TDDE07 Bayesian Learning - Lab 1

Erik Linder-Norén - erino397

2017-04-20

1. Bernoulli ... again.

(a)

Combining Bernoulli likelihood of 20 samples consisting of 14 successes and 6 failures with a Beta prior with parameters $\alpha_0 = \beta_0 = 2$ results in a Beta posterior $\theta|y \sim Beta(s + \alpha_0, f + \beta_0) \sim Beta(16, 8)$ from which to draw random numbers. When the number of draws increases the sample mean and sample standard deviation converges towards the theoretical mean and standard deviation. This convergence can be seen in figure 1 and figure 2.

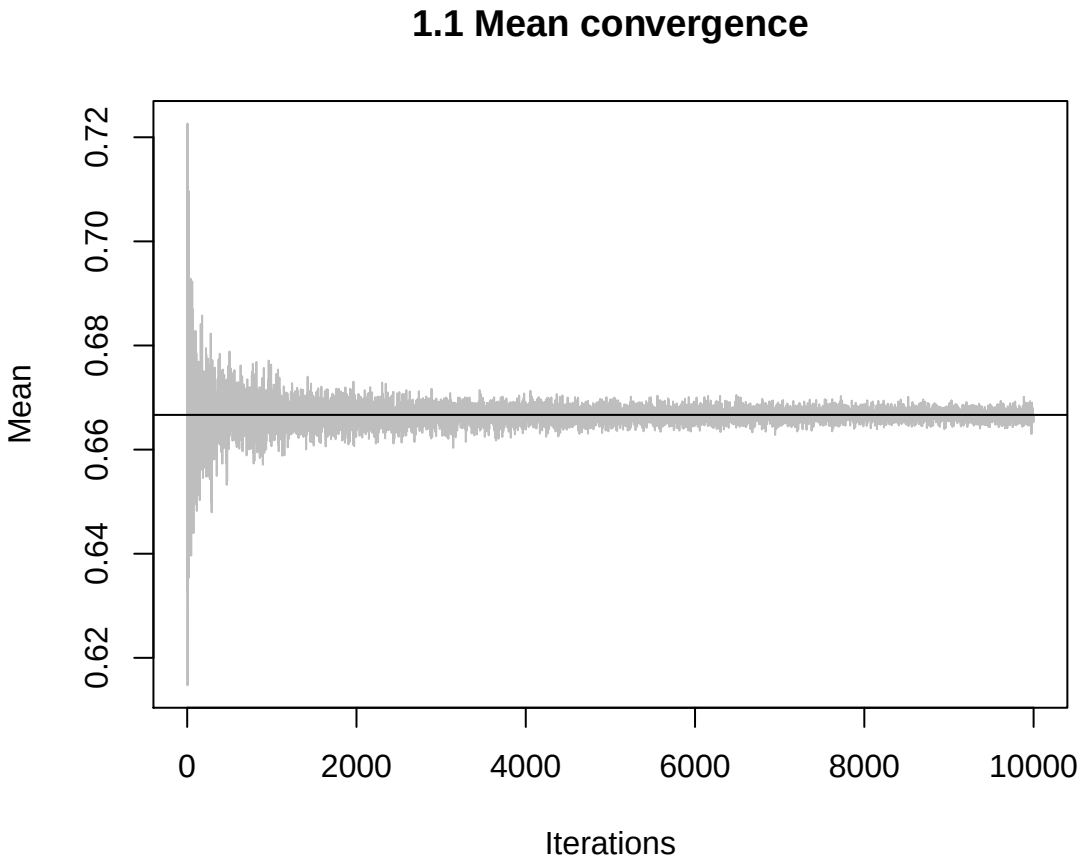

Figure 1: Mean convergence towards theoretical value

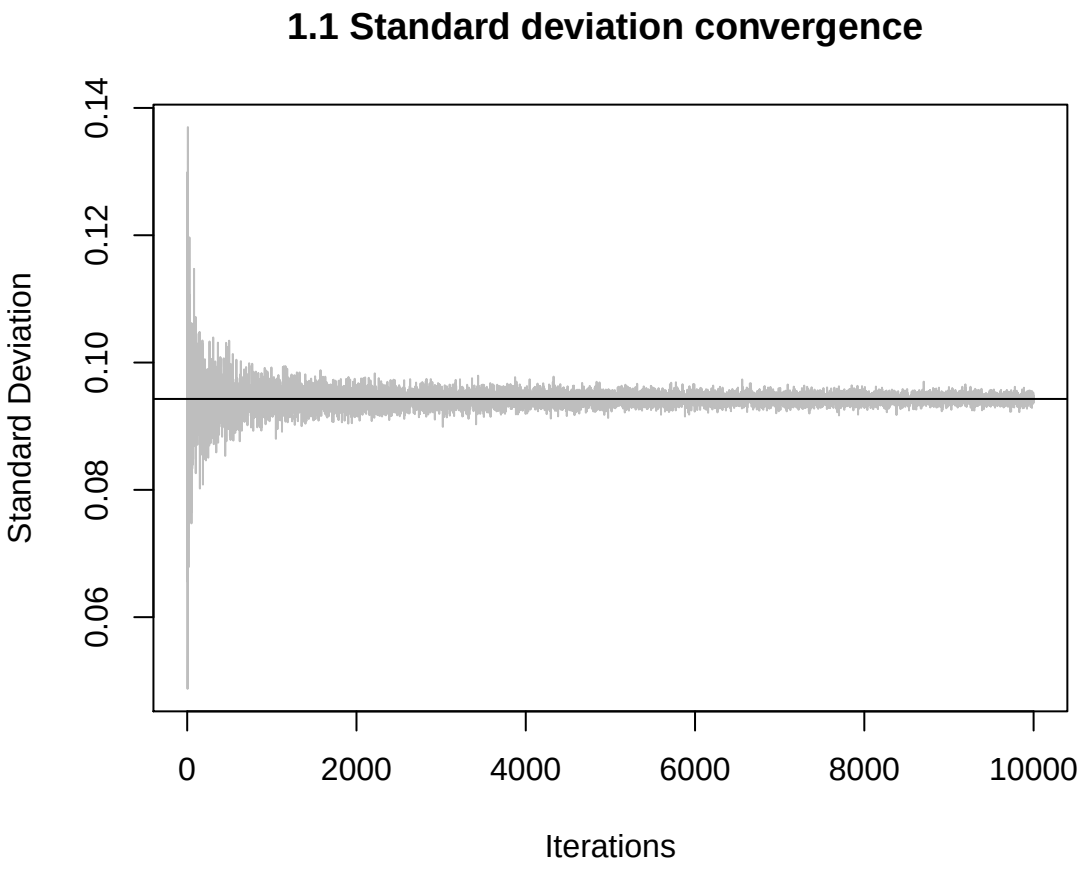

Figure 2: Standard deviation convergence towards theoretical value

(b)

I drew 10 000 samples from the posterior and calculated $Pr(\theta < 0.4|y)$. This was done a couple of times and the values ranged from 0.35 to 0.43. The exact probability was 0.397, so the probability calculated based on the simulation gave a pretty good approximation to the exact value.

(c):

After drawing 10 000 samples from the posterior and calculating the log odds as $\phi = log(\frac{\phi}{1-\phi})$ the histogram distribution was plotted against the density. The results can be seen in figure 3.

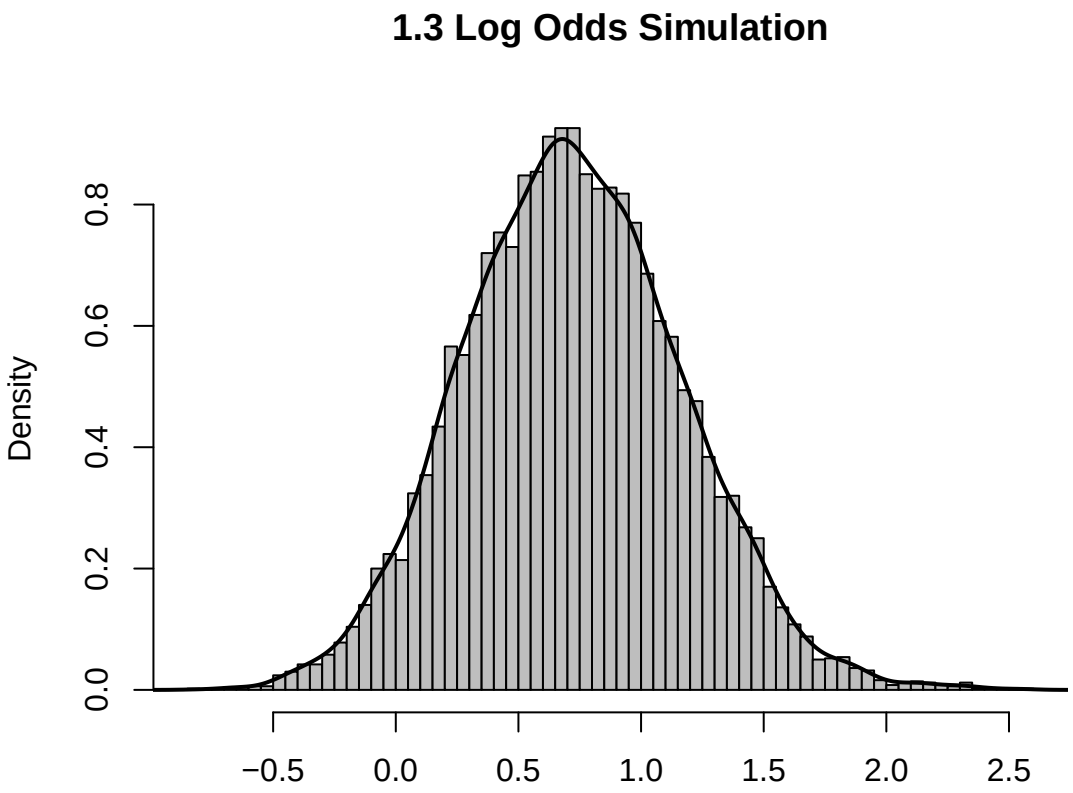

Figure 3: Log-odds

2. Log-normal distribution and the Gini coefficient

(a)

Simulating 10 000 draws from the posterior and plotting the histogram of those draws against the true posterior distribution $Inv - \chi^2(n, \tau^2)$ showed a close similarity. The results can be seen in figure 4.

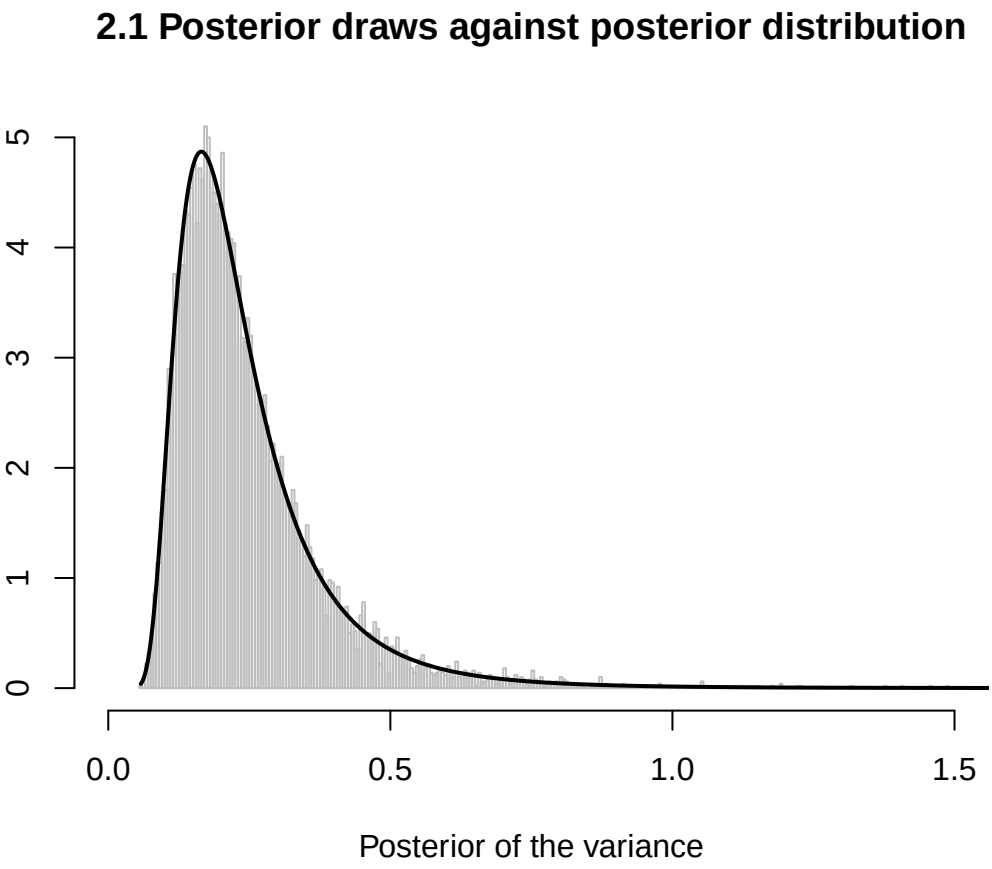

Figure 4: Simulation of the posterior distribution $Inv - \chi^2(n, \tau^2)$

(b)

Using the posterior draws for σ in (a) to determine the Gini coefficient gave the results seen in figure 5.

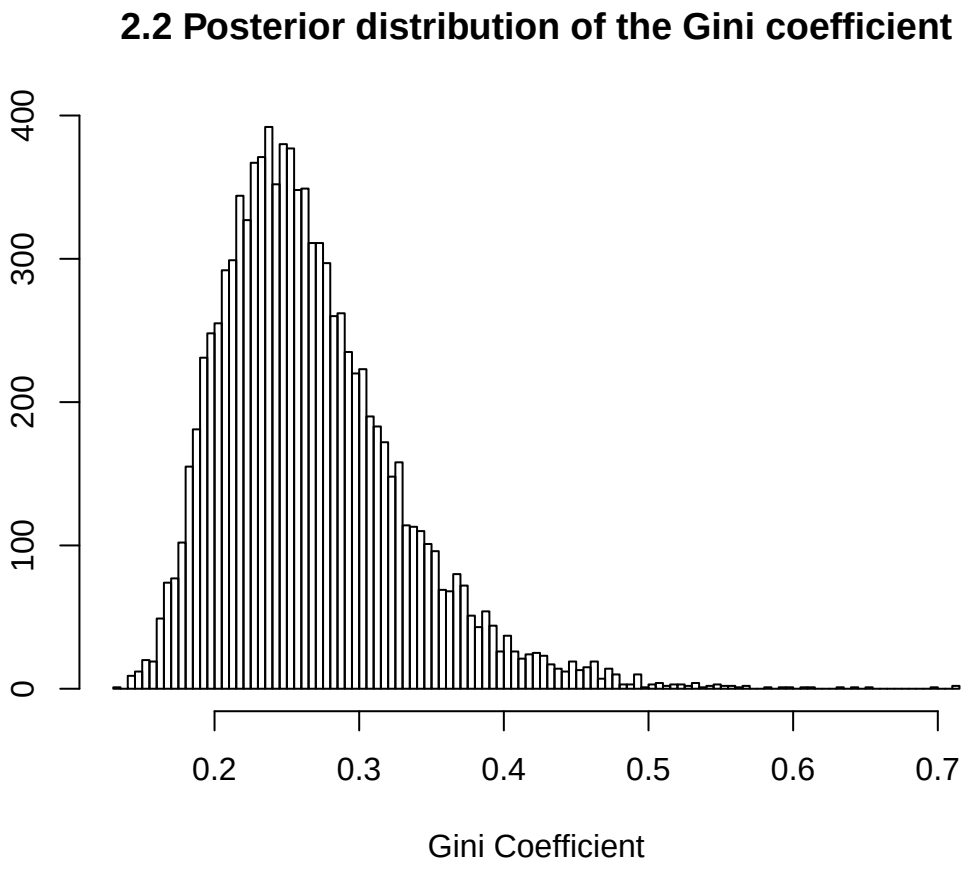

Figure 5: Gini coefficient

(c)

I calculated the Equal Tail Interval and Highest Posterior Density using the Gini coefficient calculated in (b) to get the results seen in figure 6. The right tail of the density seemed to shift the Equal Tail Interval to the right. The Highest Posterior Density interval was not affected by the tail and seemed to give better results in this case.

3. Bayesian inference for the concentration parameter in the von Mises distribution.

The posterior distribution of κ was calculated as the product of likelihoods multiplied with the prior. The posterior mode of kappa was calculated as the κ value that gives the highest probability. This value was 2.12. The plot can be seen in figure 7.

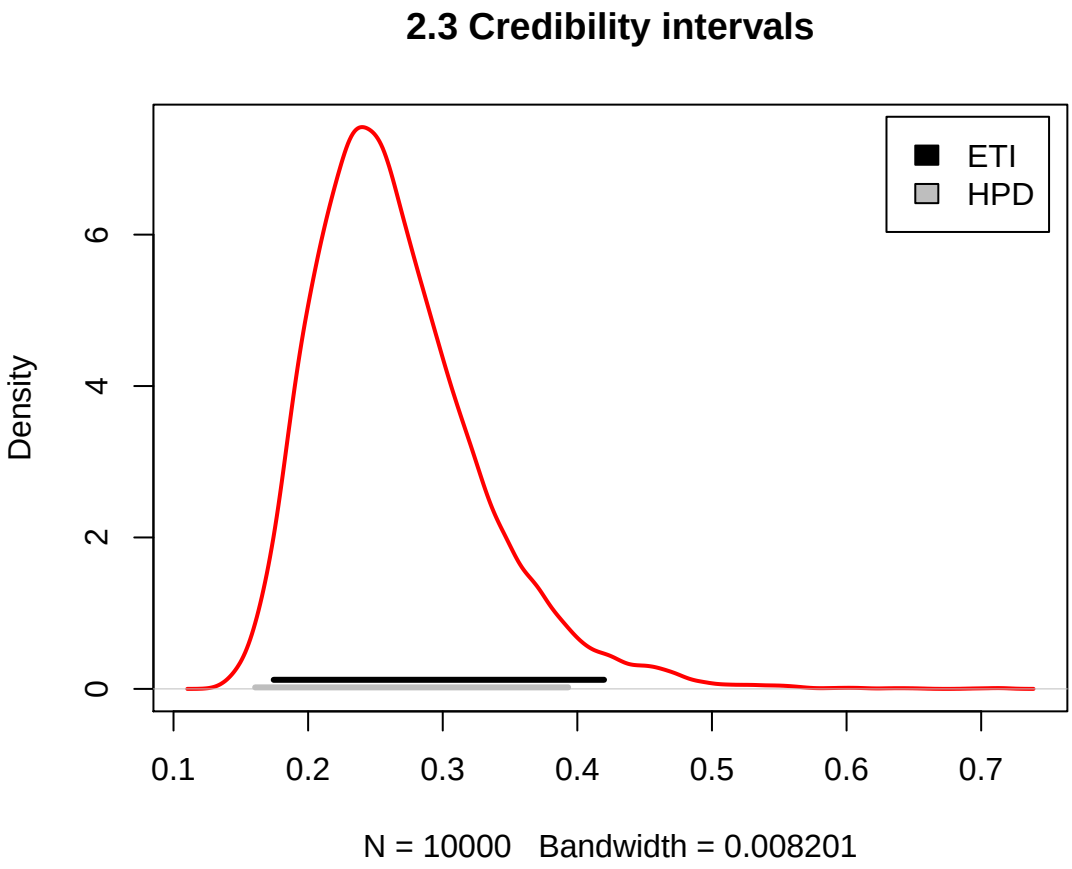

Figure 6: Credibility intervals

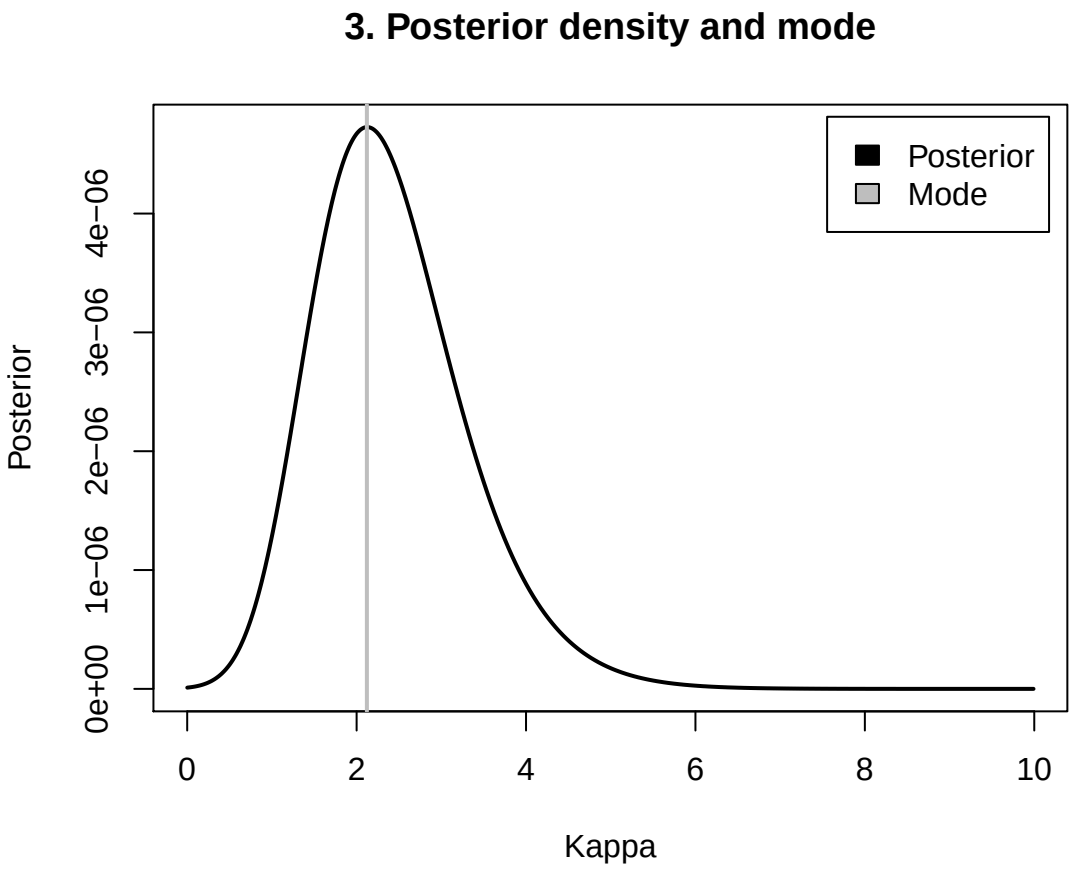

Figure 7: Von Mises posterior distribution

Assignment 1

```
grid_w = 6
grid_h = 5

# Experiment Setup
a = 2
b = 2
n = 20
s = 14
p = s / n

# Posterior alpha and beta
a_n = a + n * p
b_n = b + n * (1 - p)

# Theoretical mean and standard deviations
mean = a_n / (a_n + b_n)
std_dev = sqrt(a_n*b_n / (((a_n + b_n)**2)*(a_n + b_n + 1)))

# 1.1

calc_mean_stddev <- function(n_draw){
  post_draws = rbeta(n_draw, a_n, b_n)
  m = mean(post_draws)
  s = sd(post_draws)

  return(c(m, s))
}

# Calculate mean and variation for 2 to 10000 draws from posterior
data = sapply(2:10000, calc_mean_stddev)

# Save plot for the convergence of mean and variation

pdf("plots/1_1_mean.pdf", width=grid_w, height=grid_h)
  mean_values = data[1,]
  plot(mean_values,
       type="l",
       col="gray",
       xlab="Iterations",
       ylab="Mean",
       main = '1.1 Mean convergence')
  abline(h=mean)
dev.off()

pdf("plots/1_1_std_dev.pdf", width=grid_w, height=grid_h)
  standard_deviations = data[2,]
  plot(standard_deviations,
       type="l",
       col="gray",
       xlab="Iterations",
       ylab="Standard Deviation",
```

```
       main = '1.1 Standard deviation convergence ')
  abline(h=std_dev)
dev.off()


# 1.2

n_draws = 10000
post_draws = rbeta(n_draws, a_n, b_n)
p_draws = 100 * sum(post_draws <= 0.4) / length(post_draws)
p_true = 100 * pbeta(.4, a_n, b_n)

# 1.3

n_draws = 10000
post_draws = rbeta(n_draws, a_n, b_n)
lodds = log(post_draws/(1-post_draws))

pdf("plots/1_3_log_odds.pdf", width=grid_w, height=grid_h)
hist(lodds, 100,
     prob=TRUE,
     col="gray",
     xlab="",
     ylab="",
     main = '1.3 Log Odds Simulation')
lines(density(lodds), lwd=2)
dev.off()
```

Assignment 2

```
require("geoR")

grid_w = 6
grid_h = 5


# 2.1

y = c(14, 25, 45, 25, 30, 33, 19, 50, 34, 67)
mean = 3.5
n = length(y)
n_draws = 10000

# Sample variance
tau_sq = sum((log(y) - mean)^2) / n
# X ~ chi(n)
X_draws = rchisq(n_draws, n)
# This is a draw from inv_chi(n, tausq)
var = n * tau_sq / X_draws
interval = seq(min(var), max(var), 0.001)
invchisq = dinvchisq(interval, n, tau_sq)

pdf("plots/2_1_chi_draws.pdf", width=grid_w, height=grid_h)
# Plot draws against density
hist(var,
     500,
     border="gray",
     prob=TRUE,
     xlim=c(0,1.5),
     xlab="Posterior of the variance",
     ylab="",
     main = '2.1 Posterior draws against posterior distribution')
lines(interval, invchisq, lwd=2)
dev.off()


# 2.2

z = sqrt(var / 2)
G = 2*pnorm(z)-1

pdf("plots/2_2_gini_draws.pdf", width=grid_w, height=grid_h)
hist(G,
     100,
     xlab="Gini Coefficient",
     ylab="",
     main = '2.2 Posterior distribution of the Gini coefficient')
dev.off()

# 2.3
```

```r
# Equal tail interval
equal_tail_interval = quantile(G, probs=c(0.025, 0.975))

# Highest Posterior Density
gd = density(G)
# Order x and y by y from high to low
ordered_x = gd$x[order(-gd$y)]
ordered_y = gd$y[order(-gd$y)]

# Iterate until 95% of prob. is captured
prob_mass = 0
total_mass = sum(gd$y)
for(i in 1:length(gd$y)){
  prob_mass = prob_mass + ordered_y[i]
  if(prob_mass / total_mass >= 0.95){
    break
  }
}

# Calculate the interval
a = min(ordered_x[1:i])
b = max(ordered_x[1:i])
highest_posterior_density = c(a, b)

pdf('plots/2_3_posterior_intervals.pdf', width=grid_w, height=grid_h)
plot(gd, col="red", lwd=2, main="2.3 Credibility intervals")
lines(equal_tail_interval, rep(0.12, 2), col="black", lwd=3)
lines(highest_posterior_density, rep(0.02, 2), col="gray", lwd=3)
legend("topright",
       legend = c("ETI","HPD"),
       fill = c("black", "gray"),
       inset = 0.02)
dev.off()
```

Assignment 3

```
grid_w = 6
grid_h = 5

# 3.1

y = c(-2.44, 2.14, 2.54, 1.83, 2.02, 2.33, -2.79, 2.23, 2.07, 2.02)
mean = 2.39
k = seq(0.001,10,0.01)

calc_prob = function(k_val, y_val){
  prob = exp(k_val * cos(y_val - mean)) / (2*pi*besselI(k_val, 0))

  return (prob)
}

calc_post = function(k_val){
  probabilities = sapply(y, calc_prob, k_val=k_val)
  prior = dexp(k_val)
  posterior = prod(probabilities) * prior

  return (posterior)
}

posterior = sapply(k, calc_post)

pdf("plots/3_posterior_mode.pdf", width=grid_w, height=grid_h)
plot(k,
     posterior, type="l",
     col="black",
     lwd=2,
     xlab="Kappa",
     ylab="Posterior",
     main="3. Posterior density and mode")

# 3.2

# Find the k value which maximizes posterior
mode = k[which.max(posterior)]

# Plot vertical line where k maximizes posterior
abline(v=mode, col="gray", lwd=2)

# Legend for posterior and mode
legend('topright',
       c('Posterior', 'Mode'),
       fill=c("black", "gray"),
       inset=0.02)
dev.off()
```