ST 502 Homework 4

Eric Warren

2024-03-19

Contents

1 Problem 8.4 E

If the prior distribution of Θ is uniform on [0, 1], what is the posterior density? Plot it. What is the mode of the posterior?

Eric Warren

ST 502 Homework 4

8.4 E: Prior distr. Θ is Unif(0,1). Find posterior density and plot it. Also find the mode.

- know that posterior is $f_{\Theta|X}(\theta|x) = \frac{f_{X,\Theta}(x,\theta)}{f_X(x)} = \frac{f_{X|\Theta}(x|\theta)\cdot f_\Theta(\theta)}{f_X(x)}$

Note the density $f_{X|\Theta}(x|\theta)$ is the likelihood function since X here is just the observed sample. So $f_{X|\Theta}(x|\theta) = L(\theta) = P(0|\theta)^2 P(1|\theta)^3 P(2|\theta)^3 P(3|\theta)^2$

$= (\frac{2}{3}\theta)^2 (\frac{1}{3}\theta)^3 (\frac{2}{3}(1-\theta))^3 (\frac{1}{3}(1-\theta))^2 = \frac{2^5\cdot 1^5}{3^{10}}\theta^5(1-\theta)^5 = \frac{2^5}{3^{10}}\theta^5(1-\theta)^5$ using exponent rules.

Since $f_{X|\Theta}(x|\theta) = \frac{2^5}{3^{10}}\theta^5(1-\theta)^5$ need to find $f_X(x)$. Note since $\Theta \sim Unif(0,1)$ then $f_\Theta(\theta) = 1$ if $\theta \in [0,1]$. Knowing this, we can solve $f_X(x)$ by solving $f_X(x) = \int f_{X|\Theta}(x|\theta) f_\Theta(\theta)\, d\theta = \int_0^1 f_{X|\Theta}(x|\theta)$ since $\theta \in [0,1]$ with $f_\Theta(\theta)=1$.

So $f_X(x) = \int_0^1 f_{X|\Theta}(x|\theta) = \frac{2^5}{3^{10}}\int_0^1 \theta^5(1-\theta)^5\, d\theta = \frac{2^5}{3^{10}}\frac{\Gamma(6)\Gamma(6)}{\Gamma(12)} = \frac{2^5}{3^{10}}\frac{5!\,5!}{11!}$

$= 8/40,320,957$. Note the integral was found using kernel matching for a Beta distribution $\left(\int_0^1 x^{a-1}(1-x)^{b-1}\,dx = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)} = \frac{(a-1)!(b-1)!}{(a+b-1)!}\right)$. Now using $f_\Theta(\theta)=1$ for $\theta\in[0,1]$ and previous work, we can show that

$f_{\Theta|X}(\theta|x) = \frac{f_{X|\Theta}(x|\theta) f_\Theta(\theta)}{f_X(x)} = \frac{2^5/3^{10}\,\theta^5(1-\theta)^5\cdot 1}{8/40,320,957} = 2772\,\theta^5(1-\theta)^5$ which is proportional to a Beta density where $a=6$ and $b=6$ so our posterior is proportional to Beta(6,6).

We can find the mode by finding its global maximum. Our derivative of our posterior is

$f'_{\Theta|X}(\theta|x) = \frac{d}{d\theta}[2772\,\theta^5(1-\theta)^5] = \frac{d}{d\theta}[2772(\theta-\theta^2)^5] = 2772\cdot 5(\theta-\theta^2)^4(1-2\theta)$

$= 13860(\theta-\theta^2)^4(1-2\theta)$. Set equal to zero to find critical points.

$13860(\theta-\theta^2)^4(1-2\theta) = 0 \iff (\theta-\theta^2)^4(1-2\theta) = 0$ so $\theta-\theta^2=0$ and $1-2\theta=0$ and thus $\theta=0$, $\theta=1$, or $\theta=\frac{1}{2}$. However, $f_{\Theta|X}(0|x) = f_{\Theta|X}(1|x) = 0$ and $f_{\Theta|X}(\frac{1}{2}|x) = 2772(\frac{1}{2})^5(1-\frac{1}{2})^5 = \frac{2772}{2^{10}} \approx \frac{692}{256} > 0$ so the mode of the posterior (call θ_{MODE}) is $\theta_{MODE} = \frac{1}{2}$. Lastly let us depict our posterior density in graphical form (using R code plot(seq(0,1,100), dbeta(seq(0,1,100), 6, 6), type="l"))

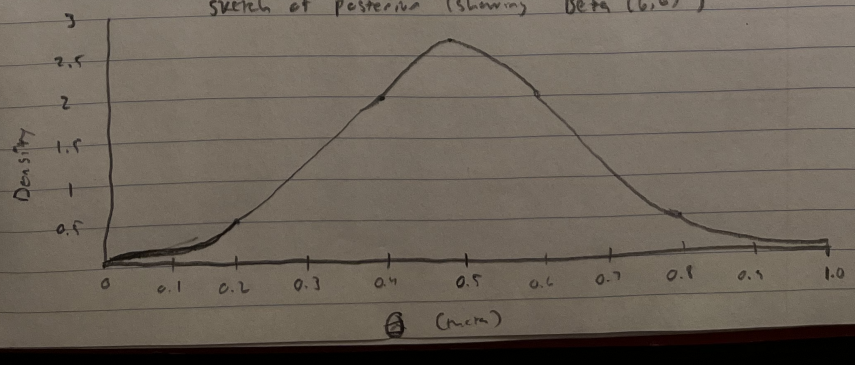
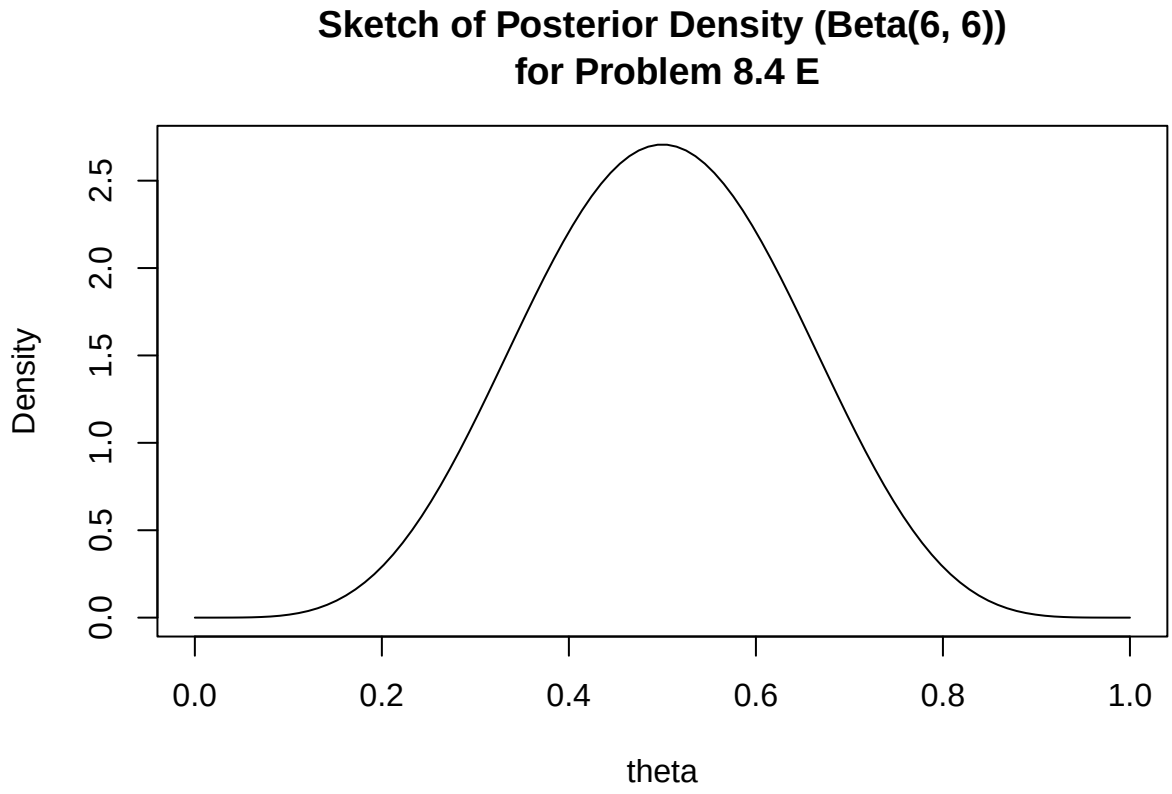


Figure 1: Work for 8.4 E

Work is done on first page, but code for graph is shown below.

```
plot(
  seq(0, 1, length = 100),
  dbeta(seq(0, 1, length = 100), 6, 6),
  type = "l",
  main = "Sketch of Posterior Density (Beta(6, 6))\n for Problem 8.4 E",
  xlab = "theta",
  ylab = "Density"
)
```



2 Problem 8.42

For each of 100 sequential time intervals of variable lengths (given in seconds), the number of gamma rays originating in a particular area of the sky was recorded. Assuming a model that the arrival times are a Poisson process with constant emission rate (λ = events per second), estimate λ. What is the estimated standard error? How might you informally check the assumption that the emission rate is constant? What is the posterior distribution of Λ if an improper gamma prior is used?

First, we should read in the data which is done below showing the first couple of observations.

```
library(tidyverse)
(rays <- read_csv("gamma-ray.csv"))
```

```
## # A tibble: 100 x 2
##    seconds count
##      <dbl> <dbl>
##  1   116       0
##  2   112       0
##  3   160       0
##  4    51.5     0
##  5   102       1
##  6    77.4     0
##  7    14.5     0
##  8  1070       3
##  9    99.1     1
## 10    49.6     0
## # i 90 more rows
```

```
# Sum number of seconds
sum(rays$seconds)
```

```
## [1] 15718.2
```

```
# Sum number of count
sum(rays$count)
```

```
## [1] 61
```

Now, The parameter λ (which is the number of events per second) can be estimated as a ratio of the total number of events to the sum of lengths of all 100 intervals. So we obtain an estimate of since the total duration of recording was 15718.2 seconds, and during that time, 61 gamma rays were observed. So $\hat{\lambda} = 61$ / 15718.2 ≈ 0.0039.

We can also find the estimated standard error for this to be $SE(\hat{\lambda}) = \sqrt{\frac{\hat{\lambda}}{n}} = \sqrt{\frac{0.0039}{15718.2}} = 0.0005$. We act as if $n = 15718.2$ is the total length of all 100 time intervals (as each second is one trial or one sampled unit).

To check to see if our emission rate is constant we can split up the intervals into 5 groups (do 1-20, 21-40, etc. up to 81-100) and see if their average emission rates are about the same. We can do this below and then show a table of the average emission results.

```
# Get list of interval values
Interval <- c("1-20", "21-40", "41-60", "61-80", "81-100")

# Get average values from each interval
value_table <- rays %>%
  mutate(index = ceiling(seq_len(n())/20)) %>%
  group_by(index, add = T) %>%
  summarise(`Average Emission` = round(mean(sum(count) / sum(seconds)), 5))

# Combine into a data frame
values_avg <- cbind(Interval, value_table) %>%
  dplyr::select(- index)

# Display table
knitr::kable(values_avg)
```

Interval	Average Emission
1-20	0.00374
21-40	0.00513
41-60	0.00165
61-80	0.00312
81-100	0.00612

Since we can see that our averages in the final column are not that far from our estimate $\hat{\lambda}$, we can say that this is a clue that the emission rate could actually be constant; however, this surely doesn't prove the claim.

Since we are given the prior distribution, we can now find the posterior distribution of Λ. Note an improper gamma is defined as $f_\Lambda(\lambda) = \frac{1}{\lambda}, 0 < \lambda < \infty$ and note the density does not integrate to 1.

Now note to find our posterior that $f_{\Lambda|X}(\lambda|x) = \frac{f_{X|\Lambda}(x|\lambda) * f_\Lambda(\lambda)}{f_X(x)}$. We can say that $f_{X|\Lambda}(x|\lambda) = \frac{(\lambda t_1)^{x_1} e^{-\lambda t_1}}{x_1!} * ... * \frac{(\lambda t_n)^{x_n} e^{-\lambda t_n}}{x_n!} = \frac{t_1^{x_1} * t_n^{x_n}}{x_1! * ... * x_n!} \lambda^{x_1+...+x_n} e^{-\lambda(t_1+...+t_n)}$ where $n = 100$ is the number of time trials, t_i is the length of our i-th time interval, and x_i is the number of observed gamma rays in our i-th interval.

Lastly, we need to find $f_X(x) = \int f_{X|\Lambda}(x|\lambda) f_\Lambda(\lambda) d\lambda = \int_0^\infty \frac{t_1^{x_1} * ... * t_n^{x_n}}{x_1! * ... * x_n!} \lambda^{x_1+...+x_n} e^{-\lambda(t_1+...+t_n)} \frac{1}{\lambda} d\lambda = \int_0^\infty \frac{t_1^{x_1} * ... * t_n^{x_n}}{x_1! * ... * x_n!} \lambda^{x_1+...+x_n-1} e^{-\lambda(t_1+...+t_n)} d\lambda$ using what we know about $f_\Lambda(\lambda)$.

Note the property that $\int_0^\infty x^{a-1} e^{-\lambda x} dx = \frac{\Gamma(a)}{\lambda^a}$. Using this to our integral, we can say that $f_X(x) = \int_0^\infty \frac{t_1^{x_1} * t_n^{x_n}}{x_1! * ... * x_n!} \lambda^{x_1+...+x_n-1} e^{-\lambda(t_1+...+t_n)} d\lambda = \frac{t_1^{x_1} * ... * t_n^{x_n}}{x_1! * ... * x_n!} \frac{\Gamma(x_1+...+x_n)}{(t_1+...+t_n)^{x_1+...+x_n}}$.

Now plugging in our values we can see that $f_{\Lambda|X}(\lambda|x) = \frac{f_{X|\Lambda}(x|\lambda) * f_\Lambda(\lambda)}{f_X(x)} = \frac{\frac{t_1^{x_1} * t_n^{x_n}}{x_1! * ... * x_n!} \lambda^{x_1+...+x_n-1} e^{-\lambda(t_1+...+t_n)}}{\frac{t_1^{x_1} * ... * t_n^{x_n}}{x_1! * ... * x_n!} \frac{\Gamma(x_1+...+x_n)}{(t_1+...+t_n)^{x_1+...+x_n}}} = \frac{(t_1+...+t_n)^{x_1+...+x_n}}{\Gamma(x_1+...+x_n)} \lambda^{x_1+...+x_n-1} e^{-\lambda(t_1+...+t_n)}$. We can recognize this as a gamma density which has the parameters $\alpha = x_1 + ...x_k = 61$ and $\beta = t_1 + ...t_n = 15718.2$. So our posterior distribution is $\Lambda|X \sim gamma(61, 15718.2)$.

3 Problem 8.63

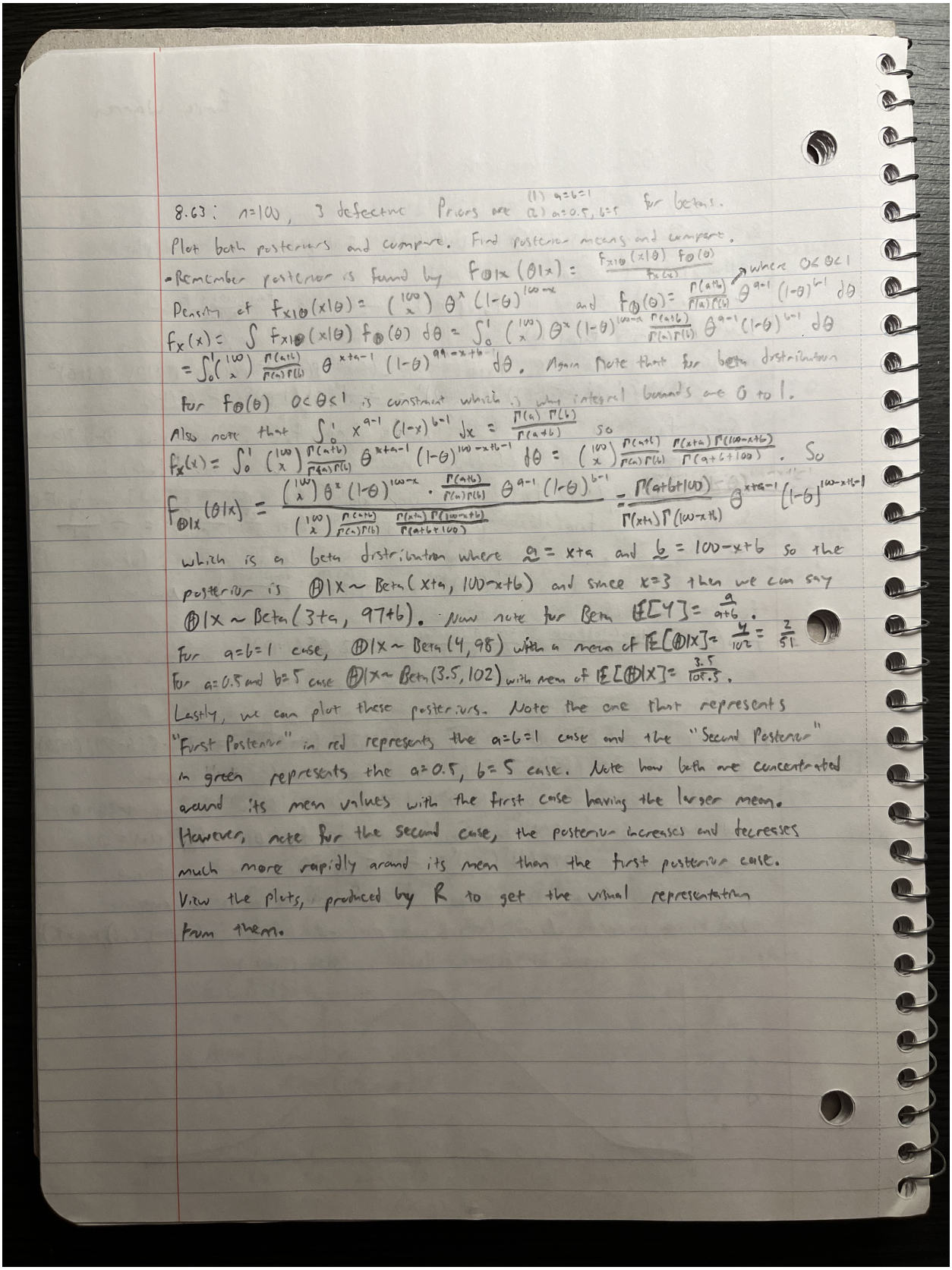
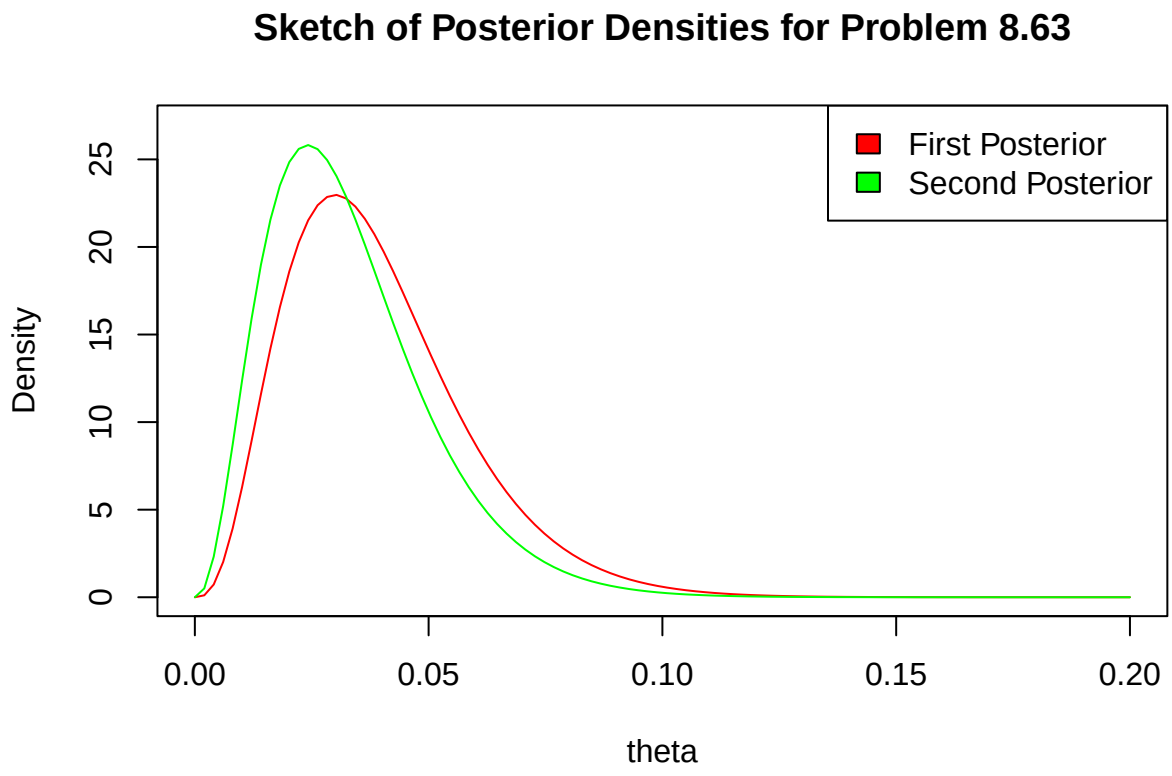


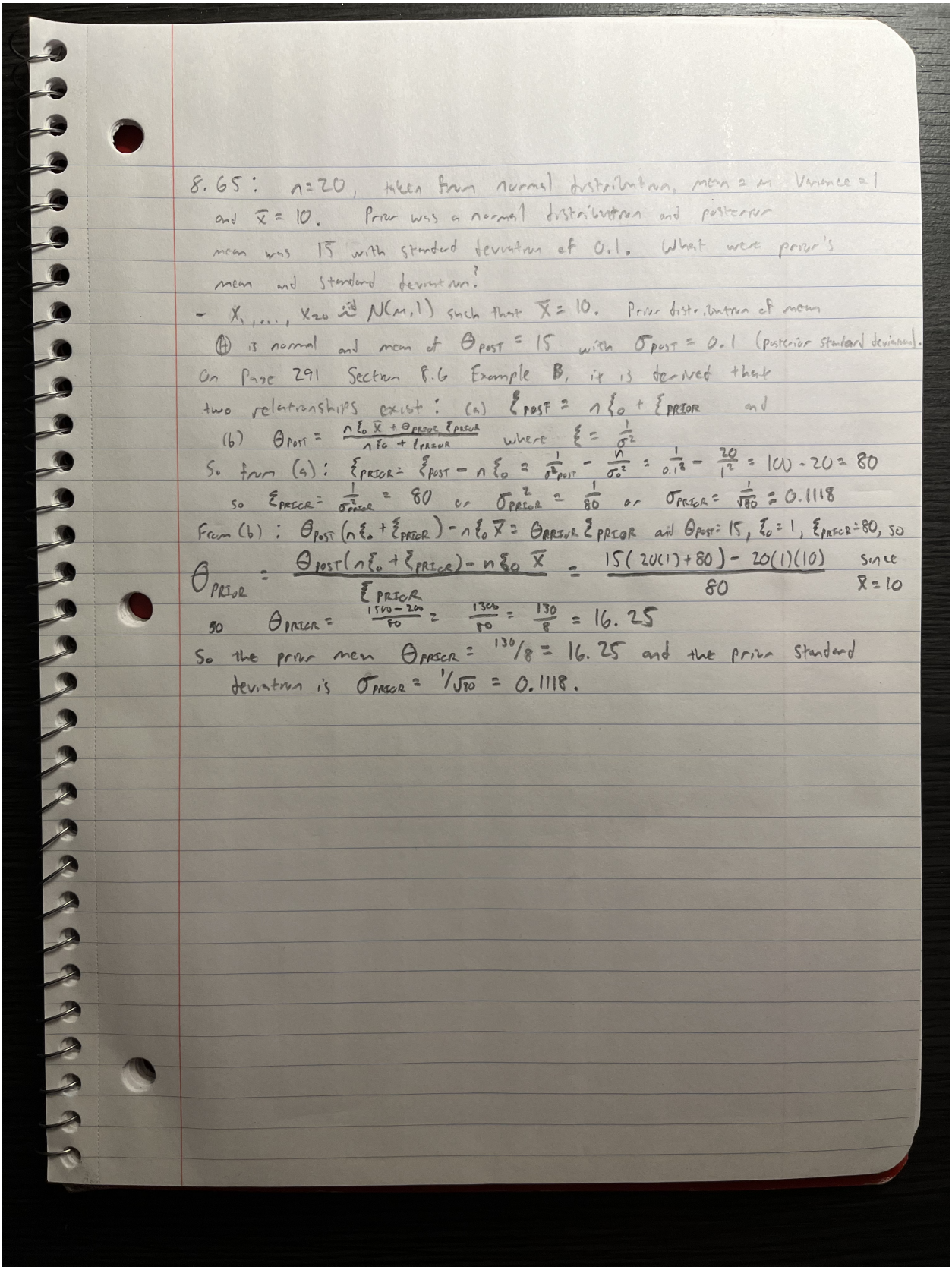
Figure 2: Work for 8.63

Here we can make the plot of our posterior distributions.

```
# Get sequence lengths for plots
seq_length <- seq(0, 0.2, length = 100)

# Make first plot where a = b = 1
plot(
  seq_length,
  dbeta(seq_length, 4, 98),
  type = "l",
  col = "red",
  main = "Sketch of Posterior Densities for Problem 8.63",
  xlab = "theta",
  ylab = "Density",
  ylim = c(0, 27)
)
lines(
  seq_length,
  dbeta(seq_length, 3.5, 102),
  col = "green",
)
legend(
  "topright",
  legend = c("First Posterior", "Second Posterior"),
  fill = c("red","green")
)
```



4 Problem 8.65



8.65: $n=20$, taken from normal distribution, mean = μ Variance = 1 and $\bar{x}=10$. Prior was a normal distribution and posterior mean was 15 with standard deviation of 0.1. What were prior's mean and standard deviation?

- $X_1, \ldots, X_{20} \overset{iid}{\sim} N(\mu, 1)$ such that $\bar{X}=10$. Prior distribution of mean μ is normal and mean of $\theta_{POST}=15$ with $\sigma_{POST}=0.1$ (posterior standard deviation).

On Page 291 Section 8.6 Example B, it is derived that two relationships exist: (a) $\xi_{POST} = n\xi_0 + \xi_{PRIOR}$ and

(b) $\theta_{POST} = \frac{n\xi_0\bar{x} + \theta_{PRIOR}\xi_{PRIOR}}{n\xi_0 + \xi_{PRIOR}}$ where $\xi = \frac{1}{\sigma^2}$

So from (a): $\xi_{PRIOR} = \xi_{POST} - n\xi_0 = \frac{1}{\sigma^2_{POST}} - \frac{n}{\sigma_0^2} = \frac{1}{0.1^2} - \frac{20}{1^2} = 100 - 20 = 80$

so $\xi_{PRIOR} = \frac{1}{\sigma^2_{PRIOR}} = 80$ or $\sigma^2_{PRIOR} = \frac{1}{80}$ or $\sigma_{PRIOR} = \frac{1}{\sqrt{80}} = 0.1118$

From (b): $\theta_{POST}(n\xi_0 + \xi_{PRIOR}) - n\xi_0\bar{x} = \theta_{PRIOR}\xi_{PRIOR}$ and $\theta_{POST}=15$, $\xi_0=1$, $\xi_{PRIOR}=80$, so

$$\theta_{PRIOR} = \frac{\theta_{POST}(n\xi_0 + \xi_{PRIOR}) - n\xi_0\bar{x}}{\xi_{PRIOR}} = \frac{15(20(1)+80) - 20(1)(10)}{80} \quad \text{since } \bar{x}=10$$

so $\theta_{PRIOR} = \frac{1500-200}{80} = \frac{1300}{80} = \frac{130}{8} = 16.25$

So the prior mean $\theta_{PRIOR} = 130/8 = 16.25$ and the prior standard deviation is $\sigma_{PRIOR} = 1/\sqrt{80} = 0.1118$.

Figure 3: Work for 8.65

5 Extra Problem

Here we are going to create a MCMC to create our own posterior distribution. We are going to do this using logistic regression. First we are going to read in the data.

```
(diabetes <- read_csv("diabetes-dataset.csv"))
```

```
## # A tibble: 2,000 x 9
##    Pregnancies Glucose BloodPressure SkinThickness Insulin   BMI
##          <dbl>   <dbl>         <dbl>         <dbl>   <dbl> <dbl>
##  1           2     138            62            35       0  33.6
##  2           0      84            82            31     125  38.2
##  3           0     145             0             0       0  44.2
##  4           0     135            68            42     250  42.3
##  5           1     139            62            41     480  40.7
##  6           0     173            78            32     265  46.5
##  7           4      99            72            17       0  25.6
##  8           8     194            80             0       0  26.1
##  9           2      83            65            28      66  36.8
## 10           2      89            90            30       0  33.5
## # i 1,990 more rows
## # i 3 more variables: DiabetesPedigreeFunction <dbl>, Age <dbl>, Outcome <dbl>
```

Now we are going to go through and set up functions for our posterior.

```
# calculate p
expit <- function(X, beta0, beta1) {
  1/(1 + exp(-beta0 - beta1*X))
}

# log prior for beta
log_prior <- function(beta, mean = 0, sd = 15) {
  -0.5*(log(sd^2) + (beta - mean)^2/sd^2)
}

# log posterior
log_post <- function(Y, X, beta0, beta1){
  P <- expit(X, beta0, beta1)
  sum(Y*log(P) + (1-Y)*log(1-P)) + log_prior(beta0) + log_prior(beta1)
}
```

Now I am going to go through and make a MCMC sampler function that will help with choosing the betas we want to keep and end up giving us an estimate of both β_0 and β_1 when we are all said and done.

```
# MCMC sampler function
mcmc_sampler <- function(Y, X, N = 100000, beta0_init = 0,
                         proposed_sd0 = 0.009, beta1_init = 0, proposed_sd1 = 0.009) {

  # make the simulation reproducible
  set.seed(999)

  # set up a data frame to store beta0 and beta1
```

```
  df <-  tibble(
    beta0 = beta0_init,
    beta1 = beta1_init
  )

  # current step - initialize our beta values
  beta0 <- beta0_init
  beta1 <- beta1_init

  # log posterior for the current step in simulation
  log_post_sim <- log_post(Y, X, beta0, beta1)

  # iteration
  for (i in 2:N) {

    # beta0: pick a new candidate
    beta0_new <- rnorm(1, beta0, proposed_sd0)

    # new log posterior for beta0
    log_post_sim_new <- log_post(Y, X, beta0_new, beta1)

    # decide whether to accept or reject the new beta0 candidate
    lnR <- log_post_sim_new - log_post_sim
    lnU <- log(runif(1))
    if(lnR > lnU){
      beta0 <- beta0_new
      log_post_sim <- log_post_sim_new
    }

    # beta1: pick a new candidate
    beta1_new <- rnorm(1, beta1, proposed_sd1)

    # New log posterior for beta1
    log_post_sim_new <- log_post(Y, X, beta0, beta1_new)

    # Decide whether to accept or reject the new beta1 candidate
    lnR <- log_post_sim_new - log_post_sim
    lnU <- log(runif(1))
    if(lnR > lnU){
      beta1 <- beta1_new
      log_post_sim <- log_post_sim_new
    }

    # save betas to our data frame to use for graphing for later
    df <- df %>% add_row(
      beta0 = beta0,
      beta1 = beta1
    )
  }

  # return all betas
  return(df)
}
```

Now that we have our MCMC sampler function, we can go through and make another function to find a way to burn in (or remove the first so many observations that we think might be inaccurate) and also use our previous data frame to go through and get important statistics like the posterior mean, median, and credible intervals. Before that though I am going to make a function that will be used for our final plots and then do our summary of MCMC function.

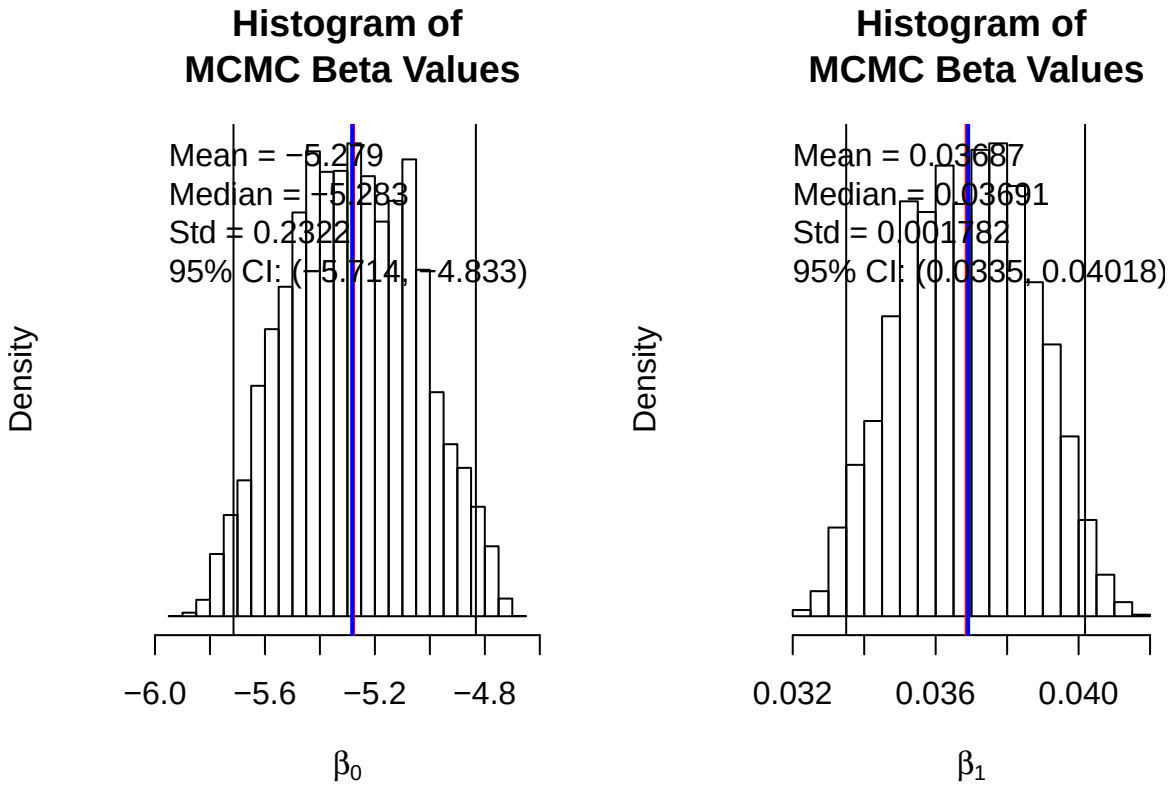
```
# Used to format the text correctly
library(gt)
library(glue)

# summarize a distribution (graphically and numerically)
summary_dist <- function(x, var_expression, pos_x = 0, pos_y = 1, adj = c(0, 1), conf_level = 0.95, plo

  # significance level
  alpha <- (1 - conf_level) / 2

  # mean, median, and intervals
  mean <- mean(x)
  median <- median(x)
  sd <- sd(x)
  CI <- quantile(x, probs = c(alpha, 1 - alpha)) %>%
    `names<-`(vec_fmt_percent(c(alpha, 1 - alpha), decimals = 1))

  if(plot){
    # plot distributions with annotations of mean, median and credible intervals (equal tail)
    hist_obj <- hist(
      x,
      probability = T,
      col = NULL,
      yaxt = "n",
      xlab = var_expression,
      main = paste("Histogram of\n MCMC Beta Values"))

    # add lines to plots
    abline(v = mean, col = "red", lwd = 2)
    abline(v = median, col = "blue", lwd = 2)
    abline(v = CI)

    # annotation
    text(
      quantile(hist_obj$breaks, probs = pos_x),
      quantile(hist_obj$density, probs = pos_y),
      glue(
        "Mean = {signif(mean, 4)}\n",
        "Median = {signif(median, 4)}\n",
        "Std = {signif(sd, 4)}\n",
        "{conf_level*100}% CI: ({toString(signif(CI, 4))})"
      ),
      adj = adj, ...)
  }

  # return summaries
  return(tibble(
    mean = mean,
```

```
    median = median,
    sd = sd,
    CI = list(CI)
  ))
}

# Do MCMC Summary function
summary_of_mcmc <- function(df, burn_in = 10000, ...){

  # Number of MCMC samples
  N <- nrow(df)

  # remove burn-in rows
  df2 <- tail(df, - burn_in)

  # plot distributions of betas with annotations of mean, median and our credible intervals
  par(mfrow = c(1,2))
  beta0 <- summary_dist(df2$beta0, var_expression = expression(beta[0]), ...)
  beta1 <- summary_dist(df2$beta1, var_expression = expression(beta[1]), ...)

  # reset par options
  par(mfrow = c(1,1))

  # combine betas and report intervals, process time and acceptance rate
  bind_rows(list(beta0 = beta0, beta1 = beta1), .id = "term") %>%
    unnest_wider(CI, names_sep = "_")
}
```

Now that we have all of our functions we can get our initial data frame and then do our get the information we want with our last function.

```
# Get the data frame of values from MCMC
df_mcmc <- mcmc_sampler(Y = diabetes$Outcome, X = diabetes$Glucose)

# Get the summary of our MCMC sampler
(mcmc_results <- summary_of_mcmc(df = df_mcmc))
```

```
## # A tibble: 2 x 6
##   term     mean  median      sd `CI_$2.5\\%$` `CI_$97.5\\%$`
##   <chr>   <dbl>   <dbl>   <dbl>         <dbl>          <dbl>
## 1 beta0 -5.28   -5.28   0.232         -5.71          -4.83
## 2 beta1  0.0369  0.0369 0.00178        0.0335         0.0402
```



We can see here that we get some important values to note for each of our beta's.

- For β_0 we get a mean and median around -5.28 with a standard deviation of around 0.232 which gets us 95% credible intervals (this is what CI stands for in this case) of roughly -5.71 and -4.83.
- For β_1 we get a mean and median around 0.0369 with a standard deviation of around 0.00178 which gets us 95% credible intervals (this is what CI stands for in this case) of roughly 0.0335 and 0.0402.

Let us see how this compares to the Frequentest view of creating a GLM with a logit function as the linking function and then getting 95% confidence intervals of that.

```
# Create the fit of this GLM
fit <- glm(Outcome ~ Glucose,
           family = binomial(link = "logit"),
           data = diabetes)

# Show the summary
summary(fit)
```

```
##
## Call:
## glm(formula = Outcome ~ Glucose, family = binomial(link = "logit"),
##     data = diabetes)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
```

```
## -2.0517  -0.7923  -0.5333   0.8647   3.2433
##
## Coefficients:
##              Estimate Std. Error z value            Pr(>|z|)
## (Intercept) -5.254362   0.257281  -20.42 <0.0000000000000002 ***
## Glucose      0.036696   0.001973   18.60 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2569.4  on 1999  degrees of freedom
## Residual deviance: 2108.1  on 1998  degrees of freedom
## AIC: 2112.1
##
## Number of Fisher Scoring iterations: 4
```

```
# Get the 95% confidence intervals
confint(fit)
```

```
##                   2.5 %      97.5 %
## (Intercept) -5.76761651 -4.75867254
## Glucose      0.03289289  0.04062958
```

As we can see the values we get for our estimates and confidence (or credible) intervals are basically the same value whether we do this via Bayesian MCMC or the Frequentest GLM modeling.