Help File

Lab 1

Assignment 1 *Daniel Bernoulli*

Let $y_1, ..., y_n \mid \theta \sim \text{Bern}(\theta)$, and assume that you have obtained a sample with s = 13 successes in n = 50 trials. Assume a Beta(α_0, β_0) prior for θ and let $\alpha_0 = \beta_0 = 5$.

Task 1a

Question:Draw random numbers from the posterior $\theta \mid y \sim Beta(\alpha_0 + s, \beta_0 + f)$, where $y = (y_1, ..., y_n)$, 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()].

The mean value and the standard deviation of the Beta distribution for θ are calculated by the below formulas:

$$\begin{aligned} E[\theta] &= \frac{a_0 + s}{a_0 + s + b_0 + f} \\ &= \frac{18}{60} \\ &= 0.3 \end{aligned}$$

$$\begin{aligned} Var[\theta] &= \frac{(a_0 + s)(b_0 + f)}{\Big((a_0 + s) + (b_0 + f)\Big)^2 \Big((a_0 + s) + (b_0 + f) + 1\Big)} \\ &= \frac{18 \cdot 42}{(18 + 42)^2 (18 + 42 + 1)} \\ &= 0.003442623 \end{aligned}$$

$$SD[\theta] = \sqrt{Var[\theta]} = 0.05867387$$

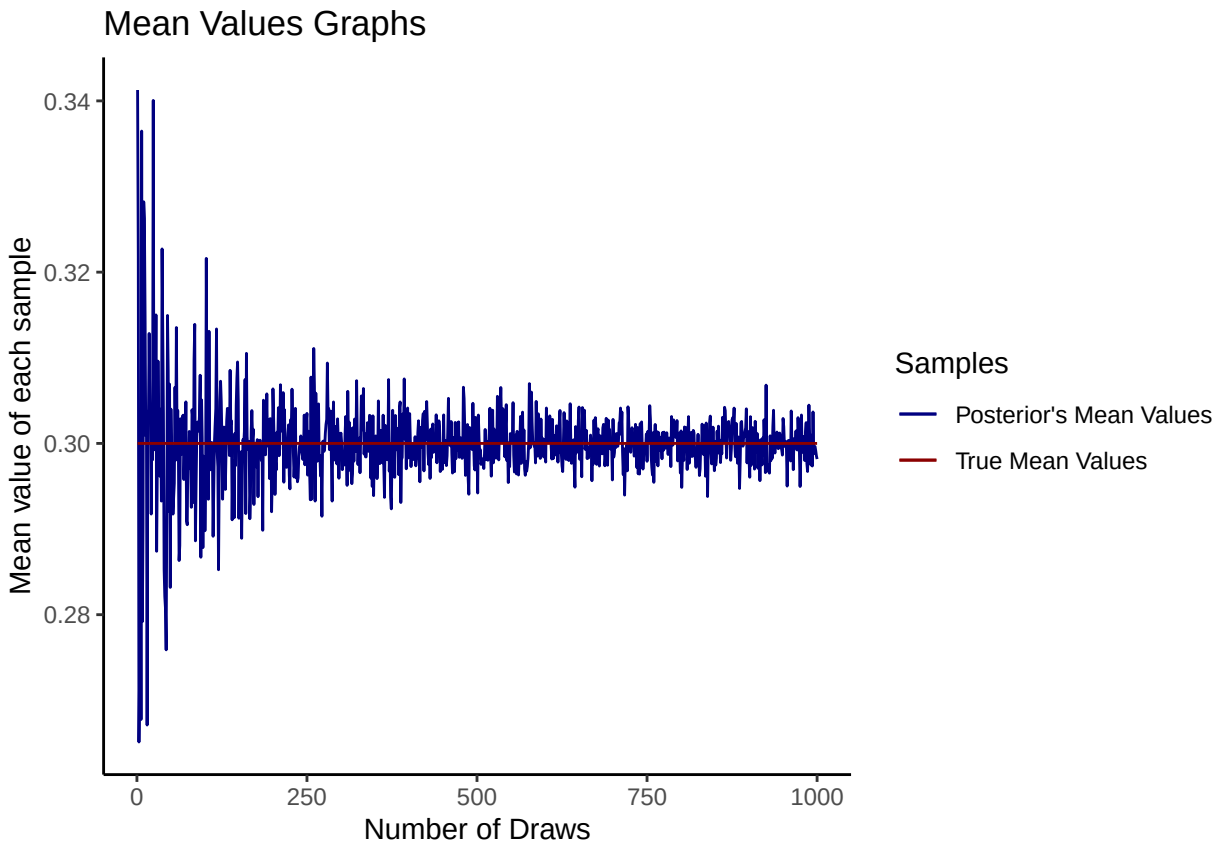
Where a_0 and b_0 are the arguments of the Beta prior, s is the number of successes and f is the number of failures.

```
#parameters
a0 <- 5
b0 <- 5
n <- 50
s <- 13
f <- n-s

#true mean
mean_true <- (a0 + s)/(a0 + s + b0 + f)
#true var
var_true <- ((a0 + s)*(b0 + f)) / (((a0 + s + b0 + f)^2)*(a0 + s + b0 + f + 1))
#true sd
sd_true <- sqrt(var_true)
```

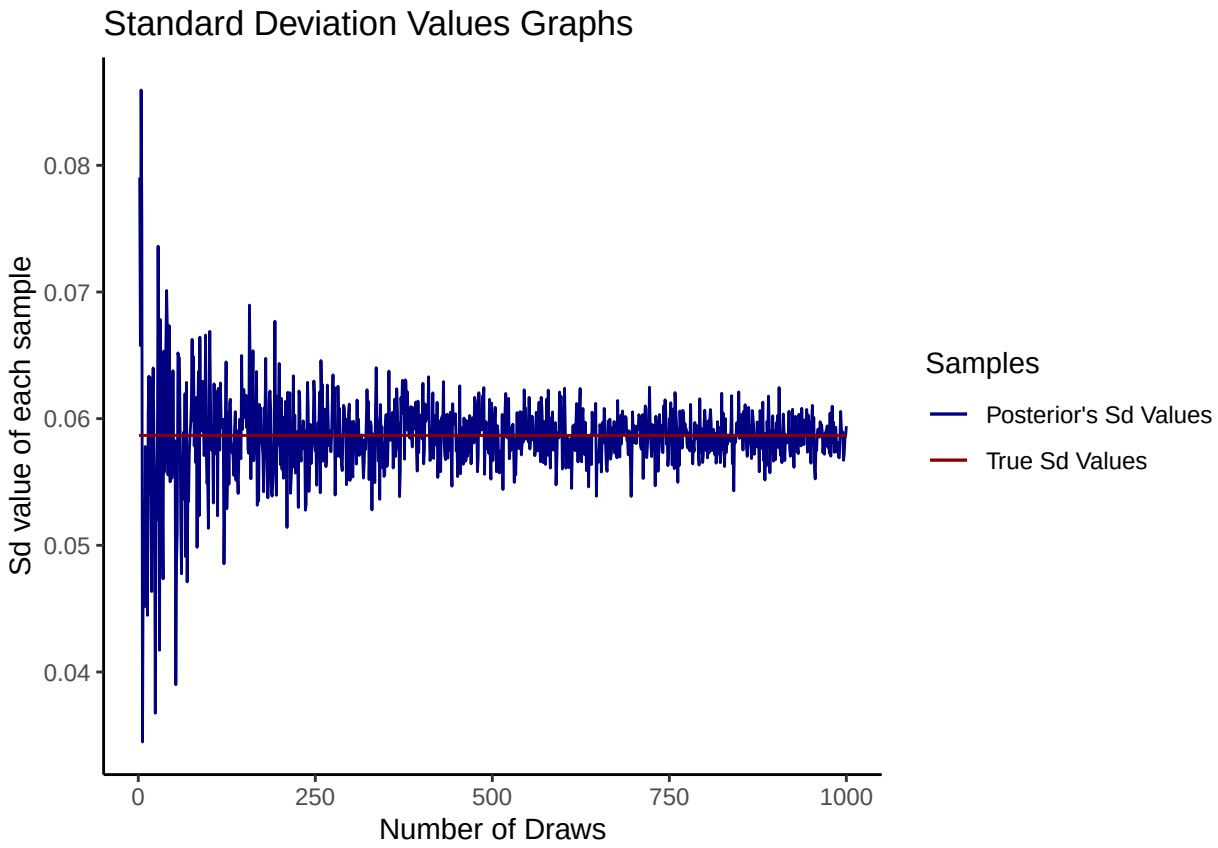
```r
set.seed(12345)

#calculate posterior's mean
mean_posterior = c()
for (i in 1:1000) {
  #rbeta generates random deviates
  mean_posterior[i] = mean(rbeta(n = i, shape1 = a0 + s, shape2 = b0 + f))
}

#calculate posterior's sd
sd_posterior = c()
for (i in 1:1000) {
  sd_posterior[i] = sd(rbeta(n = i, shape1 = a0 + s, shape2 = b0 + f))
}
```

The graphs represent the mean value and the standard deviation of the Beta distribution for θ and the mean value $E[\theta|y]$ (red line) and standard deviation $SD[\theta|y]$ from the posterior $\theta|y \sim Beta(a_0 + s, b_0 + f)$ (blue points), where $y = y_1, y_2, ..., y_n$.

```r
#data for plots
df_plot1 <- data.frame("draws" = 1:1000,
                       "mean_true" = mean_true,
                       "sd_true" = sd_true,
                       "mean_posterior" = mean_posterior,
                       "sd_posterior" = sd_posterior)

library(ggplot2)

#plot of mean values
ggplot(df_plot1) +
  geom_line(aes( x = draws, y = mean_posterior, color = "nany")) +
  geom_line(aes(x = draws, y = mean_true, color = "red4")) +
  theme(legend.position="right") +
  scale_color_manual(values=c('navy','red4'),
                     name = "Samples",
                     labels = c("Posterior's Mean Values","True Mean Values" )) +
  ggtitle("Mean Values Graphs") +
  xlab("Number of Draws") +
  ylab("Mean value of each sample") +
  theme_classic()
```




```
#plot of sd values
ggplot(df_plot1) +
  geom_line(aes( x = draws, y = sd_posterior, color = "nany")) +
  geom_line(aes(x = draws, y = sd_true, color = "red4")) +
  theme(legend.position="right") +
  scale_color_manual(values=c('navy','red4'),
                     name = "Samples",
                     labels = c("Posterior's Sd Values","True Sd Values" )) +
  ggtitle("Standard Deviation Values Graphs") +
  xlab("Number of Draws") +
  ylab("Sd value of each sample") +
  theme_classic()
```



From the above plots, it could be seen that both posterior's mean and standard deviation values converge to the actual mean and standard deviation values, respectively. More specifically, between 0 and approximately 250 draws in both graphs, some of the posterior's values abstain from true values. However, after the 250 draws, the posterior's values start to converge more and more to the true ones in both graphs.

Task 1b

Question:Use simulation (nDraws = 10000) to compute the posterior probability $\Pr(\theta < 0.3|y)$ and compare with the exact value from the Beta posterior. [Hint: use pbeta()].

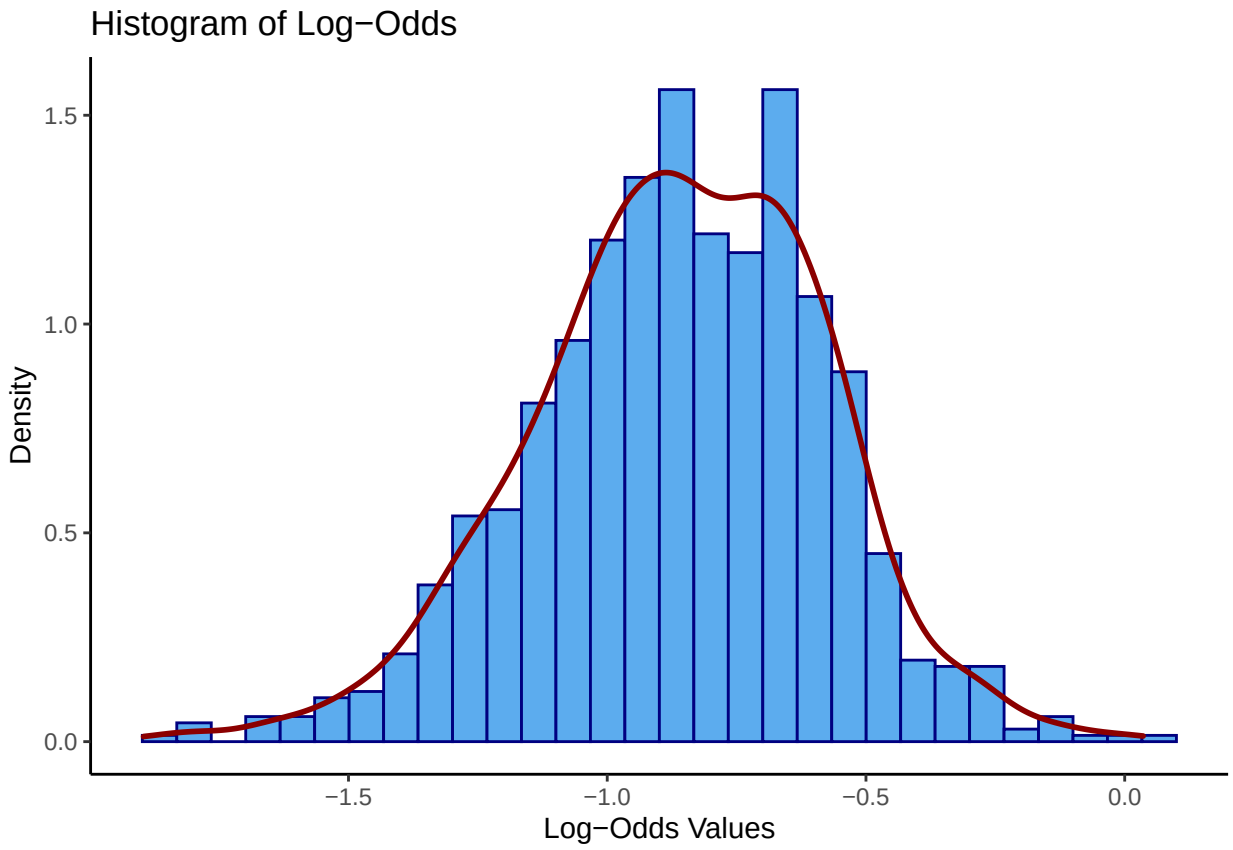
```
set.seed(12345)

#generates 1,000 random deviates.
posterior_sample <- rbeta(n = 1000, shape1 = a0+s, shape2 = b0+f)

#posterior probability
posterior_prob <- sum(posterior_sample < 0.3)/1000

#exact posterior prob
#pbeta the distribution function
exact_prob <- pbeta(q = 0.3, shape1 = a0+s, shape2 = b0+f)
```

The posterior probability $P(\theta < 0.3|y)$ equals 0.506, and the exact probability value from the Beta posterior is 0.5150226; thus, it could be assumed that both values are pretty similar.

Task 1c

Question:Simulate draws from the posterior distribution of the log-odds $\phi = log(\frac{\theta}{1-\theta})$ using simulated draws from the Beta posterior for θ and plot the posterior distribution of ϕ (nDraws = 10000). [Hint: hist() and density() can be utilized].

```
#log-odds
phi <- log(posterior_sample/(1-posterior_sample))

#data for plot
df_plot2 <- data.frame("phi" = phi)

#plot of log-odds
ggplot(df_plot2, aes(x=phi)) +
  geom_histogram(bins = 30, color = "navy", fill = "steelblue2", aes(y=..density..)) +
  geom_density(colour = "red4", size = 1) +
  ggtitle("Histogram of Log-Odds") +
  xlab("Log-Odds Values") +
  ylab("Density") +
  theme_classic()
```



Assignment 2 *Log-normal distribution and the Gini coefficient.*

Assume that you have asked 9 randomly selected persons about their monthly in- come (in thousands Swedish Krona) and obtained the following nine observations: 33, 24, 48, 32, 55, 74, 23, 76 and 17. A common model for non-negative continuous variables is the log-normal distribution. The log-normal distribution $log\,\mathcal{N}(\mu, \sigma^2)$ has density function:

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

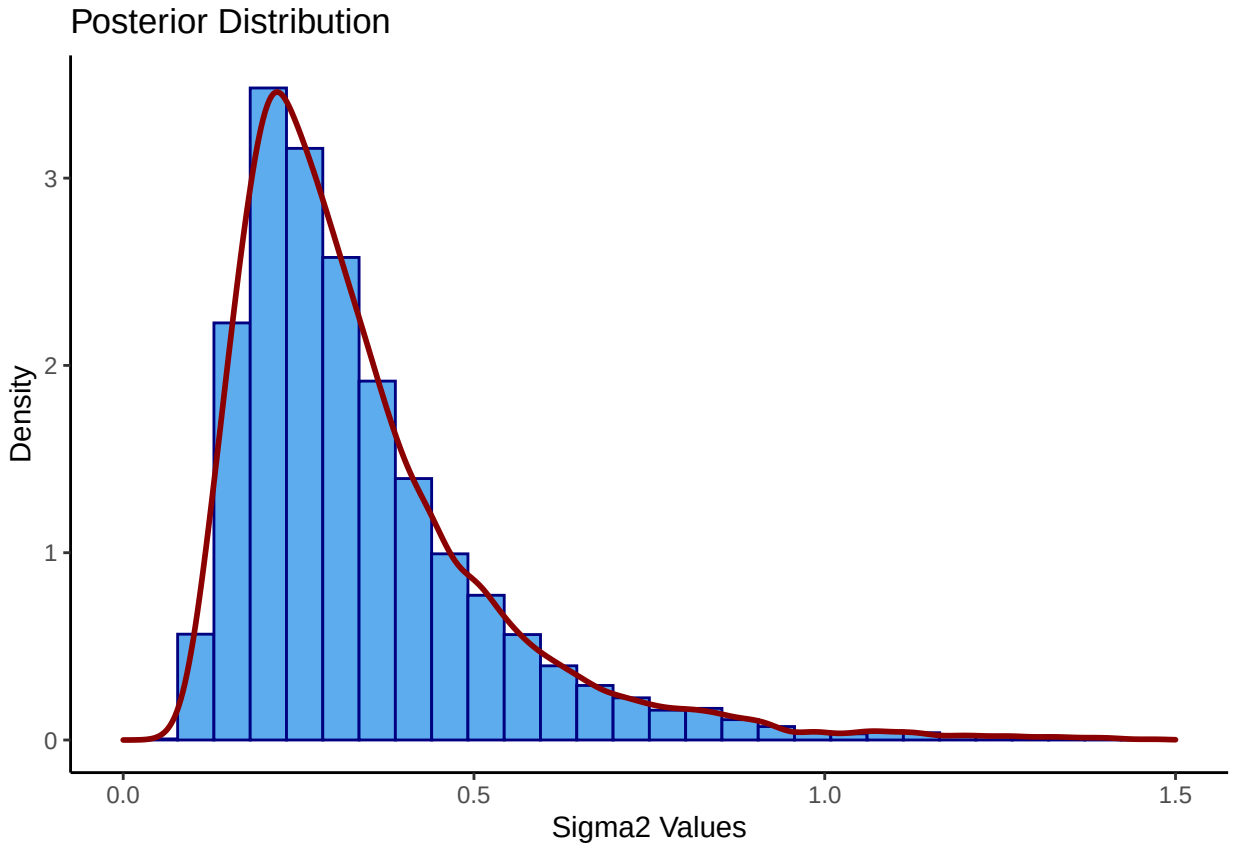
where $y > 0, \mu > 0$ ND σ^2. The log-normal distribution is related to the normal distribution as follows: if $y \sim \log\ \mathcal{N}(\mu, \sigma^2)$ then $\log\ y \sim \mathcal{N}(\mu, \sigma^2)$. Let $y_1, ..., y_n \mid \mu, \sigma^2 \overset{\text{iid}}{\sim} log\mathcal{N}(\mu, \sigma^2)$, where $\mu = 3.5$ is assumed to be known but σ^2 is unknown with non-informative prior $p(\sigma^2) \propto \frac{1}{\sigma^2}$. The posterior for σ^2 is the $Inv\ \chi^2(n, \tau^2)$ distribution, where

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

Task 2a

Question:Simulate 10, 000 draws from the posterior of σ^2 by assuming $\mu = 3.5$ and plot the posterior distribution.

```
#data for plot
df_plot2 <- data.frame("sigma2" = sigma2)

#plot
ggplot(df_plot2, aes(x=sigma2)) +
  geom_histogram(bins = 30, color = "navy", fill = "steelblue2", aes(y=..density..)) +
  geom_density(colour = "red4", size = 1) +
  scale_x_continuous(limits = c(0,1.5)) +
  ggtitle("Posterior Distribution") +
  xlab("Sigma2 Values") +
  ylab("Density") +
  theme_classic()
```



Task 2b

Question The most common measure of income inequality is the Gini coefficient, G, where $0 \leqslant G \leqslant 1$. 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 coefficient). It can be shown that $G = 2\Phi(\frac{\sigma}{\sqrt{2}}) - 1$ when incomes follow a log $\mathcal{N}(\mu, \sigma)$ 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 coefficient G for the current data set.

```
#Gini calculation
gini <- 2*pnorm(sqrt(sigma2)/sqrt(2)) - 1
```

Task 2c

Question:Use the posterior draws from b) to compute a 95% equal tail credible interval for G. A 95% equal tail 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.

```
#producing sample quantiles corresponding to the probabilities
interval <- quantile(gini, probs = c(0.025,0.975))

#table for the interval
df_intervals <- data.frame("lower_bound" = interval[1], "upper_bound" = interval[2])
colnames(df_intervals) <- c("lower bound", "upper bound")
rownames(df_intervals) <- c("95% Equal Tail Credible Interval")
knitr::kable(df_intervals)
```

	lower bound	upper bound
95% Equal Tail Credible Interval	0.197551	0.4908131

The above table illustrates the 95% equal tail credible interval for the Gini coefficient.

Task 2d

Question 2d: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.].

```
#kernel density estimation
gini_density <- density(gini)

df_density <- data.frame(
  #the n coordinates of the points where the density is estimated
  "coord" = gini_density$x,
  #the estimated density values
  "estimated_vals" = gini_density$y)

#order/sort the estimated density values
df_density <- df_density[order(gini_density$y, decreasing = TRUE),]

#calculate the probs
df_density$probs <- cumsum(df_density$estimated_vals)/sum(df_density$estimated_vals)

#get the indexes
index <- which(df_density$probs <= 0.95)

#get the probs
hdps <- df_density[index,]

#low hdpi
low_hdpi <- min(hdps$coord)
#upper hdpi
upp_hdpi <- max(hdps$coord)

intervals <- data.frame("lower_bound" = c(interval[1],low_hdpi), "upper_bound" = c(interval[2],upp_hdpi
rownames(intervals) <- c("95% Equal Tail Credible Interval","95% Highest Posterior Density Interval")

knitr::kable(intervals)
```

	lower_bound	upper_bound
95% Equal Tail Credible Interval	0.1975510	0.4908131
95% Highest Posterior Density Interval	0.1810286	0.4621952

The above table illustrates the 95% equal tail credible interval and the 95% highest posterior density interval for the Gini coefficient. It could be assumed that both intervals have almost similar lower and upper bounds.

Assignment 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 10 different days. The data are recorded in degrees: (285, 296, 314, 20, 299, 296, 40, 303, 326, 308), 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 $-\pi \leqslant y \leqslant \pi$. The 10 observations in radians are: (1.83, 2.02, 2.33, -2.79, 2.07, 2.02, -2.44, 2.14, 2.54, 2.23). Assume that these data points conditional on (μ, κ) are independent observations from the following von Mises distribution:

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

where $I_0(\kappa)$ is the modified Bessel function of the first kind of order zero. The parameter $\mu(-\pi \leqslant y \leqslant \pi)$ is the mean direction and $\kappa > 0$ is called the concentration parameter. Large κ gives a small variance around μ, and vice versa. Assume that μ is known to be 2.39. Let $\kappa \sim Exponential(\lambda = 1)$ a priori, where λ is the rate parameter of the exponential distribution (so that the mean is $1/\lambda$).

Task 3a

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

The posterior is given by the below expression:

$$p(\kappa|y, \mu) = \frac{p(y, \mu|\kappa) \cdot p(\kappa)}{\int_k p(y, \mu|\kappa) \cdot p(\kappa) d\kappa}$$

The numerator is consisted of the likelihood $p(y, \mu|\kappa)$ and the prior $p(\kappa)$. The denominator is the marginal likelihood, which is the normalising constant, ensuring that the posterior distribution of κ adds up to one.

The likelihood is given by the below formula:

$$\begin{aligned}
p(y, \mu|\kappa) &= \prod_{i=1}^{n} p(y|\mu, \kappa) \\
&= \prod_{i=1}^{n} \frac{\exp(\kappa \cdot \cos(y_i - \mu))}{2\pi \cdot I_o(\kappa)} \\
&= \Big(\frac{1}{2\pi \cdot I_o(\kappa)}\Big)^n \exp(\kappa \cdot \sum_{i=1}^{n} \cos(y_i - \mu)) \\
&= \frac{1}{(2\pi)^n} \cdot \frac{1}{I_o(\kappa)^n} \cdot \exp(\kappa \cdot \sum_{i=1}^{n} \cos(y_i - \mu)) \\
&\propto \frac{1}{I_o(\kappa)^n} \cdot \exp(\kappa \cdot \sum_{i=1}^{n} \cos(y_i - \mu))
\end{aligned}$$

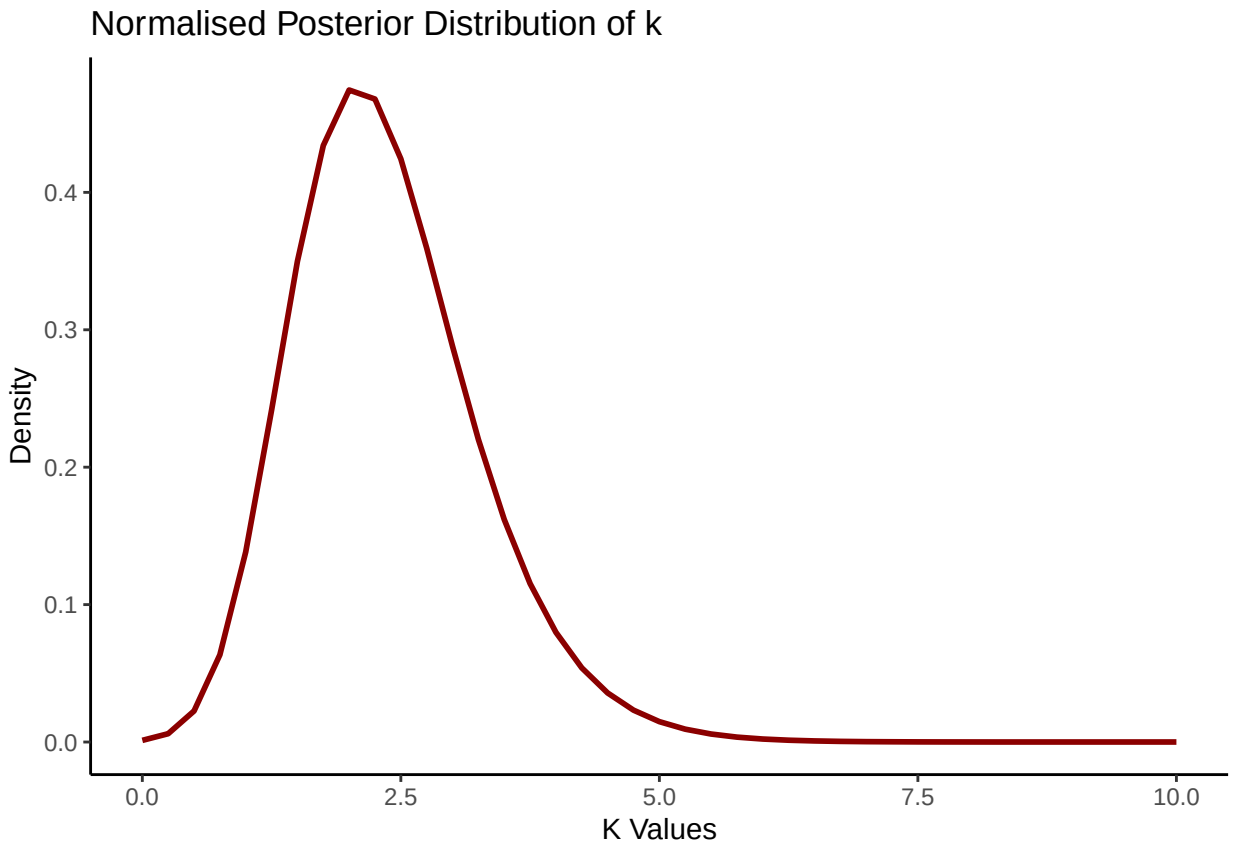
It is given that $\kappa \sim Exp(\lambda = 1)$; thus, it is only needed to calculate the probability density function of κ.

$$\begin{aligned}
p(\kappa) &= \lambda \cdot \exp(-\lambda x) \\
&= \exp(-\kappa)
\end{aligned}$$

Hence the numerator is given by the below expression:

$$p(\kappa|y,\mu) = \frac{1}{I_o(\kappa)^n} \cdot \exp(\kappa \cdot \sum_{i=1}^{n} \cos(y_i - \mu)) \cdot \exp(-\kappa)$$
$$= \frac{1}{I_o(\kappa)^n} \cdot \exp\left(\kappa \cdot \left(\sum_{i=1}^{n} \cos(y_i - \mu) - 1\right)\right)$$

```r
#data
degrees <- c(285, 296, 314, 20, 299, 296, 40, 303, 326, 308)
radians = c(-2.44, 2.14, 2.54, 1.83, 2.02, 2.33, -2.79, 2.23, 2.07, 2.02)

#kappa values
k <- seq(0,10,0.25)

#posterior
posterior <- exp(k*(sum(cos(radians-2.51))-1))/(besselI(x = k, nu=0)^10)

#normalising constant/marginal likelihood
norm_constant <- 0.25*sum(posterior)

#normalised posterior
norm_posterior <- posterior/norm_constant

#data for plotting
df_plot3 <- data.frame("k"=k, "posterior_vals"=norm_posterior)

#plot
ggplot(df_plot3) +
  geom_line(aes(x=k, y=posterior_vals),colour = "red4", size = 1) +
  ggtitle("Normalised Posterior Distribution of k") +
  xlab("K Values") +
  ylab("Density") +
  theme_classic()
```



Task 3b

Question:Find the (approximate) posterior mode of κ from the information in a).

```
#get the index of the max value
max_index <- which.max(norm_posterior)

#get the max value
k_mode <- norm_posterior[max_index]
```

The approximate posterior mode of κ equals 0.04552635

Lab 2

Assignment 1 *Linear and polynomial regression*

The dataset *TempLambohov.txt* contains daily temperatures (in Celcius degrees) at at Lambohov, Linköping over the course of the year 2019. The response variable is temp and the covariate is

$$time = \frac{the\ number\ of\ days\ since\ beginning\ of\ year}{365}$$

A Bayesian analysis of the following quadratic regression model is to be performed:

$$temp = \beta_0 + \beta_1 \cdot\ time + \beta_2 \cdot\ time^2 + \varepsilon \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2)$$

```
#reading data
temperatures <- read.table("TempLambohov.txt",header = TRUE)
```

Task 1a

Question: Use the conjugate prior for the linear regression model. The prior hyperparameters μ_0, Ω_0, u_0 and σ_0^2 shall be set to sensible values.. Start with $\mu_0 = (-10, 100, -100)^T$,$\Omega_0 = 0.02 \cdot I_3$, $u_0 = 3$, $\sigma_0^2 = 2$. Check if this prior agrees with your prior opinions by simulating draws from the joint prior of all parameters and for every draw compute the regression curve. This gives a collection of regression curves; one for each draw from the prior. Does the collection of curves look reasonable? If not, change the prior hyperparameters until the collection of prior regression curves agrees with your prior beliefs about the regression curve. [Hint: the R package *mvtnorm* will be handy. And use your $Inv - \chi^2$ simulator from Lab 1.]

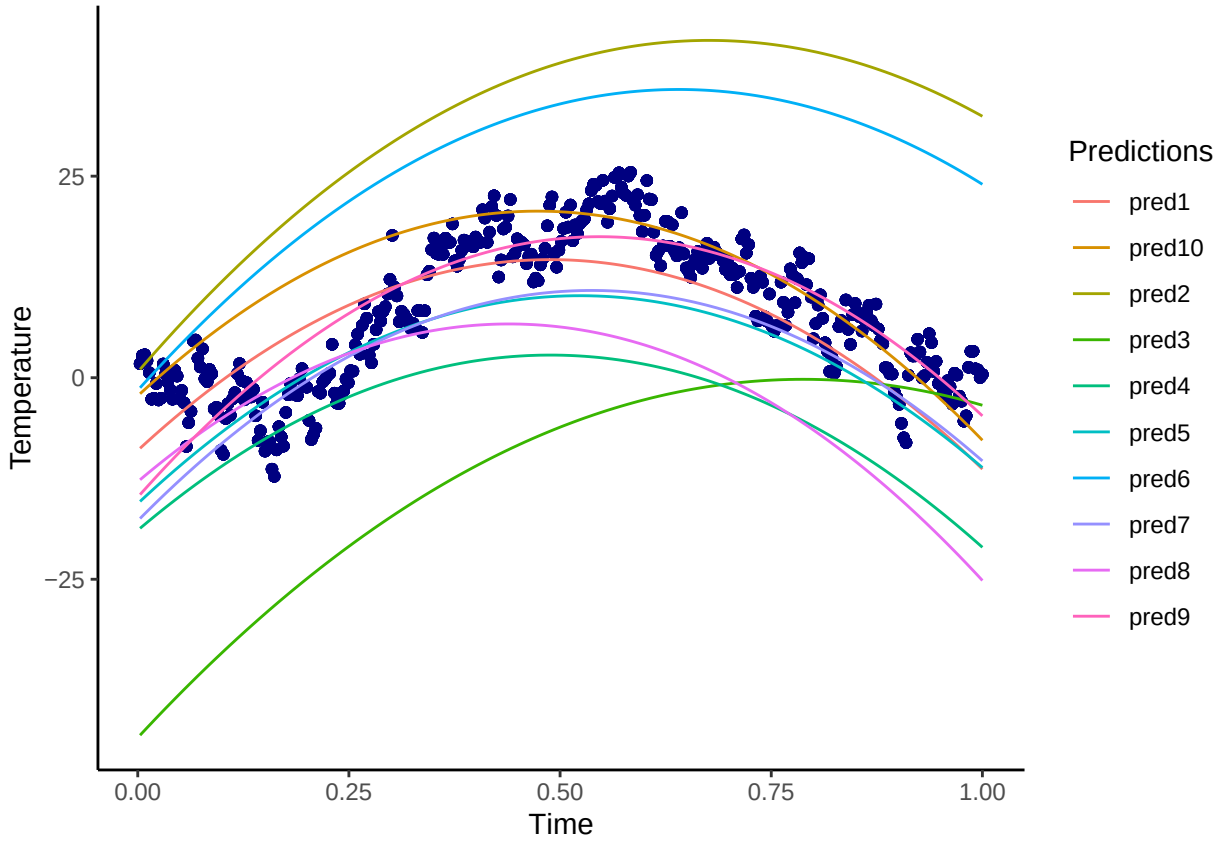
```
#given parameters
m_0 <- c(-10,100,-100)
omega_0 <- 0.02 * diag(3)
n_0 <- 3
sigma2_0 <- 2
```

```
set.seed(123456)

library(mvtnorm)

#number of draws
N <-10

#matrix to save b0, b1, b2
b <- matrix(NA,nrow = N, ncol = length(m_0))

#matrix to save the predicted temperatures
pred_temp <- matrix(NA, nrow = N, ncol = nrow(temperatures))

for(i in 1:N){

    #Inv-x simulator from lab 1
    sigma2 <- n_0 * sigma2_0 / rchisq(1,n_0)

    #generate b0,b1,b2
    b[i,] <- rmvnorm(1,mean = m_0, sigma = sigma2*solve(omega_0))

    #quadratic regression model
```

```
    pred_temp[i,]<-b[i,1] + b[i,2]*temperatures$time + b[i,3]*(temperatures$time^2) + rnorm(1,0,sigma2)
}
```

```
#preparing data for plot
rownames(pred_temp) <- c("pred1","pred2","pred3","pred4","pred5","pred6","pred7","pred8","pred9","pred1
pred_temp <- t(pred_temp)

plot_data <- cbind(temperatures,pred_temp)

library(tidyr)

plot_data <- pivot_longer(plot_data, c(3:12))
```

```
ggplot(plot_data)+
  geom_point(aes(x = time, y = temp), color = "navy") +
  geom_line(aes(x = time, y = value, color = name)) +
  theme(legend.position="right") +
  guides(color=guide_legend("Predictions")) +
  xlab("Time") +
  ylab("Temperature") +
  theme_classic()
```

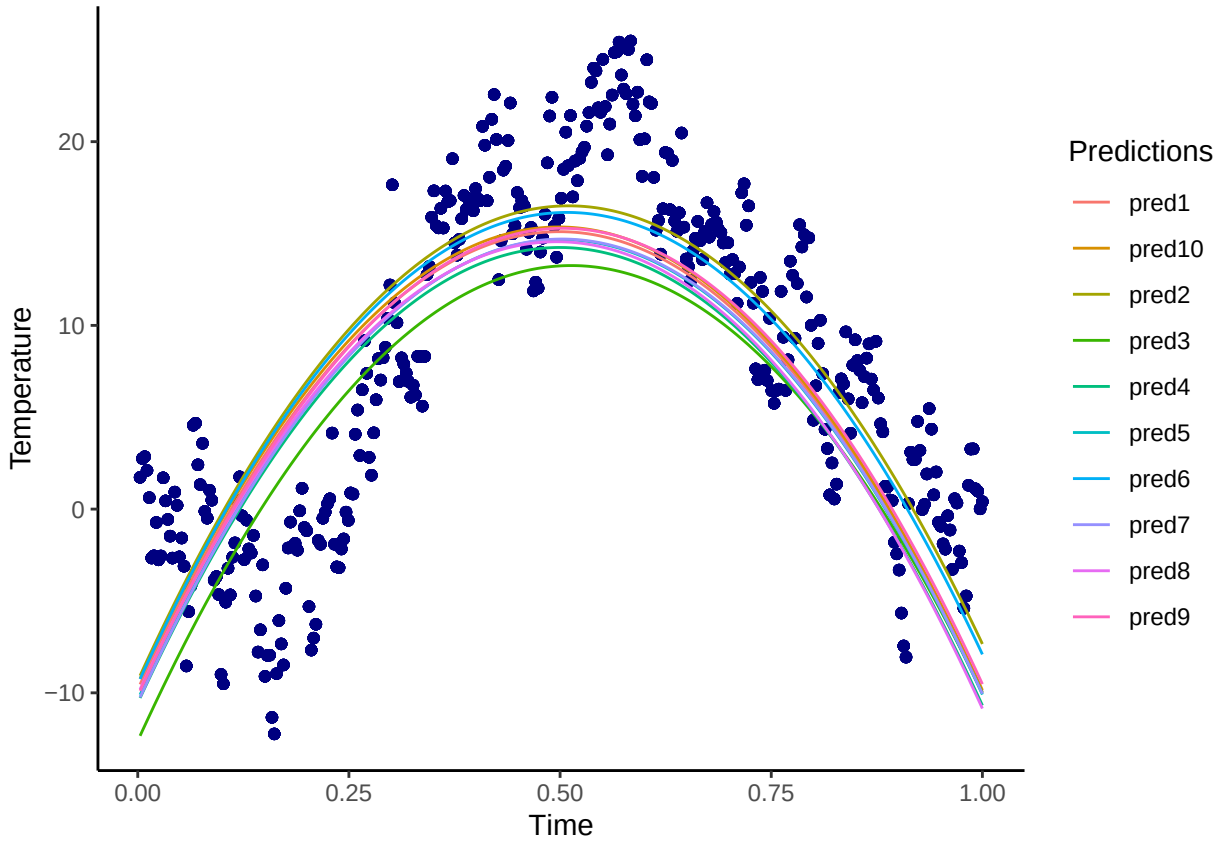


The above graph illustrates the actual temperatures (blue points) and a collection of regression curves, one for each draw from the prior. The graph provides mixed results almost half of the curves are not placed in the region of the actual temperatures, but it is not that bad. Two represent a little higher temperatures but follow the same pattern as the actual ones. Additionally, two curves represent the actual temperatures initially, but as time passes, they illustrate lower temperatures than the actual ones. Finally, only one curve shows poor results; in the very beginning has low temperatures, but in the end, the curve is in the region of

the actual temperatures. Thus, the prior hyper-parameters need small configurations.

```
#modefied parameters
m_0_mod <- c(-10,100,-100)
omega_0_mod <- 0.5 * diag(3) #before 0.2
n_0_mod <- 3
sigma2_0_mod <- 0.2 #before 2

set.seed(123456)

#number of draws
N <-10

#matrix to save b0, b1, b2
b_mod <- matrix(NA,nrow = N, ncol = length(m_0_mod))

#matrix to save the predicted temperatures
pred_temp_mod <- matrix(NA, nrow = N, ncol = nrow(temperatures))

for(i in 1:N){

    #Inv-x simulator from lab 1
    sigma2_mod <- n_0_mod * sigma2_0_mod/ rchisq(1,n_0)

    #generate b0,b1,b2
    b_mod[i,]<-rmvnorm(1,mean = m_0_mod, sigma = sigma2_mod*solve(omega_0_mod))

    #quadratic regression model
    pred_temp_mod[i,]<-b_mod[i,1] + b_mod[i,2]*temperatures$time + b_mod[i,3]*(temperatures$time^2) + r
}
```

```
#preparing data for plot
rownames(pred_temp_mod) <- c("pred1","pred2","pred3","pred4","pred5","pred6","pred7","pred8","pred9","p

pred_temp_mod<- t(pred_temp_mod)

plot_data_mod <- cbind(temperatures,pred_temp_mod)

plot_data_mod <- pivot_longer(plot_data_mod, c(3:12))

ggplot(plot_data_mod)+
  geom_point(aes(x = time, y = temp), color = "navy") +
  geom_line(aes(x = time, y = value, color = name)) +
  theme(legend.position="right") +
  guides(color=guide_legend("Predictions")) +
  xlab("Time") +
  ylab("Temperature") +
  theme_classic()
```



The above graph illustrates the actual temperatures (blue points) and a collection of regression curves, one for each draw from the prior, but with the modified hyper-parameters. Only two parameters were changed, $\Omega_0 = 0.05$ and $\sigma^2 = 0.2$. As a result, all the curves are in the actual temperature region.

Task 1b

Question:Write a function that simulates draws from the joint posterior distribution of $\beta_0, \beta_1, \beta_2$ and σ^2.
i)Plot the marginal posteriors for each parameter as a histogram.
ii)Make a scatter plot of the temperature data and overlay a curve for the posterior median of the regression function $f(\ time) = E[temp|time] = \beta_0 + \beta_1 \cdot\ time + \beta_2 \cdot\ time^2$, i.e. the median of $f(time)$ is computed for every value of time. In addition, overlay curves for the 95% equal tail posterior probability intervals of $f(time)$, i.e. the 2.5 and 97.5 posterior percentiles of $f(time)$ is computed for every value of time. Does the posterior probability intervals contain most of the data points? Should they?

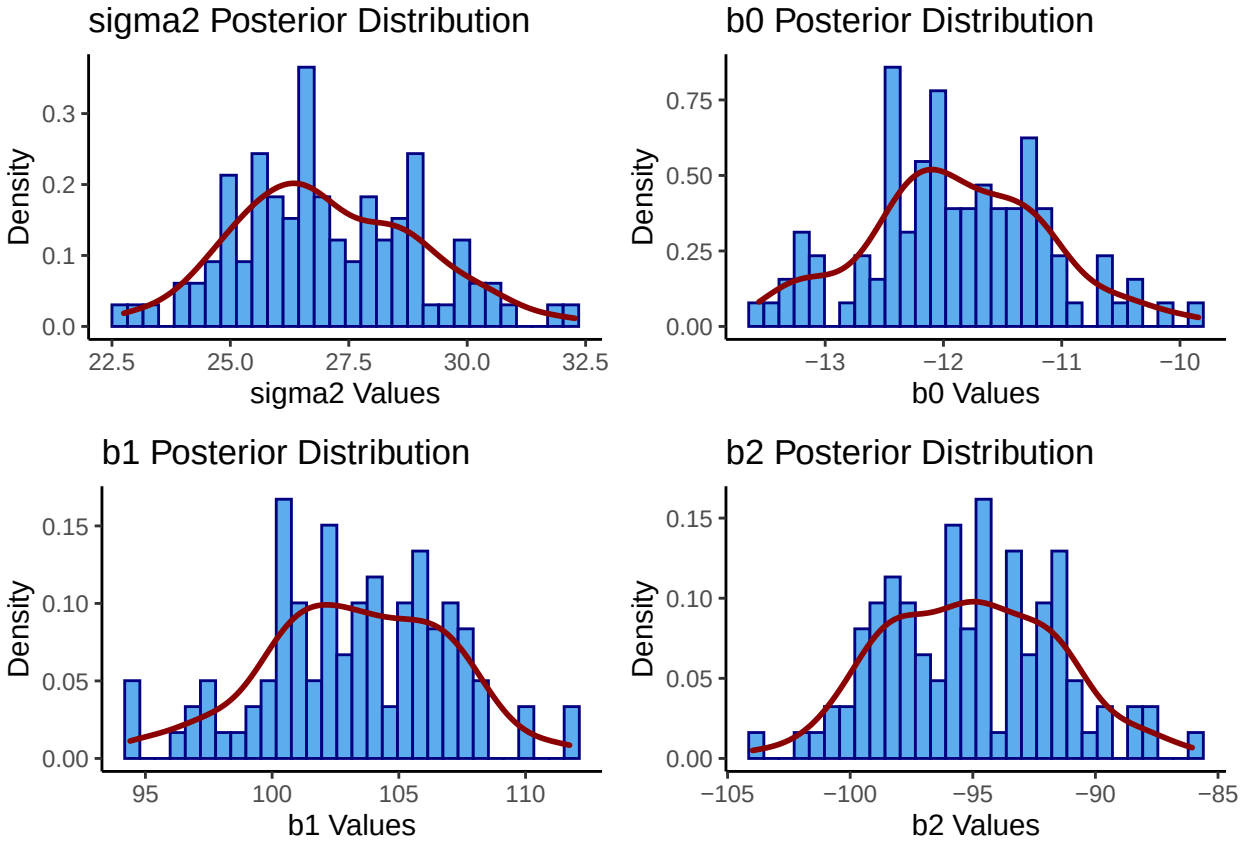
i)

```
set.seed(123456)

#matrix with the times (1st collum with 1 because of the intercept)
X <- cbind(rep(1,nrow(temperatures)), temperatures$time, temperatures$time^2)
#vector with target values
y <- temperatures$temp
#calculate b hat
b_hat<-solve(t(X)%*%X)%*%t(X)%*%y

#number of samples
N <- 100
```

```r
#matrix to store the beta values
b <- matrix(0,N,length(m_0))
#vector to store sigma2 values
sigma2 <- c()

for (i in 1:N){

  #calculate mu_n
  m_n <- solve( t(X) %*% X + omega_0 )%*%(t(X) %*% X %*% b_hat + omega_0 %*% m_0 )

  #calculate omega_n
  omega_n <- t(X) %*% X + omega_0

  #calculate n_n
  n_n <- n_0 + nrow(temperatures)

  #calculate sigma_n
  sigma2_n <- (n_0 * sigma2_0 + (t(y) %*% y + t(m_0) %*% omega_0 %*% m_0 - t(m_n) %*% omega_n %*% m_n))

  #Inv-x simulator from lab 1
  sigma2[i] <- n_n * sigma2_n / rchisq(1,n_n)

  #generate b0,b1,b2
  b[i,] <- rmvnorm(1,mean=m_n,sigma=sigma2[i]*solve(omega_n))

}
```

```r
library(gridExtra)

#df of sigma2 for plot
plot_df_sigma2 <- data.frame("sigma2" = sigma2)

#histogram of sigma2
sigma2_hist <-ggplot(plot_df_sigma2, aes(x=sigma2)) +
  geom_histogram(bins = 30, color = "navy", fill = "steelblue2", aes(y=..density..)) +
  geom_density(colour = "red4", size = 1) +
  ggtitle("sigma2 Posterior Distribution") +
  xlab("sigma2 Values") +
  ylab("Density") +
  theme_classic()

#df of betas for plot
plot_df_b <- data.frame("b0" = b[,1], "b1" = b[,2], "b2" = b[,3])

#histogram of b0
b0_hist <- ggplot(plot_df_b, aes(x=b0)) +
  geom_histogram(bins = 30, color = "navy", fill = "steelblue2", aes(y=..density..)) +
  geom_density(colour = "red4", size = 1) +
  ggtitle("b0 Posterior Distribution") +
  xlab("b0 Values") +
  ylab("Density") +
  theme_classic()
```

```r
#histogram of b1
b1_hist <-ggplot(plot_df_b, aes(x=b1)) +
  geom_histogram(bins = 30, color = "navy", fill = "steelblue2", aes(y=..density..)) +
  geom_density(colour = "red4", size = 1) +
  ggtitle("b1 Posterior Distribution") +
  xlab("b1 Values") +
  ylab("Density") +
  theme_classic()

#histogram of b2
b2_hist <-ggplot(plot_df_b, aes(x=b2)) +
  geom_histogram(bins = 30, color = "navy", fill = "steelblue2", aes(y=..density..)) +
  geom_density(colour = "red4", size = 1) +
  ggtitle("b2 Posterior Distribution") +
  xlab("b2 Values") +
  ylab("Density") +
  theme_classic()

grid.arrange(sigma2_hist,b0_hist, b1_hist, b2_hist, nrow = 2)
```

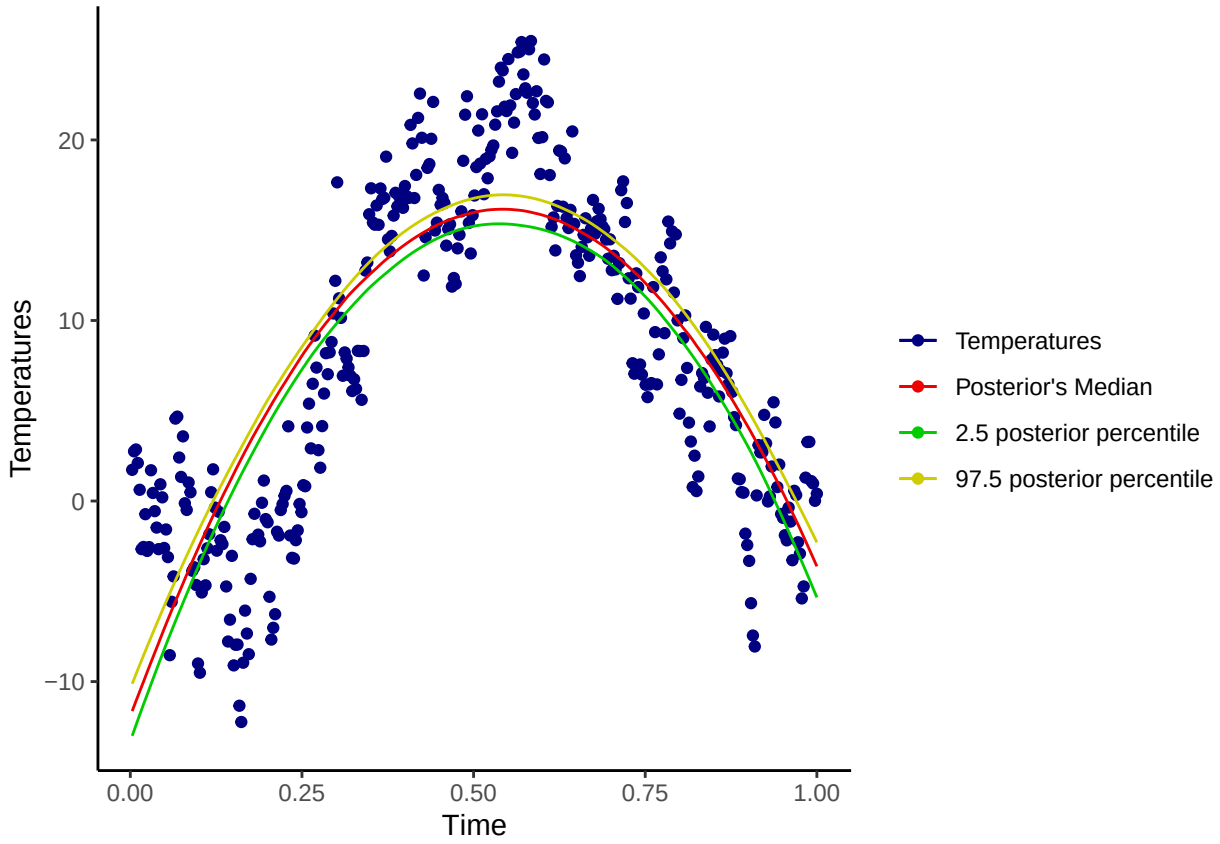


ii)

```r
#matrix to store the predicted values
pred_temp2 <- matrix(NA,nrow(b),nrow(temperatures))

#calculate the predicted values
```

```r
for (i in 1:nrow(b)){
  pred_temp2[i,]<-b[i,1] + b[i,2]*temperatures$time + b[i,3]*(temperatures$time^2)
}

#calculate posterior's median for every value of time
post_median <- c()
for (i in 1:ncol(pred_temp2)){
  post_median[i] <- median(pred_temp2[,i])
}

#calculate the 2.5 and 97.5 posterior percentiles of f (time)
intervals <- matrix(NA, nrow = 2, ncol = ncol(pred_temp2))
for(i in 1:ncol(pred_temp2)){
  intervals[,i] <- quantile(pred_temp2[,i], probs = c(0.025,0.975))
}
```

```r
#df for plot
plot_df_1b2 <- data.frame("temperatures" = temperatures$temp,
                          "time" = temperatures$time,
                          "interval1" = intervals[1,],
                          "interval2" = intervals[2,],
                          "medians" = post_median)

#scatterplot of the temperatures & curve of the posterior median & curves for the 95% equal tail poster
ggplot(data = plot_df_1b2) +
  geom_point(aes(x = time, y = temperatures, color="navy")) +
  geom_line(aes(x =time, y = medians, color = "red2")) +
  geom_line(aes(x = time, y = interval1, color = "green3")) +
  geom_line(aes(x = time, y = interval2, color = "yellow3")) +
  theme(legend.position="right") +
  scale_color_identity(guide = "legend",
                       name = "",
                       breaks=c("navy", "red2", "green3", "yellow3"),
                       labels = c("Temperatures",
                                  "Posterior's Median",
                                  "2.5 posterior percentile",
                                  "97.5 posterior percentile")) +
  xlab("Time") +
  ylab("Temperatures") +
  theme_classic()
```



It is evident that 95% equal tail posterior probability intervals do not contain most data points. It should not contain most data points because the interval illustrates the uncertainty around the median value.

Task c

Question:It is of interest to locate the time with the highest expected temperature (i.e. the time where $f(time)$ is maximal). Let's call this value $\tilde{x}$. Use the simulated draws in (b) to simulate from the posterior distribution of $\tilde{x}$. [Hint: the regression curve is a quadratic polynomial. Given each posterior draw of β_0, β_1 and β_2, you can find a simple formula for $\tilde{x}$.]

It is given that the regression function is $f(time) = \beta_0 + \beta_1 \cdot time + \beta_2 \cdot time^2$. In order to locate the time with the highest expected temperature; the time, where f(time) is maximal, is needed to be found. Thus, the derivative of $f(time)$ is needed to be found and set it equal to zero to find the maximal.

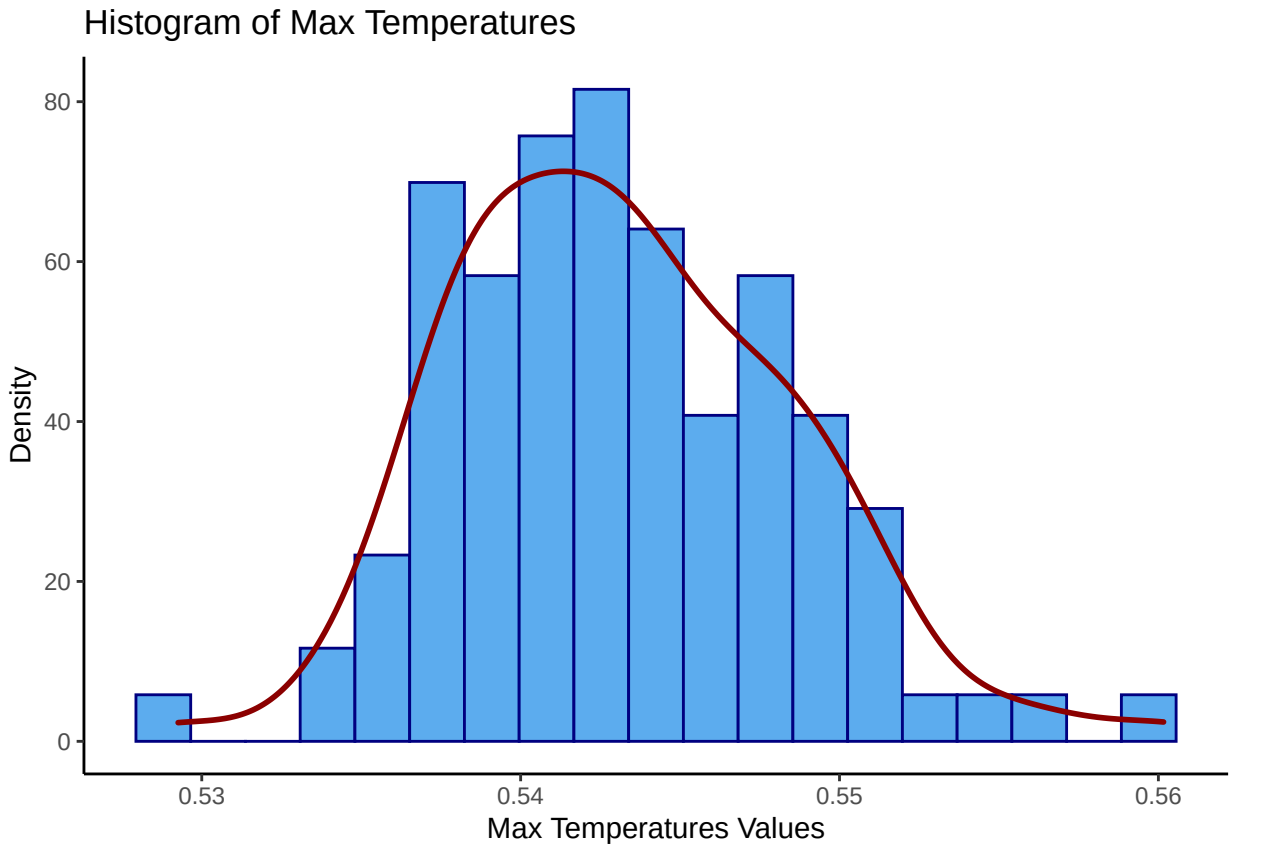
Set $time = x$; thus, $f(x) = \beta_0 + \beta_1 \cdot x + \beta_2 \cdot x^2$

Calculate the derivative, $f'(x) = \beta_1 + 2 \cdot \beta_2 \cdot x$

Find the maximal: $f'(x) = 0 \Leftrightarrow \beta_1 + 2 \cdot \beta_2 \cdot x = 0 \Leftrightarrow x = \frac{-\beta_1}{2 \cdot \beta_2}$

```
#getting the time with the highest expected temperature by using the above expression
max_temp <- c()

for (i in 1:N){
  max_temp[i] <- max(-b[i,2]/(2*b[i,3]))
}

df_max <- data.frame("max_temp" = max_temp)

#histogram of max temps
```

```
ggplot(df_max, aes(x=max_temp)) +
  geom_histogram(bins = 19, color = "navy", fill = "steelblue2", aes(y=..density..)) +
  geom_density(colour = "red4", size = 1) +
  ggtitle("Histogram of Max Temperatures") +
  xlab("Max Temperatures Values") +
  ylab("Density") +
  theme_classic()
```



Task d

Question:Say now that you want to estimate a polynomial regression of order 8, but you suspect that higher order terms may not be needed, and you worry about overfitting the data. Suggest a suitable prior that mitigates this potential problem. You do not need to compute the posterior. Just write down your prior. [Hint: the task is to specify μ_0 and $Omega_0$ in a suitable way.]

Estimating a polynomial regression of order 8 with the suspicion that higher-order terms may not be needed because it might lead to overfitting the data. The main problem that could lead to overfitting is the number of knots. A solution to this problem is to introduce a regularised prior, $\beta_i|\sigma^2 \sim N(\mu_0, \frac{\sigma^2}{\lambda})$, where $Omega_0 = \lambda \cdot I_3$. Where the parameter λ will control the variance of β_i. If λ is larger, β_i would be much closer to μ_0.

Assignment 2 *Posterior approximation for classification with logistic regression*

The dataset WomenAtWork.dat contains n = 168 observations on the following eight variables related to women:

```
#reading data
women <- read.table("WomenAtWork.dat",header=TRUE)
```

```r
#given parameteres
tau <- 5
```

Task a

Question:Consider the logistic regression model:

$$Pr(y = 1 \mid x) = \frac{\exp(x^T\beta)}{1 + \exp(x^T\beta)}$$

where y equals 1 if the woman works and 0 if she does not. x is a 7-dimensional vector containing the seven features (including a 1 to model the intercept). The goal is to approximate the posterior distribution of the parameter vector β with a multivariate normal distribution $\beta|y, x \sim N\big(\tilde{\beta}, J_y^{-1}(\tilde{\beta})\big)$, y where $\tilde{\beta}$ is the posterior mode and $J(\tilde{\beta}) = -\frac{d^2 lnp(\beta|y)}{d\beta \cdot d\beta^T}$ is the negative of the observed Hessian evaluated at the posterior mode. Note that $\frac{d^2 lnp(\beta|y)}{d\beta \cdot d\beta^T}$ is a 7×7 matrix with second derivatives on the diagonal and cross-derivatives $\frac{d^2 lnp(\beta|y)}{d\beta \cdot d\beta^T}$ on the off-diagonal. You can compute this derivative by hand, but we will let the computer do it numerically for you. Calculate both $\tilde{\beta}$ and $J(\tilde{\beta})$ by using the **optim** function in R. [Hint: You may use code snippets from my demo of logistic regression in Lecture 6.] Use the prior $\beta \sim N(0, \tau^2 I)$, where $tau^2 = 5$. Present the numerical values of $\tilde{\beta}$ and $J^{-1}(\tilde{\beta})$ for the WomenAtWork data. Compute an approximate 95% equal tail posterior probability interval for the regression coefficient to the variable NSmallChild. Would you say that this feature is of importance for the probability that a woman works? [Hint: You can verify that your estimation results are reasonable by comparing the posterior means to the maximum likelihood estimates, given by: glmModel <- glm(Work ~ 0 + ., data = WomenAtWork, family = binomial).]

```r
#the below code snippets from a demo of logistic regression in Lecture 6

#target value
y <- women$Work

#prior inputs
# Select which covariates/features to include
X <- as.matrix(women[2:8])
Xnames <- colnames(X)

Npar <- dim(X)[2]

# Setting up the prior
mu <- as.matrix(rep(0,Npar)) # Prior mean vector
Sigma <- tau^2 * diag(Npar) # Prior covariance matrix

# Functions that returns the log posterior for the logistic and probit regression.
# First input argument of this function must be the parameters we optimize on,
# i.e. the regression coefficients beta.

LogPostLogistic <- function(betas,y,X,mu,Sigma){
  linPred <- X%*%betas
  logLik <- sum( linPred*y - log(1 + exp(linPred)) )
  if (abs(logLik) == Inf) logLik = -20000 # Likelihood is not finite, stear the optimizer away from her
  logPrior <- dmvnorm(betas, mu, Sigma, log=TRUE)

  return(logLik + logPrior)
}
```

```
# Select the initial values for beta
initVal <- matrix(0,Npar,1)

# The argument control is a list of options to the optimizer optim, where fnscale=-1 means that we mini
# the negative log posterior. Hence, we maximize the log posterior.
OptimRes <- optim(initVal,LogPostLogistic,gr=NULL,y,X,mu,Sigma,method=c("BFGS"),control=list(fnscale=-1
```

```
names(OptimRes$par) <- Xnames # Naming the coefficient by covariates
approxPostStd <- sqrt(diag(-solve(OptimRes$hessian))) # Computing approximate standard deviations.
names(approxPostStd) <- Xnames # Naming the coefficient by covariates
#print('The posterior mode is:')
#print(OptimRes$par)
#print('The approximate posterior standard deviation is:')
approxPostStd <- sqrt(diag(-solve(OptimRes$hessian)))
#print(approxPostStd)
```

The below table illustrates the $J_y^{-1}(\widetilde{\beta})$.

```
invHessian <- data.frame("invhes" = -solve(OptimRes$hessian))
colnames(invHessian) <- c("Constant","HusbandInc","EducYears","ExpYears","Age","NSmallChild","NBigChild
knitr::kable(invHessian)
```

Constant	HusbandInc	EducYears	ExpYears	Age	NSmallChild	NBigChild
2.5292633	0.0039371	-0.0775073	0.0008893	-0.0341000	-0.2281033	-0.1137008
0.0039371	0.0003915	-0.0008069	-0.0000011	-0.0000516	0.0011500	-0.0000640
-0.0775073	-0.0008069	0.0073415	0.0000610	0.0000496	-0.0080245	0.0019256
0.0008893	-0.0000011	0.0000610	0.0009006	-0.0002492	-0.0009930	0.0006122
-0.0341000	-0.0000516	0.0000496	-0.0002492	0.0008072	0.0060957	0.0012787
-0.2281033	0.0011500	-0.0080245	-0.0009930	0.0060957	0.1918381	0.0094209
-0.1137008	-0.0000640	0.0019256	0.0006122	0.0012787	0.0094209	0.0222163

The below table illustrates the posterior mode values of every feature of the dataset.

```
df_post_mode <- data.frame("post_mode" = OptimRes$par)
colnames(df_post_mode) <- c("Value")
rownames(df_post_mode) <- c("Constant","HusbandInc","EducYears","ExpYears","Age","NSmallChild","NBigChi
knitr::kable(df_post_mode)
```

	Value
Constant	0.9929529
HusbandInc	-0.0343502
EducYears	0.1794176
ExpYears	0.1230628
Age	-0.0727923
NSmallChild	-1.6227735
NBigChild	-0.0838883

From the above table, it could be seen that the posterior mode of the feature *NSmallChild* has the lowest value; thus, it could be assumed that this variable plays a significant role if a woman is employed.

The below table illustrates the approximate posterior standard deviation values of every feature of the dataset.

```
df_approxPostStd <- data.frame("approxPostStd" =sqrt(diag(-solve(OptimRes$hessian))))
colnames(df_approxPostStd) <- c("Value")
rownames(df_approxPostStd) <-c("Constant","HusbandInc","EducYears","ExpYears","Age","NSmallChild","NBig
knitr::kable(df_approxPostStd)
```

	Value
Constant	1.5903658
HusbandInc	0.0197857
EducYears	0.0856826
ExpYears	0.0300097
Age	0.0284107
NSmallChild	0.4379933
NBigChild	0.1490514

The approximate 95% equal tail posterior probability interval for the regression coefficient to the variable *NSmallChild* has a lower and an upper bound less than 0; as a result, it strengthens the assumption mentioned above that the *NSmallChild* feature plays a significant role if a woman is employed.

```
#draw b
b_draws <- rmvnorm(n = 1000, mean = OptimRes$par, sigma = -solve(OptimRes$hessian))

#calculate quantiles
interval <- quantile(b_draws[,6], c(0.025, 0.975))
df_interval <- data.frame("lower_bound" = interval[1], "upper_bound" = interval[2])
colnames(df_interval) <- c("lower bound", "upper bound")
rownames(df_interval) <- c("95% Equal Tail Credible Interval")
knitr::kable(df_interval)
```

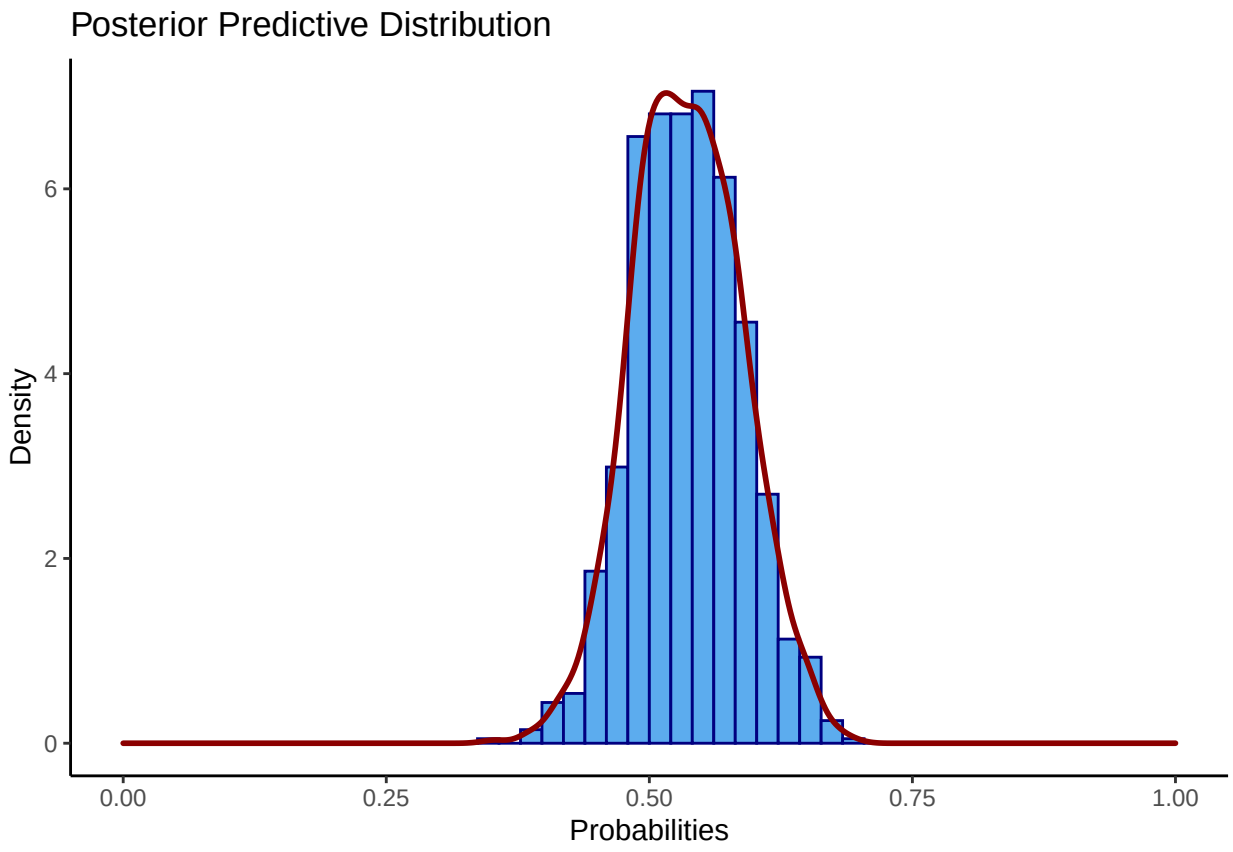
	lower bound	upper bound
95% Equal Tail Credible Interval	-2.445008	-0.8075428

Task b

Question:Use your normal approximation to the posterior from (a). Write a function that simulate draws from the posterior predictive distribution of $Pr(y = 1|x)$, where the values of x corresponds to a 43-year-old woman, with two children (7 and 10 years old), 12 years of education, 8 years of experience, and a husband with an income of 20. Plot the posterior predictive distribution of $Pr(y = 1|x)$ for this woman. [Hints: The R package mvtnorm will be useful. Remember that $Pr(y = 1|x)$ can be calculated for each posterior draw of β.]

```
set.seed(1234567)
women2 <- c(1,20,12,8,43,0,2)

#function that classifies if the women work or not
prediction <- function(N,data,posterior_mode,posterior_covariates){

  #generate betas
  b <- rmvnorm(N, mean= posterior_mode, sigma = posterior_covariates)

  #calculating the the logistic regression model
  res <- exp(data %*% t(b))/(1 + exp(data %*% t(b)))

  return(res)
```

```
}

#predictions
pred <- prediction(1000, women2, OptimRes$par, -solve(OptimRes$hessian))

#df for plot
pred_women <- data.frame("pred" = t(pred))

ggplot(pred_women, aes(x=pred)) +
  geom_histogram(bins = 50, color = "navy", fill = "steelblue2", aes(y=..density..)) +
  geom_density(colour = "red4", size = 1) +
  scale_x_continuous(limits = c(0,1)) +
  ggtitle("Posterior Predictive Distribution") +
  xlab("Probabilities") +
  ylab("Density") +
  theme_classic()
```



From the above plot, it could be assumed that a 43-year-old woman with two children (7 and 10 years old), 12 years of education, 8 years of experience, and a husband with an income of 20.000 SEK; is more likely to be employed, but there is still a considerable number of occasions that she might without work.

Task 2c

Question:Now, consider 11 women which all have the same features as the woman in (b). Rewrite your function and plot the posterior predictive distribution for the number of women, out of these 11, that are working. [Hint: Simulate from the binomial distribution, which is the distribution for a sum of Bernoulli random variables.]

```r
set.seed(1234567)

prediction2 <- function(N,data,posterior_mode,posterior_covariates, nwomen){

  #generate betas
  b <- rmvnorm(N, mean= posterior_mode, sigma = posterior_covariates)

  #calculating the the logistic regression model
  pred <- exp(data %*% t(b))/(1 + exp(data %*% t(b)))

  #vector to store results
  res <- c()

  for (i in 1:nwomen){
    #binomial distribution
    res[i] <- sum(rbinom(n = 7,size = 11,prob = pred))
  }

  return(res)
}

#predictions
pred2 <- prediction2(1000, women2, OptimRes$par, -solve(OptimRes$hessian), 11)

hist(pred2, col = "steelblue2", main = "Posterior Predictive Distribution For The Number Of Women", xlab
```