Bayesian Learning Lab 1 Report

Manu Jain [manja242] & Varun Gurupurandar [vargu125]

2025-04-19

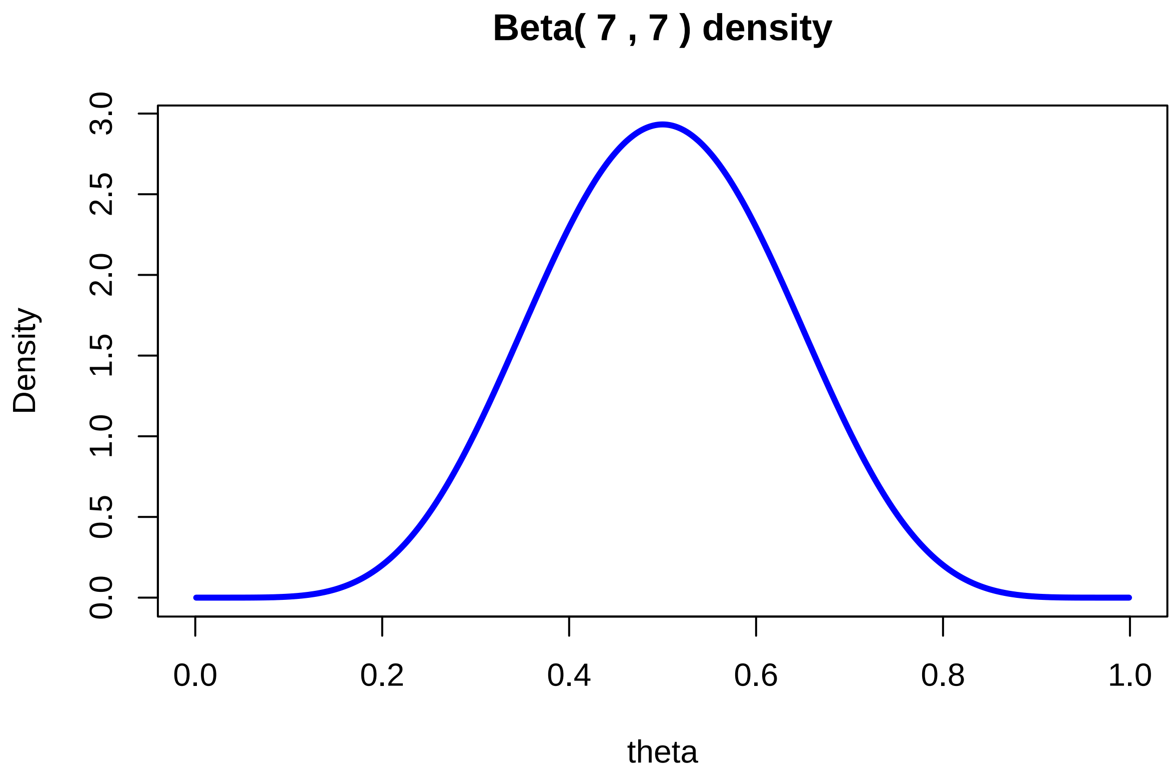
QUESTION 1

Part A)

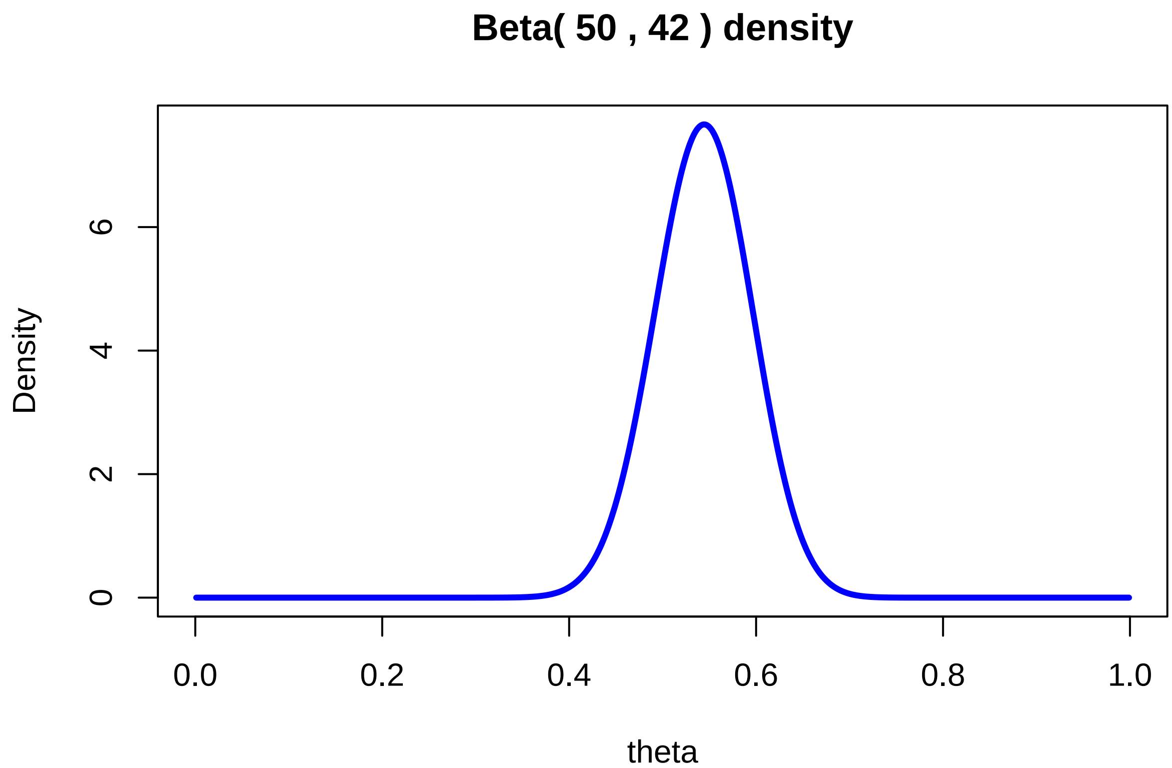
Given - Let y1,…., yn|θ ~ Bern(θ), and assume that you have obtained a sample with f = 35 failures in n = 78 trials. Assume a Beta(α0,β0) prior for θ and let $\alpha 0 = \beta 0 = 7$.

Draw 10000 random values (nDraws = 10000) from the posterior θ|y ~ Beta(α0 + s,β0 + f), where y = (y1,…,yn), and verify graphically that the posterior mean E[θ|y] and standard deviation SD[θ|y] converges to the true values as the number of random draws grows large.

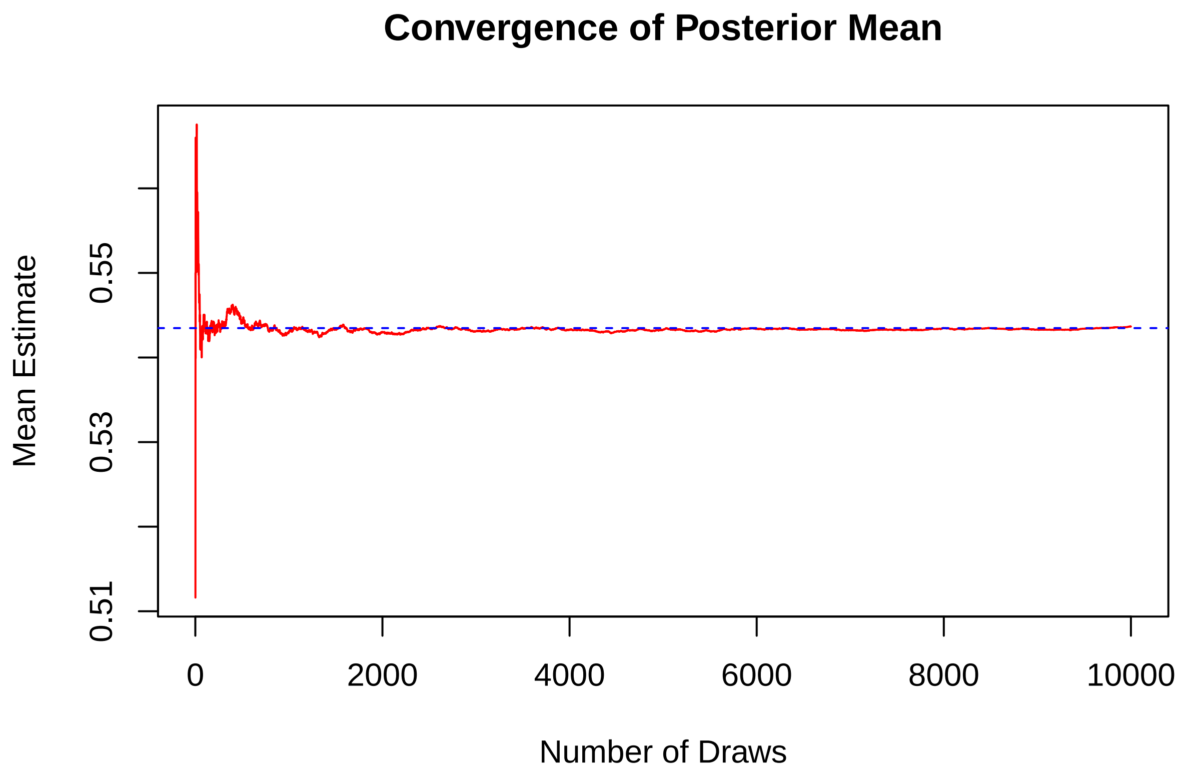
```
### Variables ###
f = 35  # Number of failures
n = 78  # Total trials
a0 = 7  # Prior alpha
b0 = 7  # Prior beta
s = n - f  # Number of successes
#HELPER FUNCTIONS WHICH WILL BE USED REPETITIVELY
meanBeta <- function(a, b) {
  return(a / (a + b))
}

varBeta <- function(a, b) {
  return((a * b) / (((a + b)^2) * (a + b + 1)))
}

BetaPlot <- function(a, b) {
  xGrid <- seq(0.001, 0.999, by = 0.001)
  prior = dbeta(xGrid, a, b)
  maxDensity <- max(prior)
  plot(xGrid, prior, type = 'l', lwd = 3, col = "blue", xlim = c(0,1), ylim = c(0, maxDensity),
       xlab = "theta", ylab = 'Density', main = paste('Beta(', a, ',', b, ') density'))
}
#Draw 10,000 samples, show posterior mean and SD converge correctly.
# & Visualize convergence of Beta posterior estimates to theoretical true values.

# Prior and Posterior Plots
BetaPlot(a0, b0)  # Prior density
```



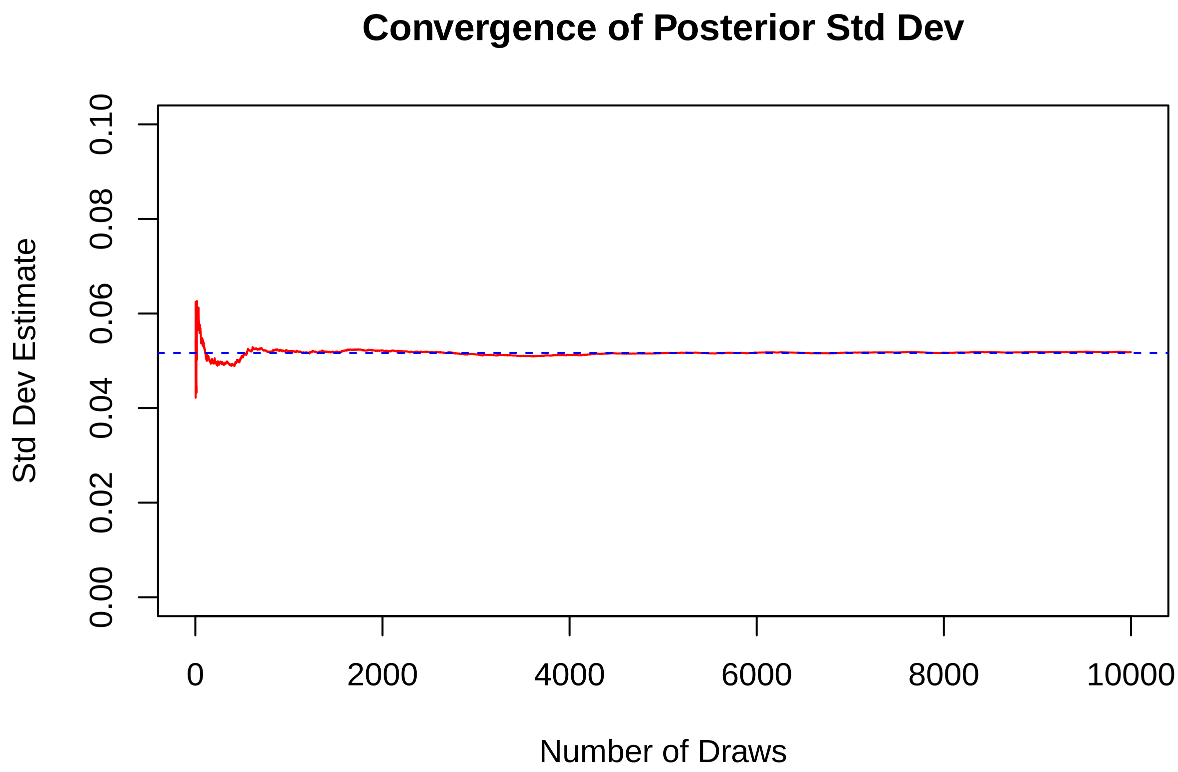

```
BetaPlot(a0 + s, b0 + f)  # Posterior density
```




```r
# Posterior Mean and Variance
meanPost = meanBeta(a0 + s, b0 + f)
varPost = varBeta(a0 + s, b0 + f)
stdPost = sqrt(varPost)

# Sampling from Posterior
Ndraws = 10000
samples = rbeta(Ndraws, shape1 = a0 + s, shape2 = b0 + f)

# Convergence of Mean and Std Dev
mean_vals = rep(0, Ndraws)
std_vals = rep(0, Ndraws)

for (i in 1:Ndraws) {
  mean_vals[i] = mean(samples[1:i])
  std_vals[i] = sd(samples[1:i])
}

plot(mean_vals, type = "l", col = "red", main = "Convergence of Posterior Mean",
     xlab = "Number of Draws", ylab = "Mean Estimate")
abline(h = meanPost, col = "blue", lty = 2)
```




```r
plot(std_vals, type = "l", col = "red", main = "Convergence of Posterior Std Dev",
     xlab = "Number of Draws", ylab = "Std Dev Estimate", ylim = c(0, 0.1))
abline(h = stdPost, col = "blue", lty = 2)
```



Part B)

Draw 10000 random values from the posterior to compute the posterior probability Pr(θ>0.5|y) and compare with the exact value from the Beta posterior.

```r
# Part (b) - Compute Probability P(theta > 0.5 | y)
threshold = 0.5
true_probability = 1 - pbeta(threshold, shape1 = a0 + s, shape2 = b0 + f)
simulated_probability = mean(samples > threshold)

print(paste("True Probability: ", round(true_probability, 4)))
```

```
## [1] "True Probability:  0.7991"
```

```r
print(paste("Simulated Probability: ", round(simulated_probability, 4)))
```

```
## [1] "Simulated Probability:  0.7986"
```

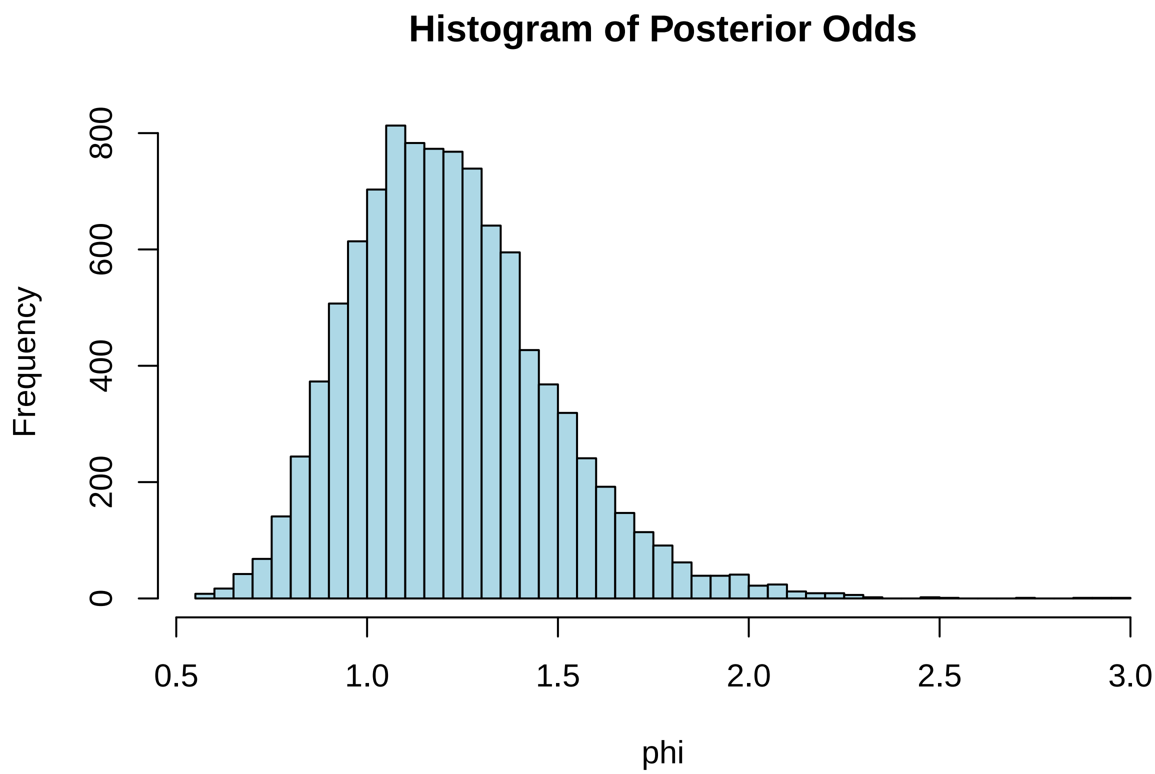
Part C)

Draw 10000 random values from the posterior of the odds $\Phi =$

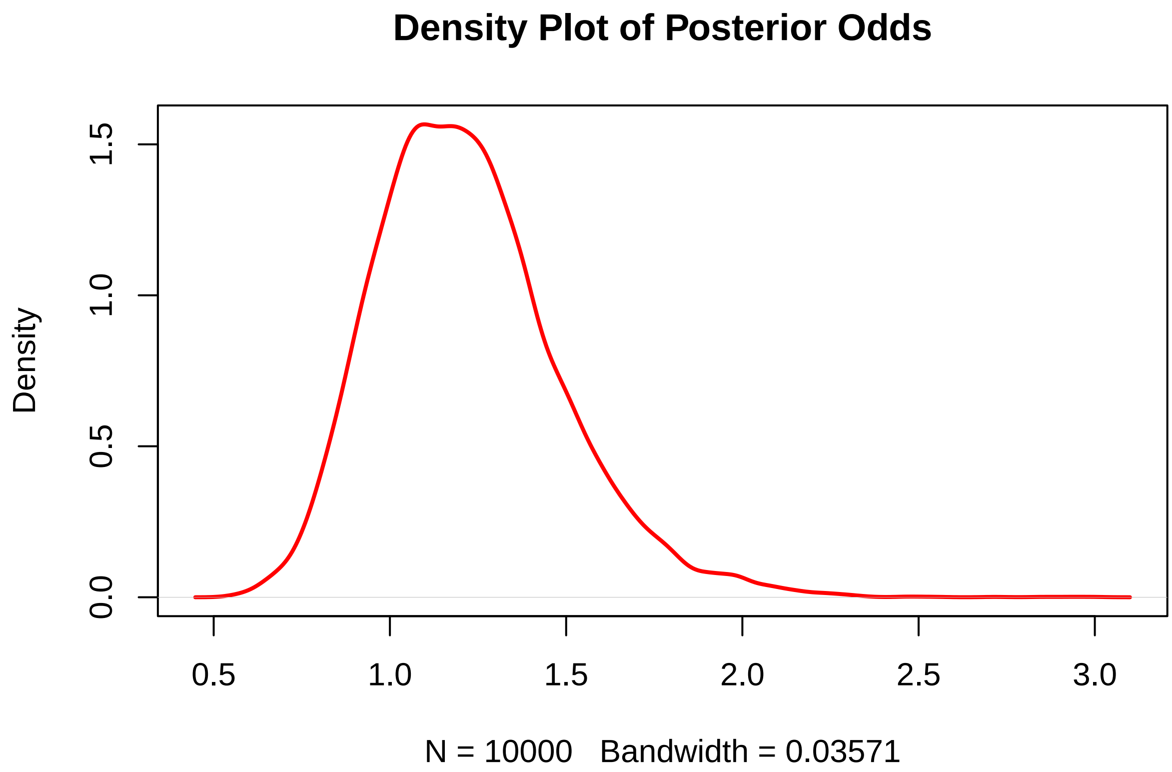
$$\theta/1-\theta$$

by using the previous random draws from the Beta posterior for θ and plot the posteriordistribution of θ.

```r
# Part (c) - Posterior Distribution of Odds phi
phi = samples / (1 - samples)
hist(phi, breaks = 50, main = "Histogram of Posterior Odds", xlab = "phi", col = "lightblue")
```




```r
plot(density(phi), main = "Density Plot of Posterior Odds", col = "red", lwd = 2)
```



QUESTION 2

Draw 10000 random values from the posterior of σ^2 by assuming $\mu = 3.65$ and plot the posterior distribution.

Part A)

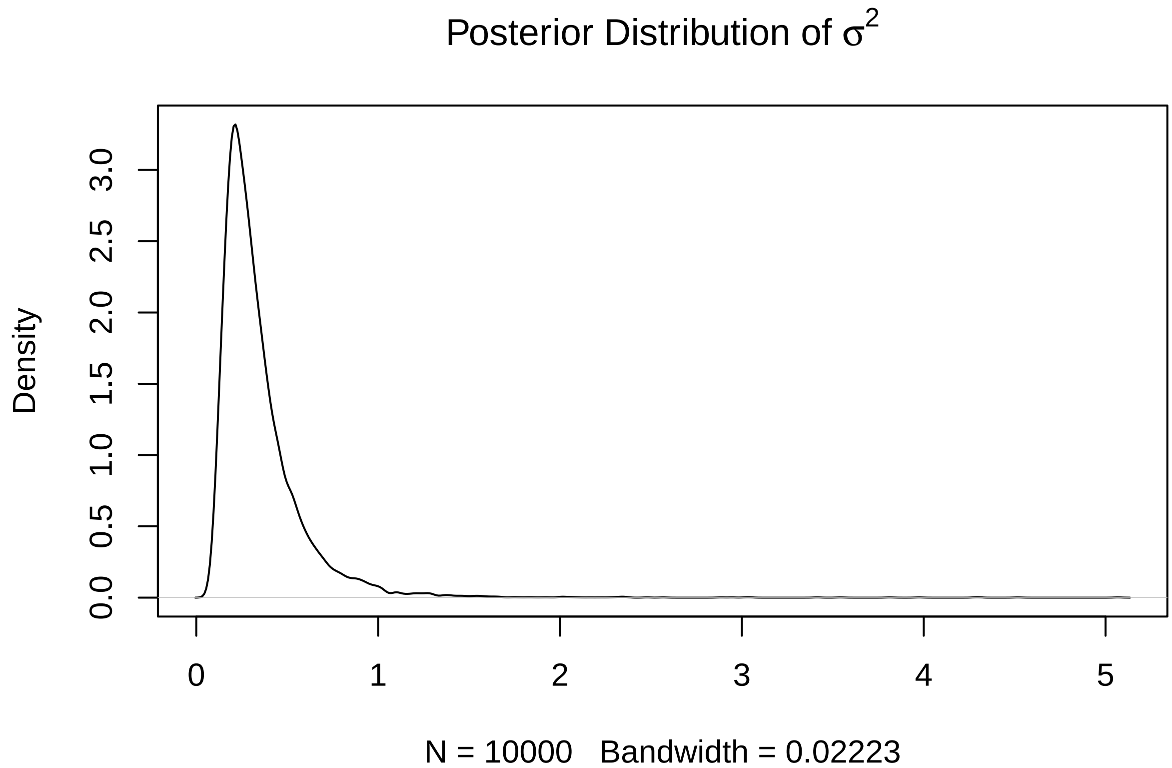
```r
# Load required libraries
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(bayestestR)  # Needed for HPDI calculation
```

```
## Warning: package 'bayestestR' was built under R version 4.4.3
```

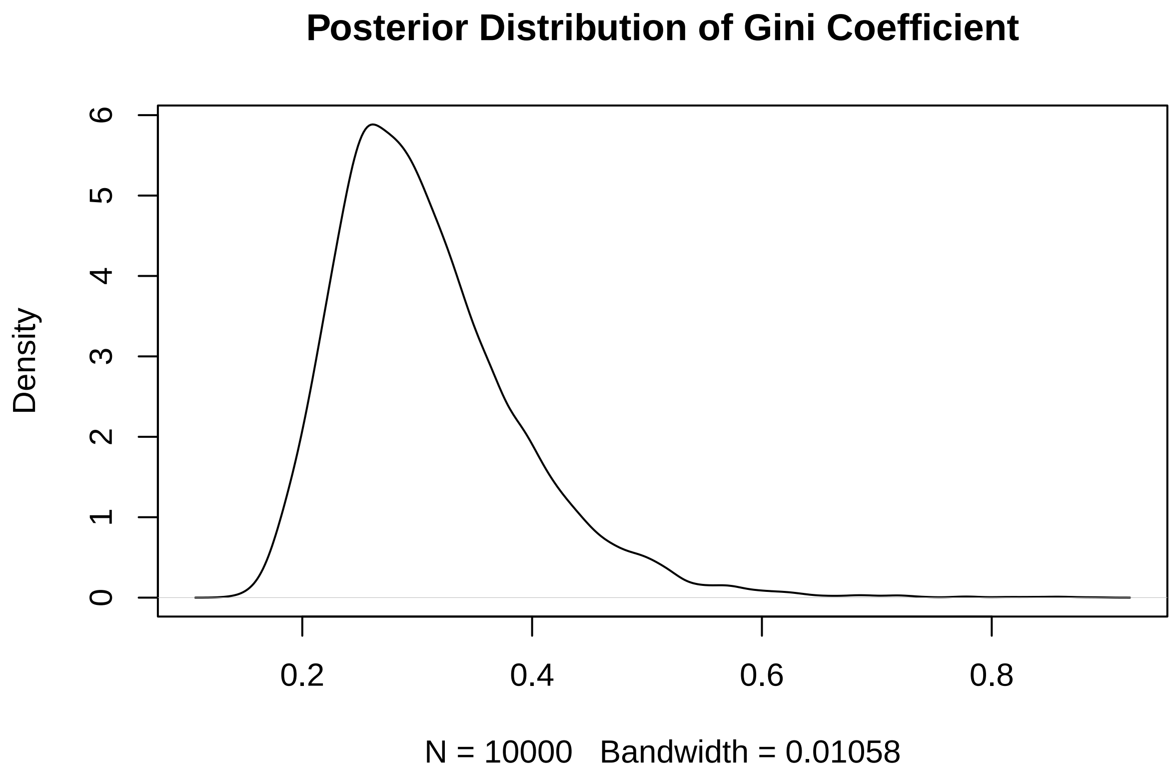
```r
# Given income data
income <- c(22, 33, 31, 49, 65, 78, 17, 24)

# (a) Compute tao^2 and draw 10,000 samples from the posterior of sigma^2
mu <- 3.65          # Known mean mu
Ndraw <- 10000      # Number of posterior samples
n <- length(income) # Sample size


# Function to compute tao² = (1/n) * Σ (log(y_i) - mu)^2
taoFunc <- function(y) {
  return(sum((log(y) - mu)^2) / length(y))
}
tao <- taoFunc(income)

# Draw from chi-squared distribution and compute sigma² posterior samples
set.seed(12345)
chi_vals <- rchisq(Ndraw, n)
sigmaSquared <- (n * tao) / chi_vals  # Posterior samples of sigma²

# Plot posterior distribution of sigma²
plot(density(sigmaSquared), main = expression(paste("Posterior Distribution of ", sigma^2)))
```



Part B)

Compute the posterior distribution of Gini coefficient G = 2 * $\Phi(\sigma$ / $\sqrt{2})$ - 1, where Φ is the CDF of standard normal

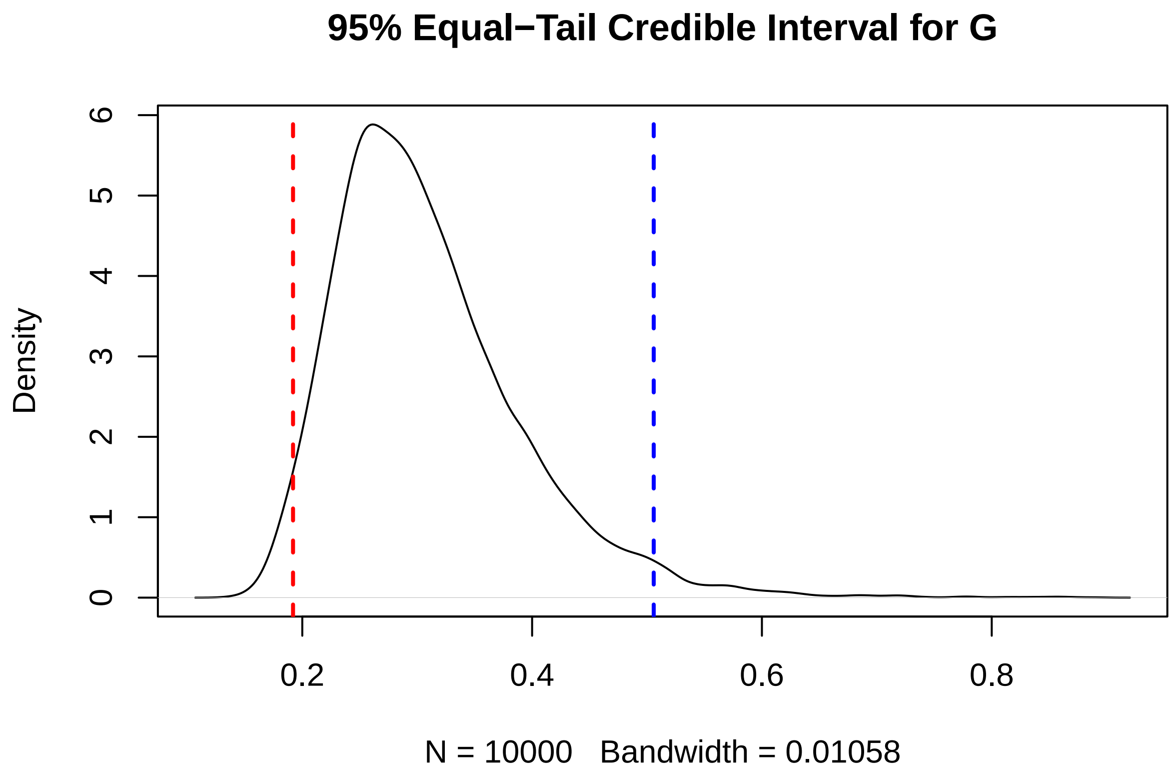
```
sigma <- sqrt(sigmaSquared)  # Get standard deviation from variance
G <- 2 * pnorm(sigma / sqrt(2)) - 1  # Compute Gini for each draw

# Plot the posterior distribution of Gini coefficient
plot(density(G), main = "Posterior Distribution of Gini Coefficient")
```



Part C)

Compute a 95% equal tail credible interval for G.

```r
# (c) Compute 95% Equal-Tail Credible Interval for Gini
sorted_G <- sort(G)
lower_ET <- sorted_G[round(Ndraw * 0.025)]
upper_ET <- sorted_G[round(Ndraw * 0.975)]
credible_interval_ET <- c(lower_ET, upper_ET)

# Plot the credible interval on Gini density
plot(density(G), main = "95% Equal-Tail Credible Interval for G")
abline(v = lower_ET, col = "red", lty = 2, lwd = 2)
abline(v = upper_ET, col = "blue", lty = 2, lwd = 2)
```




```r
# Print Equal-Tail Credible Interval
cat("95% Equal-Tail Credible Interval for Gini:\n")
```

```
## 95% Equal-Tail Credible Interval for Gini:
```

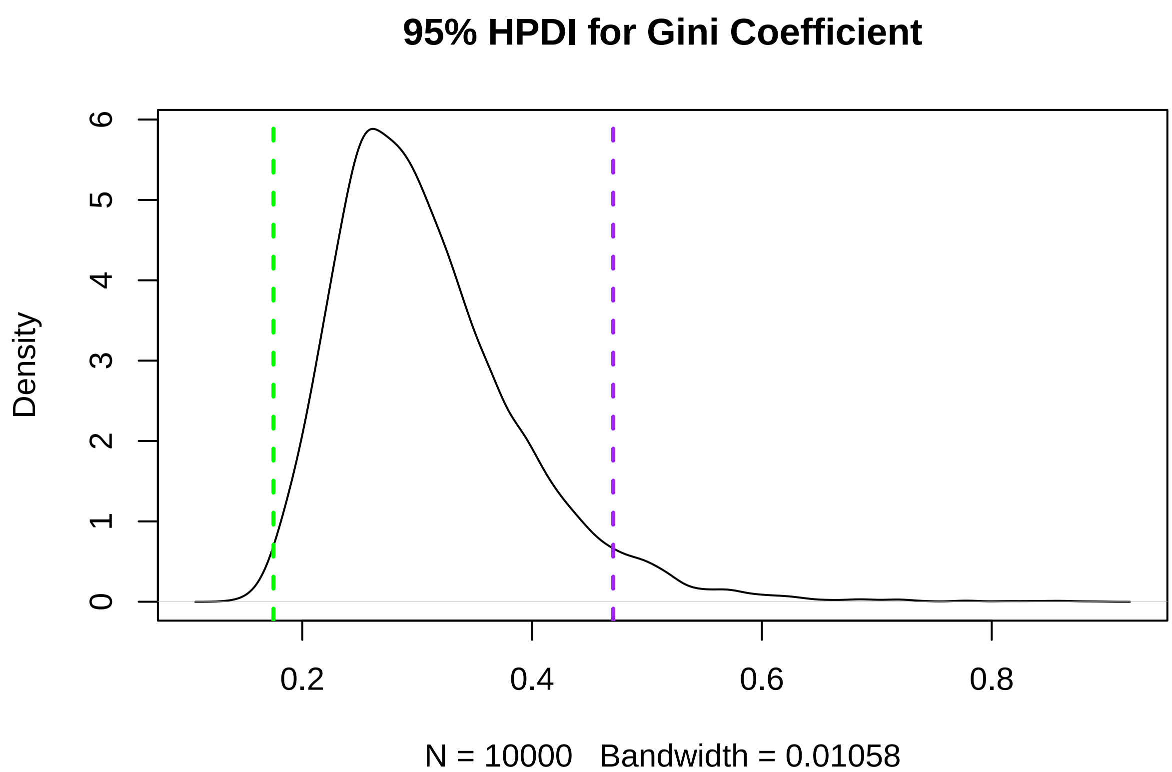
```r
print(credible_interval_ET)
```

```
## [1] 0.1919565 0.5058270
```

Part D)

Compute a 95% Highest Posterior Density Interval (HPDI) for G. Also, compare the two intervals in (c) and (d).

```r
# (d) Compute 95% Highest Posterior Density Interval (HPDI) using bayestestR
hpdi_interval <- hdi(G, ci = 0.95)

# Plot HPDI on top of Gini density
plot(density(G), main = "95% HPDI for Gini Coefficient")
abline(v = hpdi_interval$CI_low, col = "green", lty = 2, lwd = 2)
abline(v = hpdi_interval$CI_high, col = "purple", lty = 2, lwd = 2)
```




```r
# Print HPDI Interval
cat("95% Highest Posterior Density Interval (HPDI) for Gini:\n")
```

```
## 95% Highest Posterior Density Interval (HPDI) for Gini:
```

```r
print(hpdi_interval)
```

```
## 95% HDI: [0.17, 0.47]
```

```r
# Comparison of intervals
cat("\nComparison:\n")
```

```
##
## Comparison:
```

```r
cat("Equal-Tail Interval: ", round(credible_interval_ET[1], 4), "-", round(credible_interval_ET[2], 4),
```

```
## Equal-Tail Interval:  0.192 - 0.5058
```

```r
cat("HPDI Interval      : ", round(hpdi_interval$CI_low, 4), "-", round(hpdi_interval$CI_high, 4), "\n"
```

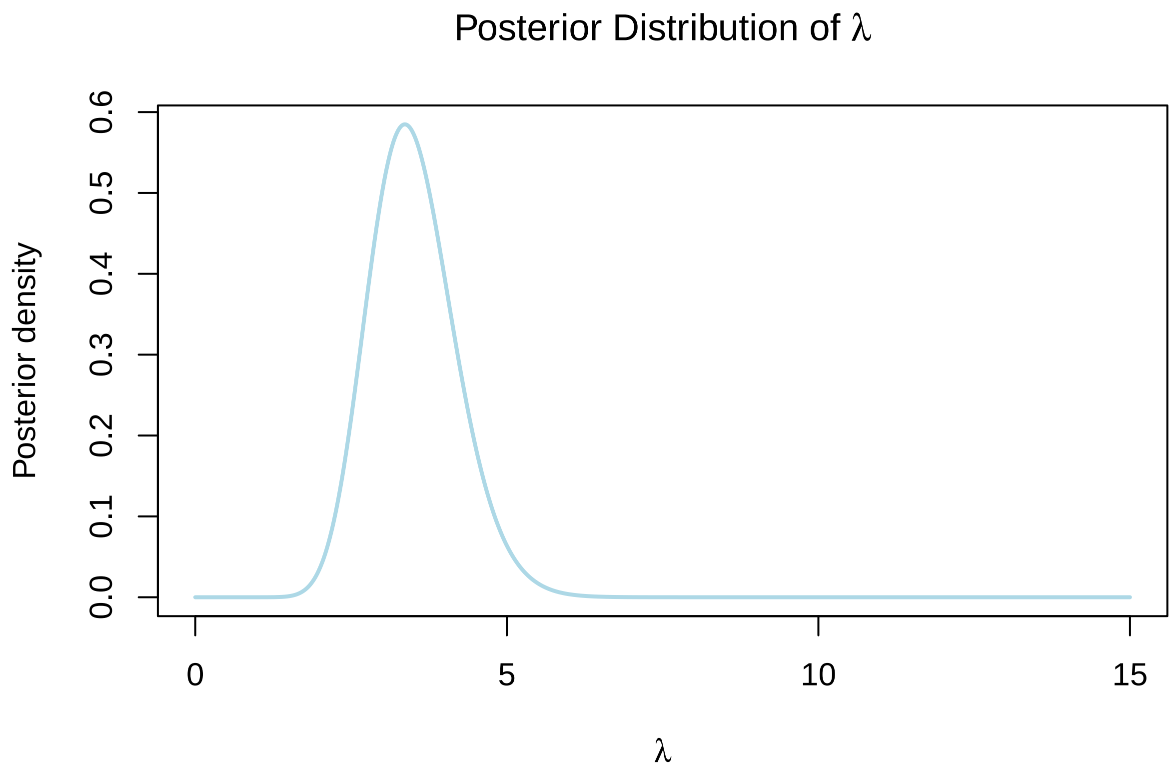
```
## HPDI Interval      :  0.1749 - 0.4706
```

QUESTION 3

Part A)

Derive the expression for what the posterior pdf p(λ|y, σ) is proportional to. Then, plot the posterior distribution of the average goals rate parameter λ over a fine grid of λ values.

```
############ Question 3 ##############

### Part A
# Data: Number of goals per match
y <- c(0, 2, 5, 5, 7, 1, 4)

# Parameters
sigma <- 5  # scale of half-normal prior
lambda_grid <- seq(0, 15, length.out = 1000)  # Grid of lambda values

# Prior: PDF of Half-normal distribution
prior <- function(lambda, sigma) {
  sqrt(2) / (sigma * sqrt(pi)) * exp(-lambda^2 / (2 * sigma^2))
}

# Likelihood: Product of Poisson
likelihood <- function(lambda, y) {
  sapply(lambda, function(l) prod(dpois(y, lambda = l)))
}

# Derivation of unnormalized posterior over the grid
unnormalized_posterior <- likelihood(lambda_grid, y) * prior(lambda_grid, sigma)

# Normalize the posterior
posterior <- unnormalized_posterior / sum(unnormalized_posterior * diff(lambda_grid)[1])
# Here, In normalization we multiply by 'diff(lambda_grid)[1]' because it is the width
# between grid points. Instead of integrating, we approximate the integral as a sum over
# small intervals.

# Plot posterior
plot(lambda_grid, posterior, type = "l", lwd = 2, col = "lightblue",
     xlab = expression(lambda), ylab = "Posterior density",
     main = expression(paste("Posterior Distribution of ", lambda)))
```



Part B)

The posterior mode of λ from the information above is -

```
### Part B
# Find posterior mode (MAP estimate)
posterior_mode <- lambda_grid[which.max(posterior)]
cat("Posterior mode of lambda:", posterior_mode, "\n")
```

```
## Posterior mode of lambda: 3.363363
```