Bayesian_Lab2

Olayemi Morrison(olamo208) Greeshma Jeev Koothuparambil (greko370)

2024-04-29

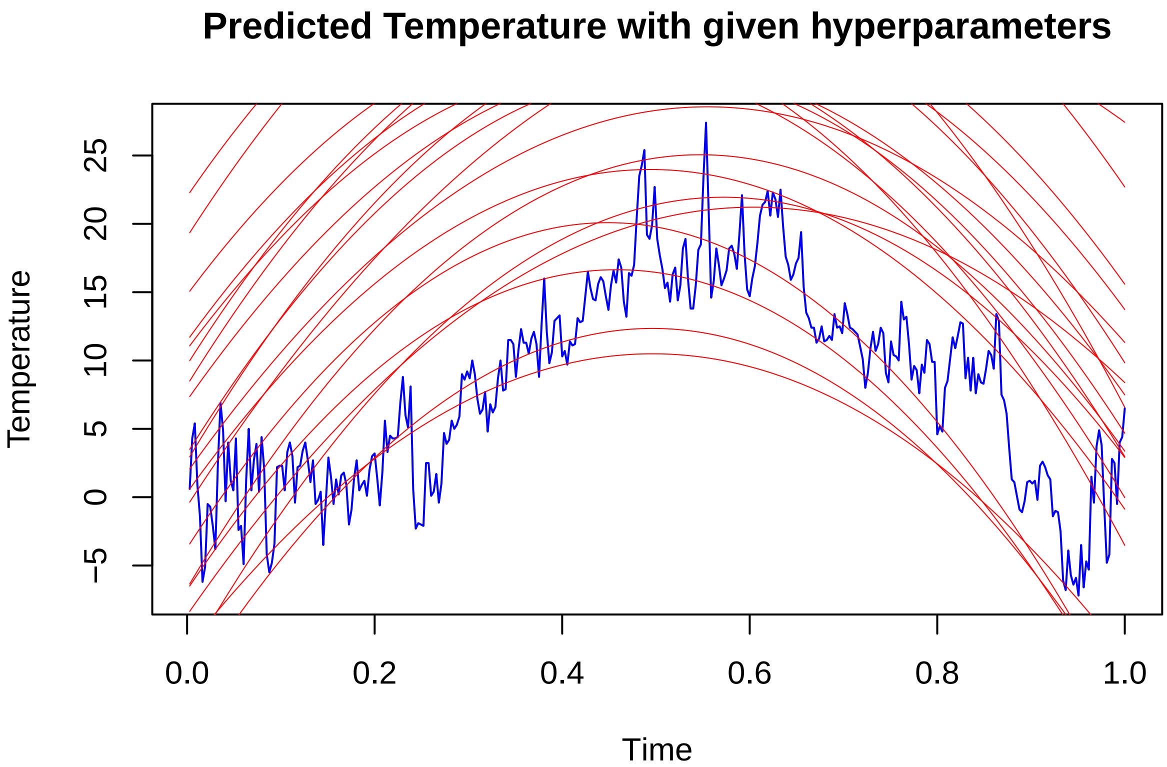
Assignment 1

Question 1a

Use the conjugate prior for the linear regression model. 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.

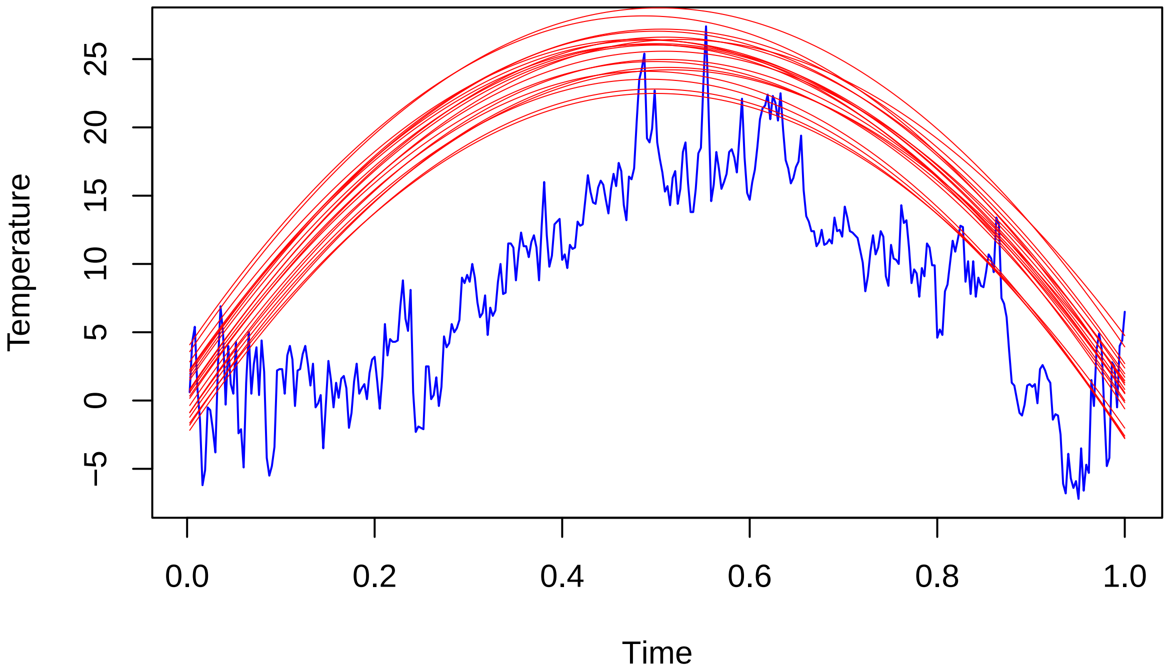
```
library(readxl)
library(mvtnorm)

tempdf <- read_xlsx("Linkoping2022.xlsx")
intercept <- rep(1,365)
data1 <- cbind(tempdf, "intercept"=intercept)
time <- c(1:365)/365
data1$time <- time
time2 <- time^2
time2 <- data1$time^2
data1 <- cbind(data1, "time2"=time2)
time_mtx <- as.matrix(data.frame(intercept,time,time2))

#Assign hyperparameters for the prior
mu0=matrix(c(0,100,-100))
omega0=diag(x=0.01, nrow=3, ncol=3)
nu0=1
sigmasq0=1

#Joint prior function
priorfunc = function(mu0,omega0,nu0,sigmasq0){
  set.seed(12345)
  for(i in 1:20){
    #using chi_sq to sample sigma^2
    chi_sample = rchisq(n=1, df=nu0)
    sigma2 = nu0*sigmasq0/chi_sample

    #using mvtnorm sample beta
    beta = rmvnorm(n=1, mean=mu0, sigma=sigmasq0*solve(omega0))

    #quadratic regression
    quad_regre= beta[1]+beta[2]*data1$time+beta[3]*(data1$time^2)+rnorm(1,mean=0, sd=sqrt(sigma2))
    lines(x=data1$time, y=quad_regre,col="red",lwd=0.5)
```

```
  }
}

### Check the given hyperparameters
plot(data1$time,data1$temp, col="blue",
     main="Predicted Temperature with given hyperparameters", ylab="Temperature",
     xlab="Time", type="l")
priorfunc( mu0, omega0, nu0, sigmasq0)
```



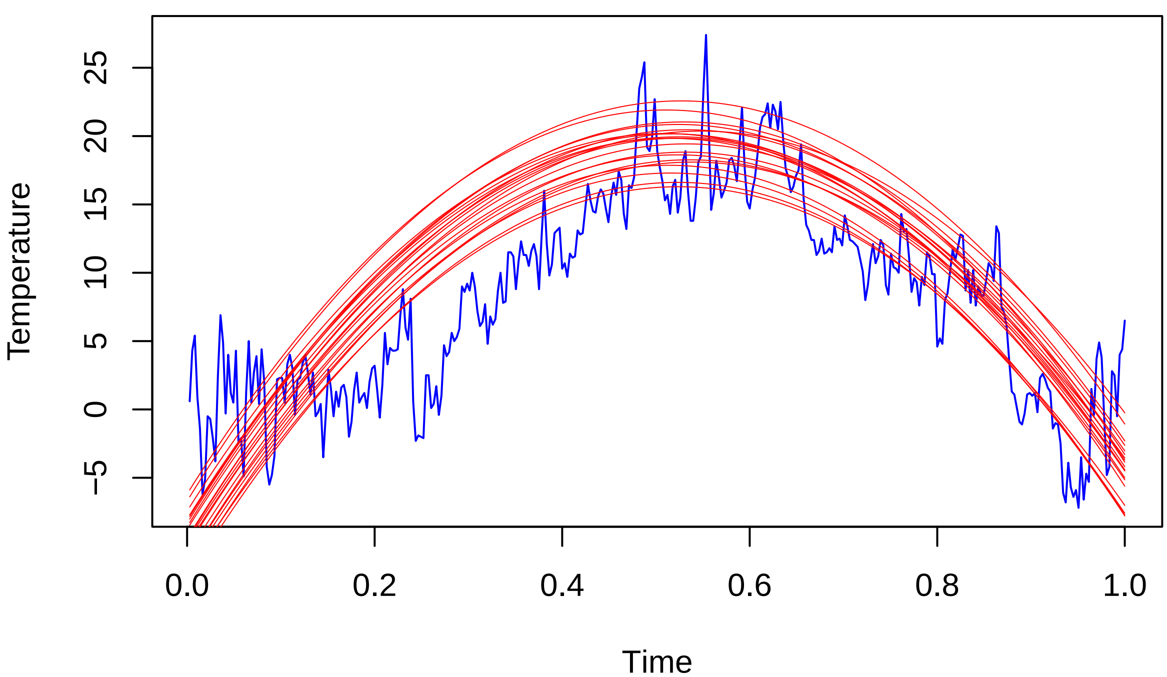

```
### change the hyperparameter sigma
plot(data1$time,data1$temp, col="blue",
     main="Predicted Temperature with given hyperparameters", ylab="Temperature",
     xlab="Time", type="l")
priorfunc( mu0, omega0, nu0, sigmasq0=0.03)
```

Predicted Temperature with given hyperparameters




```r
# Change the hyperparameter mu
plot(data1$time,data1$temp, col="blue",
     main="Predicted Temperature with changed hyperparameters", ylab="Temperature",
     xlab="Time", type="l")
priorfunc(mu0=matrix(c(-10,110,-105)), omega0, nu0, sigmasq0=0.03)
```

Predicted Temperature with changed hyperparameters



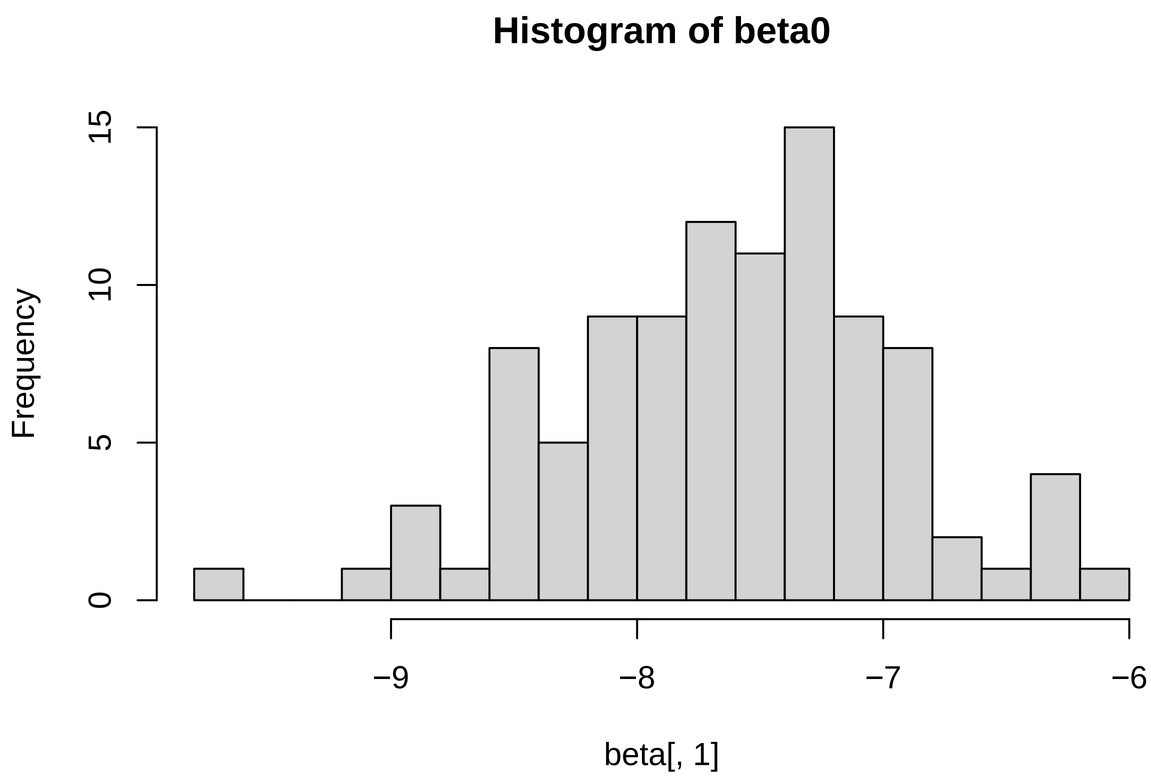
From the plots above, we can see that the prior agrees with my prior opinion to some degree, given different iterations of the hyperparameters.

Question 1b

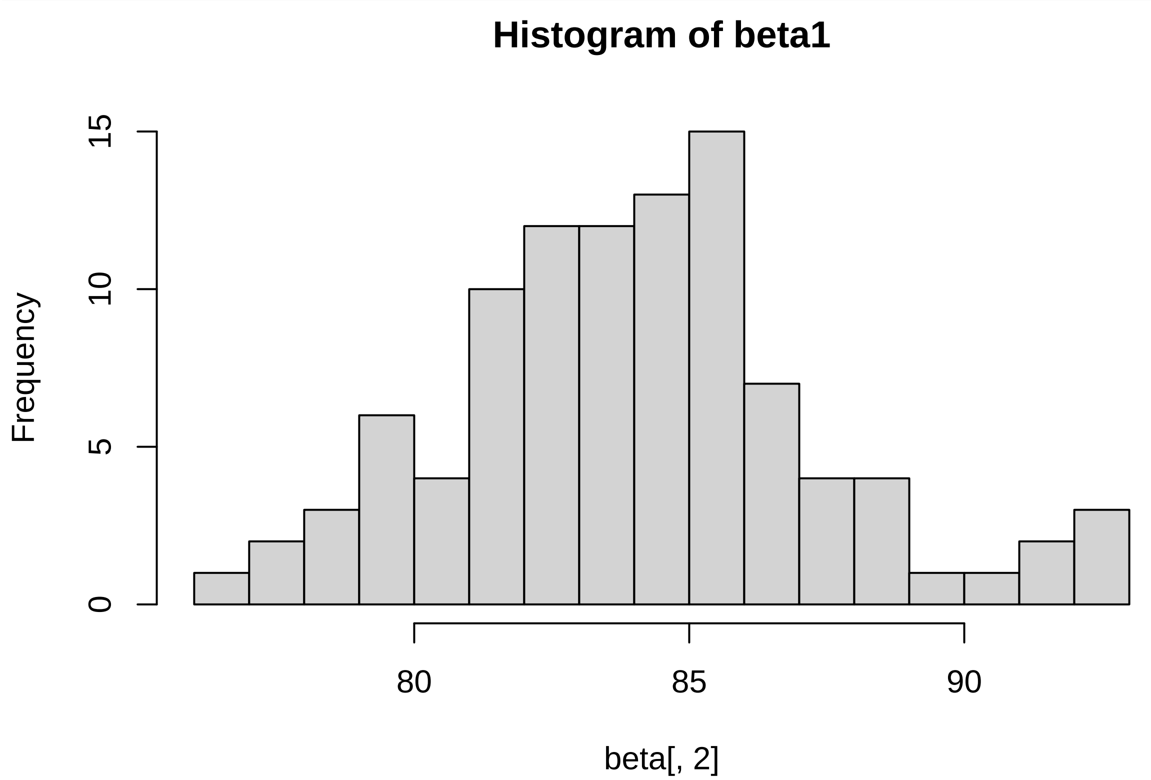
Write a function that simulate draws from the joint posterior distribu- tion of beta0, beta1, beta2 and sigma2. i. Plot a histogram for each marginal posterior of the parameters. ii. Make a scatter plot of the temperature data and overlay a curve for the posterior median of the regression function f(time) = E [tempjtime] = B0 + B1 * time + B2 * time2, i.e. the median of f (time) is computed for every value of time.

From the graph below, the parameters are simulated from the joint posterior distribution. The marginal posteriors for each parameter β_0,β_1,β_2,and σ^2 are shown below.

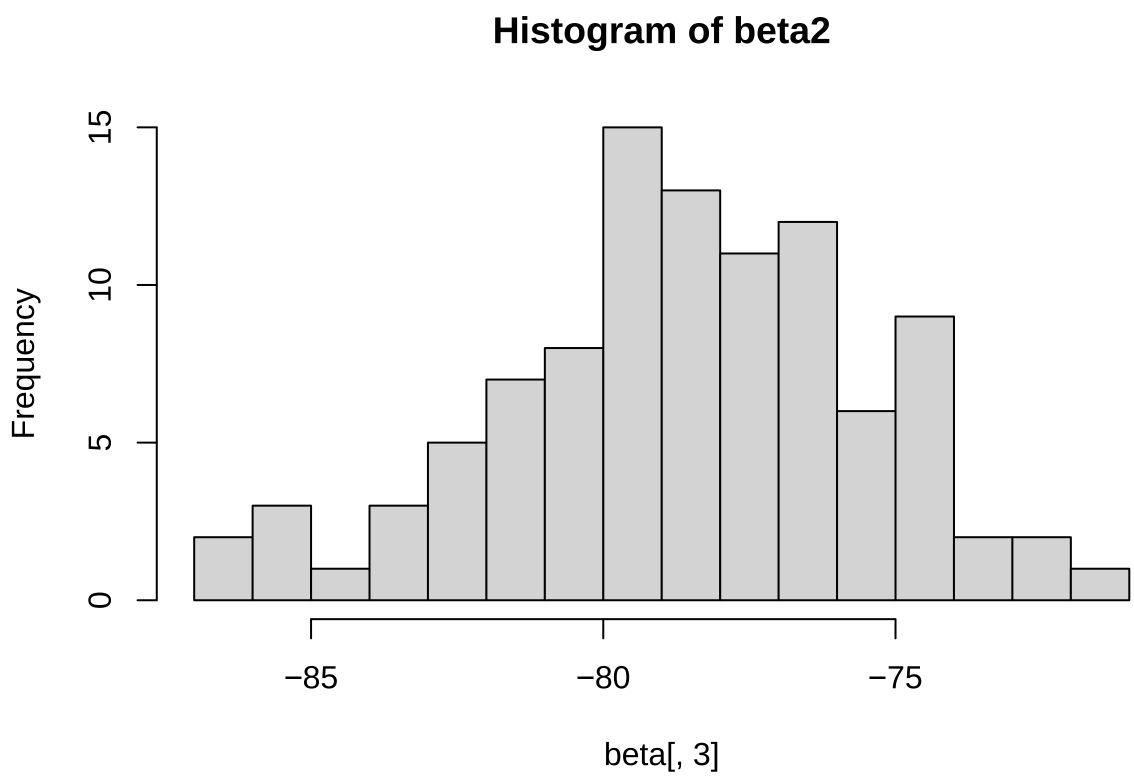
```
y <- as.matrix(tempdf$temp)
n <- 365
beta_hat <- solve(t(time_mtx) %*% time_mtx) %*% t(time_mtx) %*% y

mu_n <- solve(t(time_mtx) %*% time_mtx + omega0) %*% (t(time_mtx) %*%
time_mtx %*% beta_hat + omega0 %*% mu0)
omega_n <- t(time_mtx) %*% time_mtx + omega0
nu_n <- nu0 + n
sigmasqn <- (nu0 * sigmasq0 + (t(y) %*% y + t(mu0) %*% omega0 %*%
mu0 - t(mu_n) %*% omega_n %*% mu_n))/nu_n
sigmasqn <- as.numeric(sigmasqn)

#simulate
beta <- c()
sigma_square <- c()

for(i in 1:100){
  X <- rchisq(1,nu_n)
  sigmasqtemp <- nu_n * sigmasqn / X
  beta_temp <- rmvnorm(1,mu_n,sigmasqtemp * solve(omega_n))
  beta <- rbind(beta,beta_temp)
  sigma_square <- c(sigma_square,sigmasqtemp)
  }

#1.2.1
# plot the histogram for the first beta
hist(beta[,1],main = "Histogram of beta0",breaks = 20)
```



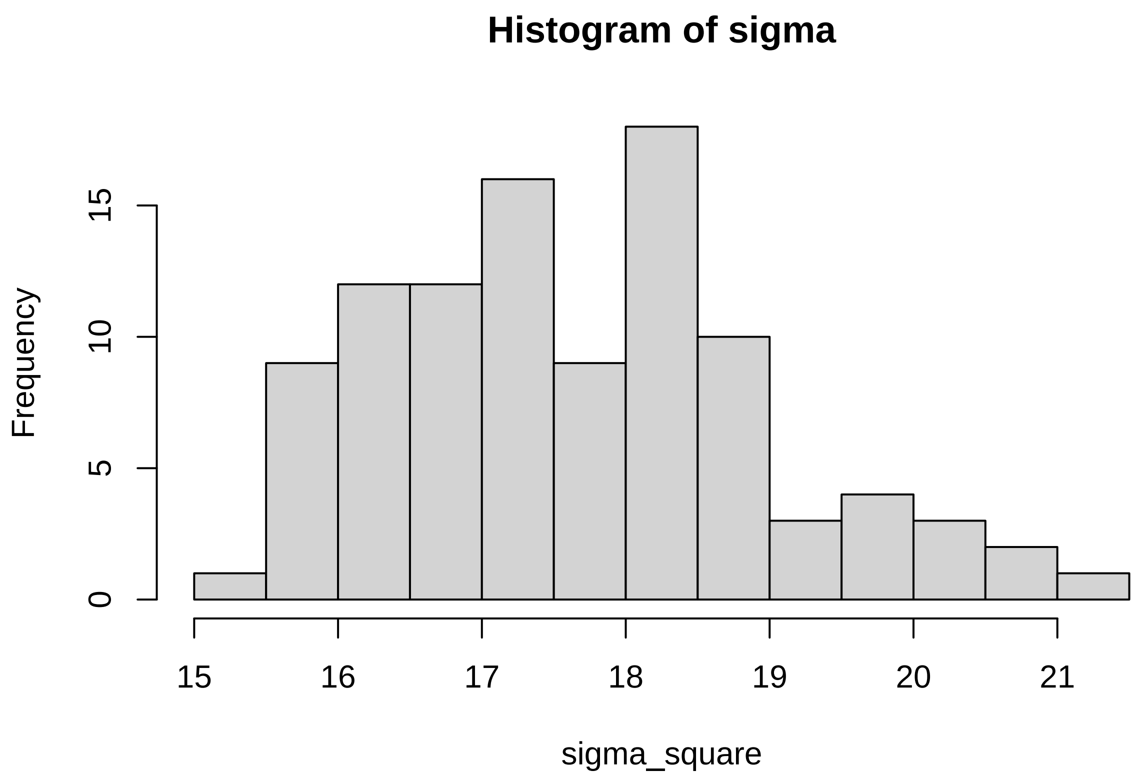

```r
# plot the histogram for the second beta
hist(beta[,2],main = "Histogram of beta1",breaks = 20)
```




```r
# plot the histogram for the third beta
hist(beta[,3],main = "Histogram of beta2",breaks = 20)
```

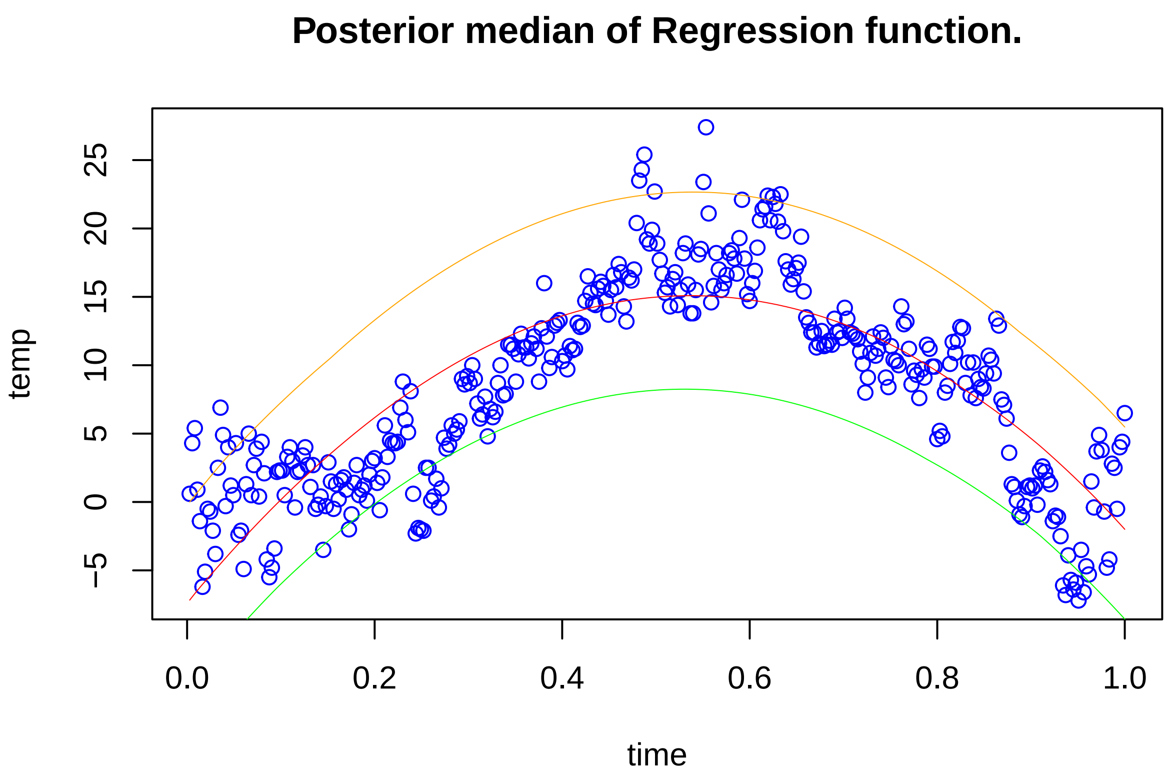



```r
#plot the histogram of sigma2
hist(sigma_square, main = "Histogram of sigma", breaks = 20)
```



Here is a scatter plot of the temperature data with the median and credible interval curves. We can see that most of the data points are contained in the 95% posterior credible interval

```r
#1.2.2
beta <- as.matrix(beta)
vector_epsilon <- unlist(lapply(sqrt(sigma_square),rnorm,n=1,mean=0))
temp <- time_mtx %*% t(beta)
temp <- + t(vector_epsilon + t(temp))# This is used for adding epsilon.
temp_median <- apply(temp,1,median)
temp_lower <- apply(temp,1,quantile,prob = 0.05)
temp_upper <- apply(temp,1,quantile,prob = 0.95)
plot(time,tempdf$temp,col="blue",xlab = "time",ylab = "temp",
     main = "Posterior median of Regression function.")
lines(time,temp_upper,type = "l",col="orange",lwd=0.5)
lines(time,temp_median,type = "l",col="red",lwd=0.5)
lines(time,temp_lower,type = "l",col="green",lwd=0.5)
```



Question 1c

Use the simulated draws in (b) to simulate from the posterior distribution of ~x.

```r
x_t <- -beta[,2] /(2 * beta[,3])
x_t
```

```
##   [1] 0.5287881 0.5305752 0.5397926 0.5297453 0.5345200 0.5310584 0.5338911
##   [8] 0.5321127 0.5308086 0.5363104 0.5365492 0.5345279 0.5318595 0.5340564
##  [15] 0.5307931 0.5376987 0.5328220 0.5327717 0.5258547 0.5371251 0.5315817
##  [22] 0.5311884 0.5363124 0.5320369 0.5309653 0.5337370 0.5373160 0.5390145
##  [29] 0.5311133 0.5393293 0.5335500 0.5340333 0.5387040 0.5408619 0.5286990
##  [36] 0.5304720 0.5320866 0.5338300 0.5382889 0.5309476 0.5413826 0.5373838
##  [43] 0.5311744 0.5400344 0.5271287 0.5339576 0.5302972 0.5348646 0.5368985
##  [50] 0.5350382 0.5368839 0.5319633 0.5334079 0.5378284 0.5394316 0.5370353
##  [57] 0.5399745 0.5405128 0.5314936 0.5479576 0.5407440 0.5333599 0.5393065
```

```
## [64] 0.5360932 0.5316760 0.5367229 0.5321840 0.5310863 0.5351674 0.5348822
## [71] 0.5350126 0.5302974 0.5387914 0.5412546 0.5367090 0.5228581 0.5413357
## [78] 0.5271468 0.5297267 0.5271232 0.5368974 0.5231314 0.5286561 0.5344778
## [85] 0.5358522 0.5311878 0.5363560 0.5359670 0.5360432 0.5428863 0.5251920
## [92] 0.5408308 0.5277097 0.5374673 0.5399620 0.5400942 0.5314077 0.5412248
## [99] 0.5239700 0.5332546
```

Question 1d

Say now that you want to estimate a polynomial regression of order 10, 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.

To prevent overfitting we suggest adding a regularization term. The proposed prior would like as follows:

$$\beta_i|\sigma^2 \sim^{iid} N(0, \frac{\sigma^2}{\lambda})$$

where λ will be the smoothness/shrinkage/regularization term. Ω_0 and λ are relate as $\Omega_0 = \lambda I$. A change in λ does not directly affect μ_0. However, it does influence μ_n through Ω_0. The larger λ is, the more shrinkage.

2a) Posterior approximation for classification with logistic regression.

The code for the posterior distribution is as shown below:

```r
#2.a

library("mvtnorm") # reads the mvtnorm package into R's memory. We can now use the necessary function d

### Prior and data inputs ###
Covs <- c(2:8) # Select which covariates/features to include
standardize <- TRUE # If TRUE, covariates/features are standardized to mean 0 and variance 1
tau <- 2# scaling factor for the prior of beta

# Loading the women dataset
WomenData <- data.frame(read.csv("WomenAtWork.dat",sep = " ")) # read data from file
Nobs <- dim(WomenData)[1] # number of observations
y <- as.matrix(WomenData$Work) # y=1 if the quality of wine is above 5, otherwise y=0.

X <- as.matrix(WomenData[,Covs]);
Xnames <- colnames(X)
#if (standardize){
#  Index <- 2:(length(Covs)-1)
#  X[,Index] <- scale(X[,Index])
#}
Npar <- dim(X)[2]

# Setting up the prior
mu <- matrix(0,Npar,1) # Prior mean vector
Sigma <- tau^2*diag(Npar) # Prior covariance matrix

# Functions that returns the log posterior for the logistic 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 he
  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

# Printing the results to the screen
PostCov <- solve(-OptimRes$hessian)
colnames(PostCov) <-Xnames
row.names(PostCov) <-Xnames
PostMode <- OptimRes$par[1:7]
names(PostMode) <- 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 approximate posterior standard deviation is:\n')
```

```
## [1] "The approximate posterior standard deviation is:\n"
```

```
print(approxPostStd)
```

```
##    Constant  HusbandInc   EducYears    ExpYears         Age NSmallChild
##  1.38198486  0.02198474  0.08920960  0.03335982  0.02747315  0.47746764
##   NBigChild
##  0.16401959
```

```
print('The posterior mode is:\n')
```

```
## [1] "The posterior mode is:\n"
```

```
print(PostMode)
```

```
##    Constant  HusbandInc   EducYears    ExpYears         Age NSmallChild
## -0.04036943 -0.03730689  0.17868950  0.12073637 -0.04618995 -1.47248930
##   NBigChild
## -0.02014458
```

```
print("The covariance matrix is :\n")
```

```
## [1] "The covariance matrix is :\n"
```

```
print(PostCov)
```

```
##                 Constant    HusbandInc     EducYears      ExpYears
## Constant     1.909882159  4.032517e-03 -6.280726e-02  1.041874e-03
## HusbandInc   0.004032517  4.833287e-04 -9.147892e-04 -2.666479e-05
```

```
## EducYears   -0.062807260 -9.147892e-04  7.958354e-03  5.508998e-05
## ExpYears     0.001041874 -2.666479e-05  5.508998e-05  1.112877e-03
## Age         -0.025755999 -6.428480e-05 -3.181372e-04 -2.845111e-04
## NSmallChild -0.137712005  1.585545e-03 -1.438778e-02 -1.336628e-03
## NBigChild   -0.088876440  4.986972e-06  1.133513e-04  7.206537e-04
##                       Age  NSmallChild     NBigChild
## Constant    -0.0257559994 -0.137712005 -8.887644e-02
## HusbandInc  -0.0000642848  0.001585545  4.986972e-06
## EducYears   -0.0003181372 -0.014387780  1.133513e-04
## ExpYears    -0.0002845111 -0.001336628  7.206537e-04
## Age          0.0007547741  0.005548132  1.044935e-03
## NSmallChild  0.0055481315  0.227975343  1.122711e-02
## NBigChild    0.0010449347  0.011227114  2.690243e-02
```

```r
#glm model evaluation for comparison purpose
glmModel <- glm(Work ~  0+., data = WomenData, family = binomial)

summary(glmModel)
```

```
##
## Call:
## glm(formula = Work ~ 0 + ., family = binomial, data = WomenData)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## Constant     0.02263    1.93083   0.012 0.990649
## HusbandInc  -0.03796    0.02229  -1.703 0.088573 .
## EducYears    0.18447    0.10007   1.844 0.065253 .
## ExpYears     0.12132    0.03353   3.618 0.000297 ***
## Age         -0.04858    0.03323  -1.462 0.143686
## NSmallChild -1.56485    0.51078  -3.064 0.002187 **
## NBigChild   -0.02526    0.17716  -0.143 0.886618
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 182.99  on 132  degrees of freedom
## Residual deviance: 146.73  on 125  degrees of freedom
## AIC: 160.73
##
## Number of Fisher Scoring iterations: 4
```

The values from glm Model and the Posterior distribution approximation on parameters are almost similar.

```r
#Calculating the credible interval of equal tail  95% for NSmallChild
womenDataPost <- rmvnorm(10000, mean = PostMode, sigma = PostCov)
NSmallChildPost <- womenDataPost[,6]
NSmallChildCI <- quantile(NSmallChildPost,c(0.025,0.975))
print("The credible interval of equal tail 95% for NsmallChild is :\n")
```

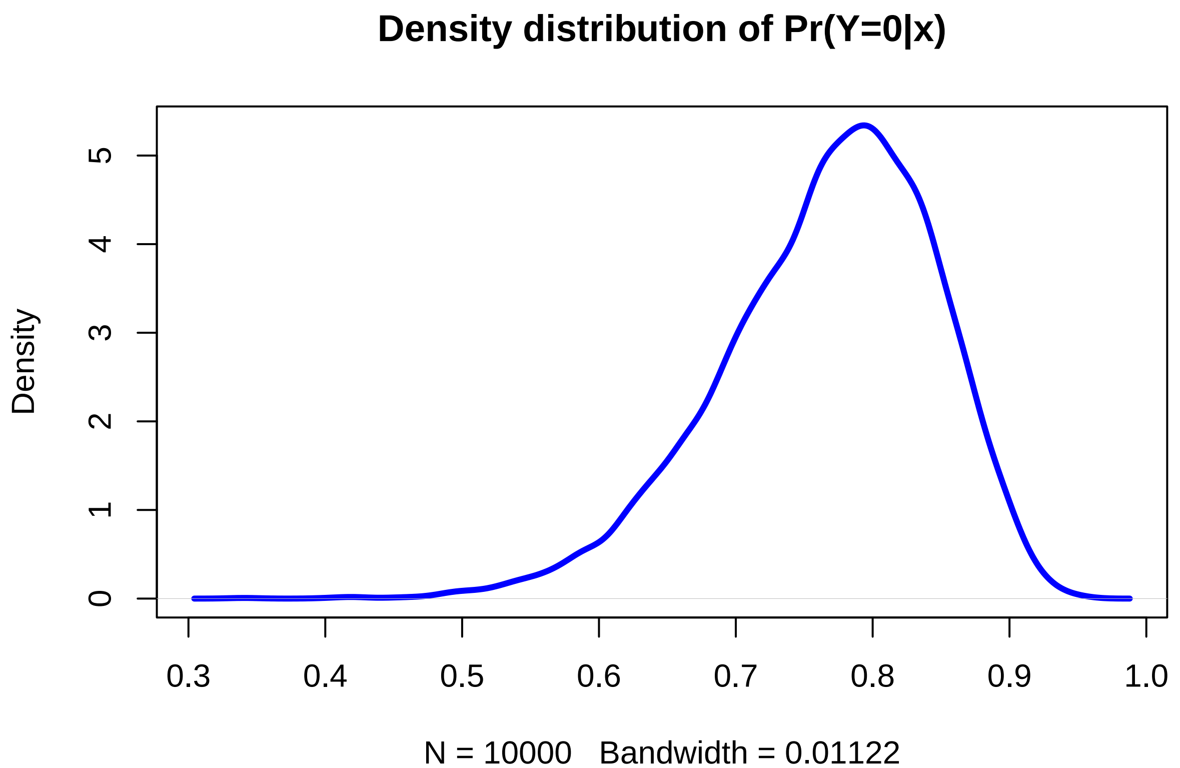
```
## [1] "The credible interval of equal tail 95% for NsmallChild is :\n"
```

```r
print(NSmallChildCI)
```

```
##       2.5%      97.5%
## -2.4166634 -0.5208677
```

2b) Function that simulate draws from the posterior predictive distribution of Pr(y = 0|x).

```
#2.b
postPredDistZero  <- function(sampleSize, mode, cov, xvalues) {
  sampleData <- rmvnorm(sampleSize, mean = mode, sigma = cov)
  yOfSample <- xvalues%*%t(sampleData)
  yProbZero <- 1-(exp(yOfSample)/(1+exp(yOfSample)))

  return(yProbZero)

}

xvals <- c(1,18,11,7,40,1,1)

probability <- postPredDistZero(10000, PostMode,PostCov, xvals)
plot(density(probability), type = 'l', lwd = 3, col = "blue",main = "Density distribution of Pr(Y=0|x)")
```

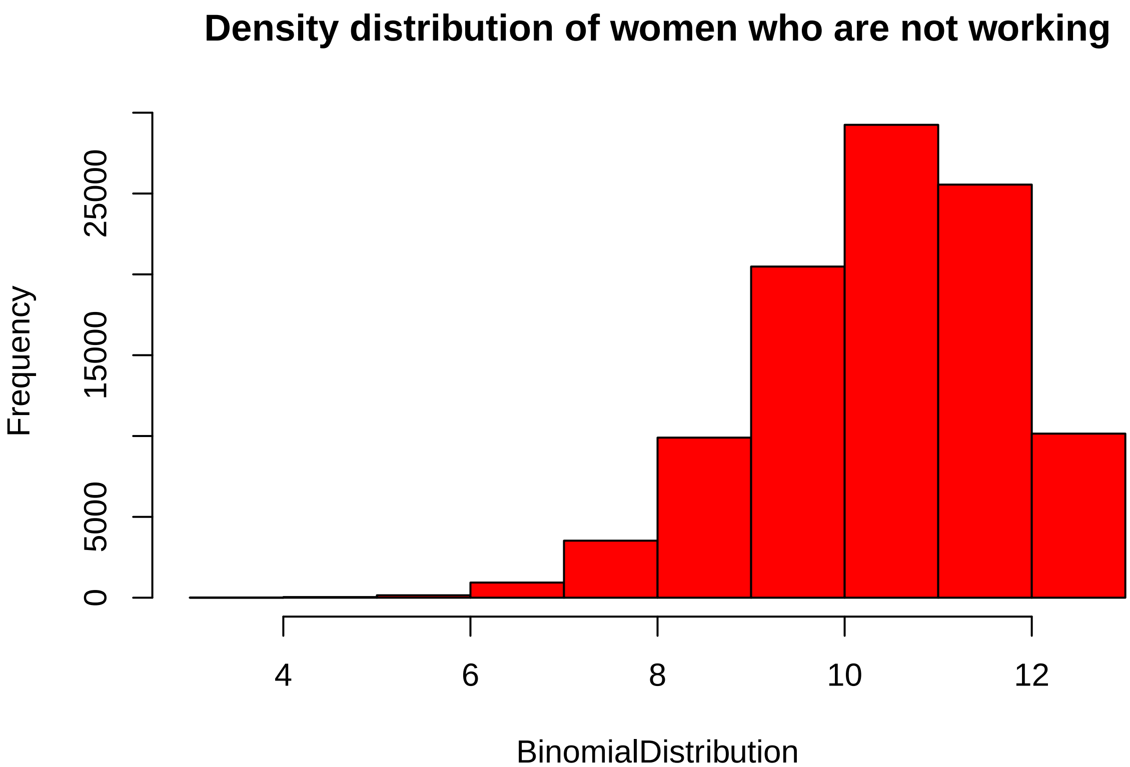


The plot shows that the probability of a 40-year-old woman, with two children (4 and 7 years old), 11 years of education, 7 years of experience, and a husband with an income of 18 not to work is extremely high.

2c) Rewrite your function and plot the posterior predictive distribution for the number of women, out of these 13, that are not working.

```
#2.c
#Additional function to find mode
FindMode <- function(listvalues) {
  uniqueValues <- unique(listvalues)
  modeValue <- uniqueValues[which.max(tabulate(match(listvalues, uniqueValues)))]
```

```
  return(modeValue)
}

#Modified function
postPredDistZeroBinomial  <- function(sampleSize, mode, cov, xvalues) {
  sampleData <- rmvnorm(sampleSize, mean = mode, sigma = cov)
  yOfSample <- xvalues%*%t(sampleData)
  yProbZero <- 1-(exp(yOfSample)/(1+exp(yOfSample)))
  yProbZero <- yProbZero[1,]
  #Finding Mode
  Mode <- FindMode(yProbZero)
  returnList <- list("Mode" = Mode ,"Probability" =yProbZero)
  return(returnList)

}

xvals <- c(1,18,11,7,40,1,1)

set.seed(1245)
ModeAndProbability <- postPredDistZeroBinomial(10000, PostMode,PostCov, xvals)
#setting the mode value of the Pr(Y=0|x) to be the probability of women do not work
set.seed(456)
BinomialDistribution <- rbinom(100000,13, ModeAndProbability$Mode)
hist(BinomialDistribution, col = "red",breaks = 13, main = "Density distribution of women who are not w
```



From the plot, around 10-12 out of 13 40-year-old women, with two children (4 and 7 years old), 11 years of education, 7 years of experience, and a husband with an income of 18 do not work.
