or the number of samples.

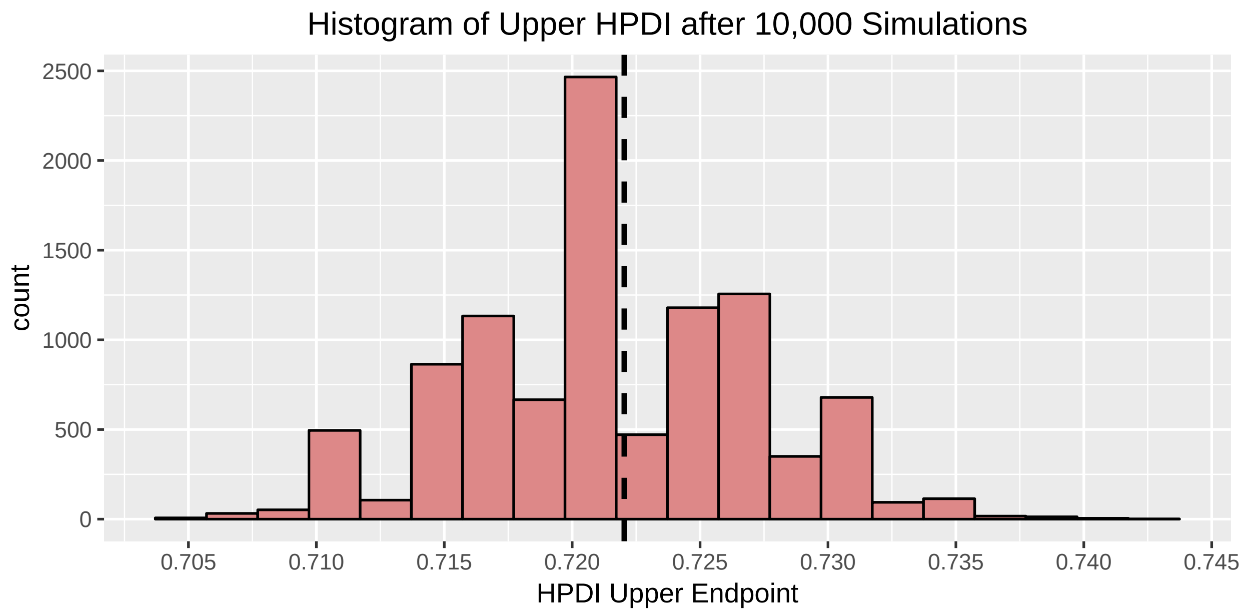
```r
#Histogram function to assist with plotting the endpoints
histogram_func <- function(df, title, lower = TRUE){
  if(lower){
    df$results <- df$lower
    x_label <- "HPDI Lower Endpoint"
    text_annotation <- "Mean Lower Endpoint: "
  } else{
    df$results <- df$upper
    x_label <- "HPDI Upper Endpoint"
    text_annotation <- "Mean Upper Endpoint: "
  }

  mean_prob <- mean(df$results)
  mean_prob_text <- paste0(text_annotation, round(mean(df$results), 4))

  plot <- ggplot(df, aes(x = results)) +
          geom_histogram(bins = 20, colour="black", fill="#DD8888") +
          scale_x_continuous(breaks = pretty(df$results, n = 10)) +
          geom_vline(aes(xintercept=mean_prob),
                     color="black", linetype="dashed", size=1) +
          xlab(x_label) +
          ggtitle(title) +
          theme(plot.title = element_text(hjust = 0.5))

  return(plot)
}
```

Running 10,000 simulations:

```r
p_grid <- seq(from = 0, to = 1, length.out = 1000)
prior <- rep(1, length(p_grid))
data <- rep("W", 8)
data <- c(data, rep("L", 7))

grid_approx <- compute_grid_approximate_posterior(p_grid, data, prior)

HPDI_df <- data.frame(lower = rep(0, 10000), upper = rep(0, 10000))
for(i in 1:10000){
  samples <- sample(grid_approx[[1]]$p_grid,
                    prob = grid_approx[[1]]$posterior,
                    size = 1e4, replace = TRUE)
  HPDI <- HPDI(samples, prob = 0.9)
  HPDI_df[i, 1] <- HPDI[1]
  HPDI_df[i, 2] <- HPDI[2]
}
```

```r
print(paste0("Mean HPDI interval: ",
      mean(HPDI_df[,1]), ", ", mean(HPDI_df[,2])))

## [1] "Mean HPDI interval: 0.335438838838839, 0.722031531531532"

print(paste0("Standard deviation of lower bound of HPDI: ",
             sd(HPDI_df[,1])))

## [1] "Standard deviation of lower bound of HPDI: 0.00585061961183533"

print(paste0("Standard deviation of upper bound of HPDI: ",
             sd(HPDI_df[,2])))

## [1] "Standard deviation of upper bound of HPDI: 0.00565161454745305"

lower_plot <- histogram_func(HPDI_df,
               "Histogram of Lower HPDI after 10,000 Simulations")
upper_plot <- histogram_func(HPDI_df,
               "Histogram of Upper HPDI after 10,000 Simulations",
               lower = FALSE)

grid.arrange(lower_plot, upper_plot, nrow = 2)
```



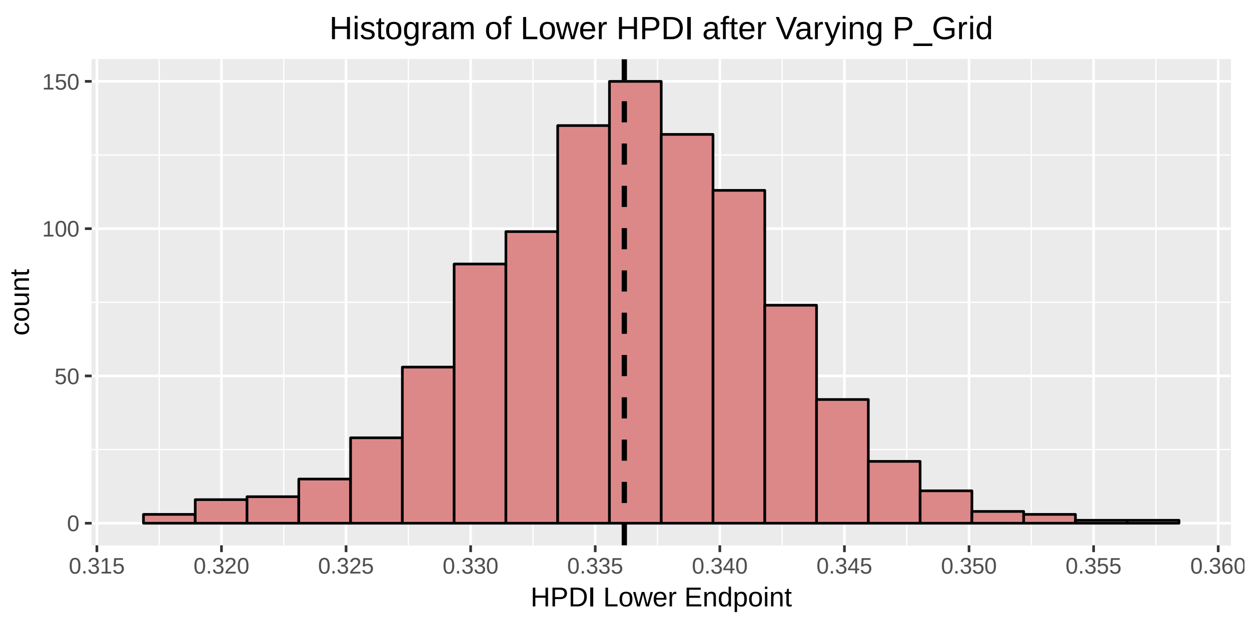


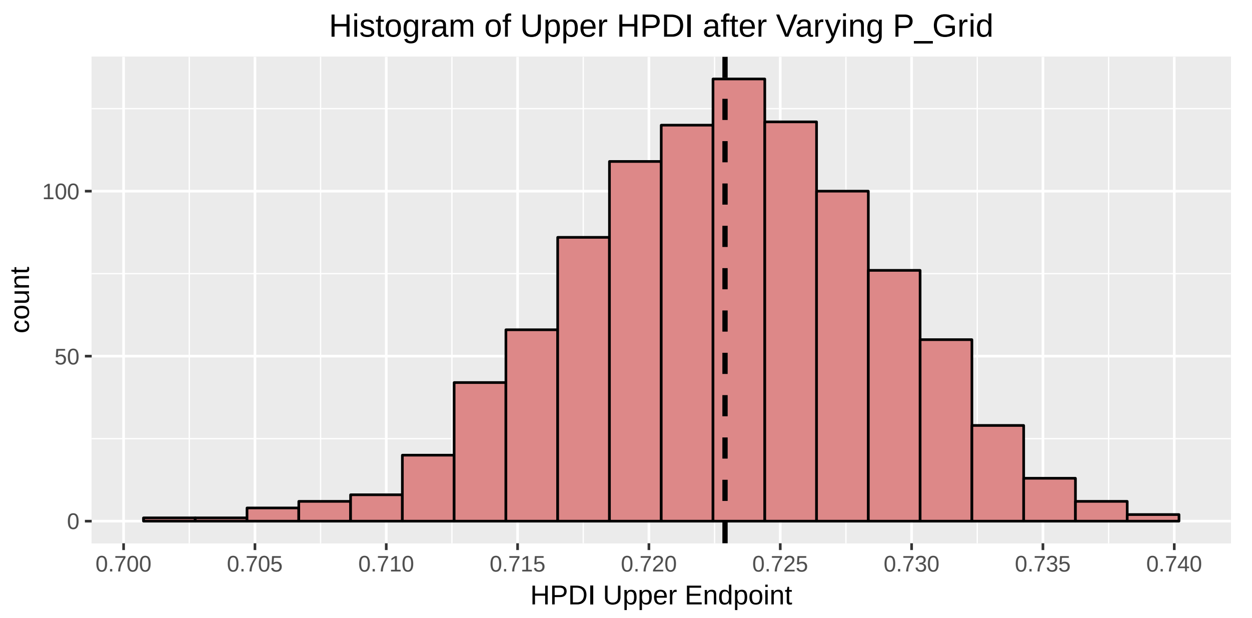
We can see that the lower endpoint appears to vary more than upper endpoint.

If we vary the length of p_grid:

```
sequence <- seq(from = 100, to = 10000, by = 10)

HPDI_df <- data.frame(lower = rep(0, length(sequence)),
                      upper = rep(0, length(sequence)),
                      length_p_grid = sequence)

count <- 1
for(i in sequence){
  p_grid <- seq(from = 0, to = 1, length.out = i)
```

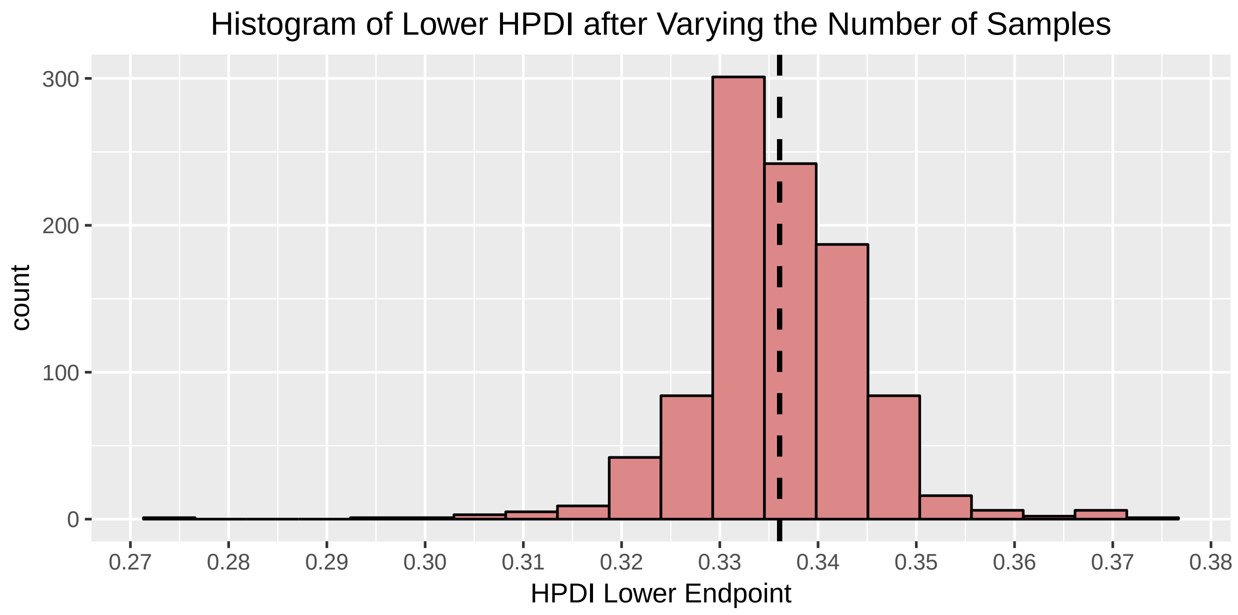
```r
  prior <- rep(1, length(p_grid))
  data <- rep("W", 8)
  data <- c(data, rep("L", 7))

  grid_approx <- compute_grid_approximate_posterior(p_grid, data, prior)

  samples <- sample(grid_approx[[1]]$p_grid,
                    prob = grid_approx[[1]]$posterior,
                    size = 1e4, replace = TRUE)
  HPDI <- HPDI(samples, prob = 0.9)
  HPDI_df[count, 1] <- HPDI[1]
  HPDI_df[count, 2] <- HPDI[2]
  count <- count + 1
}

print("After varying length of of p_grid")

## [1] "After varying length of of p_grid"

print(paste0("Mean HPDI interval: ",
      mean(HPDI_df[,1]), ", ", mean(HPDI_df[,2])))

## [1] "Mean HPDI interval: 0.336166437982061, 0.722897197411659"

print(paste0("SD of lower bound of HPDI: ", sd(HPDI_df[,1])))

## [1] "SD of lower bound of HPDI: 0.00584574453796885"

print(paste0("SD of upper bound of HPDI: ", sd(HPDI_df[,2])))

## [1] "SD of upper bound of HPDI: 0.00588149272981486"

lower_plot <- histogram_func(HPDI_df,
               "Histogram of Lower HPDI after Varying P_Grid")
upper_plot <- histogram_func(HPDI_df,
               "Histogram of Upper HPDI after Varying P_Grid",
               lower = FALSE)

grid.arrange(lower_plot, upper_plot, nrow = 2)
```

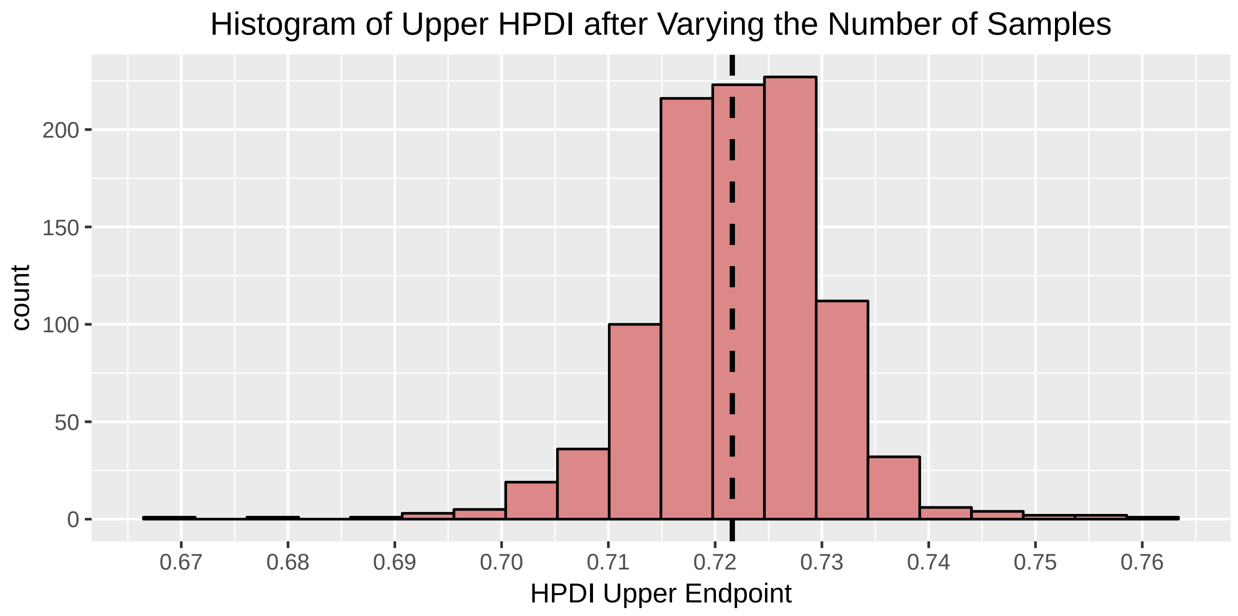




Now if we vary the number of samples:

```
p_grid <- seq(from = 0, to = 1, length.out = 1000)
prior <- rep(1, length(p_grid))
data <- rep("W", 8)
data <- c(data, rep("L", 7))

grid_approx <- compute_grid_approximate_posterior(p_grid, data, prior)

sequence <- seq(from = 100, to = 10000, by = 10)

HPDI_df <- data.frame(lower = rep(0, length(sequence)),
                      upper = rep(0, length(sequence),
                                  num_samples = sequence))

count <- 1
for(i in sequence){
  samples <- sample(grid_approx[[1]]$p_grid,
                    prob = grid_approx[[1]]$posterior,
                    size = i, replace = TRUE)
  HPDI <- HPDI(samples, prob = 0.9)
  HPDI_df[count, 1] <- HPDI[1]
  HPDI_df[count, 2] <- HPDI[2]

  count <- count + 1
}

print("After varying number of samples")

## [1] "After varying number of samples"

print(paste0("Mean HPDI interval: ",
      mean(HPDI_df[,1]), ", ", mean(HPDI_df[,2])))

## [1] "Mean HPDI interval: 0.336086843654957, 0.721608591437047"

print(paste0("SD of lower bound of HPDI: ", sd(HPDI_df[,1])))

## [1] "SD of lower bound of HPDI: 0.00864784890462752"

print(paste0("SD of upper bound of HPDI: ", sd(HPDI_df[,2])))

## [1] "SD of upper bound of HPDI: 0.00840545532680845"

lower_plot <- histogram_func(HPDI_df,
               "Histogram of Lower HPDI after Varying the Number of Samples")
upper_plot <- histogram_func(HPDI_df,
               "Histogram of Upper HPDI after Varying the Number of Samples",
               lower = FALSE)

grid.arrange(lower_plot, upper_plot, nrow = 2)
```





Problem 2.
From 3H5: Make a 2x2 contingency table of birth1 and birth2. Run the appropriate hypothesis test to check if the rows and columns are independent (eg. Fisher's exact test, χ^2 test). What do you conclude? How does it compare with the analysis in 3H5? Which do you trust more? Justify your answer.

```
data(homeworkch3)

boy_girl <- length(birth2[birth1 == 1 & birth2 == 0])
```

```r
boy_boy <- length(birth2[birth1 == 1 & birth2 == 1])

girl_boy <- length(birth2[birth1 == 0 & birth2 == 1])

girl_girl <- length(birth2[birth1 == 0 & birth2 == 0])

contingency_table <- data.frame(boy = c(boy_boy, boy_girl),
                                girl = c(girl_boy, girl_girl))

rownames(contingency_table) <- c("boy", "girl")

print(contingency_table)

##      boy girl
## boy   21   39
## girl  30   10
```

The assumptions for the χ^2 test (taken from "McHugh ML. The chi-square test of independence. Biochem Med (Zagreb). 2013;23(2):143-9.") are:

1. The data in the cells should be frequencies, or counts of cases rather than percentages or some other transformation of the data.

2. The levels (or categories) of the variables are mutually exclusive. That is, a particular subject fits into one and only one level of each of the variables.

3. Each subject may contribute data to one and only one cell in the χ^2. If, for example, the same subjects are tested over time such that the comparisons are of the same subjects at Time 1, Time 2, Time 3, etc., then χ^2 may not be used.

4. The study groups must be independent. This means that a different test must be used if the two groups are related. For example, a different test must be used if the researcher's data consists of paired samples, such as in studies in which a parent is paired with his or her child.

5. There are 2 variables, and both are measured as categories, usually at the nominal level. However, data may be ordinal data. Interval or ratio data that have been collapsed into ordinal categories may also be used. While Chi-square has no rule about limiting the number of cells (by limiting the number of categories for each variable), a very large number of cells (over 20) can make it difficult to meet assumption 6 below, and to interpret the meaning of the results.

6. The value of the cell expecteds should be 5 or more in at least 80% of the cells, and no cell should have an expected of less than one. This assumption is most likely to be met if the sample size equals at least the number of cells multiplied by 5. Essentially, this assumption specifies the number of cases (sample size) needed to use the χ^2 for

any number of cells in that χ^2. This requirement will be fully explained in the example of the calculation of the statistic in the case study example.

The assumptions for Fisher's exact test (from statisticssolutions.com) are:

1. The row and column totals are fixed, not random.
2. Sampling or allocation are random and observations are mutually independent within the constraints of fixed marginal totals.
3. Each observation is mutually exclusive - in other words each observation can only be classified in one cell.

It seems that the data satisfies the assumptions for both tests. We will inspect the results of both:

```
chisq <- chisq.test(contingency_table)

print(chisq)

##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  contingency_table
## X-squared = 13.807, df = 1, p-value = 0.0002026

fisher <- fisher.test(contingency_table)

print(fisher)

##
##  Fisher's Exact Test for Count Data
##
## data:  contingency_table
## p-value = 0.0001022
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##  0.06590571 0.47305403
## sample estimates:
## odds ratio
##  0.1828989
```

Both tests demonstrate that the rows and columns (ie. first and second births) are not independent, which agrees with the original graphical analysis in 3H5.