Bayesian Learning

Report-Lab 2

Jonathan Dorairaj,Yi Hung Chen

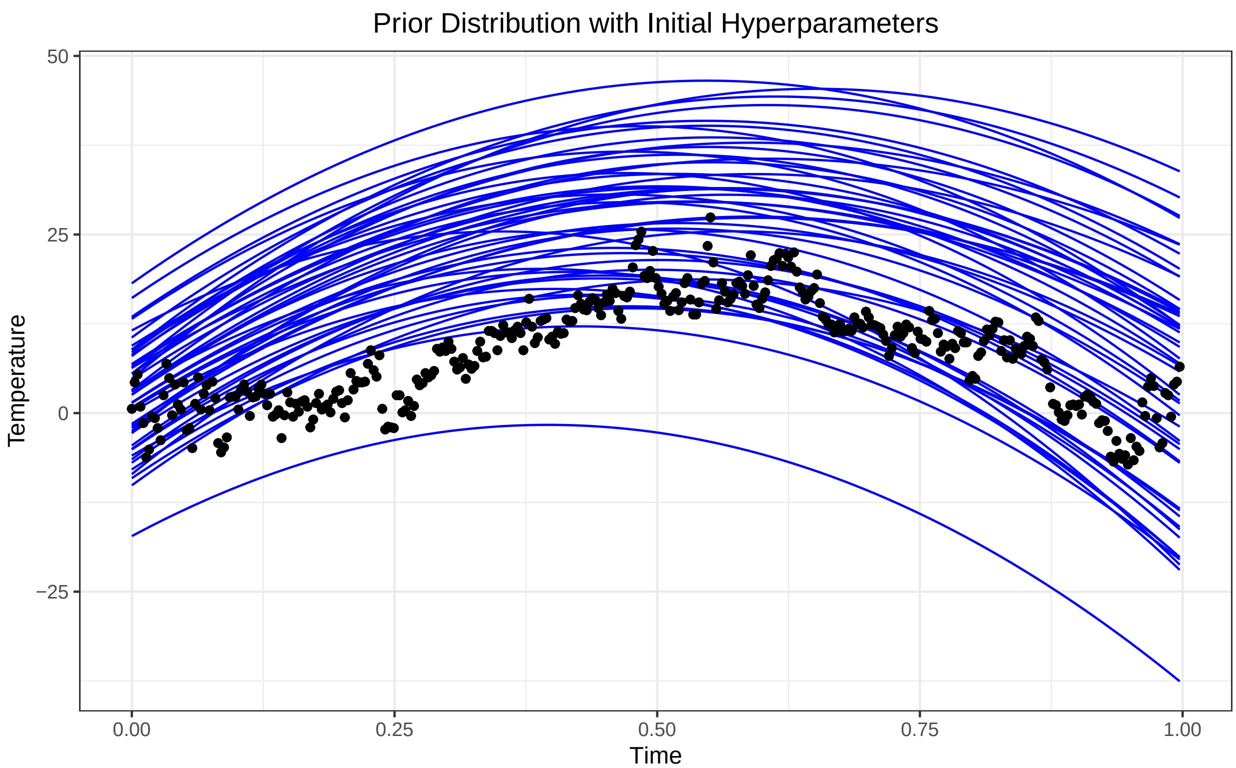
2023-05-01

Linear and Polynomial regression

1a)

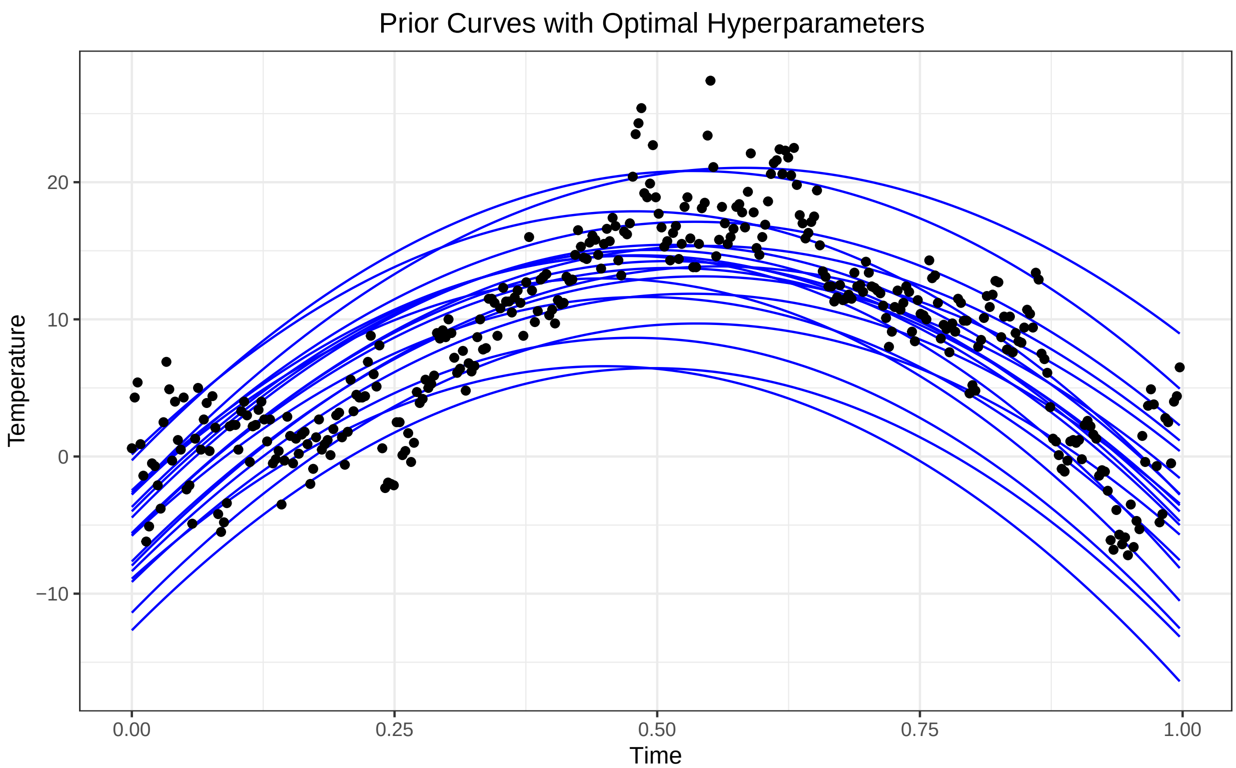
```
temperature_data <- read_xlsx('Linkoping2022.xlsx')
temperature_data$datetime <- as.Date(temperature_data$datetime)

create_time <- function(x)
{
  res <- as.numeric(x - as.Date('2022-01-01')) / 365

  res
}

temperature_data$time <- create_time(temperature_data$datetime)
temperature_data$time2 <- temperature_data$time^2

time_mat <- as.matrix(cbind("bias" = 1,"time" = temperature_data$time,
                            "time_2" = (temperature_data$time)^2))
# lecture 5
# get sigma^2 from inv chi-sq simlator using v0 & sigma_0^2

#Question 1a

n <- dim(temperature_data)[1]-1
InvChiSq <- function(sample_size,n,tau2)
{
  X <- rchisq(sample_size,df = n)
  xs <- (n*tau2)/X
  return(xs)
}

# prior variance
prior_var <- function(v0,s2)
{
  pvar <- InvChiSq(sample_size = 1,n = v0,tau2 = s2)
  return(pvar)
}

# now get beta given variance from mvt norm distribution
```

```r
# joint prior
prior_beta <- function(mu0,sigma_2,omega0_inv){
  betaprior <- rmvnorm(1,mean = mu0, sigma = sigma0_2*omega0_inv)
  return(betaprior)
}

# initialize starting hyperparameters
sigma0_2 <- 1
v0 <- 1
mu0 <- matrix(c(0,100,-100),nrow = 3,ncol = 1)
omega0_inv <- solve(diag(x = 0.01,nrow = 3,ncol = 3))

prior_draws <- c()

plot_df <- list()
#set.seed(123)
p <- ggplot()
  for(i in 1:50){
  sigma_2 <- prior_var(v0 = 1,s2 = 1)
  val <- prior_beta(mu0 = mu0,sigma_2 = sigma_2,omega0_inv = omega0_inv)
  prior_draws <- c(prior_draws,val)
  y <- time_mat %*% t(val)

  plot_df[[i]] <- data.frame(cbind("x" = temperature_data$time,"y" = y))
  p <- p + geom_line(data = plot_df[[i]], aes(x = x, y = V2), col = "blue")+
    theme_bw() +
  labs(x = "Time",y= "Temperature",title = "Prior Distribution with Initial Hyperparameters")+
  theme(plot.title = element_text(hjust = 0.5))
  }
p + geom_point(data = temperature_data,aes(x = time,y = temp))
```



We observe the prior regression curves with initially specified hyperparameters seem to capture most of data, but some curves don't intersect with a single data point, therefore we tweak the hyperparameters.Rerunning with optimal hyperparameters :

```r
# rerun with changed hyperparameters
v0 <- 4
sigma0_2 <- 12
mu0 <- matrix(c(-6,75,-75),nrow = 3,ncol = 1)
omega0_inv <- solve(diag(x = 0.6,nrow = 3,ncol = 3))

# different results for every run,since seed is different
prior_draws <- c()
sigma_2 <- prior_var(v0 = v0,s2 = sigma0_2)
plot_df <- list()
p <- ggplot()
for(i in 1:20){
  val <- prior_beta(mu0 = mu0,sigma_2 = sigma_2,omega0_inv = omega0_inv)
  prior_draws <- c(prior_draws,val)
  y <- time_mat %*% t(val)
  plot_df[[i]] <- data.frame(cbind("x" = temperature_data$time,"y" = y))
  p <- p + geom_line(data = plot_df[[i]], aes(x = x, y = V2), col = "blue")+
    theme_bw()
}
p + geom_point(data = temperature_data,aes(x = time,y = temp)) +
  labs(x = "Time",y= "Temperature",title = "Prior Curves with Optimal Hyperparameters ")+
  theme(plot.title = element_text(hjust = 0.5))
```


Prior Curves with Optimal Hyperparameters
Temperature
20
10
0
−10
0.00
0.25
0.50
0.75
1.00
Time

1b)

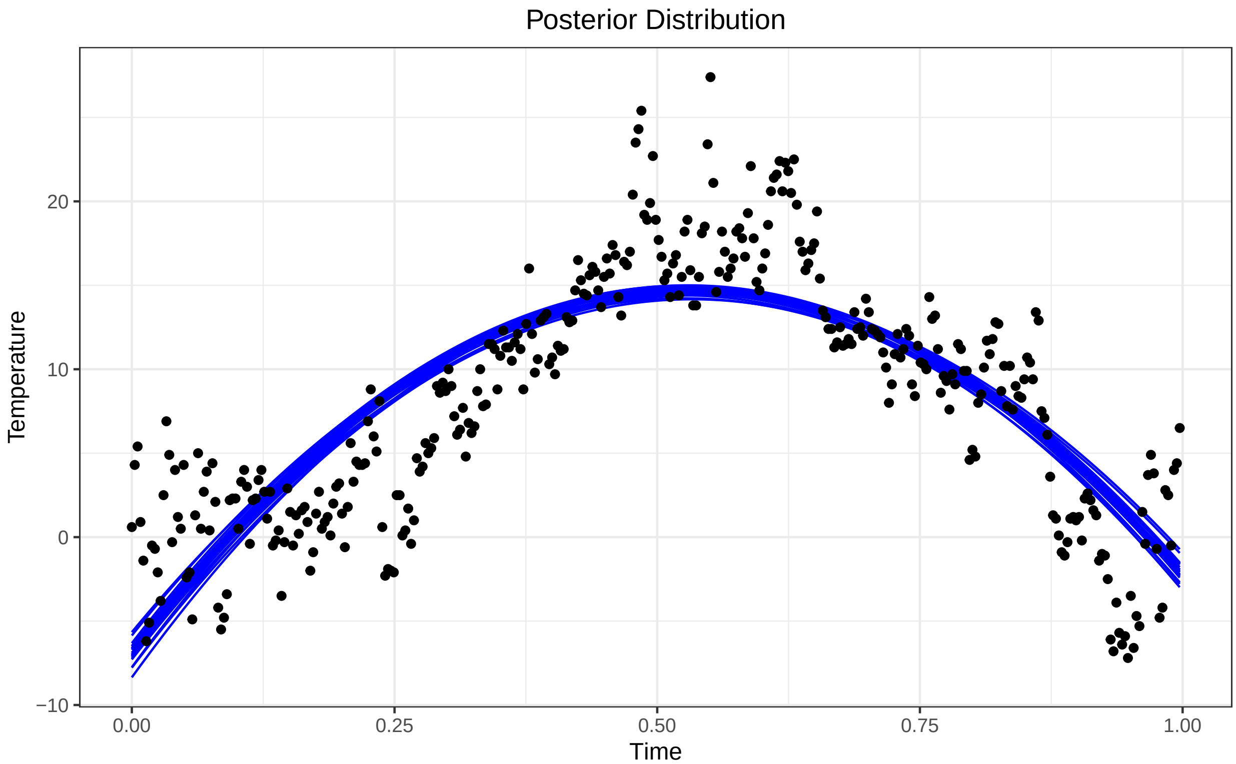
Using the formulas for the conjugate posterior from the lecture slides, we calculate the posterior.

$$\beta \mid \sigma^2, \mathrm{y} \sim N\left[\mu_n, \sigma^2 \Omega_n^{-1}\right]$$
$$\sigma^2 \mid \mathrm{y} \sim \mathrm{Inv}-\chi^2\left(\nu_n, \sigma_n^2\right)$$
$$\mu_n = \left(\mathrm{X}'\mathrm{X} + \Omega_0\right)^{-1}\left(\mathrm{X}'\mathrm{X}\hat{\beta} + \Omega_0\mu_0\right)$$
$$\Omega_n = \mathrm{X}'\mathrm{X} + \Omega_0$$
$$v_n = \nu_0 + n$$
$$v_n\sigma_n^2 = v_0\sigma_0^2 + \left(\mathrm{y}'\mathrm{y} + \mu_0'\Omega_0\mu_0 - \mu_n'\Omega_n\mu_n\right)$$
$$\hat{\beta} = \left(X'X\right)^{-1}X'y$$

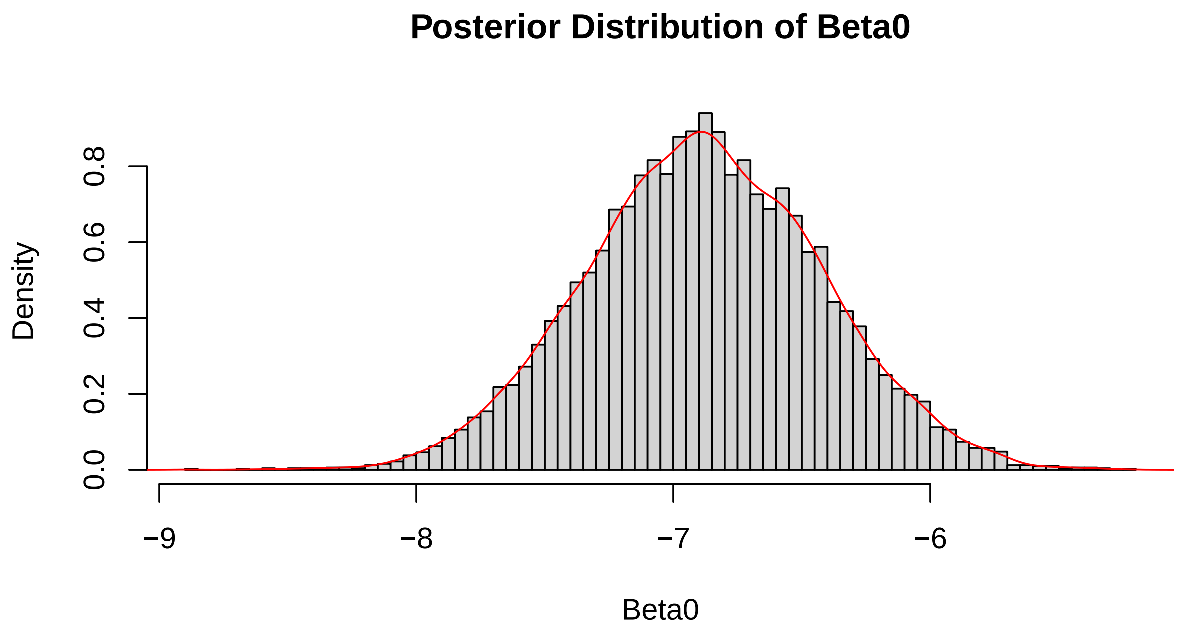
where X is the matrix contains the columns (1,time,$time^2$).

```
#formulas from lecture slide 5
# compute posterior
v_n <- v0 + n
omega_n <- (t(time_mat)%*%time_mat) + solve(omega0_inv)

beta_hat <- solve(t(time_mat)%*%time_mat) %*% t(time_mat) %*% temperature_data$temp

mu_n <- (solve((t(time_mat)%*%time_mat) + solve(omega0_inv))) %*%
  (((t(time_mat)%*%time_mat)%*%beta_hat) + solve(omega0_inv)%*%mu0)

v_n_sigma2_n <- v0*sigma0_2 + ((t(temperature_data$temp)%*%temperature_data$temp) +
                                  (t(mu0)%*%solve(omega0_inv)%*%mu0 )- (t(mu_n)%*%omega_n%*%mu_n))
sigma_2 <- v_n_sigma2_n/v_n

# posterior variance
posterior_var <- function(v_n,sigma_2)
{
  pvar <- InvChiSq(sample_size = 1,n = v_n,tau2 = sigma_2)
  return(pvar)
}

# posterior betas
posterior_beta <- function(mu_n,sigma_2,omega_n){
  betaposterior<- rmvnorm(1,mean = mu_n, sigma = as.vector(sigma_2)*solve(omega_n))
  return(betaposterior)
}

## plotting posterior
posterior_draws <- c()
plot_df <- list()
p <- ggplot()
for(i in 1:20){
  sigma_2 <- posterior_var(v_n = v_n,sigma_2 = sigma_2 )
  val <- posterior_beta(mu_n = mu_n,sigma_2 = sigma_2,omega_n = omega_n)
  posterior_draws <- c(posterior_draws,val)
  y <- time_mat %*% t(val)
  plot_df[[i]] <- data.frame(cbind("x" = temperature_data$time,"y" = y))
  p <- p + geom_line(data = plot_df[[i]], aes(x = x, y = V2), col = "blue") +
```

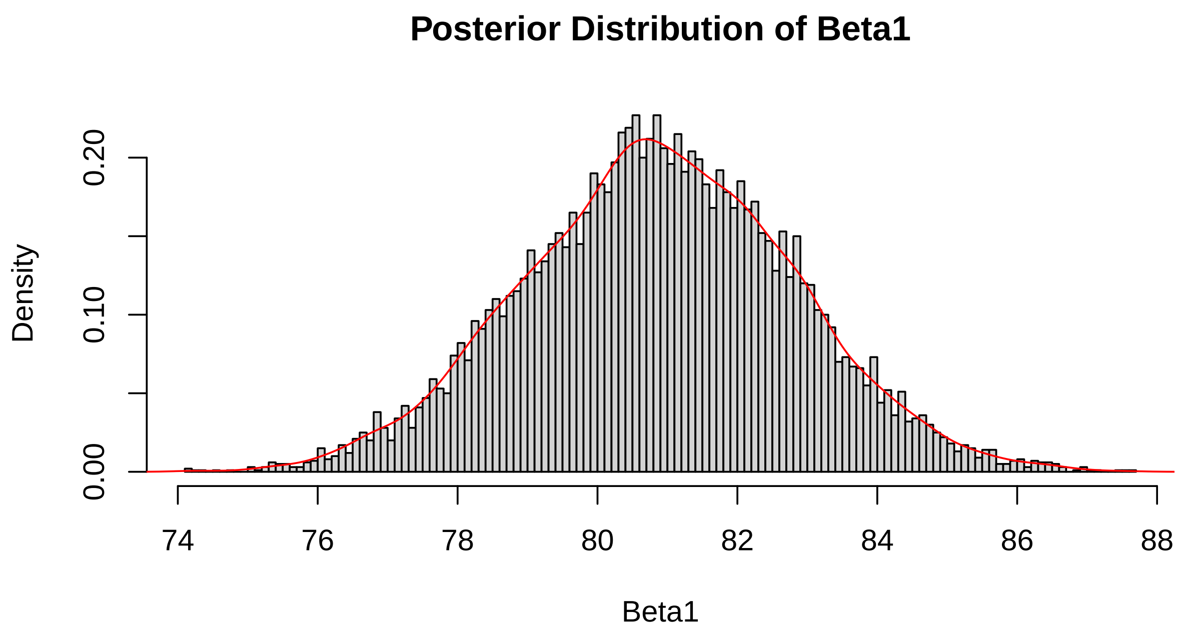
```r
    theme_bw()

}
p + geom_point(data = temperature_data,aes(x = time,y = temp)) +
  labs(x = "Time",y= "Temperature",title = "Posterior Distribution") +
  theme(plot.title = element_text(hjust = 0.5))
```



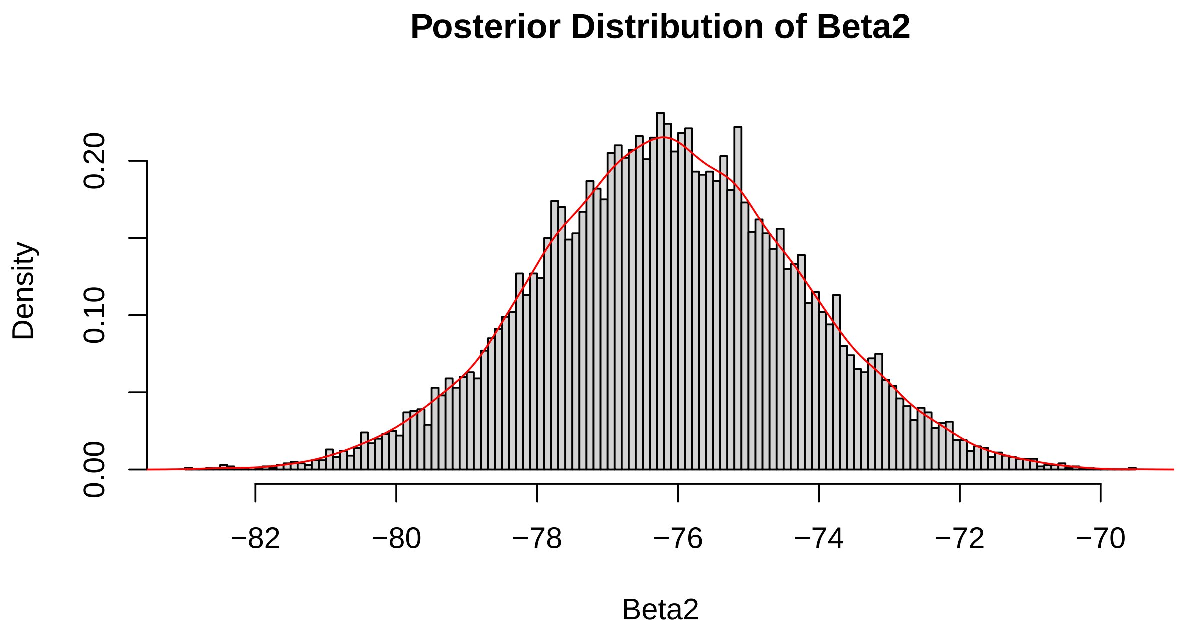

```r
## sample from posterior, then plot marginal densities
posterior_mat <- matrix(nrow = 10000,ncol = 3)
posterior_variance <- c()
for(i in 1:10000){
  posterior_mat[i,] <- posterior_beta(mu_n = mu_n,sigma_2 = sigma_2,omega_n = omega_n)
  posterior_variance[i] <- posterior_var(v_n = v_n,sigma_2 = sigma_2)

}
colnames(posterior_mat) <- c('b0','b1','b2')

# posterior plots for b0,b1 and b2
hist(posterior_mat[,1],breaks = 100,main = 'Posterior Distribution of Beta0',
     probability = TRUE,xlab = 'Beta0')
lines(density(posterior_mat[,1]),col = 'red')
```



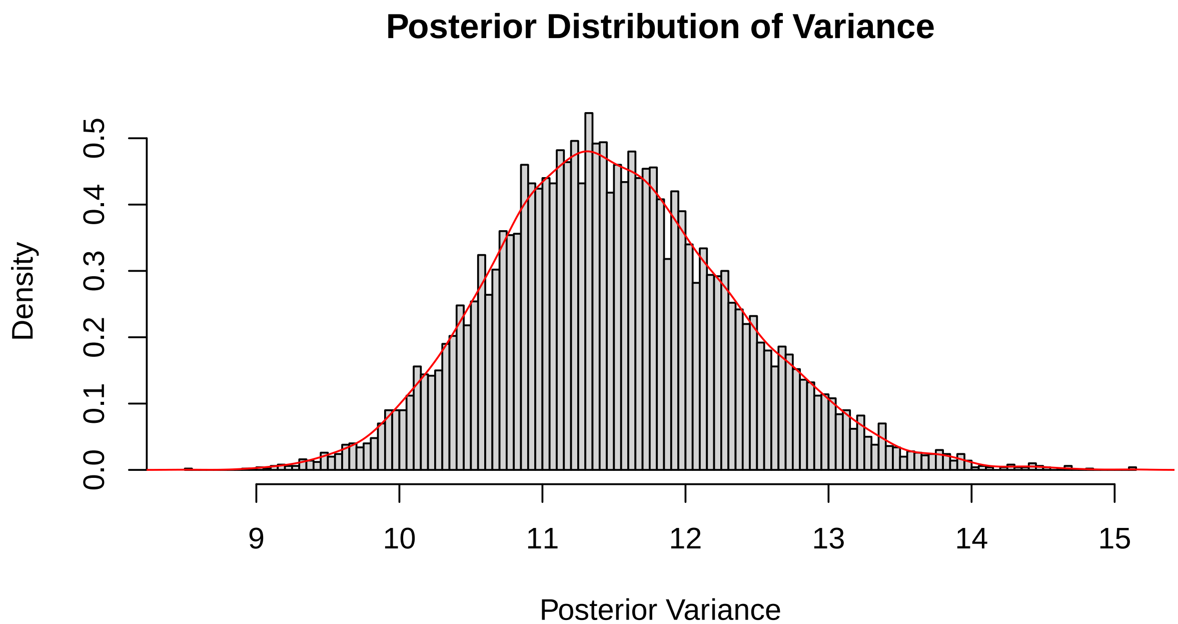

```
hist(posterior_mat[,2],breaks = 100,main = 'Posterior Distribution of Beta1',
     probability = T,xlab = 'Beta1')
lines(density(posterior_mat[,2]),col = 'red')
```



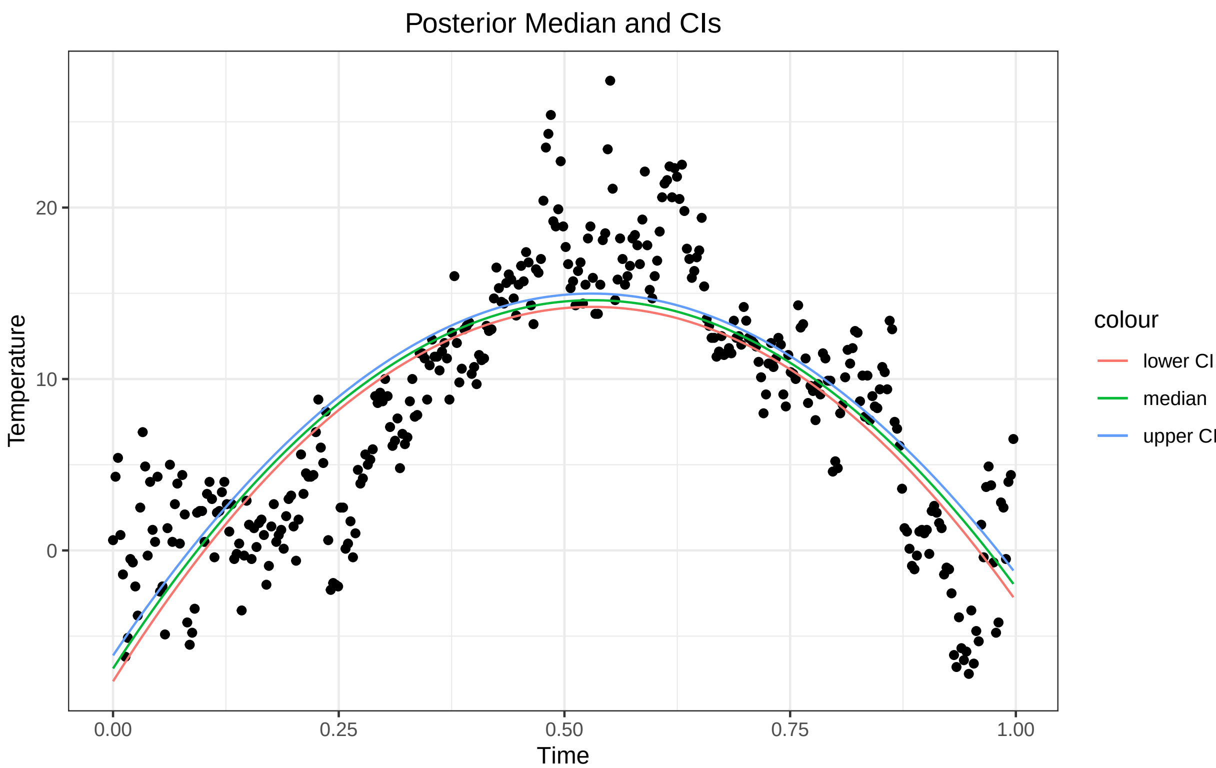
```r
hist(posterior_mat[,3],breaks = 100,main = 'Posterior Distribution of Beta2',
     probability = T,xlab = 'Beta2')
lines(density(posterior_mat[,3]),col = 'red')
```




```r
# posterior plot for sigma2
hist(posterior_variance,breaks = 100,main = 'Posterior Distribution of Variance',
     probability = T,xlab = 'Posterior Variance')
lines(density(posterior_variance),col = 'red')
```




```
f_time  <- time_mat%*%t(posterior_mat)

stat_df <- data.frame('lower' = 0,'median' = 0,'upper' = 0)

for(i in seq(0,dim(f_time)[1])){
  #calculate lower,median and upper CI values
  lower <- quantile(f_time[i,],probs = 0.05)
  med <- median(f_time[i,])
  upper <- quantile(f_time[i,],probs = 0.95)
  stat_df[i,] <- c(lower,med,upper)
}

#combine with temperature data for plotting
combined_df <- data.frame(cbind(temperature_data,stat_df))


p <- ggplot() + geom_point(data = combined_df,aes(x = time,y = temp))+
  geom_line(data =  combined_df,aes(x = time, y = lower, col = "lower CI "))+
  geom_line(data =  combined_df,aes(x = time, y = median, col = "median"))+
  geom_line(data =  combined_df,aes(x = time, y = upper, col = "upper CI"))+
  theme_bw() + labs(x = "Time",y= "Temperature",title = "Posterior Median and CIs") +
  theme(plot.title = element_text(hjust = 0.5))

p
```



The posterior probability interval curve also do not contain the majority of the data points. This is because there is noise in the data that is not modeled in the posterior. If there is an addition of a noise term ϵ, we can model the posterior probability intervals to capture more of the data points.

1c)

The regression curve is given by the forumula

$$f(time) = \beta_0 + \beta_1 x + \beta_2 x^2$$

where x is time.

Differentiating the above formula with respect to x gives us

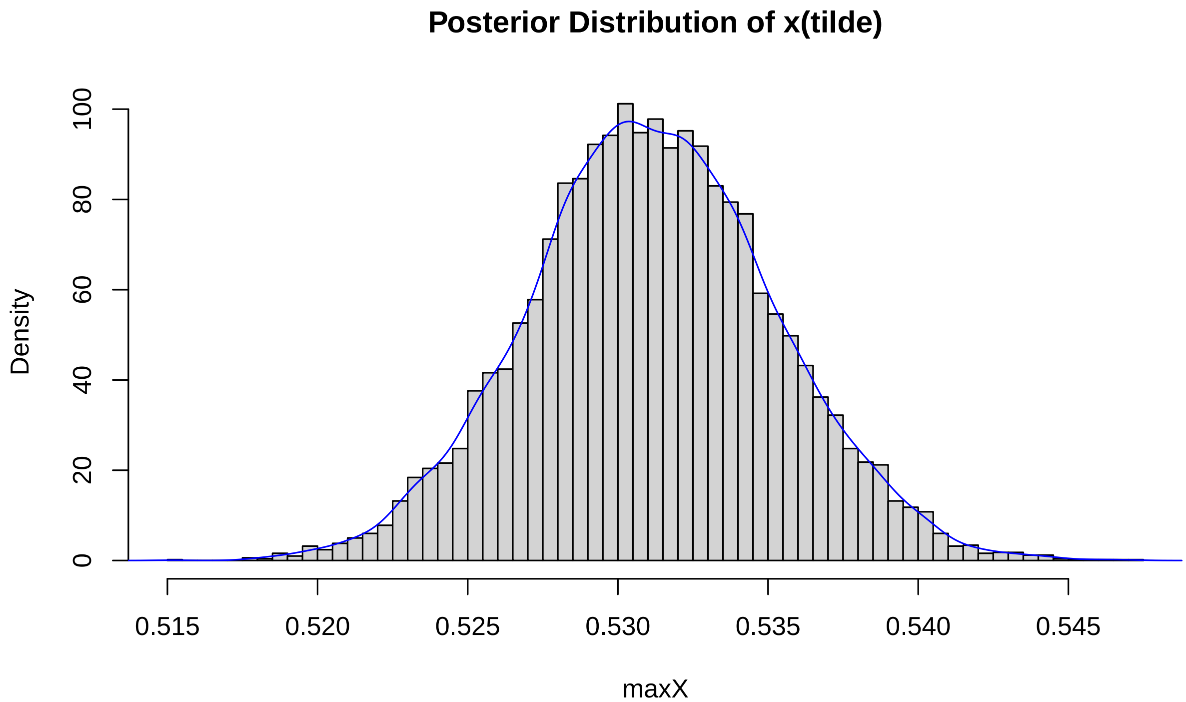
$$\beta_1 + 2x\beta_2$$

Setting this result to 0 and solving for x gives us

$$\widetilde{x} = \frac{-\beta_1}{2\beta_2}$$

Using the formula in R gives us :

```
maxX <- -(posterior_mat[,2])/(2 * posterior_mat[,3])
hist(maxX,breaks = 100,main = 'Posterior Distribution of x(tilde)',probability = T)
lines(density(maxX),col = 'blue')
```



1d)

To avoid overfitting the data, we use a regularization prior,

$$\beta_k|\sigma^2 \sim N\left(\mu_0, \sigma^2\Omega_0^{-1}\right)$$

where $\omega_0 = \lambda I$ and λ is the shrinkage parameter.

Question 2a

1.Present the numerical values of $\bar{\beta}$ and $J_y^{-1}(\bar{\beta})$ for the WomenAtWork data.
2.Compute an approximate 95% equal tail posterior probability interval for the regression coefficient to the variable NSmallChild.

```r
library(mvtnorm)
library(ggplot2)

WomenAtWork <- read.delim("WomenAtWork.dat", header = TRUE, sep="")
glmModel<- glm(Work ~ 0 + ., data = WomenAtWork, family = binomial)
#summary(glmModel)

women_df <- WomenAtWork[,2:ncol(WomenAtWork)]
lable <- WomenAtWork[, 1]
Npar <- dim(women_df)[2]

# Initialize prior
mu <- as.matrix(rep(0, Npar))
tau <- 2
Sigma <- tau^2 * diag(Npar) #tau^2I

LogPostLogistic <- function(betas,y,X,mu,Sigma){
  X = as.matrix(X)
  linPred <- X%*%betas
  logLik <- sum( linPred*y - log(1 + exp(linPred)) )
  logPrior <- dmvnorm(betas, mu, Sigma, log=TRUE)
  return(logLik + logPrior)
}

# Initialize
initVal <- matrix(0, Npar, 1)

# Opt
OptimRes <- optim (initVal, LogPostLogistic, gr = NULL, y = lable, X = women_df,
                   mu = mu, Sigma = Sigma, method=c("BFGS"),
                   control=list(fnscale=-1),hessian=TRUE)

beta_mode <- OptimRes$par

# Printing results
posterior_df <- as.data.frame(OptimRes$par)
colnames(posterior_df) <- "posterior mode "
posterior_df$glmest <- glmModel$coefficients
row.names(posterior_df) <- colnames(women_df)

approxPostStd <- sqrt(diag(solve(-OptimRes$hessian))) # Computing approximate standard deviations.
approxPostStd_df <- as.data.frame(approxPostStd)
colnames(approxPostStd_df) <- "approxPostStd"
approxPostStd_df$glmstd <- summary(glmModel)$coefficients[, 2]
row.names(approxPostStd_df) <- colnames(women_df)

kbl(posterior_df)
```

	posterior mode	glmest
Constant	-0.0403694	0.0226293
HusbandInc	-0.0373069	-0.0379631
EducYears	0.1786895	0.1844741
ExpYears	0.1207364	0.1213176
Age	-0.0461900	-0.0485817
NSmallChild	-1.4724893	-1.5648514
NBigChild	-0.0201446	-0.0252606

```
kbl(approxPostStd_df)
```

	approxPostStd	glmstd
Constant	1.3819849	1.9308324
HusbandInc	0.0219847	0.0222923
EducYears	0.0892096	0.1000661
ExpYears	0.0333598	0.0335345
Age	0.0274732	0.0332250
NSmallChild	0.4774676	0.5107825
NBigChild	0.1640196	0.1771613

Compare the result using optim function in R and the result from glm summary, we can see that both the posterior mode and posterior standard divination are similar.

```
upper<-beta_mode+1.96*approxPostStd
lower<-beta_mode-1.96*approxPostStd
cat("The intervals for the variable NSmallChild are upper=",upper[6],"lower=",lower[6])
```

```
## The intervals for the variable NSmallChild are upper= -0.5366527 lower= -2.408326
```

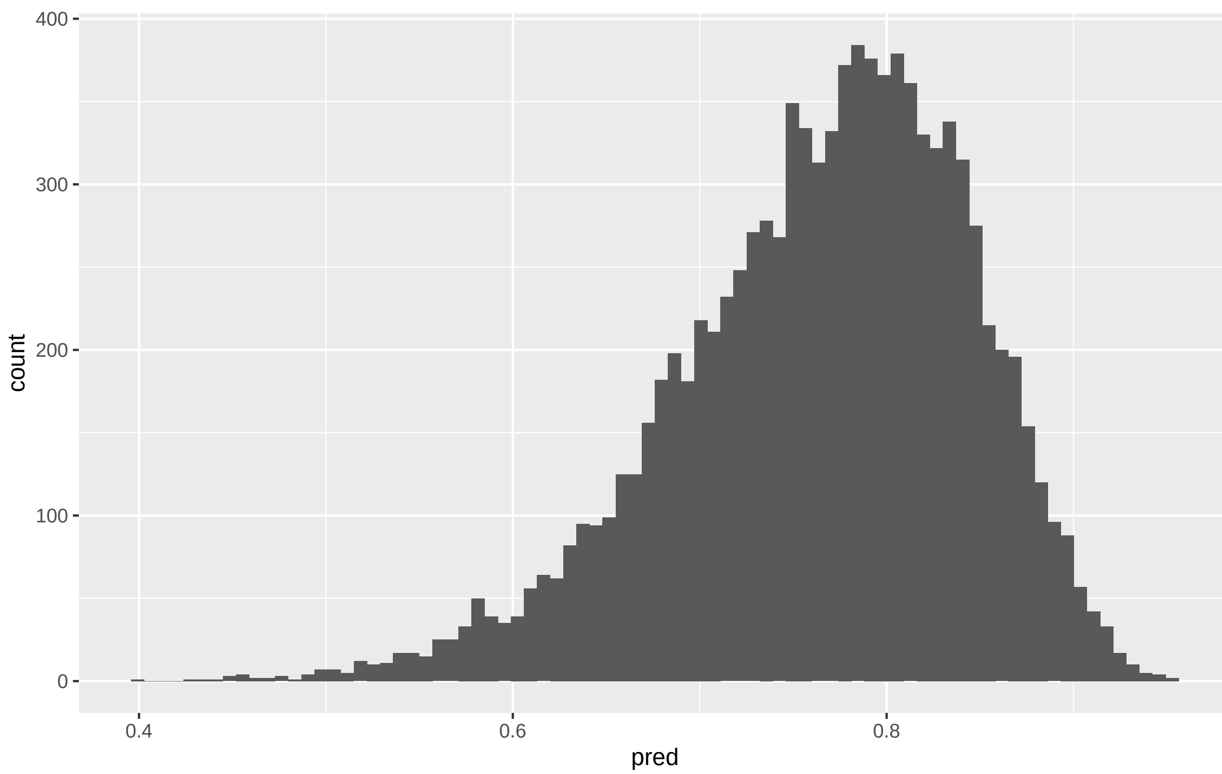
Comparing all the posterior mode for each features and compute the 95% interval of NSmallChild, we can say that NSmallChild has important affect to the probability that a woman works. To be more precise, it has negative impact of it, if the woman has child that has the age ≤ 6 years the chance of she is working decrease.

Question 2b

Use your normal approximation to the posterior from (a). Write a function that simulate draws from the posterior predictive distribution of $\Pr(y = 0|x)$, where the values of x corresponds to a 40-year-old woman, with two children (4 and 7 years old), 11 years of education, 7 years of experience, and a husband with an income of 18. Plot the posterior predictive distribution of $\Pr(y = 0|x)$ for this woman.

```
woman <- c(1,18,11,7,40,1,1)
sigma = solve(-OptimRes$hessian) #since the input of rmvnorm is covariance not sd

sim_draw <- function(x,mean,sigma){
  #convert x
  x <- as.matrix(x)
  beta <- rmvnorm(n=1, mean = mean, sigma =sigma)

  #logistic regression
```

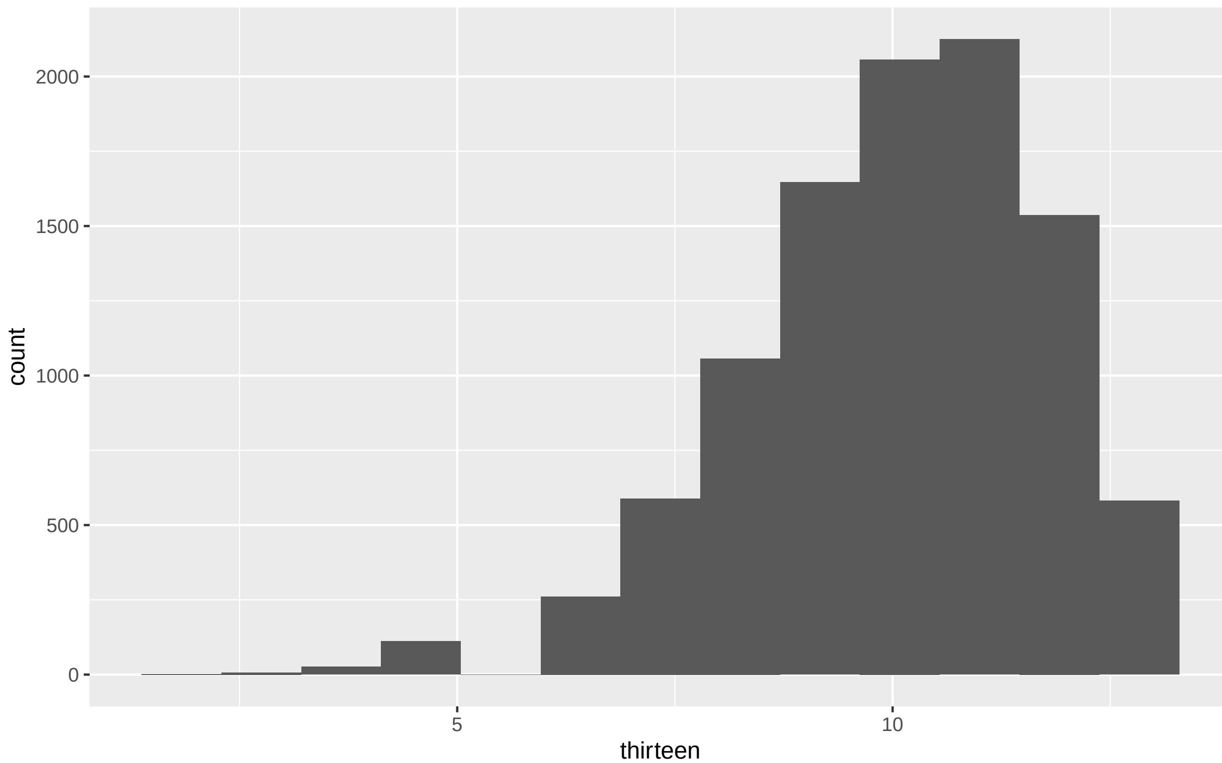
```r
  #the transpose of beta is to make sure the dimension correct
  elem1 <- exp(t(x)%*%t(beta))
  #1-,Since we are looking for "not-working" and the equation given is for "working"
  draw <- 1-(elem1/(1 + elem1))

  return(draw)
}

pred <- replicate(10000, sim_draw(woman,beta_mode,sigma))

plotdf <- as.data.frame(pred)

ggplot(data = plotdf)+geom_histogram(aes(x = pred),bins = 80)
```



Question 2c

Now, consider 13 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 13, that are not working. [Hint: Simulate from the binomial distribution, which is the distribution for a sum of Bernoulli random variables.]

```r
thirteen <- replicate(10000,rbinom(1,13,sample(pred)))
thirteen_df <- as.data.frame(thirteen)
ggplot(data = thirteen_df)+geom_histogram(aes(x = thirteen),bins = 13)
```


2000
1500
1000
500
0
count
5
10
thirteen

Appendix

```r
# Assignment 1

temperature_data <- read_xlsx('Linkoping2022.xlsx')
temperature_data$datetime <- as.Date(temperature_data$datetime)

create_time <- function(x)
{
  res <- as.numeric(x - as.Date('2022-01-01')) / 365

  res
}

temperature_data$time <- create_time(temperature_data$datetime)
temperature_data$time2 <- temperature_data$time^2

time_mat <- as.matrix(cbind("bias" = 1,"time" = temperature_data$time,
                            "time_2" = (temperature_data$time)^2))
# lecture 5
# get sigma^2 from inv chi-sq simlator using v0 & sigma_0^2

n <- dim(temperature_data)[1]-1
InvChiSq <- function(sample_size,n,tau2)
{
  X <- rchisq(sample_size,df = n)
  xs <- (n*tau2)/X
  return(xs)
}

# prior variance
prior_var <- function(v0,s2)
{
  pvar <- InvChiSq(sample_size = 1,n = v0,tau2 = s2)
  return(pvar)
}

# now get beta given variance from mvt norm distribution
# joint prior
prior_beta <- function(mu0,sigma_2,omega0_inv){
  betaprior <- rmvnorm(1,mean = mu0, sigma = sigma0_2*omega0_inv)
  return(betaprior)
}

# initialize starting hyperparameters
sigma0_2 <- 1
v0 <- 1
mu0 <- matrix(c(0,100,-100),nrow = 3,ncol = 1)
omega0_inv <- solve(diag(x = 0.01,nrow = 3,ncol = 3))

prior_draws <- c()

plot_df <- list()
```

```r
#set.seed(123)
p <- ggplot()
  for(i in 1:50){
  sigma_2 <- prior_var(v0 = 1,s2 = 1)
  val <- prior_beta(mu0 = mu0,sigma_2 = sigma_2,omega0_inv = omega0_inv)
  prior_draws <- c(prior_draws,val)
  y <- time_mat %*% t(val)

  plot_df[[i]] <- data.frame(cbind("x" = temperature_data$time,"y" = y))
  p <- p + geom_line(data = plot_df[[i]], aes(x = x, y = V2), col = "blue")+
    theme_bw() +
  labs(x = "Time",y= "Temperature",
       title = "Prior Distribution with Initial Hyperparameters")+
    theme(plot.title = element_text(hjust = 0.5))
  }
p + geom_point(data = temperature_data,aes(x = time,y = temp))

# rerun with changed hyperparameters
v0 <- 4
sigma0_2 <- 12
mu0 <- matrix(c(-6,75,-75),nrow = 3,ncol = 1)
omega0_inv <- solve(diag(x = 0.6,nrow = 3,ncol = 3))

# different results for every run,since seed is different
prior_draws <- c()
sigma_2 <- prior_var(v0 = v0,s2 = sigma0_2)
plot_df <- list()
p <- ggplot()
for(i in 1:20){
  val <- prior_beta(mu0 = mu0,sigma_2 = sigma_2,omega0_inv = omega0_inv)
  prior_draws <- c(prior_draws,val)
  y <- time_mat %*% t(val)
  plot_df[[i]] <- data.frame(cbind("x" = temperature_data$time,"y" = y))
  p <- p + geom_line(data = plot_df[[i]], aes(x = x, y = V2), col = "blue")+
    theme_bw()
}
p + geom_point(data = temperature_data,aes(x = time,y = temp)) +
  labs(x = "Time",y= "Temperature",title = "Prior Curves with Optimal Hyperparameters ")+
  theme(plot.title = element_text(hjust = 0.5))

#formulas from lecture slide 5
# compute posterior
v_n <- v0 + n
omega_n <- (t(time_mat)%*%time_mat) + solve(omega0_inv)

beta_hat <- solve(t(time_mat)%*%time_mat) %*% t(time_mat) %*% temperature_data$temp

mu_n <- (solve((t(time_mat)%*%time_mat) + solve(omega0_inv))) %*%
  (((t(time_mat)%*%time_mat)%*%beta_hat) + solve(omega0_inv)%*%mu0)

v_n_sigma2_n <- v0*sigma0_2 +
  ((t(temperature_data$temp)%*%temperature_data$temp) +
     (t(mu0)%*%solve(omega0_inv)%*%mu0 ) - (t(mu_n)%*%omega_n%*%mu_n))
```

```r
sigma_2 <- v_n_sigma2_n/v_n

# posterior variance
posterior_var <- function(v_n,sigma_2)
{
  pvar <- InvChiSq(sample_size = 1,n = v_n,tau2 = sigma_2)
  return(pvar)
}

# posterior betas
posterior_beta <- function(mu_n,sigma_2,omega_n){
  betaposterior<- rmvnorm(1,mean = mu_n, sigma = as.vector(sigma_2)*solve(omega_n))
  return(betaposterior)
}

## plotting posterior
posterior_draws <- c()
plot_df <- list()
p <- ggplot()
for(i in 1:20){
  sigma_2 <- posterior_var(v_n = v_n,sigma_2 = sigma_2 )
  val <- posterior_beta(mu_n = mu_n,sigma_2 = sigma_2,omega_n = omega_n)
  posterior_draws <- c(posterior_draws,val)
  y <- time_mat %*% t(val)
  plot_df[[i]] <- data.frame(cbind("x" = temperature_data$time,"y" = y))
  p <- p + geom_line(data = plot_df[[i]], aes(x = x, y = V2), col = "blue") +
    theme_bw()

}
p + geom_point(data = temperature_data,aes(x = time,y = temp)) +
  labs(x = "Time",y= "Temperature",title = "Posterior Distribution") +
  theme(plot.title = element_text(hjust = 0.5))

## sample from posterior, then plot marginal densities
posterior_mat <- matrix(nrow = 10000,ncol = 3)
posterior_variance <- c()
for(i in 1:10000){
  posterior_mat[i,] <- posterior_beta(mu_n = mu_n,sigma_2 = sigma_2,omega_n = omega_n)
  posterior_variance[i] <- posterior_var(v_n = v_n,sigma_2 = sigma_2)

}
colnames(posterior_mat) <- c('b0','b1','b2')

# posterior plots for b0,b1 and b2
hist(posterior_mat[,1],breaks = 100,main = 'Posterior Distribution of Beta0',
     probability = TRUE,xlab = 'Beta0')
lines(density(posterior_mat[,1]),col = 'red')
hist(posterior_mat[,2],breaks = 100,main = 'Posterior Distribution of Beta1',
     probability = T,xlab = 'Beta1')
lines(density(posterior_mat[,2]),col = 'red')
hist(posterior_mat[,3],breaks = 100,main = 'Posterior Distribution of Beta2',
     probability = T,xlab = 'Beta2')
lines(density(posterior_mat[,3]),col = 'red')
```

```r
# posterior plot for sigma2
hist(posterior_variance,breaks = 100,main = 'Posterior Distribution of Variance',
     probability = T,xlab = 'Posterior Variance')
lines(density(posterior_variance),col = 'red')

f_time  <- time_mat%*%t(posterior_mat)

stat_df <- data.frame('lower' = 0,'median' = 0,'upper' = 0)

for(i in seq(0,dim(f_time)[1])){
  #calculate lower,median and upper CI values
  lower <- quantile(f_time[i,],probs = 0.05)
  med <- median(f_time[i,])
  upper <- quantile(f_time[i,],probs = 0.95)
  stat_df[i,] <- c(lower,med,upper)
}

#combine with temperature data for plotting
combined_df <- data.frame(cbind(temperature_data,stat_df))


p <- ggplot() + geom_point(data = combined_df,aes(x = time,y = temp))+
  geom_line(data =  combined_df,aes(x = time, y = lower, col = "lower CI "))+
  geom_line(data =  combined_df,aes(x = time, y = median, col = "median"))+
  geom_line(data =  combined_df,aes(x = time, y = upper, col = "upper CI"))+
  theme_bw() + labs(x = "Time",y= "Temperature",title = "Posterior Median and CIs") +
  theme(plot.title = element_text(hjust = 0.5))

p

maxX <- -(posterior_mat[,2])/(2 * posterior_mat[,3])
hist(maxX,breaks = 100,main = 'Posterior Distribution of x(tilde)',probability = T)
lines(density(maxX),col = 'blue')
```

```r
#Assignment 2
#Question 2a
set.seed(12345)
library(mvtnorm)
library(ggplot2)
WomenAtWork <- read.delim("WomenAtWork.dat", header = TRUE, sep="")
glmModel<- glm(Work ~ 0 + ., data = WomenAtWork, family = binomial)
#summary(glmModel)

women_df <- WomenAtWork[,2:ncol(WomenAtWork)]
lable <- WomenAtWork[, 1]
Npar <- dim(women_df)[2]

# Initialize prior
mu <- as.matrix(rep(0, Npar))
tau <- 2
Sigma <- tau^2 * diag(Npar) #tau^2I

LogPostLogistic <- function(betas,y,X,mu,Sigma){
  X = as.matrix(X)
  linPred <- X%*%betas
  logLik <- sum( linPred*y - log(1 + exp(linPred)) )
  logPrior <- dmvnorm(betas, mu, Sigma, log=TRUE)
  return(logLik + logPrior)
}

# Initialize
initVal <- matrix(0, Npar, 1)

# Opt
OptimRes <- optim (initVal, LogPostLogistic, gr = NULL, y = lable, X = women_df,
                   mu = mu, Sigma = Sigma, method=c("BFGS"),
                   control=list(fnscale=-1),hessian=TRUE)

beta_mode <- OptimRes$par

# Printing results
posterior_df <- as.data.frame(OptimRes$par)
colnames(posterior_df) <- "posterior mode "
posterior_df$glmest <- glmModel$coefficients
row.names(posterior_df) <- colnames(women_df)

approxPostStd <- sqrt(diag(solve(-OptimRes$hessian))) # Computing approximate standard deviations.
approxPostStd_df <- as.data.frame(approxPostStd)
colnames(approxPostStd_df) <- "approxPostStd"
approxPostStd_df$glmstd <- summary(glmModel)$coefficients[, 2]
row.names(approxPostStd_df) <- colnames(women_df)

kbl(posterior_df)
kbl(approxPostStd_df)

upper<-beta_mode+1.96*approxPostStd
lower<-beta_mode-1.96*approxPostStd
```

```r
cat("The intervals for the variable NSmallChild are upper=",upper[6],
    "lower=",lower[6])

#Question 2b
woman <- c(1,18,11,7,40,1,1)
sigma = solve(-OptimRes$hessian) #since the input of rmvnorm is covariance not sd

sim_draw <- function(x,mean,sigma){
  #convert x
  x <- as.matrix(x)
  beta <- rmvnorm(n=1, mean = mean, sigma =sigma)

  #logistic regression
  #the transpose of beta is to make sure the dimension correct
  elem1 <- exp(t(x)%*%t(beta))
  #1- ,Since we are looking for "not-working" and the equation given is for "working"
  draw <- 1-(elem1/(1 + elem1))

  return(draw)
}

pred <- replicate(10000, sim_draw(woman,beta_mode,sigma))

plotdf <- as.data.frame(pred)

ggplot(data = plotdf)+geom_histogram(aes(x = pred),bins = 100)

#Qusetion 2c

thirteen <- replicate(10000,rbinom(1,13,sample(pred)))
thirteen_df <- as.data.frame(thirteen)
ggplot(data = thirteen_df)+geom_histogram(aes(x = thirteen),bins = 13)
```