Bayesian Learning Lab 1

Mohamed Ali - Mohal954

2023-05-06

Question 1 Daniel Bernoulli

let $y_1, ..., y_n|\theta \sim Bern(\theta)$ and assume that you have obtained a sample with s = 22 successes in n = 70 trials. Assume a $Beta(\alpha_0; \beta_0)$ prior for θ and let $\alpha_0 = \beta_0 = 8$.

First we calculate the mean and the standar deviation from the below equation to compare the with sample means and standar deviations of θ as function of the accumulating number of drawn values.

$$E(\theta|y) = a/a + b$$
$$E(\theta|y) = ab/(a + b)^2(a + b + 1)$$

A

Draw 10000 random values (nDraws = 10000) from the posterior $\theta|y \sim \beta(\alpha_0 + s, \beta_0 + f)$, where y = (y1,..,yn); and verify graphically that the posterior mean $E(\theta|y)$ and standard deviation $SD(\theta|y)$ converges to the true values as the number of random draws grows large.
[Hint: use rbeta() to draw random values and make graphs of the sample means and standard deviations of θ as a function of the accumulating number of drawn values].

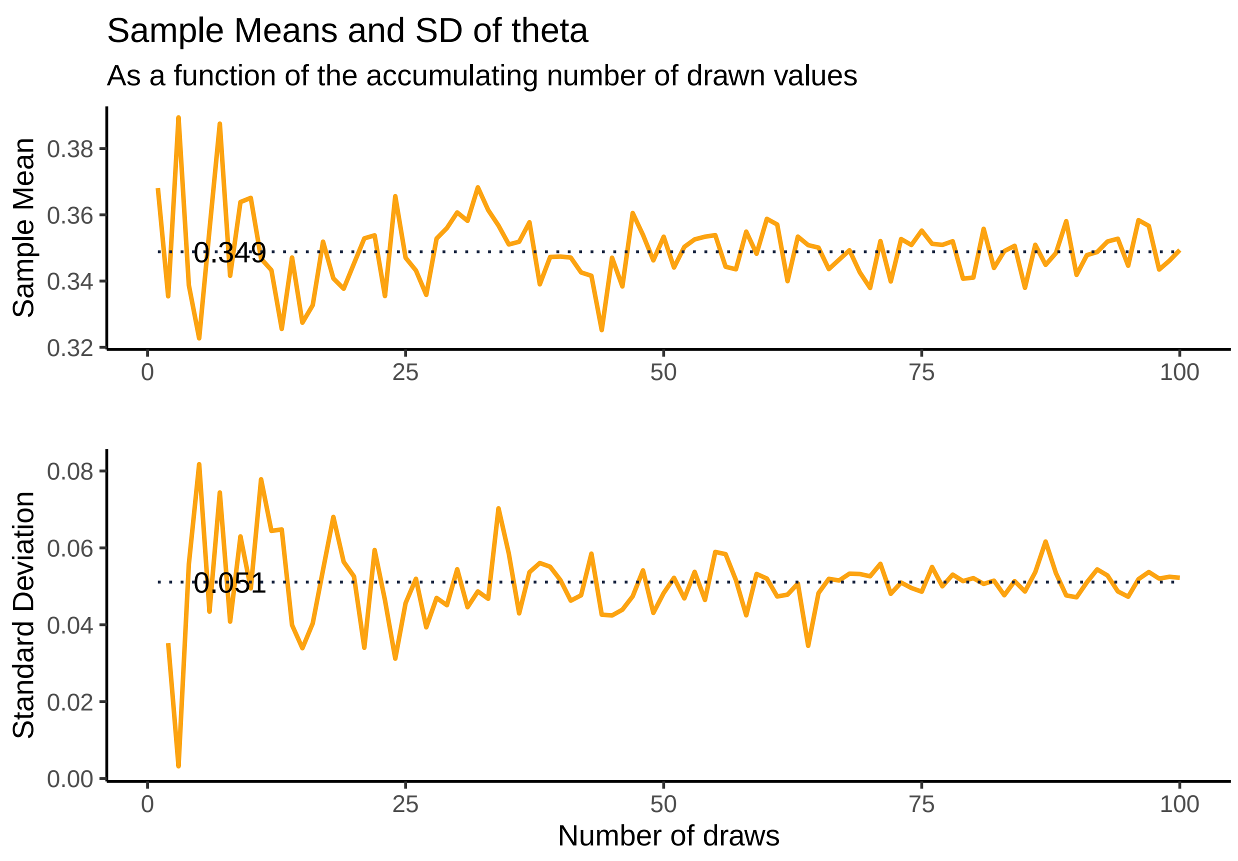
The figure below shows how the values of the Mean and Sd converges to the true values as the number of draws grows large

```r
ber_fun<-function(n_d,n,s,a,b) {
  #Initial Value of the function givan from the question
  t_n = n
  s   = s
  f   = n-s
  a   = a
  b   = b
  #Beta(alpha+s,beta+s)
  a_new = a+s
  b_new = b+f
  Mean_true= a_new/(a_new+b_new)
  #we take the sqrt to get the Sd insted of the Var
  Sd_true=  sqrt((a_new*b_new)/(((a_new+b_new)^2) * (a_new+b_new+1)))
  # Beta(alpha+s,beta+s)
  mean_theta = c()
  sd_theta = c()
  n_draws = 1:n_d
  #For loop to fill the values of mean and sd based on the draws
  for (i in 1:n_d){
    mean_theta[i]=mean(rbeta(i,a_new,b_new))
    sd_theta[i]=sd(rbeta(i,a_new,b_new))}
  #Binding everything togther
  df<-cbind.data.frame(n_draws,mean_theta,sd_theta)
  #Plot of the mean
  mean<-ggplot(df,aes(x=n_draws))+geom_line(aes(y=mean_theta)
                                           , color='#FCA311', size=.8)+
              geom_line(aes(y=Mean_true), color='#14213D',linetype=3)+
              annotate(geom = "text", x = 8, y = Mean_true,
              label = paste0(format(round(Mean_true, 3), nsmall = 3)))+
              labs(title = 'Sample Means and SD of theta ',
       subtitle = 'As a function of the accumulating number of drawn values',
                           x= ' ', y='Sample Mean')+ theme_classic()
  #Plot of the Sd
  sd<-ggplot(df,aes(x=n_draws))+geom_line(aes(y=sd_theta),
                                         color='#FCA311', size=.8)+
              geom_line(aes(y=Sd_true), color='#14213D',linetype=3)+
              annotate(geom = "text", x = 8, y = Sd_true,
              label = paste0(format(round(Sd_true, 3), nsmall = 3)))+
              labs(x= 'Number of draws', y='Standard Deviation')+
              theme_classic()
  #grid.arrange(mean,sd)
  return(grid.arrange(mean,sd))
}


ber_fun(100,70,22,8,8)
```

```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

```
## Warning: Removed 1 row containing missing values (‘geom_line()‘).
```



B

Draw 10000 random values from the posterior to compute the posterior probability $Pr(\theta > 0.3|y)$ and compare with the exact value from the Beta posterior. [Hint: use pbeta()].

First we find the value from the beta posterior using the function *pbeta* we get the value of theta which can be used to compute the probability that a random variable from a beta distribution is less than or equal to a given value, or greater than a given value, depending on the value of the lower.tail argument. in our case we use lower.tail as False because we need the values grater than.

```r
prob<-function(n_d,n,s,a,b,prob){
        t_n = n
        s   = s
        f   = n-s
        a   = a
        b   = b
        #Beta(alpha+s,beta+s)
        a_new = a+s
        b_new = b+f
        p<-pbeta(prob,a_new,b_new,lower.tail = F)
        post<-mean(rbeta(n_d,a_new,b_new)>prob)
        return(list(paste('random values from the posterior with the given condition',post)
        ,paste('The exact value from the Beat posterior', round(p,2))))}
prob(100,70,22,8,8,.3)
```

```
## [[1]]
## [1] "random values from the posterior with the given condition 0.85"
##
## [[2]]
## [1] "The exact value from the Beat posterior 0.83"
```

```
#Example Code from lec

# ########################################################################################
# # Generates samples from the joint posterior distribution of the parameters
# # in the x1,....xn iid Normal(theta,sigma^2) model with
# # prior p(theta,sigma^2) propto 1/sigma^2
# ########################################################################################
# NormalNonInfoPrior <- function(NDraws,Data){
#   Datamean <- mean(Data)
#   s2 <- var(Data)
#   n <- length(Data)
#   PostDraws <- matrix(0,NDraws,2)
#   PostDraws[,2] <- ((n-1)*s2)/rchisq(NDraws,n-1)
#   PostDraws[,1] <- rnorm(NDraws,mean=Datamean,sd=sqrt(PostDraws[,2]/n))
#
#   return(PostDraws)
# }
#
# Nobs <- 10000
# Ndraws <- 10000
# Data <- rnorm(Nobs,5,10)  # Sampling Nobs observations from the N(5,10) density##
# PostDraws <- NormalNonInfoPrior(Ndraws,Data) # Generating draws from the joint posterior of mu and si
# hist(PostDraws[,1])           # Plotting the histogram of mu-draws
# hist(PostDraws[,2])       # Plotting the histogram of sigma^2-draws
#
# # Examples of probability calculations
# mean(PostDraws[,1]>4.9 & PostDraws[,1]<5.1) # Approximate posterior probability of 4.9 < mu < 5.1
# mean(PostDraws[,2]>99 & PostDraws[,2]<101)  # Approximate posterior probability of 99 < sigma^2 < 101
#
#
# ###########################################################################################
# # Generate samples from the joint posterior distribution of theta=(theta_1,...,theta_K)
# # for the multinomial model with K categories and a Dirichlet prior for theta.
# ###########################################################################################
# Dirichlet <- function(NDraws,y,alpha){
#   K <- length(alpha)
#   xDraws <- matrix(0,NDraws,K)
#   thetaDraws <- matrix(0,NDraws,K) # Matrix where the posterior draws of theta are stored
#   for (j in 1:K){
#     xDraws[,j] <- rgamma(NDraws,shape=alpha[j]+y[j],rate=1)
#   }
#   for (ii in 1:NDraws){
#     thetaDraws[ii,] <- xDraws[ii,]/sum(xDraws[ii,])
#   }
#   return(thetaDraws)
# }
```

```
#
# ###########   Setting up data and prior  #################
# y <- c(180,230,62,41) # Data of counts for each category
# p <- y/sum(y)
# alpha_const <- 1
# alpha <- alpha_const*c(15,15,10,10) # Dirichlet prior hyperparameters
# NDraws <- 10000 # Number of posterior draws
#
# ###########   Posterior sampling from Dirichlet  #################
# thetaDraws <- Dirichlet(NDraws,y,alpha)
#
# K <- length(y)
# ########### Summary statistics from the posterior sample ########
# for (k in 1:K){
#   mean(thetaDraws[,k])
#   sqrt(var(thetaDraws[,k]))
# }
#
# sum(thetaDraws[,2]>thetaDraws[,1])/NDraws # p(theta2>theta1 | y)
# # Posterior probability that Android has largest share, i.e. p(theta_2 > max(theta_1,theta_3,theta_4)
# Index_max <- matrix(0,NDraws,1)
# for (ii in 1:NDraws){
# Index_max[ii,1] <- which.max(thetaDraws[ii,])
# }
# mean(Index_max==2)
#
# # Plot histograms of the posterior draws
# plot.new() # Opens a new graphical window
# par(mfrow = c(2,2)) # Splits the graphical window in four parts (2-by-2 structure)
# hist(thetaDraws[,1],25) # Plots the histogram of theta[,1] in the upper left subgraph
# hist(thetaDraws[,2],25)
# hist(thetaDraws[,3],25)
# hist(thetaDraws[,4],25)

# marginal likelihood for the model
# beta(post_alpha,post_beta)/beta(alpha,beta) # Ratio of beta functions
```

C

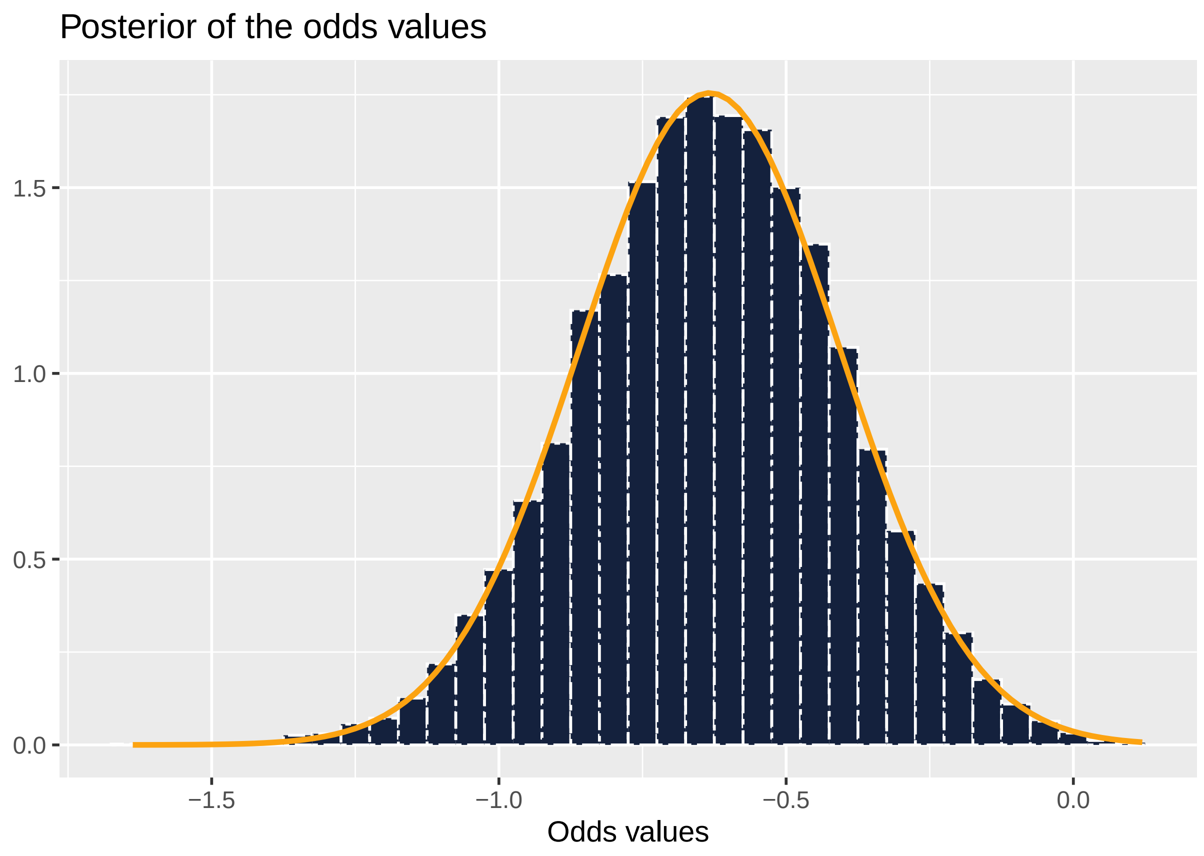
Draw 10000 random values from the posterior of the odds $\phi = \frac{\theta}{1-\theta}$ by using the previous random draws from the Beta posterior for $\theta$and plot the posterior distribution of ϕ.
[Hint: hist() and density() can be utilized].

Now we want to draw 10000 random values from the posterior of the odds

```
grp<-function(n_d,n,s,a,b){
        t_n = n
        s   = s
        f   = n-s
        a   = a
        b   = b
        #Beta(alpha+s,beta+s)
        a_new = a+s
        b_new = b+f
        res<-rbeta(n_d,a_new,b_new)
        res2<-data.frame(x=log(res/(1-res)))
        plt<-ggplot(res2,aes(x=x))+geom_histogram(aes(y=..density..),
                                      color="white",linetype=6,
                                      fill='#14213D',binwidth = 0.05)+
                                      labs(x='Odds values',y=' ',
                                      title ='Posterior of the odds values')+
                                      stat_function(fun = dnorm,
                                          args = list(mean = mean(res2$x),
                                          sd = sd(res2$x)),
                                          color = "#FCA311", size = 1)
        return(plt)}

grp(10000,70,22,8,8)
```

```
## Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.
## i Please use `after_stat(density)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Question 2 Log-Normal distribution and Gini Coefficient

Assume that you have asked 8 randomly selected persons about their monthly income (in thousands Swedish Krona) and obtained the following eight observations: 33, 24, 48, 32, 55, 74, 23, and 17. A common model for non-negative continuous variables is the log-normal distribution.
The log-normal distribution $logN(\mu, \sigma^2)$ has density function:

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

where $y > 0, -inf < \mu < inf$ and σ^2 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 $y_1, ..., y_n|\mu, \sigma^2 \sim logN(\mu, \sigma^2)$, where $\mu = 3.6$ is assumed to be known but σ^2 is unknown with non-informative prior $p(\sigma^2) \propto 1/\sigma^2$ The posterior for σ^2 is the $inv - \chi^2(n, \tau^2)$ distribution, where:

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

A

Draw 10000 random values from the posterior of σ^2 by assuming $\mu = 3:6$ and plot the posterior distribution.

First we find the value of tau, the question assume that we have 8 randomly selected persons thus we have n = 8 and we have the value of mu = 3.6 Given, to find the value of tau we use this equation:

$$\tau^2 \ = \sum_1^n (log\ y_i \ - \mu)^2/n$$

,
Note A non-informative prior is a prior distribution that is chosen to express little to no prior information about the parameters. Non-informative priors are often chosen to avoid introducing bias or strong assumptions into the model, and to allow the data to have a greater influence on the posterior distribution. Examples of non-informative priors include the uniform distribution, the Jeffreys prior, and the reference prior.
It's important to note that a non-informative prior is not necessarily a prior with no information at all, but rather one that expresses a minimal amount of information that is consistent with our knowledge and beliefs before observing the data. In practice, the choice of prior distribution often depends on the specific problem and the available prior knowledge.
in the question we have inverse chi distribution is our posterior, the inverse-chi-squared distribution (or inverted-chi-square distribution[1]) is a continuous probability distribution of a positive-valued random variable. It is closely related to the chi-squared distribution. It arises in Bayesian inference, where it can be used as the prior and posterior distribution for an unknown variance of the normal distribution.

the inverse chi distribution is The inverse chi-square distribution with degrees of freedom n and scale parameter s^2 is closely related to the inverse gamma distribution with shape parameter α = n/2 and scale parameter $\beta = 1/(2s^2)$.
In fact, if X is Inv-χ^2(n,s^2) distributed, then Y = (ns^2)/X is Inv-γ(α,β) distributed, and vice versa. then we use the function rinvgamma from r and we change on the pramters shape and scale.

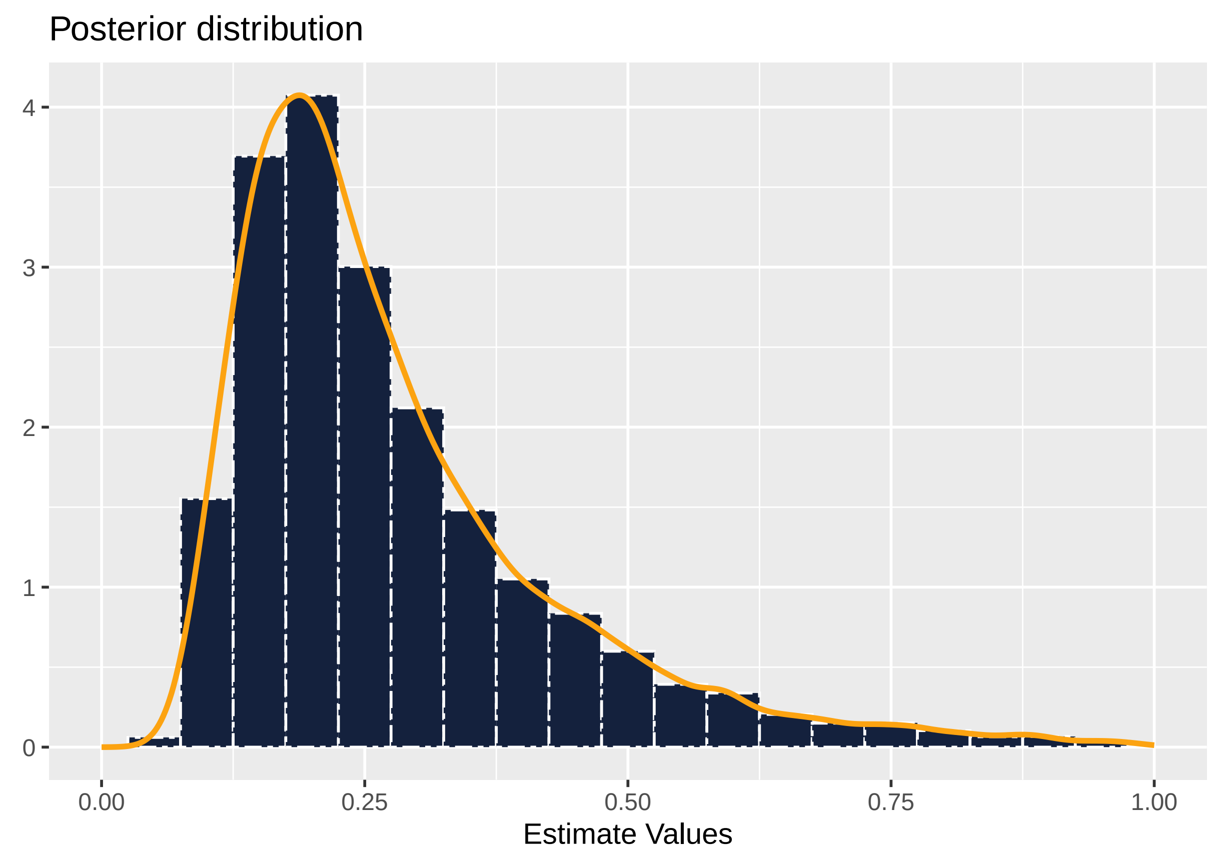
```r
y<-c(33,24,48,32,55,74,23,17)
#n is the sample size
n=8
#the number of draws wanted
nDraws=10000
```

```r
mu= 3.6
logy= (log(y)-mu)^2
tau2=sum(logy)/n
# then we use the function rinvgamma from r and we change on the pramters shape and scale.
df<-data.frame(x=rinvgamma(nDraws, shape=n/2, scale=n*tau2/2))
plt<-ggplot(df,aes(x=x))+geom_histogram(aes(y=..density..),color="white",
                                        linetype=6,
                                     fill='#14213D',binwidth = 0.05)+
                                        labs(x='Estimate Values',y=' ',
                                        title ='Posterior distribution')+
                                        xlim(0,1)+
  geom_density(color = "#FCA311", size = 1)
plt
```

```
## Warning: Removed 125 rows containing non-finite values (`stat_bin()`).

## Warning: Removed 125 rows containing non-finite values (`stat_density()`).

## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```



B

The most common measure of income inequality is the Gini coeffcient, G, where 0 <= G <= 1. G = 0 means a completely equal income distribution, whereas G = 1 means complete income inequality (see

e.g. Wikipedia for more information about the Gini coeffcient).
It can be shown that $G = 2\phi(\sigma/\sqrt{2}) - 1$ when incomes follow a *log* $N(\mu, \sigma^2)$ distribution. $\phi(z)$ 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 coeffcient G for the current data set.

We define The Gini coefficient as a measure of inequality in a distribution, typically used to measure income inequality. It ranges from 0 (perfect equality, where everyone has the same income) to 1 (perfect inequality, where one person has all the income).
A Gini coefficient of 0.5, for example, indicates that 50% of the population has 50% of the total income, while the other 50% of the population has the remaining 50% of the income.

Steps:

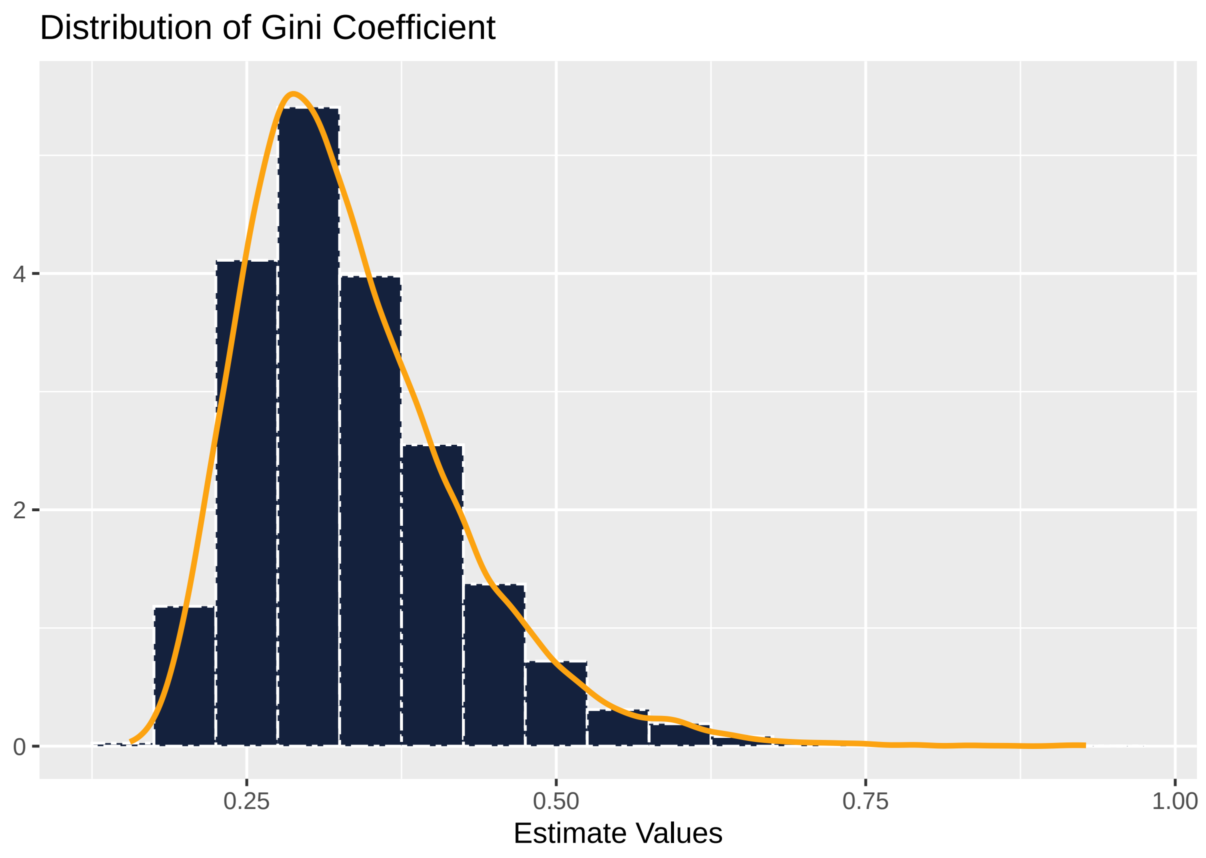
1- We find the value of $\phi(\sigma/\sqrt{2})$ by using the values of σ^2 from the estimated values in A using the formula $\sigma/\sqrt{2}$.
2- We use the function *pnorm* to find the values of ϕ where:
- q: the quantile(s) at which to evaluate the CDF.
mean: the mean of the normal distribution (default value is 0).
- sd: the standard deviation of the normal distribution (default value is 1).
- lower.tail: a logical value indicating whether to compute the lower tail probability (TRUE, default) or the upper tail probability (FALSE).
- log.p: a logical value indicating whether to return the natural logarithm of the probability density (TRUE) or the probability density (FALSE, default).
Note that: Quantiles are points in a probability distribution that divide the distribution into intervals of equal probability.In our case we use the probabilities from the function $x = rinvgamma(nDraws, shape = n/2, scale = n * tau2/2)$.

3- We calculate the value of G by pluggin 1 and 2.

```r
y<-c(33,24,48,32,55,74,23,17)
#n is the sample size
n=length(y)
#the number of draws wanted
nDraws=10000
mu= 3.6
logy= (log(y)-mu)^2
tau2=sum(logy)/n
# then we use the function rinvgamma from r and we change on the pramters shape and scale.
df<-data.frame(x=rinvgamma(nDraws, shape=n/2, scale=n*tau2/2))
# Calculating the value of Phi_arg which is simply the squar root of the estimated sigma
df$phi_arg<-df$x/sqrt(2)
# Calculating the value of Gini index by appling the formula G=2*phi(sigma/sqrt(2))-1

# Xgrid
# x1_grid <- seq(min(X[,2]),max(X[,2]),0.1)
# Mu_draws <- matrix(0,length(x1_grid),2)
# for (ii in 1:length(x1_grid)){
# CurrMu <- BostonRes$betaSample %*% c(1,x1_grid[ii],XNewHouse[-1:-2])
# Mu_draws[ii,] <- quantile(CurrMu,probs=c(0.025,0.975))
# }
# plot(x1_grid,Mu_draws[,1],"n",main="95 % posterior probability intervals as a function of crim",
# xlab="crim", ylab="",ylim=c(10,30))
# lines(x1_grid,Mu_draws[,1],col="blue")
# lines(x1_grid,Mu_draws[,2],col="blue")

df$G<-2*pnorm(sqrt(df$phi_arg),0,1)-1
#Ploting the Gini distribution
ggplot(df,aes(x=G))+geom_histogram(aes(y=..density..),color="white",
                                   linetype=6,
                                   fill='#14213D',binwidth = 0.05)+
  labs(x='Estimate Values',y=' ',
      title ='Distribution of Gini Coefficient')+
  geom_density(color = "#FCA311", size = 1)
```



C

Use the posterior draws from b) to compute a 95% equal tail credible interval for G. A 95% equal tail credible interval (a; b) cuts off 2:5% percent of the posterior probability mass to the left of a, and 2:5% to the right of b.

To find the 95% CI we use the function *qnorm*()in R with mean and Sd drived from the Gini distribution we found in the previous data. here we indicates that 0,025 is the 2,5% cut off point for both uppbe and lower limits.

```r
sample_mean= mean(df$G)
sample_sd= sd(df$G)
# Compute the 95% confidence interval
lower_ci <- qnorm(0.025, mean = sample_mean, sd = sample_sd)
upper_ci <- qnorm(1-0.025, mean = sample_mean, sd = sample_sd)

#In case of utility fun
# nIter <- 1e4
# Mean_log_mu <- mean(log(rnorm(nIter,mean = 92,sd = 2)))
# ExpectedUtility <- function(c, Mean_log_mu){
#   EU <- 60 + sqrt(c)*Mean_log_mu - c
#   return(EU)
# }
#
# cGrid <- seq(0,20,by = 0.01)
# EU <- rep(NA,length(cGrid),1)
# count <- 0
# for (c in cGrid){
#   count <- count + 1
#   EU[count] = ExpectedUtility(c, Mean_log_mu)
# }
# plot(cGrid, EU, type = "l")
# cOpt = cGrid[which.max(EU)] # This is the optimal c
# points(cOpt,ExpectedUtility(c=cOpt, Mean_log_mu), col = "red",pch=19)
# cOpt
# Print the confidence interval
cat("95% confidence interval: [", round(lower_ci, 2), ",", round(upper_ci, 2), "]", "\n")
```

```
## 95% confidence interval: [ 0.16 , 0.5 ]
```

D

Use the posterior draws from b) to compute a 95% Highest Posterior Density Interval (HPDI) for G. Compare the two intervals in (c) and (d). [Hint: 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 the HPDI. Note that you need to order/sort the estimated density values to obtain the HPDI.].

We define the Highest Posterior Density (HPD) interval is a type of confidence interval in Bayesian inference that contains the most credible values for a parameter based on the observed data. Specifically, it is the narrowest interval that contains a specified proportion (usually 95% or 99%) of the posterior distribution of the parameter.
To do so First we generate a sample using the posterior mean and Sd then We use the function HDI to find the 95% CI from library(HDInterval).

```r
library(HDInterval)
```

```
## Warning: package 'HDInterval' was built under R version 4.1.3
```

```r
#First we generate a sample using the posterior mean and Sd
post_G <- rnorm(n = 10000, mean = mean(df$G), sd = sd(df$G))
post_G_density <- density(post_G)

#We use the function HDI to find the 95% CI
hdpi=hdi(post_G, conf = 0.95)

cat("95% equal tail interavl for G : [", round(lower_ci, 2), ",",
    round(upper_ci, 2), "]", "\n")
```

```
## 95% equal tail interavl for G : [ 0.16 , 0.5 ]
```

```r
cat("95% Highest Posterioe Density Interval for G: [", round(hdpi[1], 2), ",",
    round(hdpi[2], 2), "]", "\n")
```

```
## 95% Highest Posterioe Density Interval for G: [ 0.17 , 0.5 ]
```

```r
# Simulate and find the pi dist
# nSim <- 1e5
# x_pred <- matrix(0,nSim,1)
# for (jj in 1:nSim){
#   mu_draw <- rnorm(1,mean = 92,sd = 2)
#   x_pred[jj,1] <- rnorm(1,mean = mu_draw,sd = sqrt(50))
# }
# plot(density(x_pred),type="l",main="Posterior distribution of x_{n+1}",
#xlab="x_{n+1}",ylab="")
```

Question 3 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: (20, 314, 285, 40, 308, 314, 299, 296, 303, 326) where North is located at zero degrees (see Figure 1 on the next page, where the angles are measured clockwise).
To fit with Wikipedia's description of probability distributions for circular data we convert the data into radians $-\pi <= y <= \pi$ .The 10 observations in radians are (-2.79, 2.33, 1.83, -2.44, 2.23, 2.33, 2.07, 2.02, 2.14, 2.54). Assume that these data points conditional on $(\mu, k]$ are independent observations from the following von Mises distribution:

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

where $I_0(k)$ is the modified Bessel function of the first kind of order zero [see ?besselI in R]. The parameter $\mu(-\pi <= y <= \pi)$ 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 2:4. Let $k \sim Exponential(\lambda = 0.5)$ a priori, where λ is the rate parameter of the exponential distribution (so that the mean is $1/\lambda$).

A

Derive the expression for what the posterior $p(k|y, \mu)$ is proportional to. Hence, derive the function f (k) such that $p(k|y, \mu) \propto f(k)$.
Then, plot the posterior distribution of k for the wind direction data over a fine grid of k values. [Hint: you need to normalize the posterior distribution of k so that it integrates to one.]

B

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

First to we drive our joint distribution to find the liklehood function from the model distribution.
We have our model distribution $p(y|\mu, k)$ defined as:

$$P(y|\mu, k) = \prod_{i=1}^{n} \frac{1}{2\pi I_0(k)} * exp(k * Cos(y - \mu))$$
$$= \frac{1}{(2\pi)^n I_0(k)^n} * exp(k * Cos(\sum_{i=1}^{n} y_i - \mu))$$

We have the prior k follows the $exp(\lambda = 0.5)$

$$p(k) = 0.5 * exp(-\lambda * k)$$

now we drive the postrior distribution using bayse formula

$$p(\theta|x) \propto p(x|\theta)\ p(\theta)$$

We plug the liklehood function drived above with the prior we get:

$$p(k|\mu, y) \propto \frac{1}{(2\pi)^n I_0(k)^n} * exp(k * Cos(\sum_{i=1}^{n} y_i - \mu))\ .\ 0.5 * exp(-\lambda * k)$$
$$\propto \frac{1}{I_0(k)^n} * exp(k * Cos(\sum_{i=1}^{n} y_i - \mu) - \frac{k}{2})$$

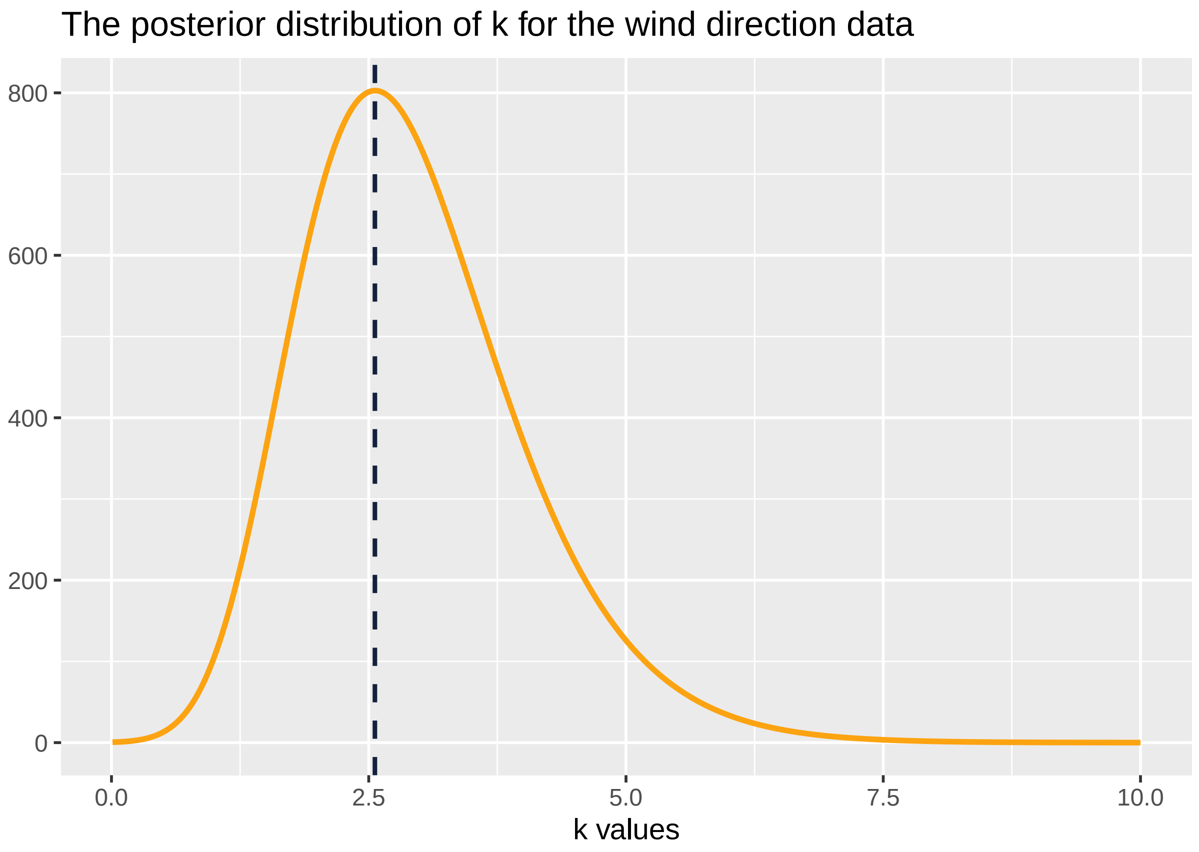
Now we implement the code in R:

```r
# The y and mu values given from the question
y<-c(-2.79,2.33,1.83,-2.44,2.23,2.33,2.07,2.02,2.14,2.54)
mu<- 2.51
n=length(y)

#We generate a sequeance of k values by 0.1
k<-seq(0.01,10,by=0.01)

#The drived posterior function can be expressed by
#(0.5/(besselI(k,0))^n)*exp(k*sum(cos(y-mu))-(k*0.5)
df<- data.frame(x=k,y=(0.5/(besselI(k,0))^n)*exp(k*sum(cos(y-mu))-(k*0.5)))

#Plotting function:
plt<-ggplot(df,aes(x=x,y=y))+geom_line(aes(),
                                               color="#FCA311",linetype=1,size=1)+
  labs(x='k values',y=' ',
       title ='The posterior distribution of k for the wind direction data')+
         geom_vline(xintercept =df$x[which.max(df$y)],
                    # we use df$x[which.max(df$y)] to find the mode
                    linetype = "dashed", color = "#14213D",size=.8)
plt
```



```
#compare the probs
#mean(Betas[,2]>0 & Betas[,3]>0)
#For betas we find the quatiles and interpretated the CI
# Effect_x1x2 <- Betas[,6]
# plot(density(Effect_x1x2),main="Posterior distribution",xlab="beta_5", ylab="")
# #quantile(Effect_x1x2,probs=c(0.025,0.975))
```

References:

1- Bertil Wegmann (2023). Bayesian Learning [Lecture notes]. 732A73, Department of Computer and Information Science,LiU University.

Code Appendix

```r
knitr::opts_chunk$set(echo = TRUE)
library(ggplot2)
library(gridExtra)
library(LaplacesDemon)
ber_fun<-function(n_d,n,s,a,b) {
  #Initial Value of the function givan from the question
  t_n = n
  s   = s
  f   = n-s
  a   = a
  b   = b
  #Beta(alpha+s,beta+s)
  a_new = a+s
  b_new = b+f
  Mean_true= a_new/(a_new+b_new)
  #we take the sqrt to get the Sd insted of the Var
  Sd_true=  sqrt((a_new*b_new)/(((a_new+b_new)^2) * (a_new+b_new+1)))
  # Beta(alpha+s,beta+s)
  mean_theta = c()
  sd_theta = c()
  n_draws = 1:n_d
  #For loop to fill the values of mean and sd based on the draws
  for (i in 1:n_d){
    mean_theta[i]=mean(rbeta(i,a_new,b_new))
    sd_theta[i]=sd(rbeta(i,a_new,b_new))}
  #Binding everything togther
  df<-cbind.data.frame(n_draws,mean_theta,sd_theta)
  #Plot of the mean
  mean<-ggplot(df,aes(x=n_draws))+geom_line(aes(y=mean_theta)
                                            , color='#FCA311', size=.8)+
              geom_line(aes(y=Mean_true), color='#14213D',linetype=3)+
              annotate(geom = "text", x = 8, y = Mean_true,
              label = paste0(format(round(Mean_true, 3), nsmall = 3)))+
              labs(title = 'Sample Means and SD of theta ',
         subtitle = 'As a function of the accumulating number of drawn values',
                      x= ' ', y='Sample Mean')+ theme_classic()
  #Plot of the Sd
  sd<-ggplot(df,aes(x=n_draws))+geom_line(aes(y=sd_theta),
                                          color='#FCA311', size=.8)+
              geom_line(aes(y=Sd_true), color='#14213D',linetype=3)+
              annotate(geom = "text", x = 8, y = Sd_true,
              label = paste0(format(round(Sd_true, 3), nsmall = 3)))+
              labs(x= 'Number of draws', y='Standard Deviation')+
              theme_classic()
  #grid.arrange(mean,sd)
  return(grid.arrange(mean,sd))
}


ber_fun(100,70,22,8,8)
prob<-function(n_d,n,s,a,b,prob){
        t_n = n
        s   = s
        f   = n-s
        a   = a
        b   = b
        #Beta(alpha+s,beta+s)
        a_new = a+s
        b_new = b+f
        p<-pbeta(prob,a_new,b_new,lower.tail = F)
        post<-mean(rbeta(n_d,a_new,b_new)>prob)
        return(list(paste('random values from the posterior with the given condition',post)
        ,paste('The exact value from the Beat posterior', round(p,2))))}
prob(100,70,22,8,8,.3)

#Example Code from lec

# #####################################################################################
# # Generates samples from the joint posterior distribution of the parameters
# # in the x1,....xn iid Normal(theta,sigma^2) model with
# # prior p(theta,sigma^2) propto 1/sigma^2
# #####################################################################################
# NormalNonInfoPrior <- function(NDraws,Data){
#   Datamean <- mean(Data)
#   s2 <- var(Data)
#   n <- length(Data)
#   PostDraws <- matrix(0,NDraws,2)
#   PostDraws[,2] <- ((n-1)*s2)/rchisq(NDraws,n-1)
#   PostDraws[,1] <- rnorm(NDraws,mean=Datamean,sd=sqrt(PostDraws[,2]/n))
#
#   return(PostDraws)
# }
#
# Nobs <- 10000
# Ndraws <- 10000
# Data <- rnorm(Nobs,5,10)  # Sampling Nobs observations from the N(5,10) density##
# PostDraws <- NormalNonInfoPrior(Ndraws,Data) # Generating draws from the joint posterior of mu and si
# hist(PostDraws[,1])           # Plotting the histogram of mu-draws
# hist(PostDraws[,2])       # Plotting the histogram of sigma^2-draws
#
# # Examples of probability calculations
# mean(PostDraws[,1]>4.9 & PostDraws[,1]<5.1) # Approximate posterior probability of 4.9 < mu < 5.1
# mean(PostDraws[,2]>99 & PostDraws[,2]<101)  # Approximate posterior probability of 99 < sigma^2 < 101
#
#
# ########################################################################################
# # Generate samples from the joint posterior distribution of theta=(theta_1,...,theta_K)
# # for the multinomial model with K categories and a Dirichlet prior for theta.
# ########################################################################################
# Dirichlet <- function(NDraws,y,alpha){
#   K <- length(alpha)
#   xDraws <- matrix(0,NDraws,K)
#   thetaDraws <- matrix(0,NDraws,K) # Matrix where the posterior draws of theta are stored
#   for (j in 1:K){
#     xDraws[,j] <- rgamma(NDraws,shape=alpha[j]+y[j],rate=1)
#   }
```

```r
#   for (ii in 1:NDraws){
#     thetaDraws[ii,] <- xDraws[ii,]/sum(xDraws[ii,])
#   }
#   return(thetaDraws)
# }
#
# ###########   Setting up data and prior  #################
# y <- c(180,230,62,41) # Data of counts for each category
# p <- y/sum(y)
# alpha_const <- 1
# alpha <- alpha_const*c(15,15,10,10) # Dirichlet prior hyperparameters
# NDraws <- 10000 # Number of posterior draws
#
# ###########   Posterior sampling from Dirichlet  #################
# thetaDraws <- Dirichlet(NDraws,y,alpha)
#
# K <- length(y)
# ########### Summary statistics from the posterior sample ########
# for (k in 1:K){
#   mean(thetaDraws[,k])
#   sqrt(var(thetaDraws[,k]))
# }
#
# sum(thetaDraws[,2]>thetaDraws[,1])/NDraws # p(theta2>theta1 | y)
# # Posterior probability that Android has largest share, i.e. p(theta_2 > max(theta_1,theta_3,theta_4)
# Index_max <- matrix(0,NDraws,1)
# for (ii in 1:NDraws){
# Index_max[ii,1] <- which.max(thetaDraws[ii,])
# }
# mean(Index_max==2)
#
# # Plot histograms of the posterior draws
# plot.new() # Opens a new graphical window
# par(mfrow = c(2,2)) # Splits the graphical window in four parts (2-by-2 structure)
# hist(thetaDraws[,1],25) # Plots the histogram of theta[,1] in the upper left subgraph
# hist(thetaDraws[,2],25)
# hist(thetaDraws[,3],25)
# hist(thetaDraws[,4],25)

# marginal likelihood for the model
# beta(post_alpha,post_beta)/beta(alpha,beta) # Ratio of beta functions
grp<-function(n_d,n,s,a,b){
        t_n = n
        s   = s
        f   = n-s
        a   = a
        b   = b
        #Beta(alpha+s,beta+s)
        a_new = a+s
        b_new = b+f
        res<-rbeta(n_d,a_new,b_new)
        res2<-data.frame(x=log(res/(1-res)))
        plt<-ggplot(res2,aes(x=x))+geom_histogram(aes(y=..density..),
                                              color="white",linetype=6,
                                              fill='#14213D',binwidth = 0.05)+
                                              labs(x='Odds values',y=' ',
                                              title ='Posterior of the odds values')+
                                              stat_function(fun = dnorm,
                                                  args = list(mean = mean(res2$x),
                                                  sd = sd(res2$x)),
                                                  color = "#FCA311", size = 1)
        return(plt)}

grp(10000,70,22,8,8)
y<-c(33,24,48,32,55,74,23,17)
#n is the sample size
n=8
#the number of draws wanted
nDraws=10000
mu= 3.6
logy= (log(y)-mu)^2
tau2=sum(logy)/n
# then we use the function rinvgamma from r and we change on the pramters shape and scale.
df<-data.frame(x=rinvgamma(nDraws, shape=n/2, scale=n*tau2/2))
plt<-ggplot(df,aes(x=x))+geom_histogram(aes(y=..density..),color="white",
                                                    linetype=6,
                                              fill='#14213D',binwidth = 0.05)+
                                                    labs(x='Estimate Values',y=' ',
                                                    title ='Posterior distribution')+
                                                    xlim(0,1)+
  geom_density(color = "#FCA311", size = 1)
plt
y<-c(33,24,48,32,55,74,23,17)
#n is the sample size
n=length(y)
#the number of draws wanted
nDraws=10000
mu= 3.6
logy= (log(y)-mu)^2
tau2=sum(logy)/n
# then we use the function rinvgamma from r and we change on the pramters shape and scale.
df<-data.frame(x=rinvgamma(nDraws, shape=n/2, scale=n*tau2/2))
# Calculating the value of Phi_arg which is simply the squar root of the estimated sigma
df$phi_arg<-df$x/sqrt(2)
# Calculating the value of Gini index by appling the formula G=2*phi(sigma/sqrt(2))-1

# Xgrid
# x1_grid <- seq(min(X[,2]),max(X[,2]),0.1)
# Mu_draws <- matrix(0,length(x1_grid),2)
# for (ii in 1:length(x1_grid)){
# CurrMu <- BostonRes$betaSample %*% c(1,x1_grid[ii],XNewHouse[-1:-2])
# Mu_draws[ii,] <- quantile(CurrMu,probs=c(0.025,0.975))
# }
# plot(x1_grid,Mu_draws[,1],"n",main="95 % posterior probability intervals as a function of crim",
# xlab="crim", ylab="",ylim=c(10,30))
# lines(x1_grid,Mu_draws[,1],col="blue")
# lines(x1_grid,Mu_draws[,2],col="blue")

df$G<-2*pnorm(sqrt(df$phi_arg),0,1)-1
#Ploting the Gini distribution
ggplot(df,aes(x=G))+geom_histogram(aes(y=..density..),color="white",
                                   linetype=6,
                                   fill='#14213D',binwidth = 0.05)+
  labs(x='Estimate Values',y=' ',
       title ='Distribution of Gini Coefficient')+
  geom_density(color = "#FCA311", size = 1)
sample_mean= mean(df$G)
sample_sd= sd(df$G)
# Compute the 95% confidence interval
lower_ci <- qnorm(0.025, mean = sample_mean, sd = sample_sd)
upper_ci <- qnorm(1-0.025, mean = sample_mean, sd = sample_sd)

#In case of utility fun
# nIter <- 1e4
# Mean_log_mu <- mean(log(rnorm(nIter,mean = 92,sd = 2)))
# ExpectedUtility <- function(c, Mean_log_mu){
#   EU <- 60 + sqrt(c)*Mean_log_mu - c
#   return(EU)
# }
#
# cGrid <- seq(0,20,by = 0.01)
# EU <- rep(NA,length(cGrid),1)
# count <- 0
# for (c in cGrid){
#   count <- count + 1
#   EU[count] = ExpectedUtility(c, Mean_log_mu)
# }
# plot(cGrid, EU, type = "l")
# cOpt = cGrid[which.max(EU)] # This is the optimal c
# points(cOpt,ExpectedUtility(c=cOpt, Mean_log_mu), col = "red",pch=19)
# cOpt
# Print the confidence interval
cat("95% confidence interval: [", round(lower_ci, 2), ",", round(upper_ci, 2), "]", "\n")

library(HDInterval)

#First we generate a sample using the posterior mean and Sd
post_G <- rnorm(n = 10000, mean = mean(df$G), sd = sd(df$G))
post_G_density <- density(post_G)

#We use the function HDI to find the 95% CI
hdpi=hdi(post_G, conf = 0.95)

cat("95% equal tail interavl for G : [", round(lower_ci, 2), ",",
    round(upper_ci, 2), "]", "\n")

cat("95% Highest Posterioe Density Interval for G: [", round(hdpi[1], 2), ",",
    round(hdpi[2], 2), "]", "\n")
```

```
# Simulate and find the pi dist
# nSim <- 1e5
# x_pred <- matrix(0,nSim,1)
# for (jj in 1:nSim){
#   mu_draw <- rnorm(1,mean = 92,sd = 2)
#   x_pred[jj,1] <- rnorm(1,mean = mu_draw,sd = sqrt(50))
# }
# plot(density(x_pred),type="l",main="Posterior distribution of x_{n+1}",
#xlab="x_{n+1}",ylab="")

# The y and mu values given from the question
y<-c(-2.79,2.33,1.83,-2.44,2.23,2.33,2.07,2.02,2.14,2.54)
mu<- 2.51
n=length(y)

#We generate a sequeance of k values by 0.1
k<-seq(0.01,10,by=0.01)

#The drived posterior function can be expressed by
#(0.5/(besselI(k,0))^n)*exp(k*sum(cos(y-mu))-(k*0.5)
df<- data.frame(x=k,y=(0.5/(besselI(k,0))^n)*exp(k*sum(cos(y-mu))-(k*0.5)))

#Plotting function:
plt<-ggplot(df,aes(x=x,y=y))+geom_line(aes(),
                                          color="#FCA311",linetype=1,size=1)+
  labs(x='k values',y=' ',
       title ='The posterior distribution of k for the wind direction data')+
         geom_vline(xintercept =df$x[which.max(df$y)],
                    # we use df$x[which.max(df$y)] to find the mode
                    linetype = "dashed", color = "#14213D",size=.8)
plt
#compare the probs
#mean(Betas[,2]>0 & Betas[,3]>0)
#For betas we find the quatiles and interpretated the CI
# Effect_x1x2 <- Betas[,6]
# plot(density(Effect_x1x2),main="Posterior distribution",xlab="beta_5", ylab="")
# #quantile(Effect_x1x2,probs=c(0.025,0.975))


######## Example code y
# Mu <- Betas[,1] + Betas[,2]*50 + Betas[,3]*50**2 + Betas[,4]*25 + Betas[,5]*25**2 + Betas[,6]*50*25
# Sigma <- sqrt(Sigma2)
# y_Vals <- rnorm(10000,Mu,Sigma)
# plot(density(y_Vals),main="Posterior predictive distribution of y",xlab="y", ylab="")
######## Example code pi
# x_obs <- as.vector(c(1,40,1))
# lin_pred <- Betas%*%x_obs
# p_i <- exp(lin_pred)/(1+exp(lin_pred))
# plot(density(p_i),type="l",main="Posterior distribution of p_i",xlab="p_i",ylab="")

######

# Example code Pr(Y)
```

```r
# T_y <- max(y)
# T_y_rep <- matrix(0,nIter,1)
# Mu <- Betas %*% t(X)
# for (ii in 1:nIter){
# y_Vals <- rnorm(length(y),Mu[ii,],Sigma[ii])
# T_y_rep[ii,1] <- max(y_Vals)
# }
# mean(T_y_rep >= T_y)
# New val x
# alpha_n <- 2326
# beta_n <- 7
# theta <- rgamma(1e4,shape = alpha_n,rate = beta_n)
# Q_6 <- rpois(1e4,theta)
# hist(Q_6,main="Posterior distribution",xlab="Q_6",ylab="")
# mean(Q_6 > 350)
```