Lab1

Andreas Stasinakis(andst745) & Mim Kemal Tekin(mimte666)

April 8, 2019

Contents

LAB 1

Question 1 Bernoulli, log odds by simulation.

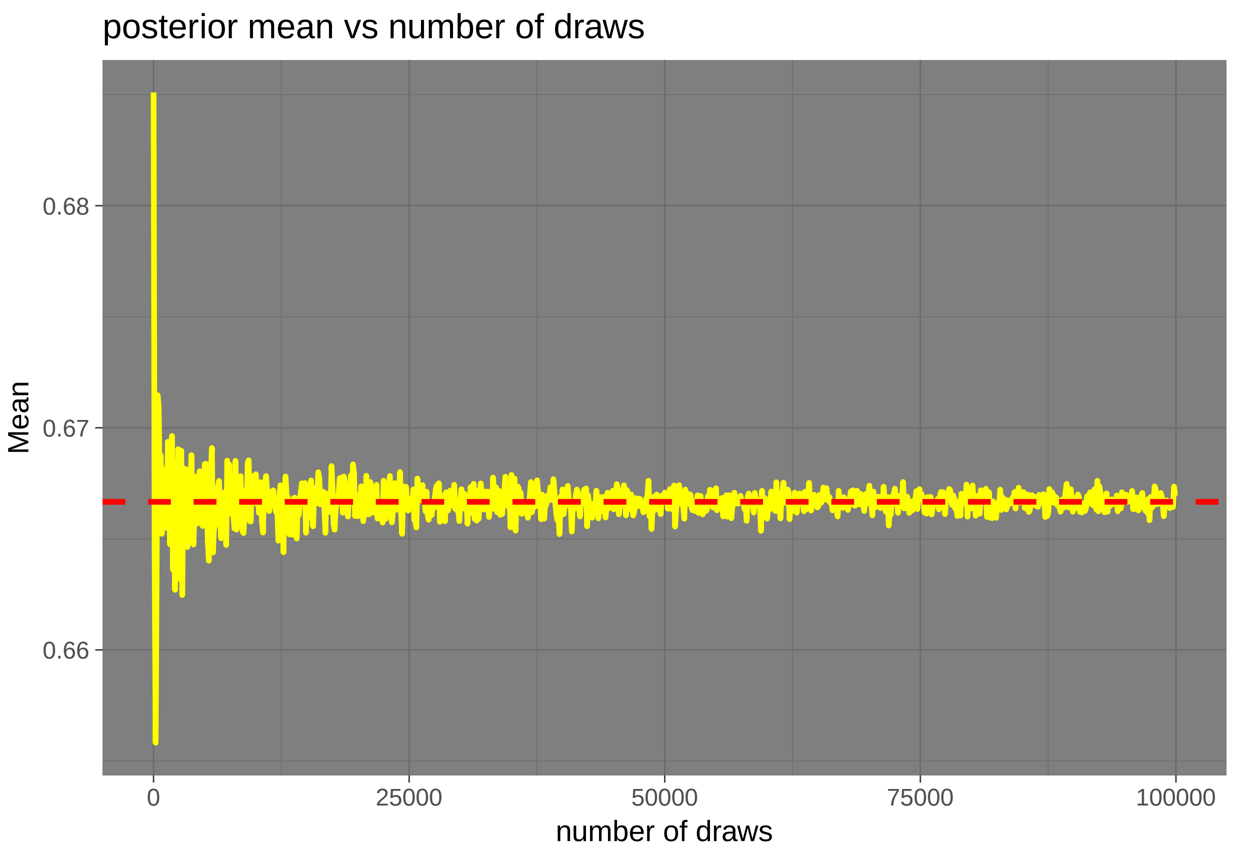
Let $y_1, y_2 ... y_n|\theta \sim Bern(\theta)$, and assume that you have obtained a sample with $s = 14$ successes in $n = 20$ trials. Assume a $Beta(\alpha_0, \beta_0)$ prior for θ and let $\alpha_0 = 2$,$\beta_0 = 2$

a) Random numbers from posterior, Posterior vs sample.

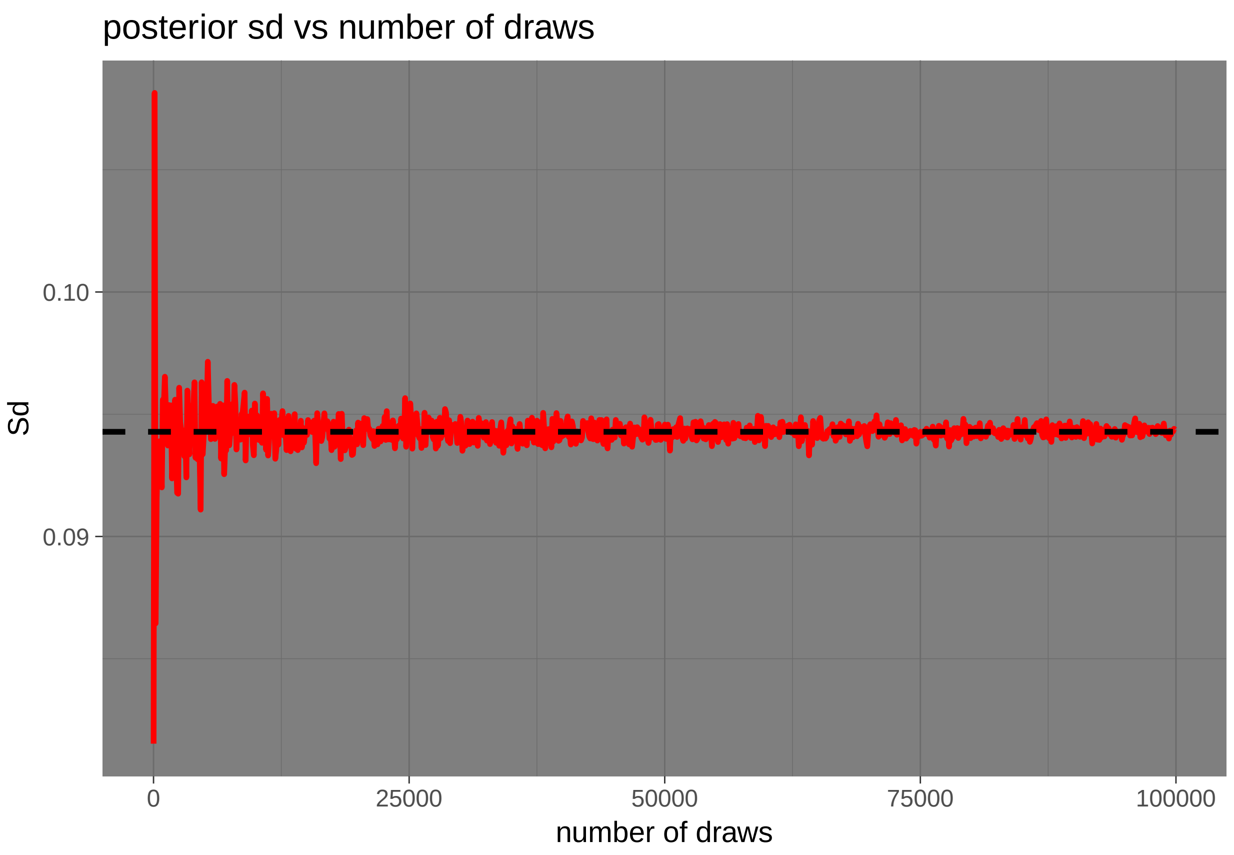
Draw random numbers from the posterior $\theta|y \sim Beta(\alpha_0 + s, \beta_0 + f), y = (y_1, .., y_n)$, and verify graphically that the posterior mean and standard deviation converges to the true values as the number of random draws grows large.

```
set.seed(12345)
library(ggplot2)
a = 2 #prior par
b = 2 #prior par
n = 20 #data size
s = 14 #successes
f = n - s #failures

a_pos = a+s #posterior parameter
b_pos = b + f #posterior parameter
real_mean = (a_pos)/(b_pos + a_pos) #theoritical mean using the beta distr
real_variance = (a_pos*b_pos)/(((b_pos + a_pos)^2)*(b_pos + a_pos+1)) #theor var
real_sd = sqrt(real_variance) #theoritical sd

#posterior beta(a0+s, b0 + f)
# we want to generate  samples with different size
# create a seq to check it
samp = seq(10, 100000,100) #increase the sample size in every iteration

# a matrix where we store the sample size, the mean and the sd of each sample
beta_stat = matrix(ncol = 3,nrow = length(samp)) #
i = 1

#for loop to generate all samples

for (n in samp) {

  # generate sample from the posterior of size n
  beta_samples= rbeta(n, a_pos, b_pos)

  # store the sample size, mena and sd of each sample
  beta_stat[i,] = c(n,mean(beta_samples),sd(beta_samples))
  i = i+1
}

beta_stat = as.data.frame(beta_stat)
colnames(beta_stat) = c("samp","mean","sd")

# Plot the mean and the standard deviation of each sample
# first the mean
```

```r
ggplot(beta_stat)+
  geom_line(mapping = aes(x = beta_stat$samp,y = beta_stat$mean),
            color = "yellow", size =1)+
  geom_hline(yintercept = real_mean, size = 1, color = "red", lty = 2)+
  labs(title = "posterior mean vs number of draws", x = "number of draws",
       y = "Mean") +
  theme_dark()
```



```r
#now the sd
ggplot(beta_stat)+
  geom_line(mapping = aes(x = beta_stat$samp,y = beta_stat$sd),
            color = "red",size =1)+
   geom_hline(yintercept = real_sd, size = 1,color = "black", lty = 2)+
  labs(title = "posterior sd vs number of draws", x = "number of draws",
       y = "Sd") +
  theme_dark()
```



The true value for the mean and the variance for the beta distribution Beta($\alpha = 16, \beta = 6$) are given by the formula below:

$$\text{mean} = \frac{\alpha}{\alpha + \beta} = \frac{16}{22}$$

which is equal to 0.6666667

$$\text{variance} = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} = \frac{16 \cdot 6}{22^2 \cdot 23} = 0.008623787$$

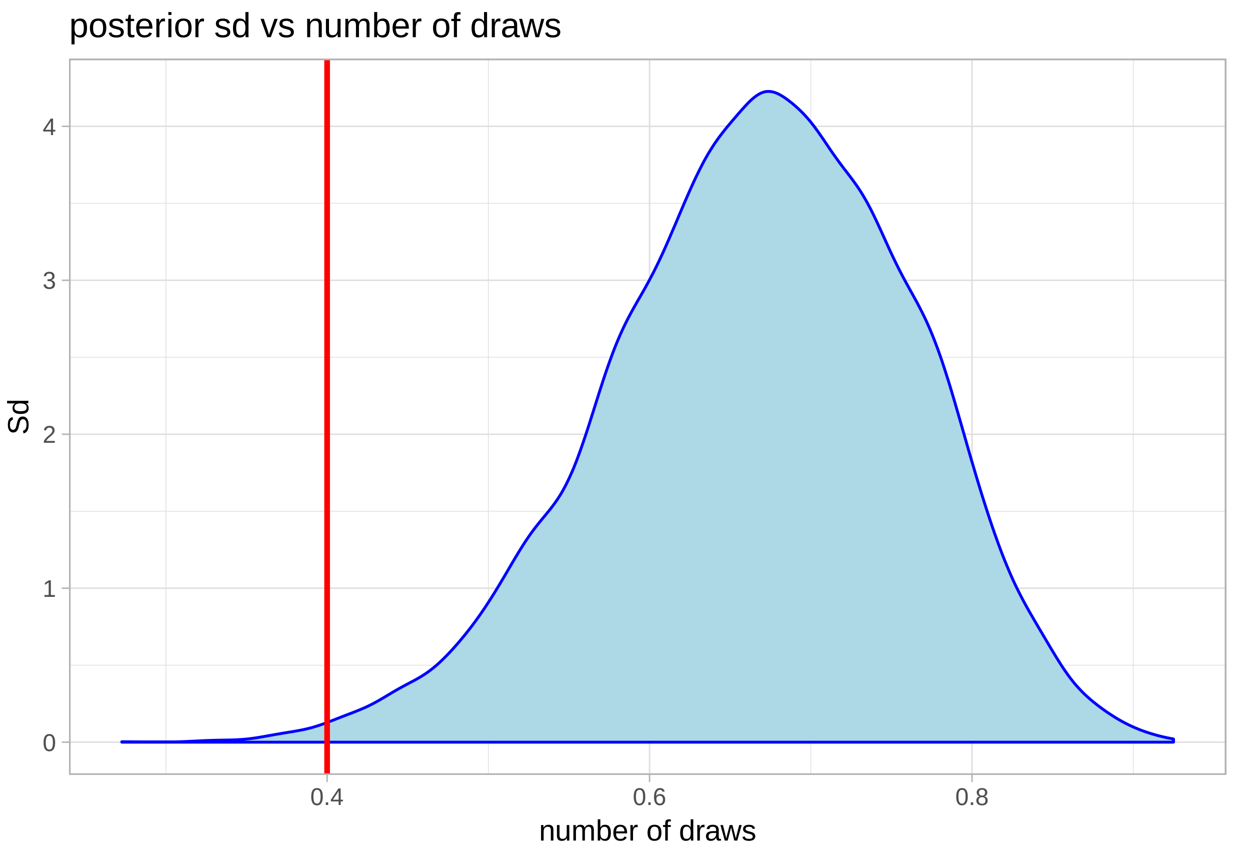
So the standard deviation is equal to 0.0942809.

As we can see from the two plots above, while the number of random draws grows large, both the mean and the standard deviation converge to the true values. For small samples, the estimation is not so accurate but the difference is not that important.

b) Posterior probability and compare with real value

Use simulation (nDraws = 10000) to compute the posterior probability $Pr(\theta < 0.4|y)$ and compare with the exact value [Hint: pbeta()].

```
#generate a sample size 10000 from the posterior distribution
sim = rbeta(n = 10000,shape1 = a_pos,b_pos)

# just a plot to visualize the area we want to estimate
ggplot(as.data.frame(sim))+
  geom_density(mapping = aes(sim),color = "blue", fill = "lightblue")+
   geom_vline(xintercept = 0.4, size = 1,color = "red")+
  labs(title = "posterior sd vs number of draws", x = "number of draws",
       y = "Sd") +
  theme_light()
```



```
# we have the posterior distribution of the parameter
# so we can proportionaly find the probability we want using the sample

prob = sum((sim<0.4))/length(sim)
theoretical_prob = pbeta(0.4, a_pos, b_pos)
```

In the plot we can visualy see what is the probability we want to calculate(the area before the red vertical line). The posterior probability is 0.0035 while the theoretical one is 0.0039727. As we expected their values are really close to each other.

c) Posterior distribution, Histogram and QQ plot

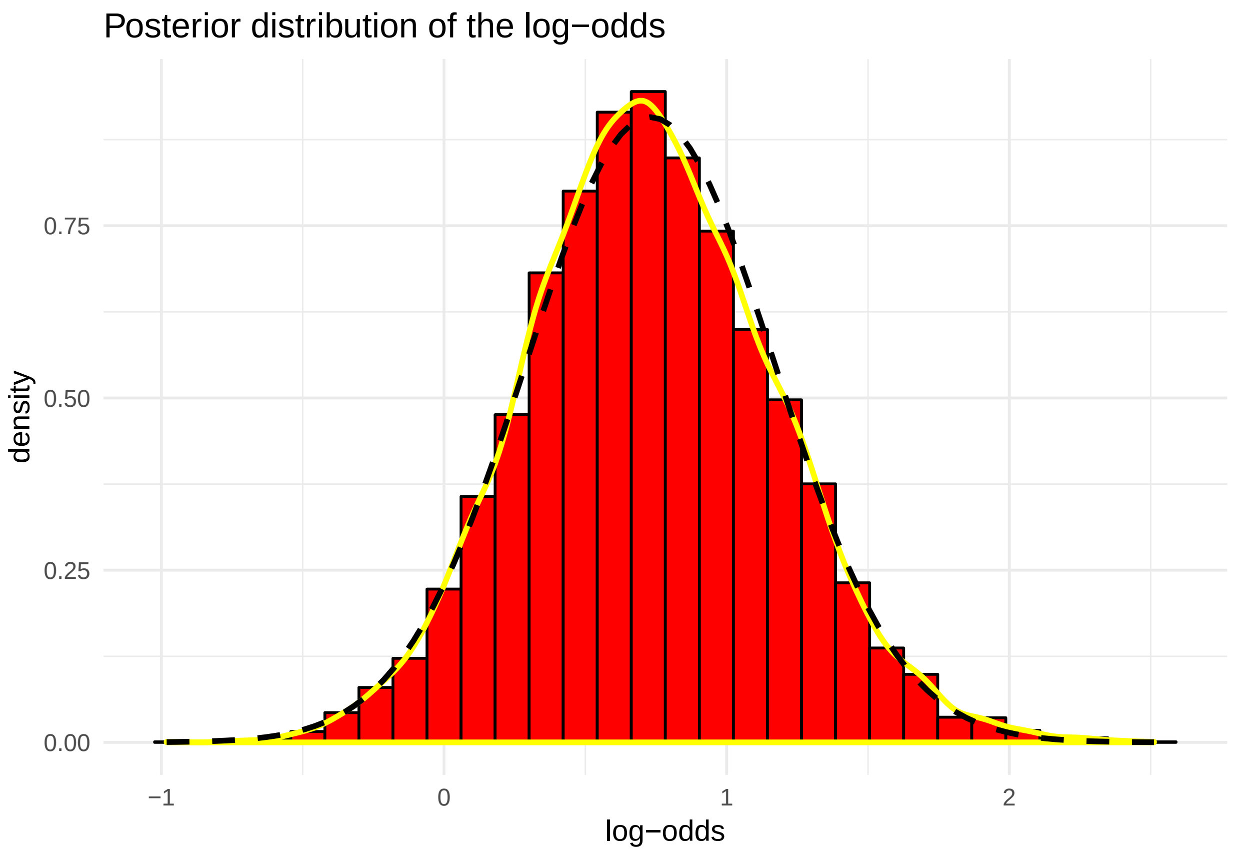
Compute the posterior distribution of the log-odds $\phi = log(\frac{\theta}{1-\theta})$ by simulation (nDraws = 10000). [Hint: hist() and density() might come in handy]

```
# library for quantile-quantile plot
library(ggpubr)
```

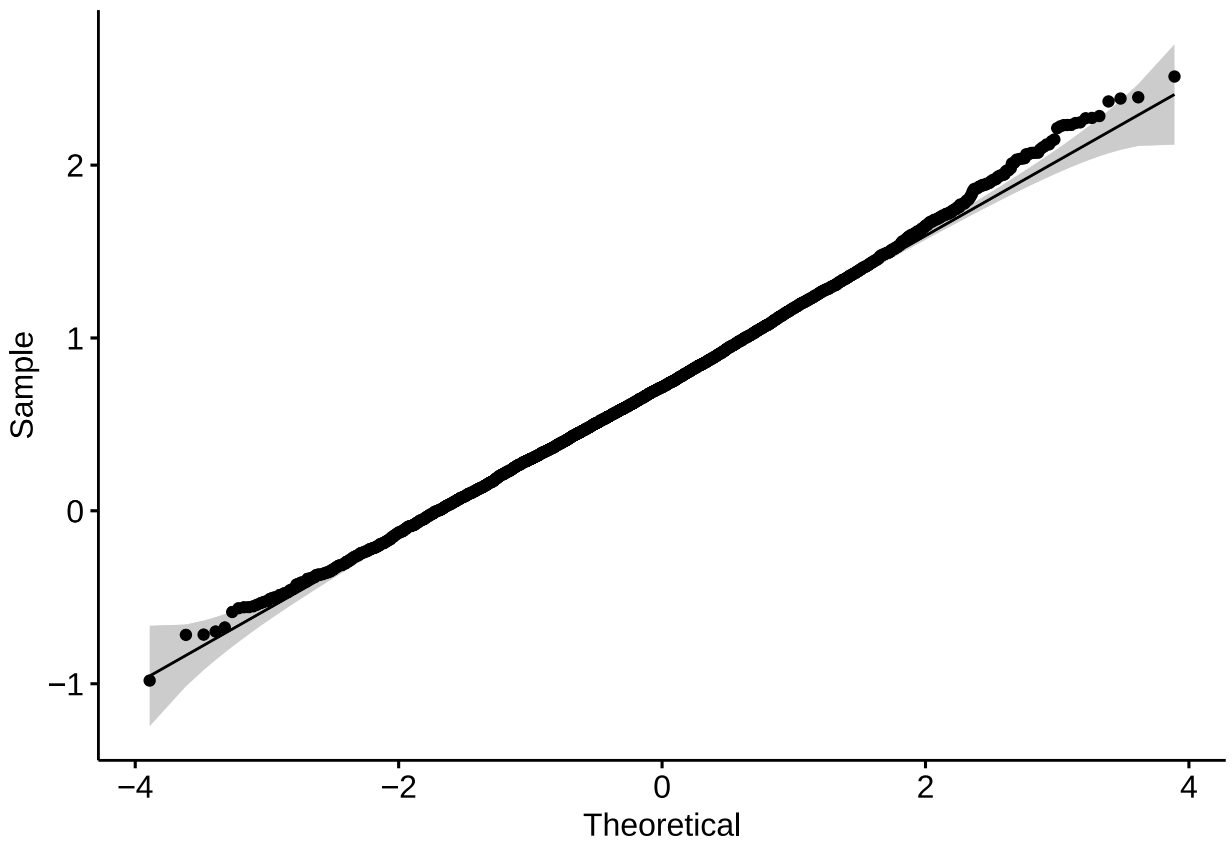
```
## Loading required package: magrittr
```

```
# re - parametrize the sample
log_odds = as.data.frame(log(sim/(1-sim)))
colnames(log_odds) = "x"
```

```r
# Plot the histogramm of the log odds

ggplot(log_odds)+
  geom_histogram(mapping = aes(x = log_odds$x, y = ..density..), bins = 30,
                 fill = "red", color = "black")+
  geom_density(mapping =aes(x = log_odds$x),color = "yellow", size = 1)+
  stat_function(fun = dnorm, lty = 2, size = 1,
                args = list(mean = mean(log_odds$x), sd = sd(log_odds$x))) +
  labs(title = "Posterior distribution of the log-odds", x = "log-odds")+
  theme_minimal()
```



```r
# QQ plot
ggqqplot(log_odds$x)
```



In the first plot we have the histogramm of the posterior distribution of the log-odds and its density(yellow line). We can observe that it looks like a normal distribution with mean close to 0.7. We also plot the density of a real normal distribution calculating the parameters from the data(dash line). The two densities look really similar, which is another reason to assume that it is most likely that the posterior distribution of the log-odds is normal.

The second plot is the quantile quantile plot. In the x-axis we have the Theoritical values for a normal distribution while in the y-axis the sample values. From this plot, we can verify that the posterior probably follows a normal distribution because most of the sample points lie in the theoritical line.

Question 2 Log-normal distribution(mean known, variance unknown) and the Gini coefficient.

Assume that you have asked 10 randomly selected persons about their monthly income (in thousands Swedish Krona) and obtained the following ten observations: 14, 25, 45, 25, 30, 33, 19, 50, 34 and 67. A common model for non-negative continuous variables is the log-normal distribution. The log-normal distribution logN(μ, σ^2) has density function:

$$p(y|\mu,\sigma^2) = \frac{1}{y \cdot \sqrt{2\pi\sigma^2}} \exp\Big[-\frac{1}{2\sigma^2}(\log y - \mu)^2\Big]$$

, for $y, \mu, \sigma^2 > 0$. The log-normal distribution is related to the normal distribution as follows: if $y \sim \log N(\mu, \sigma^2)$ then $\log y \sim N(\mu, \sigma^2)$. Let $y_1, y_2 ... y_n|\mu, \sigma^2 \sim \log N(\mu, \sigma^2)$, where $\mu = 3.5$ is assumed to be known but σ^2 is unkown with non-informative prior $p(\sigma^2) \propto 1/\sigma^2$. The posterior for σ^2 is the Inv-x$^2(n, \tau^2)$ distribution, where

$$\tau^2 = \frac{\sum_{i=1}^{n}(logy_i - \mu)^2}{n}$$

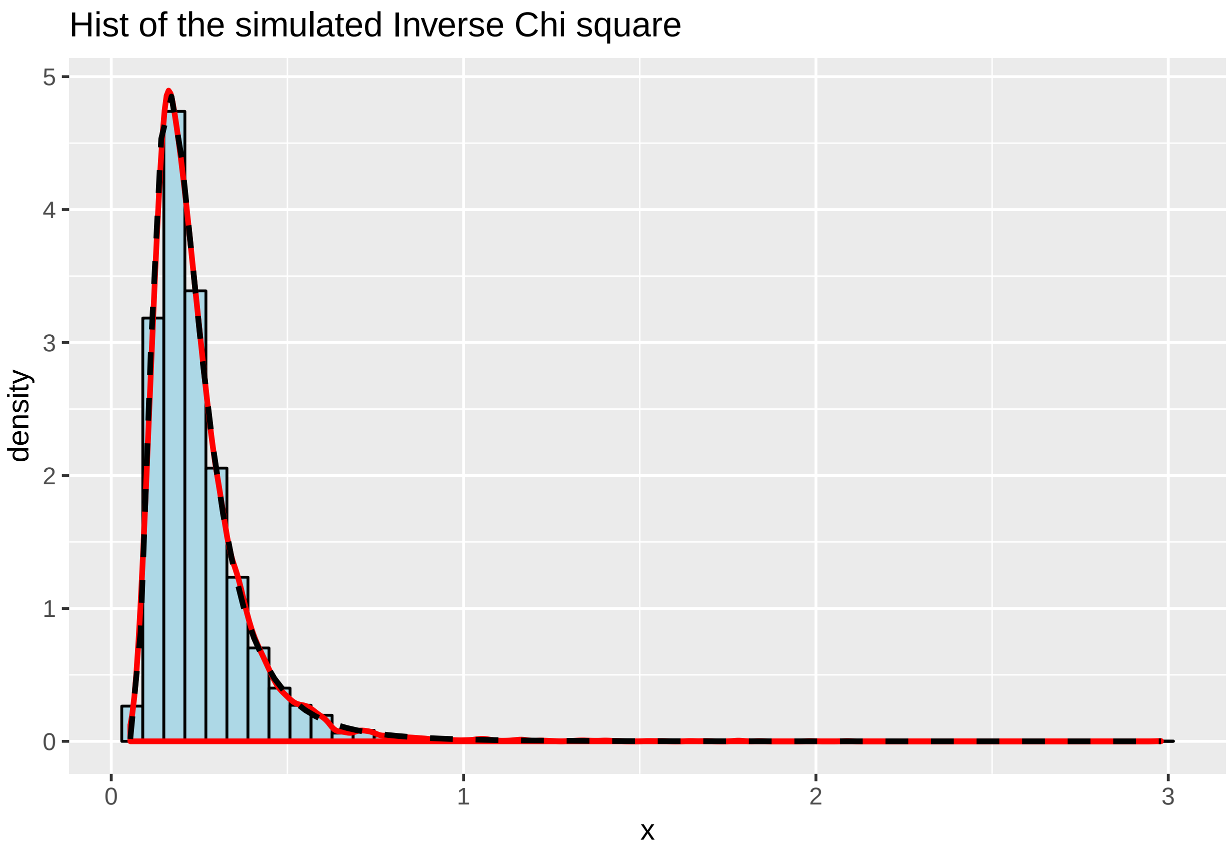
a) Simulate from the posterior inv - chi, compare with the theor. one using stat_function and some statistics.

Simulate 10,000 draws from the posterior of σ^2 (assuming μ = 3.5) and compare with the theoretical Inv-$x^2(n, \tau^2)$ posterior distribution.

```
set.seed(12345)

library(gridExtra)
library(geoR)
```

```
## --------------------------------------------------------------
##  Analysis of Geostatistical Data
##  For an Introduction to geoR go to http://www.leg.ufpr.br/geoR
##  geoR version 1.7-5.2.1 (built on 2016-05-02) is now loaded
## --------------------------------------------------------------
```

```
# We know the posterior is distributed as inv - chi
# function simulate one sample of size len from inv-chi
rinvChi = function(len, mu, t_sq, n){
  # generate a random sample from Chi- square
  X = rchisq(len,df = n)

  res = (n*t_sq)/X
  return(res)
}

# in order to plot the real inverse Chi square we need the density
# function for the density, we can also use the package from laplace
invChi_pdf = function(n,x,tau){
  fraction1 = (((tau)*(n/2))^(n/2))/gamma(n/2)
  fraction2 = exp(-n*tau/(2*x))/(x^((n/2)+1))
  res = fraction1*fraction2
  return(res)
}

# we create our data
#in this case just 10 observations
data = c(14,25,45,25,30,33,19,50,34,67)
n = length(data)

nDraws = 10000 # size of the sample we will draw
mu = 3.5 # known parameter
t_sq = sum((log(data) - mu)^2)/n # parameter of the inv-chi, we can compute it.


#plot the inverse Chi square sample
#generate the sample
x = rinvChi(nDraws,mu,t_sq,n)

post_var = as.data.frame(x)
colnames(post_var) = "x"

# plot the hist of the sample and the theoritical density using stat_function
ggplot(post_var)+
  geom_histogram(mapping = aes(x = post_var$x,y = ..density..), bins = 50,
                 color = "black", fill = "lightblue")+
  geom_density(mapping = aes(x = post_var$x,y = ..density..),
               color = "red", size = 1)+
  stat_function(mapping = aes(x = x),
                fun = invChi_pdf,args = list(n = n, tau = t_sq),
                size = 1,color = "black", lty = 2)+
  labs(title = "Hist of the simulated Inverse Chi square", x = "x")
```



```
# sample and theoritical statistics

sim_mean = mean(post_var$x) #sample
sim_var = var(post_var$x) # sample
theor_mean = (n*t_sq)/(n-2) #theoritical mean
theor_var = (2*(n^2)*t_sq^2)/(((n-2)^2)*(n-4))# theoritical variance

cat(paste("The simulation mean is",sim_mean,
            "\nThe simulation variance is", sim_var,
            "\nThe theoritical mean is", theor_mean,
            "\nThe theoritical variance is",theor_var))
```

```
## The simulation mean is 0.246717829216389
## The simulation variance is 0.0207204283697603
## The theoritical mean is 0.247349686559943
## The theoritical variance is 0.0203939558137673
```

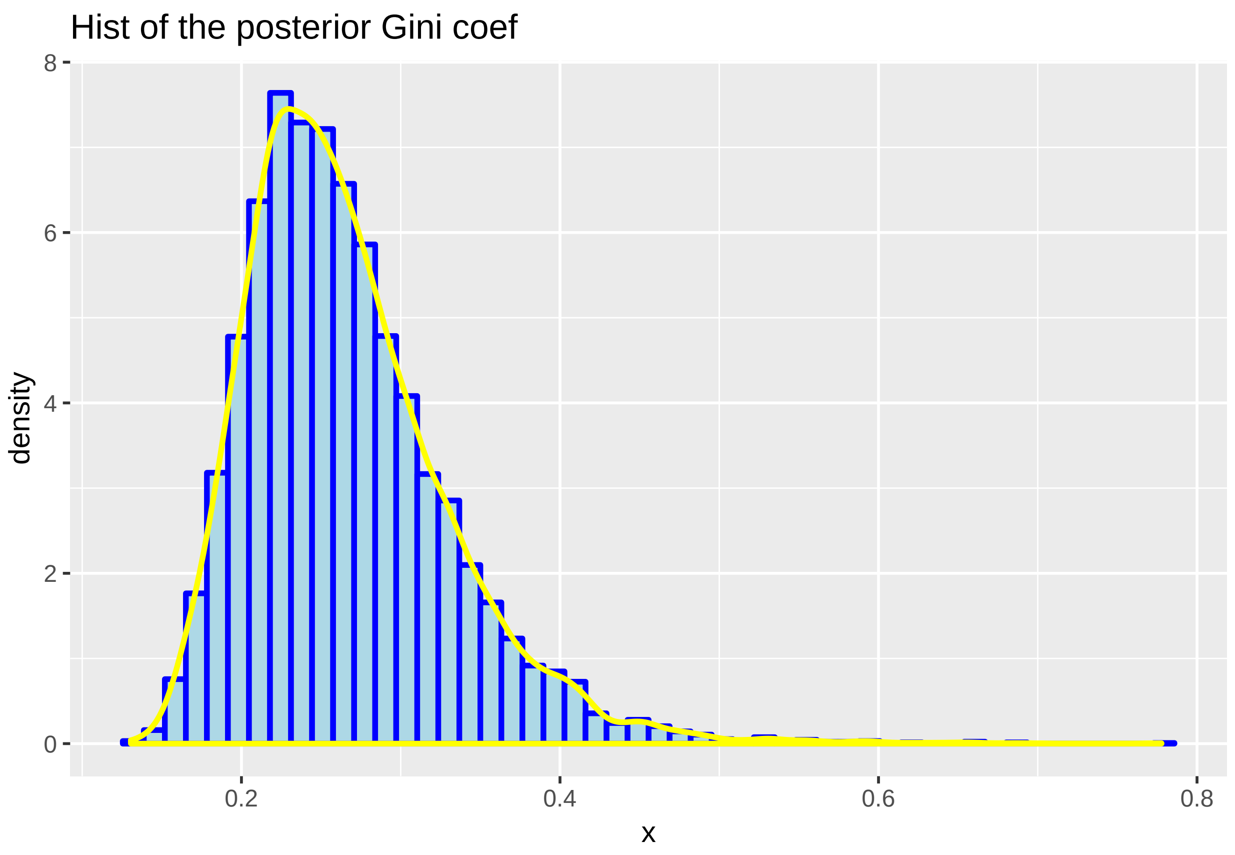
In this task we simulate 10000 values of the posterior Inv-x$^2(n, \tau^2)$ and we plot the histogramm of those values. We also plot its density(red line) and the density of the theoretical posterior distribution. As can be seen the both two densities are really similar, so our estimation seems really accurate.

We also print some statistics above. As one can see both the simulation and theoritical mean and variance are really close to each other.

b) Gini Coefficient

The most common measure of income inequality is the Gini coefficient, G, where 0 < = G < = 1. G = 0 means a completely equal income distribution, whereas G = 1 means complete income inequality. See Wikipedia for more information. It can be shown that $G = 2\Phi(\sigma/\sqrt{2}) - 1$ *when incomes follow a* $LogN(\mu, \sigma^2)$ *ditribution.* $\Phi(z)$ *is the cumulative distribution function (CDF) for the standard normal distribution with mean zero and unit variance. Use the posterior draws in a) to compute the posterior distribution of the Gini coefficient G for the current data set.*

```r
# we have generate a sample from the posterior of the variance
# we use that in order to estimate the gini coef from the data set.
# we can do that because our data are log normally distributed
# function for the posterior distribution of Gini coef

posterior_Gini = function(var){
  #cdf of a standard normal
  fi = pnorm(sqrt(var)/sqrt(2))

  res = 2*fi - 1
  return(res)
}

# data frame for the posterior gini coef
gini_coef = posterior_Gini(post_var$x)

plot_G = ggplot(as.data.frame(gini_coef))+
  geom_histogram(mapping = aes(x = gini_coef,y = ..density..), bins = 50,
                 color = "blue", fill = "lightblue",size =1)+
  geom_density(mapping = aes(x = gini_coef,y = ..density..), color = "yellow",
               size=1)+
  labs(title = "Hist of the posterior Gini coef", x = "x")

plot_G
```



c) Credible intervals(equal tails,HPD), plots with different labels

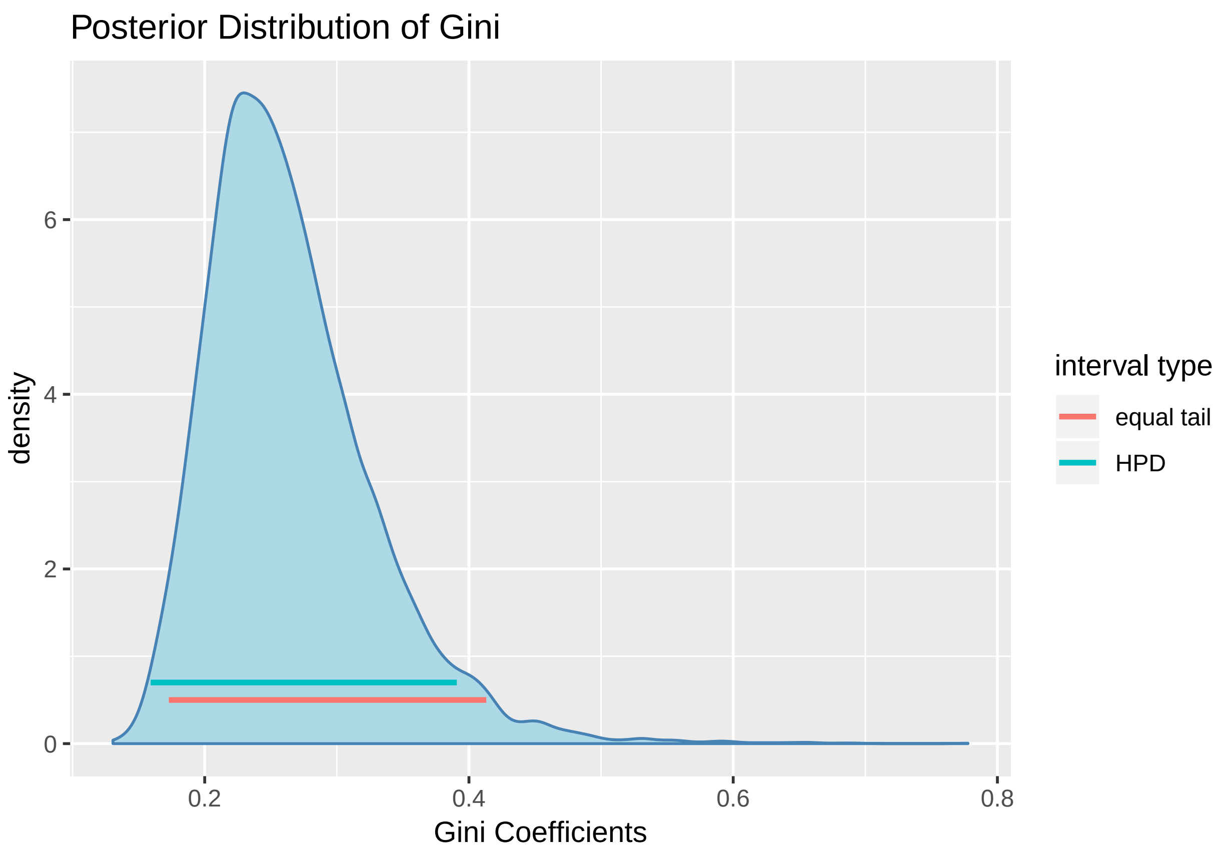
Use the posterior draws from b) to compute a 95% equal tail credible interval for G. An 95% equal tail interval (a, b) cuts off 2.5% percent of the posterior probability mass to the left of a, and 97.5% to the right of b. Also, do a kernel density estimate of the posterior of G using the density function in R with default settings, and use that kernel density estimate to compute a 95% Highest Posterior Density interval for G. Compare the two intervals.

```
set.seed(12345)

# equal tail confidence interval
# if it was a normal
#equal_tail_int = c(mean(gini_coef)-1.96*sd(gini_coef),mean(gini_coef)+1.96*sd(gini_coef))
# in the equal tail we exclude the same quantity both from the two tails
# not good for non symmetric

lower_equal = quantile(gini_coef,0.025)
upper_equal = quantile(gini_coef,0.975)
equal_tail = c(lower_equal,upper_equal)


# Highest posterior density method for the CI
kernel_density = density(gini_coef)

#Now we have the density values and we store them in a data frame
kern_df = data.frame(x = kernel_density$x, y = kernel_density$y)
```

```r
# we sort the data frame, because we want the threshold
# for the points that have the lower density
sorted_df = kern_df[order(kern_df$y,decreasing = TRUE),]

# we have to add the cumsum now to the dataframe
sorted_df$cumsum = cumsum(sorted_df$y)

# threshold for the 95% CI

# density function gives back 512 observations (beacause of default n argument)
# so we need the 95% of the sorted points
# we have to calculate 95% of the density
# in this case we take 0.95 of the highest cumulative sum.
a = sorted_df$cumsum[dim(sorted_df)[1]]*0.95

# delete the observations which are not belong to this interval
final_df = sorted_df[which(sorted_df$cumsum<a),]

# now we have the points with the highest density
lower = final_df[which.min(final_df$x),]$x # left interval
upper = final_df[which.max(final_df$x),]$x # right interval
HPD = c(lower,upper)

# data frame with all information for the plot
df = data.frame(x = c(lower_equal,upper_equal,lower,upper),
                y= c(0.5,0.5,0.7,0.7),
                label = c("equal tail", "equal tail", "HPD", "HPD"))


plot_intervals = ggplot() +
  geom_density(aes(x = gini_coef, y = ..density..),
               color = "steelblue", fill = "lightblue") +
  geom_line(mapping = aes(x = df$x, y = df$y, color = df$label), size =1)+
  labs(title = "Posterior Distribution of Gini",
       #subtitle = "95% Equal Tail Interval (a,b)",
       x = "Gini Coefficients", color = "interval type")


cat(paste("The equal tail interval is ( ",equal_tail[1], ", ", equal_tail[2]," )",
            "\nThe HPD interval is ( ",HPD[1], ", ", HPD[2]," )"))
```

```
## The equal tail interval is (  0.172854643796486 ,  0.413130979236925  )
## The HPD interval is (  0.159106626507391 ,  0.390773071734713  )
```

```r
plot_intervals
```



In equal tail intervals we exclude 2.5% of the density from both sides of the plot. When we do this as we can see we actually exclude biger amount of high density zone than HDP. The reason of this, we exlude the areas equally and we have lost some good estimations that have higher density than the estimations on right side. In HDP, we can include the estimations which have highest density in this 95% of the distribution. In general HDP has more accuracy in nonsymmetric distributions like this one.

Question 3 Von Mises with known mean and unknown k(concentration parameter).

Bayesian inference for the concentration parameter in the von Mises distribution. This exercise is concerned with directional data. The point is to show you that the posterior distribution for somewhat weird models can be obtained by plotting it over a grid of values. The data points are observed wind directions at a given location on ten different days. The data are recorded in degrees:

$$(40, 303, 326, 285, 296, 314, 20, 308, 299, 296)$$

, where North is located at zero degrees (see Figure 1 on the next page, where the angles are measured clockwise). To fit with Wikipedias description of probability distributions for circular data we convert the data into radians $-\pi <= y <= \pi$. The 10 observations in radians are

$$(-2.44, 2.14, 2.54, 1.83, 2.02, 2.33, -2.79, 2.23, 2.07, 2.02)$$

.

Assume that these data points are independent observations following the von Mises distribution

$$p(y|\mu, k) = \frac{exp\big[k \cdot cos(y - \mu)\big]}{2\pi I_0(k)}$$

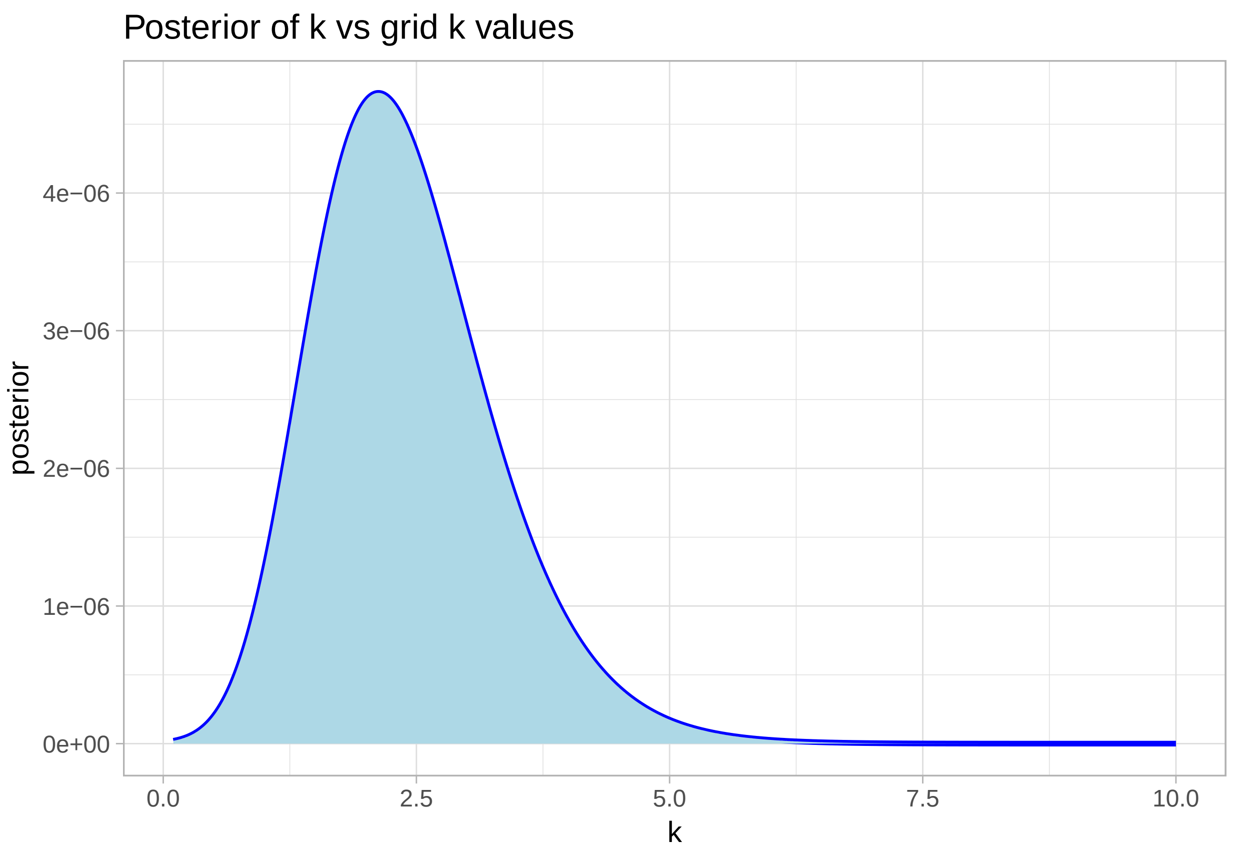
, $-\pi <= y <= \pi$,

where $I_0(k)$ is the modified Bessel function of the first kind of order zero [see ?besselI in R]. The parameter $\mu(-\pi <= \mu <= \pi)$ is the mean direction and $k > 0$ is called the concentration parameter. Large k gives a small variance around μ, and vice versa. Assume that μ is known to be 2.39. Let $k \sim$ Exponential$(\lambda = 1)$ a priori, where λ is the rate parameter of the exponential distribution (so that themean is $\frac{1}{\lambda}$).

a) Plot a posterior using grid k values, if the model is too weird.

Plot the posterior distribution of k for the wind direction data over a fine grid of k values.

```r
#the parameters and data  we know
set.seed(12345)
mu = 2.39
data = c(-2.44,2.14,2.54,1.83,2.02,2.33,-2.79,2.23,2.07,2.02)
lambda = 1

#We did the calculations in order to have the posterior
posterior_k = function(dt,mu,lambda,k){
  n = length(dt)

  den = (2*pi*besselI(k,0))^n
  step2 = sum(cos(dt- mu)) - 1
  num = exp(k*step2)
  res = num/den
  return(res)


}

posterior = function(k, data, mu=2.39, lambda=1){
  n = length(data)
  num = exp(k * (sum(cos(data-mu)) - 1))
  denum = (2 * pi * besselI(k, 0))^n
  return(num / denum)
}


# use some grid k values in order to plot the posterior density
all_k = seq(0.1,10,0.01)

d = posterior_k(dt = data,mu = mu,lambda = lambda,k = all_k)

df = data.frame = data.frame(k = all_k, posterior = d)

ggplot(df)+
  geom_line(mapping = aes(x = df$k,y = df$posterior), size = 1,color = "blue")+
  geom_area((mapping = aes(x = df$k,y = df$posterior)), fill = "lightblue")+
  labs(title = "Posterior of k vs grid k values", x = "k", y = "posterior")+
  theme_light()
```



In order to generate numbers from the posterior distribution, we have to calculate its formula first. We use the known formula :

$$\text{Posterior} \propto \text{Likelihood} \cdot \text{Prior}$$

In our case we have :

$$P(k|y,\mu) \propto P(y|k,\mu) \cdot P(k)$$

.

More specific:

Prior:

$$p(\kappa) \sim Exp(\lambda = 1) = \lambda \cdot e^{-\lambda\kappa}$$

Model:

$$p(y \mid \mu, \kappa) = \frac{\exp[\kappa \cdot cos(y-\mu)]}{2\pi I_0(\kappa)}$$

So we take the product for all data points y_i.

Likelihood:

$$p(y_i \mid \mu, \kappa) = \prod_{i=1}^{n} \frac{\exp[\kappa \cdot cos(y_i-\mu)]}{2\pi I_0(\kappa)} = \left[\frac{1}{I_0(\kappa)}\right]^n \cdot \exp\left[\sum_{i=1}^{n} \kappa \cdot cos(y_i-\mu)\right]$$

Finaly, in order to find the posterior we multiply the prior and the likelihood.

Therefore,

$$p(\kappa \mid y_1, y_2, ..., y_n) \propto \left[\frac{1}{I_0(\kappa)}\right]^n \cdot \exp\left[\kappa\Big(\sum_{i=1}^{n} \cdot cos(y_i-\mu) - \lambda\Big)\right]$$

We also know that $\lambda = 1$ and $\mu = 2.39$, so the final posterior formula is :

$$p(\kappa \mid y_1, y_2, ..., y_n) \propto \left[\frac{1}{I_0(\kappa)}\right]^n \cdot \exp\left[\kappa\Big(\sum_{i=1}^{n} \cdot cos(y_i - 2.39) - 1\Big)\right]$$

We now have the posterior distribution, so we can plot it. We need to choose a reasonable grid of k values. We start plotting for different values and we realized that the majority of the density occurs between 0 and 6. After that the density values are close to 0. For that reason, in our final plot we use k from 0 to 10 with a step 0.01.

b) approximate posterior mode

Find the (approximate) posterior mode of k from the information in a).

```
# find the approximate posterior mode for k
pos_mode = df[which.max(df$posterior),]$k
cat(pos_mode)
```

```
## 2.12
```