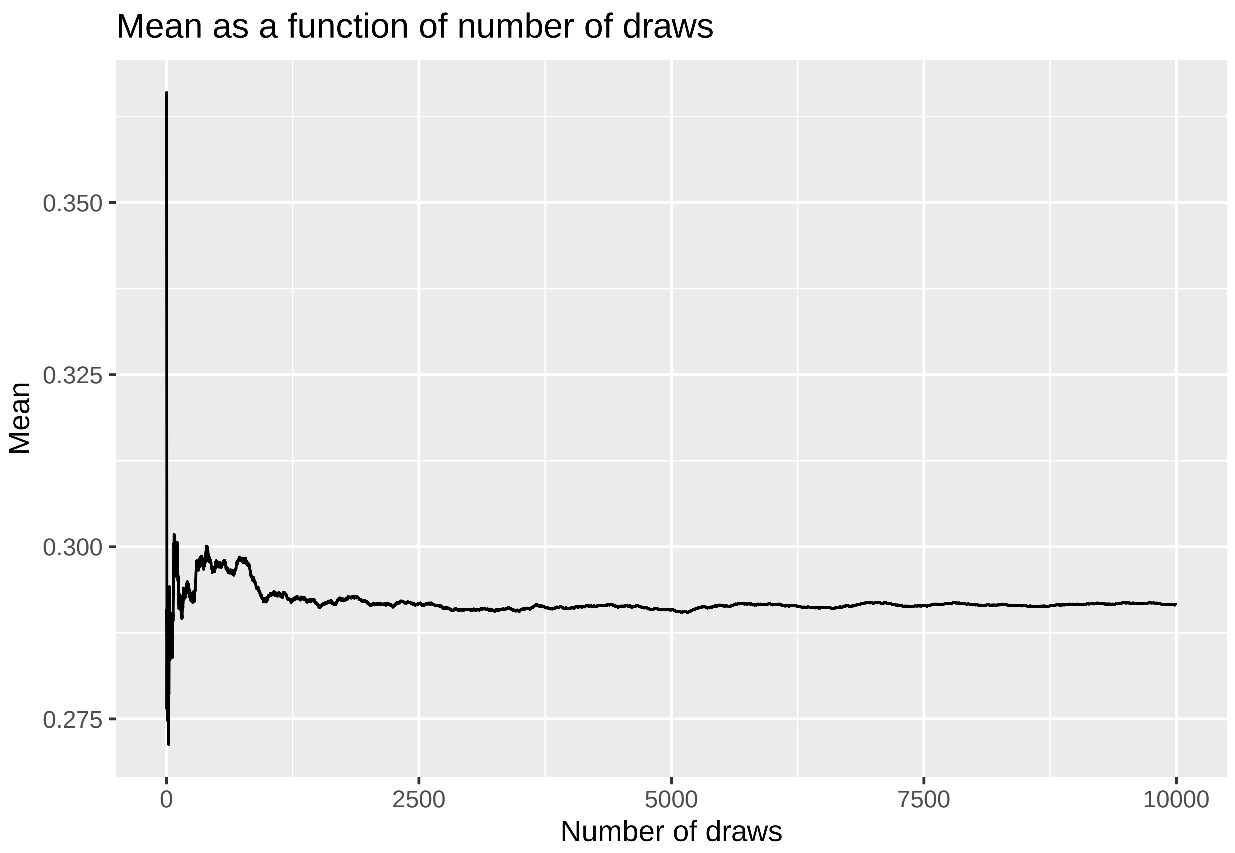
Lab 1 - TDDE07

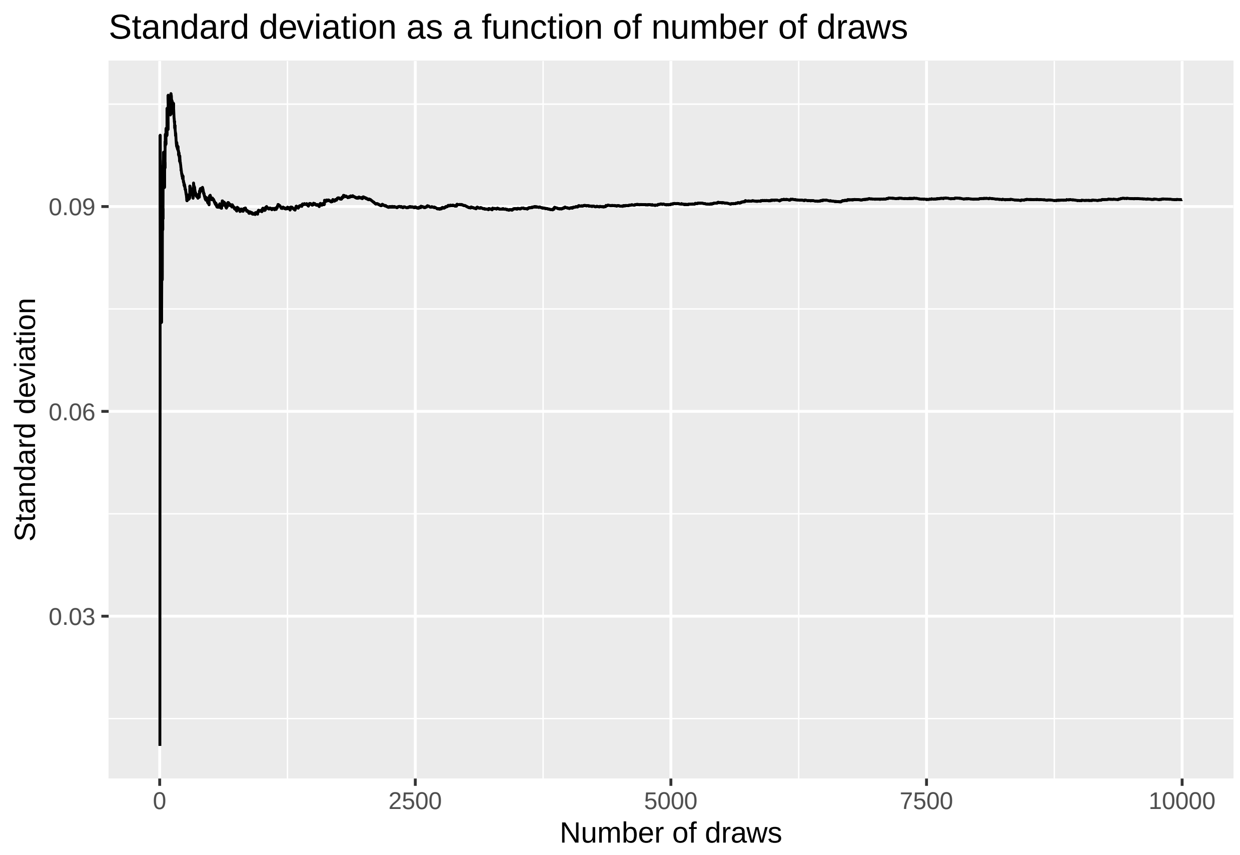
Axel Holmberg (axeho681), Wilhelm Hansson (wilha431)

1.

a)



Above is a graph that shows the mean as a function of the number of draws. As one can see it converges to about 0.292. To calculate the true value for the mean of the posterior the following values are used $\alpha = \alpha_o + s = 2 + 5$ and $\beta = \beta_0 + f = 2 + 15$ and the formula to calculate the mean of a beta distribution is $E(\theta|y) = \alpha/(\alpha + \beta)$ where $\theta|y \sim Beta(a_0 + s, b_0 + f) = Beta(7, 17)$. The true value of the mean is 0.2917.



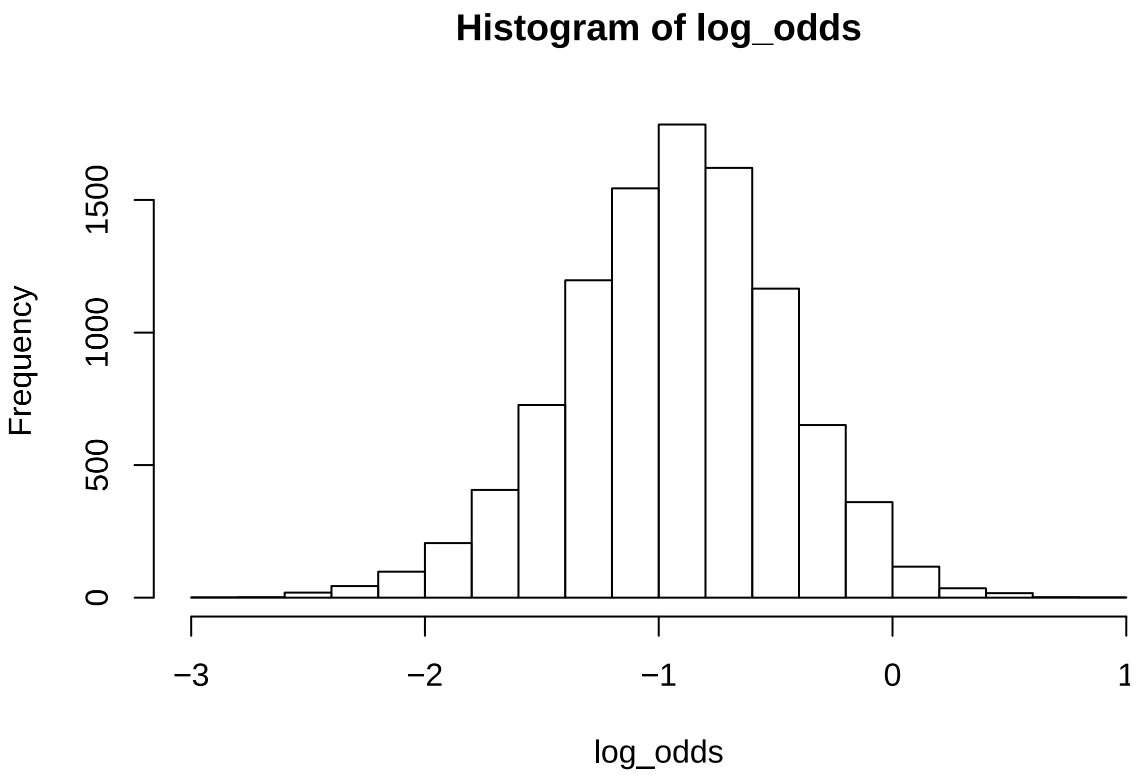
Above is a graph that shows the standard deviation as a function of the number of draws. As one can see it converges to about 0.091. To calculate the true value for the standard deviation of the posterior the following values are used $\alpha = \alpha_o + s = 2 + 5$ and $\beta = \beta_0 + f = 2 + 15$ and the formula to calculate the standard deviation of a beta distribution is $SD(\theta|y) = \sqrt{V(\theta|y)} = \sqrt{\alpha\beta/((\alpha+\beta)^2(\alpha+\beta+1))}$ where $\theta|y \sim Beta(a_0 + s, b_0 + f) = Beta(7, 17)$. The true value of the standard deviation is 0.0901.

b)

```
##      1
## 0.4392
```

The true value of the probability is 0.4399472 and the the calculated probability is 0.4392. They are almost equal.

c)

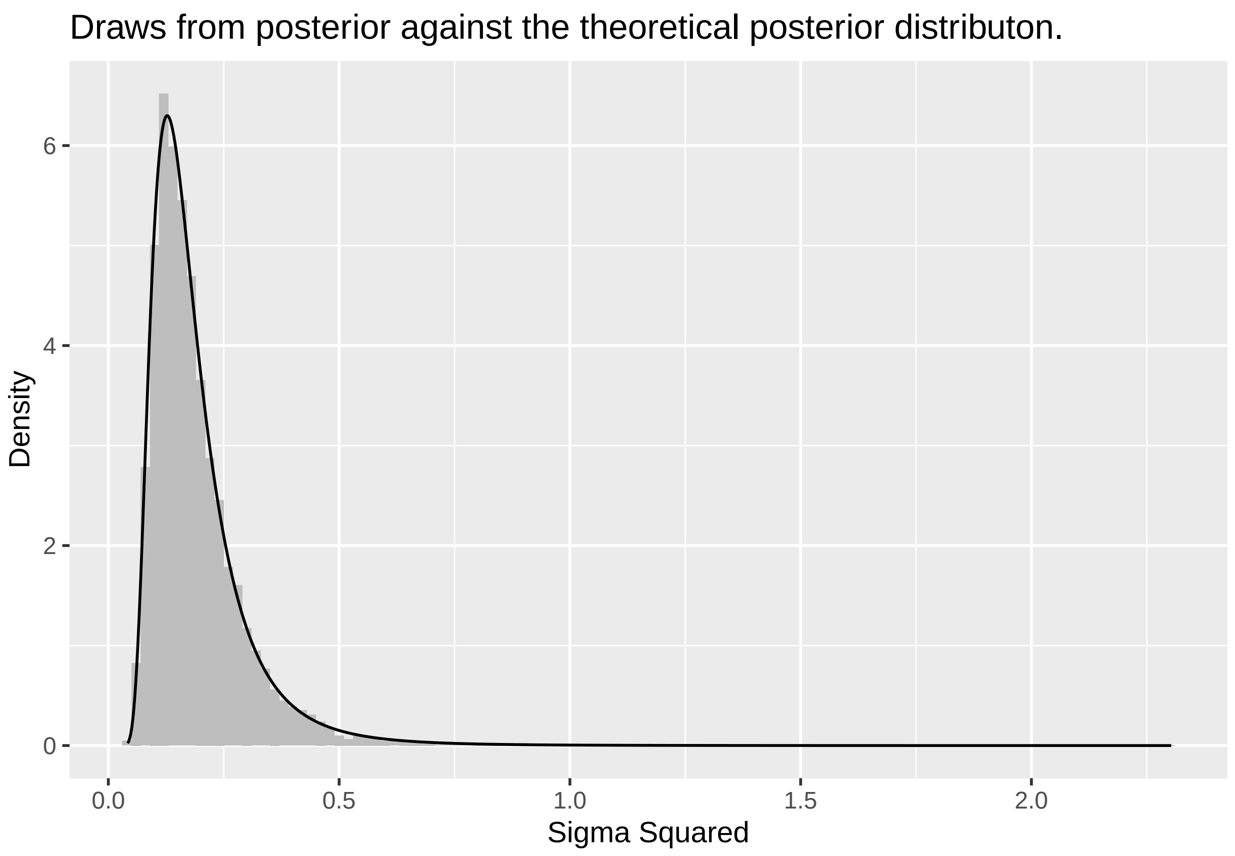


```
##
## Call:
##  density.default(x = log_odds)
##
## Data: log_odds (10000 obs.); Bandwidth 'bw' = 0.06469
##
##        x                 y
##  Min.   :-3.17277   Min.   :0.0000071
##  1st Qu.:-2.08423   1st Qu.:0.0027689
##  Median :-0.99569   Median :0.0560336
##  Mean   :-0.99569   Mean   :0.2294409
##  3rd Qu.: 0.09285   3rd Qu.:0.4063222
##  Max.   : 1.18139   Max.   :0.8987902
```

Above is a histogram of the log-odds. The log-odds seems to be normally distributed with a mean of about -1.

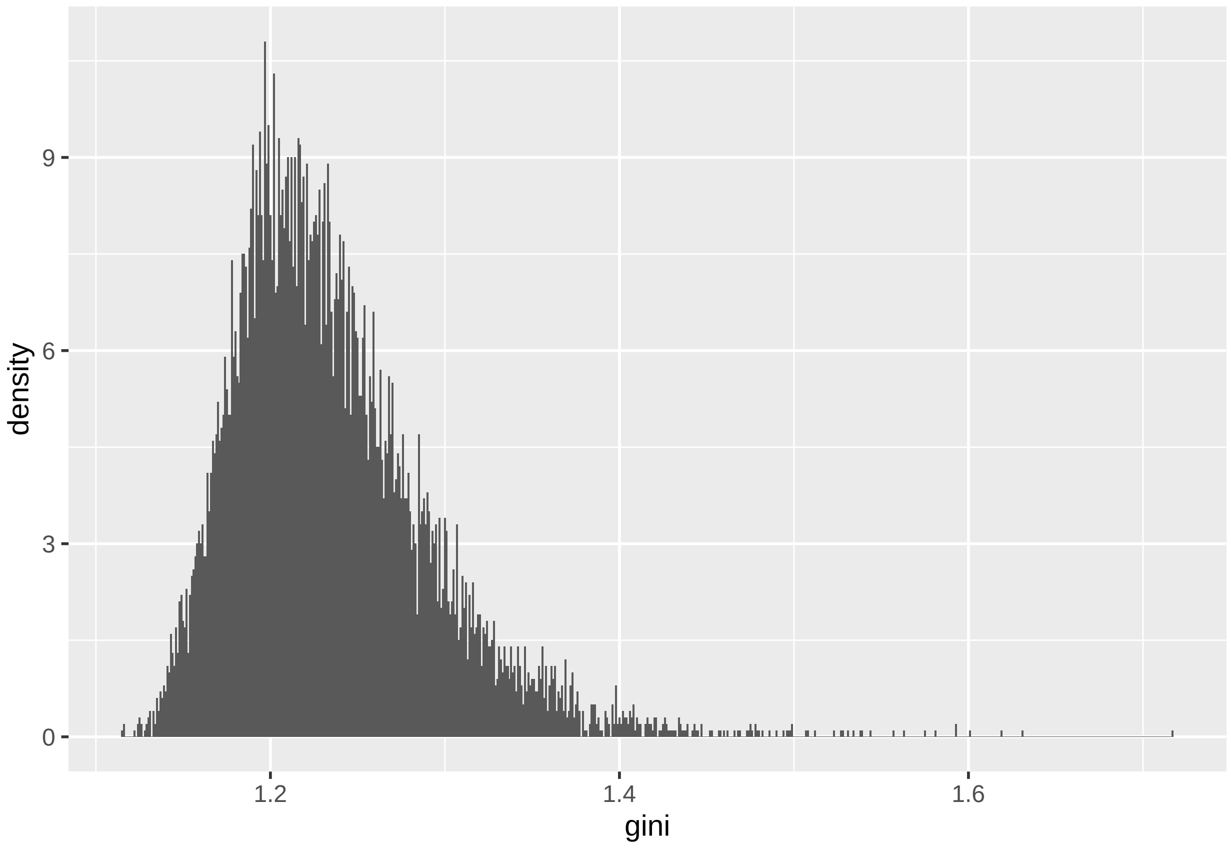
2.

a)



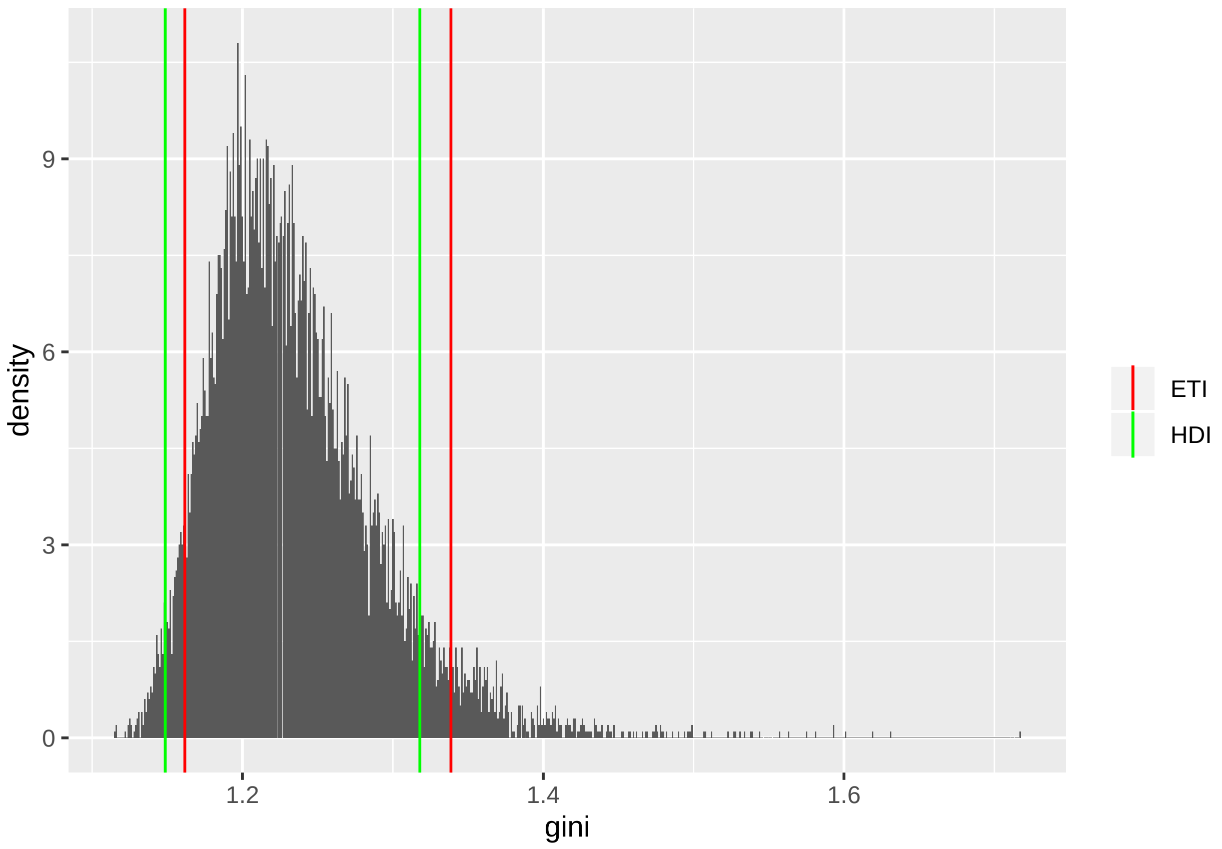
Above is the graph for the draws from posterior plotted against the theoretical posterior distribution. What this shows is that the draws from the posterior fits the theoretical posterior distribution well.

b)



Plot show density of calculated gini coefficients. The coefficients are calculated from the draws of a).

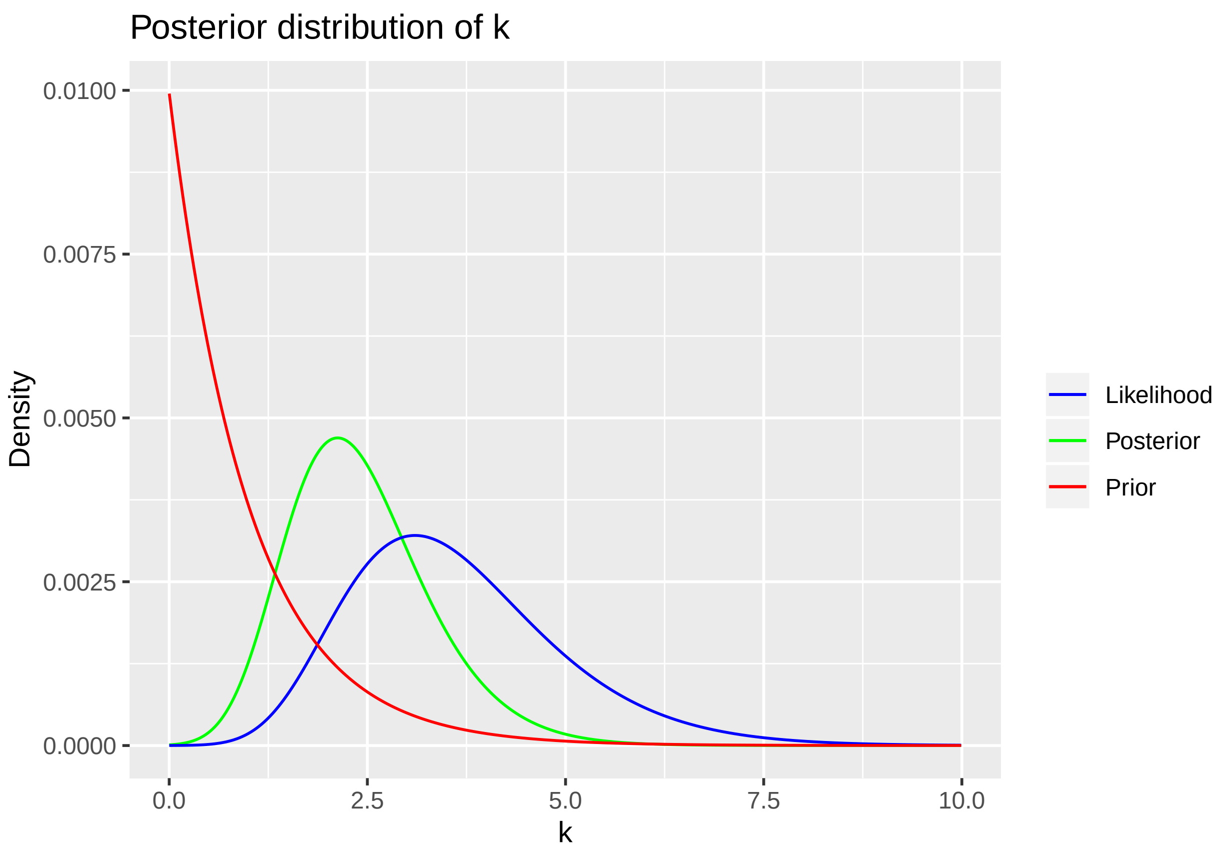
c)



In the graph above both the ETI and the HDI interval. Where the HDI-interval is a CI with respect to the density and the ETI just takes the CI of the values without respect to the density. If one has a non-symmetrical distribution, one should use a HDI rather than an ETI when calculating CI as it takes the density into consideration.

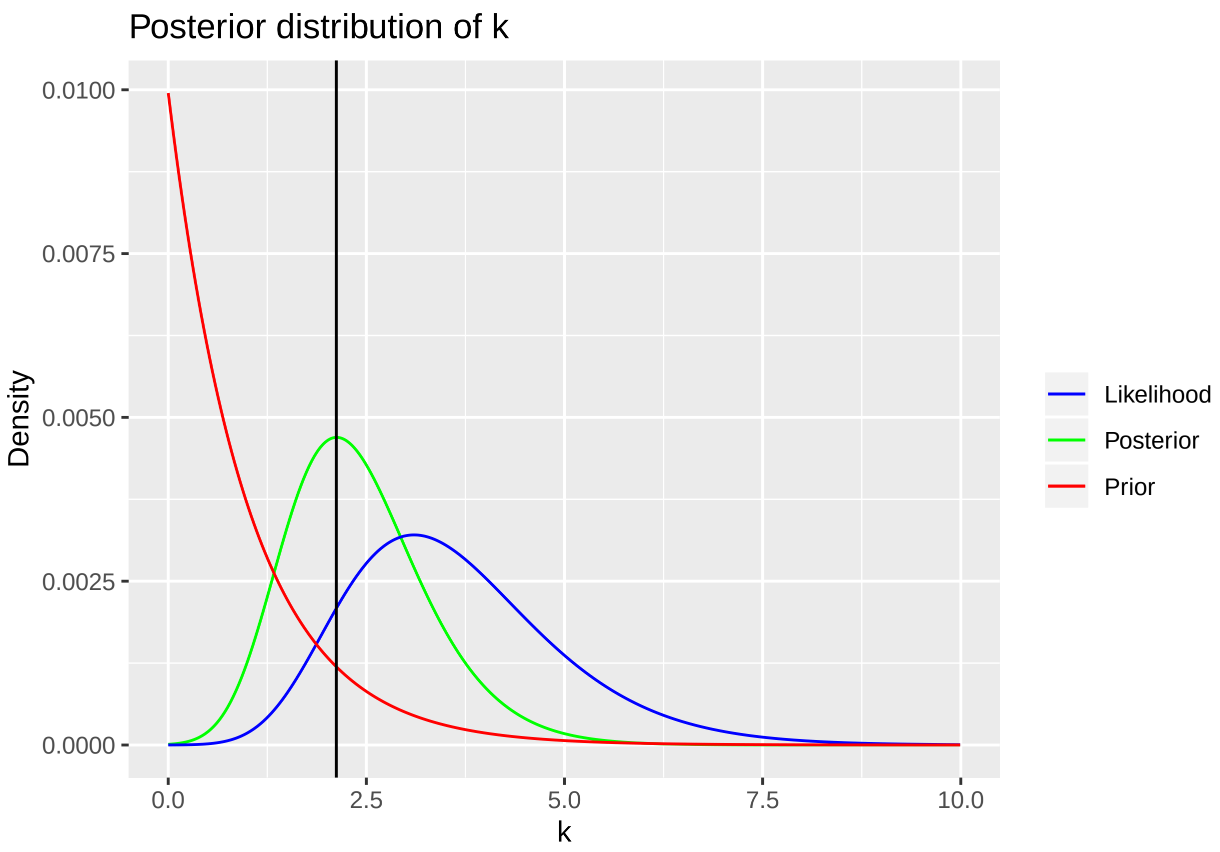
3.

a)



The posterior distribution of κ is as expected based on the density of the prior and the likelihood.

b)



The mode of the posterior is 2.121.

Appendix for code

```r
library(ggplot2)
set.seed(12345)
#1 a)
a0 = 2
b0 = 2
s = 5
f = 15
x_vector = seq(1, 10000, 1)
set.seed(12345)
distribution =rbeta(x_vector, a0+s, b0+f)

y_calced_mean = c(1:length(x_vector))
for (i  in x_vector) {
    y_calced_mean[i] = mean(distribution[0:i])
}

qplot(x=x_vector,y=y_calced_mean, geom="line")+
    labs(x="Number of draws", y="Mean", title="Mean as a function of number of draws")


y_calced_sd = c(1:length(x_vector))
for (i  in x_vector) {
    y_calced_sd[i] = sd(distribution[0:i])
}

qplot(x=x_vector,y=y_calced_sd, geom="line")+
    labs(x="Number of draws", y="Standard deviation", title="Standard deviation as a function of number

#1 b)

classified = ifelse(distribution > 0.3, 1, 0)
classified_table = table(classified)
counted_classified = classified_table[names(classified_table)==1]

classified_probabilty = counted_classified/length(distribution)
classified_probabilty

true_probability_value = 1- pbeta(0.3, a0+s, b0+f)

#1 c)

log_odds = log(distribution/(1-distribution))
hist(log_odds)

density(log_odds)

#2 a)

library(LaplacesDemon)
observations <- c(44, 25, 45, 52, 30, 63, 19, 50, 34, 69)
n <- length(observations)
my <- 3.7
```

```r
number_of_draws <- 10000
set.seed(12345)


tausquared <-
    sum((log(observations) - my) ^ 2) / length(observations)

sigma_sq <- rinvchisq(number_of_draws, n, tausquared)

interval = seq(min(sigma_sq), max(sigma_sq), 0.001)
invchisq = dinvchisq(interval, n, tausquared)


plot <-
    ggplot() + geom_histogram(aes(x = sigma_sq, y = ..density..), fill = "gray", binwidth = 0.02) + geon

plot

#2 b)

gini <- 2*pnorm(sqrt(sigma_sq)/sqrt(2))

plot_gini <- ggplot() + geom_histogram(aes(x=gini, y=..density..), binwidth = 0.001)
plot_gini

#2 c)

library(HDInterval)

interval_pin <- p.interval(gini, HPD=FALSE, prob=0.9)

gini_density <- density(gini)

interval_hdi <- hdi(gini_density, credMass=0.9)



plot_CI <- plot_gini + geom_vline(aes(xintercept=c(interval_pin[1,1], interval_pin[1,2]),color="ETI")) +
    geom_vline(aes(xintercept=c(interval_hdi["lower"], interval_hdi["upper"]),color="HDI")) + scale_colo




plot_CI

#3 a)

directions <-
    c(-2.44, 2.14, 2.54, 1.83, 2.02, 2.33, -2.79, 2.23, 2.07, 2.02)

mu <- 2.39

p_y_given_k_mu <- c()
```

```r
k_v = seq(0.001,10,0.01)

posterior <- c()
likelihood <- c()
prior <- c()

for (o in 1:length(k_v)) {
    probs <- c()
    k <- k_v[o]
    for (i in 1:length(directions)) {
        y <- directions[i]

        probs[i] <- exp(k * cos(y - mu)) / (2 * pi * besselI(k, 0)) #likelihood
    }
    likelihood[o] <- prod(probs)
    posterior[o] <- prod(probs) * dexp(k,1) #product of likelihood for y1...yn * prior for k
    prior[o] <- dexp(k,1)
}

plotposterior <-
    ggplot() + geom_line((aes(
        x = k_v,
        y = posterior / sum(posterior),
        color = "Posterior"
    ))) + geom_line((
        aes(
            x = k_v,
            y = likelihood / sum(likelihood),
            color = "Likelihood"
        )
    ))  + geom_line((aes(
        x = k_v,
        y = prior / sum(prior),
        color = "Prior"
    ))) + labs(x = "k", y = "Density", title = "Posterior distribution of k") + scale_color_manual(
        name = '',
        values = c(
            'Prior' = 'red',
            'Posterior' = 'green',
            'Likelihood' = 'blue'
        )
    )
plotposterior

#3 b)

plotkmode <- plotposterior + geom_vline(aes(xintercept=k_v[which.max(posterior)]))
plotkmode

print(k_v[which.max(posterior)])
```