Bayesian Lab2

Dinesh (dinsu875) and Umamaheswarababu (umama339)

Question 1: Linear and polynomial regression

The dataset Linkoping2022.xlsx contains daily average temperatures (in degree Celcius) in Linköping over the course of the year 2022. Use the function read_xlsx(), which is included in the R package readxl (install.packages("readxl")), to import the dataset in R. The response variable is temp and the covariate time that you need to create yourself is defined by

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

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

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

task a)

Use the conjugate prior for the linear regression model. The prior hyperparameters μ_0, Ω_0, v_0 *and* σ_0^2 shall be set to sensible values. Start with $\mu_0 = (0, 100, -100)^T, \Omega_0 = 0.01 \cdot I_3, v_0 = 1$ *and* $\sigma_0^2 = 1$. 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: R package **mvtnorm** can be used and your $Inv - \chi^2$ simulator of random draws from Lab 1.]

- To find the regression curve, we used the *linear regression* model which has the following formula:

$$Y = X\beta + \epsilon, \quad where \;\; \epsilon \overset{iid}{\sim} N(0, \sigma^2)$$

Where β and σ^2 are unknown. And to simulate them for *task a* and *task b*, we have the joint prior and joint posterior distributions, respectively, as stated below:

- The *joint prior* distribution for both β and σ^2, is as follows, respectively:

$$\beta|\sigma^2 \sim N(\mu_0, \sigma^2\Omega_0^{-1})$$
$$\sigma^2 \sim Inv - \chi^2(\nu_0, \sigma_0^2)$$

- The *joint posterior* distribution for both β and σ^2, is as follows, respectively:

$$\beta|\sigma^2, y \sim N(\mu_n, \sigma^2\Omega_n^{-1})$$
$$\sigma^2|y \sim Inv - \chi^2(\nu_n, \sigma_n^2)$$

Where,

$$\mu_n = (X^T X + \Omega_0)^{-1}(X^T X\hat{B} + \Omega_0\mu_0)$$
$$\hat{\beta} = (X^T X)^{-1} X^T Y$$

$$\Omega_n = X^T X + \Omega_0$$

$$\nu_n = \nu_0 + n$$
$$\sigma_n^2 = \frac{\nu_0\sigma_0^2 + (Y^T Y + \mu_0^T\Omega_0\mu_0 - \mu_n^T\Omega_n\mu_n)}{\nu_n}$$

```
set.seed(1234)
library(mvtnorm)
library(ggplot2)
library(reshape2) # melt function
```

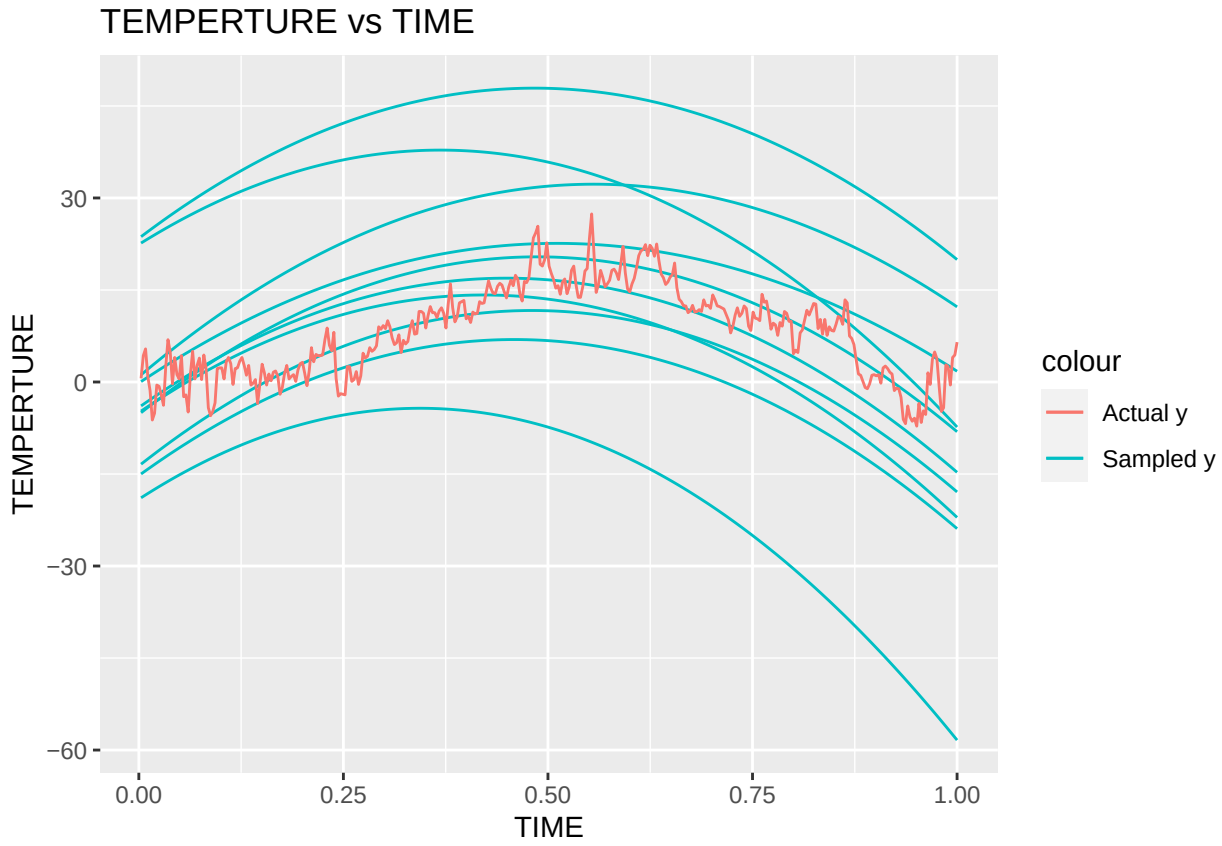
```
## Warning: package 'reshape2' was built under R version 4.2.2
```

```
library("readxl")
```

```
## Warning: package 'readxl' was built under R version 4.2.3
```

```
#Reading the excel data
data= read_excel("Linkoping2022.xlsx")
data$time <- (1:365)/ 365

Y=data$temp
X=cbind("intercept"=1,"time_1"=data$time, "time_2"=data$time^2)

# Initial Parameters
mu_not = c(0, 100, -100)
omega_not = 0.01 * diag(1,3,3)
nu_not = 15
sigma_sq_not = 1
n=365 # number of observations
n_sample=10

# prior
x_prior = rchisq(10, nu_not)
sigma_prior = (nu_not * sigma_sq_not)/x_prior
sample_beta_prior=c()
for (draw in sigma_prior) {
  temp=rmvnorm(1, mean = mu_not, sigma = (draw * solve(omega_not)))
  sample_beta_prior=c(sample_beta_prior,temp)
}

# By transforming the beta samples into a 3x10 matrix, it becomes possible to utilize it for the purpos
sample_beta_prior=matrix(sample_beta_prior, 3,10)
y_hat_prior=X%*%sample_beta_prior
```

```r
yhat_pr=melt(y_hat_prior)

# For every draw, a regression curve can be plotted.
ggplot()+
  geom_line(aes(x=rep(data$time, n_sample),color="Sampled y",
                y=yhat_pr$value,
                group=yhat_pr$Var2))+
  geom_line(aes(x=data$time, y=data$temp, color="Actual y"))+
  labs(title="TEMPERTURE vs TIME", x="TIME", y="TEMPERTURE")
```



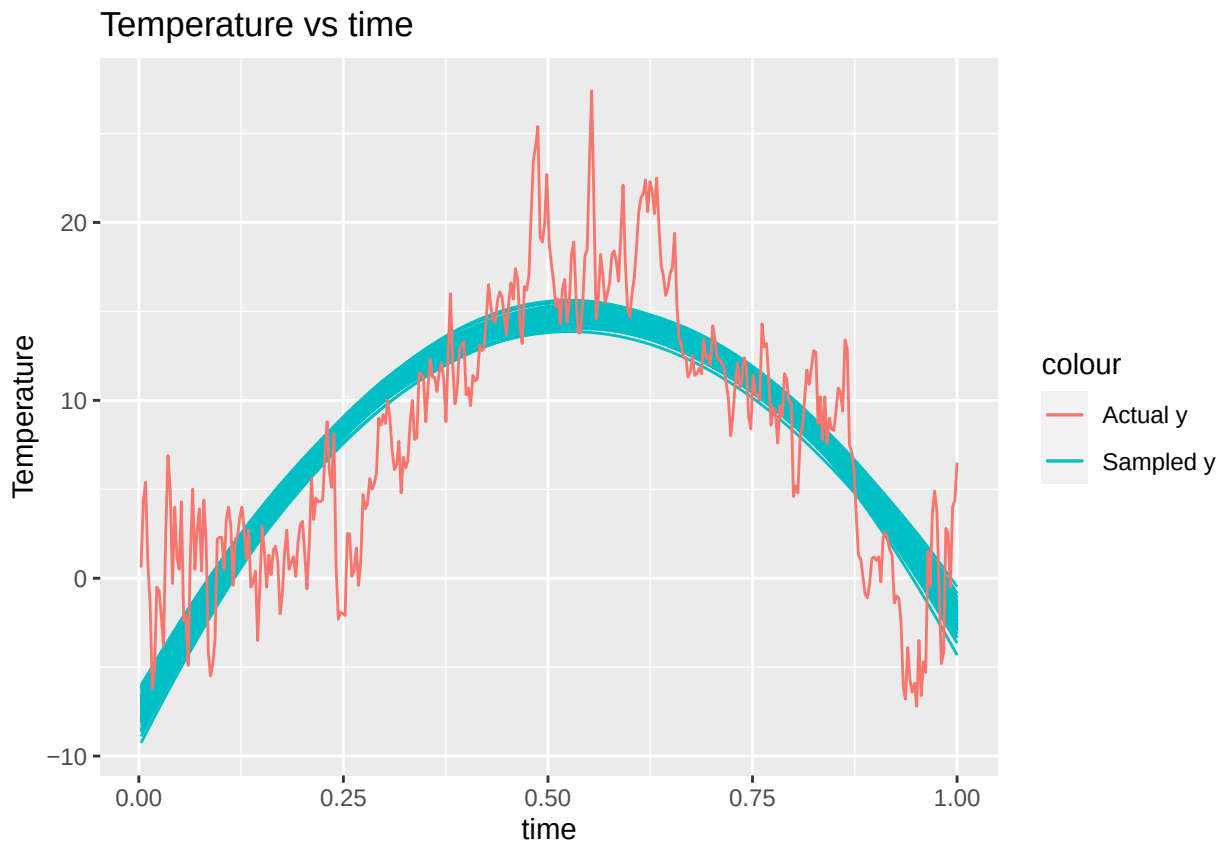
After plotting the regression curves using the initial hyperparameter values to predict the temperature, it appears that the curves are quite dispersed. Despite trying out various hyperparameter values, there were no significant changes observed in the behavior of the curves, except for a slight improvement when gradually increasing the degrees of freedom to higher values. As a result, v_0 was ultimately set to 15 i.e, $v_0 = 15$.

task b)

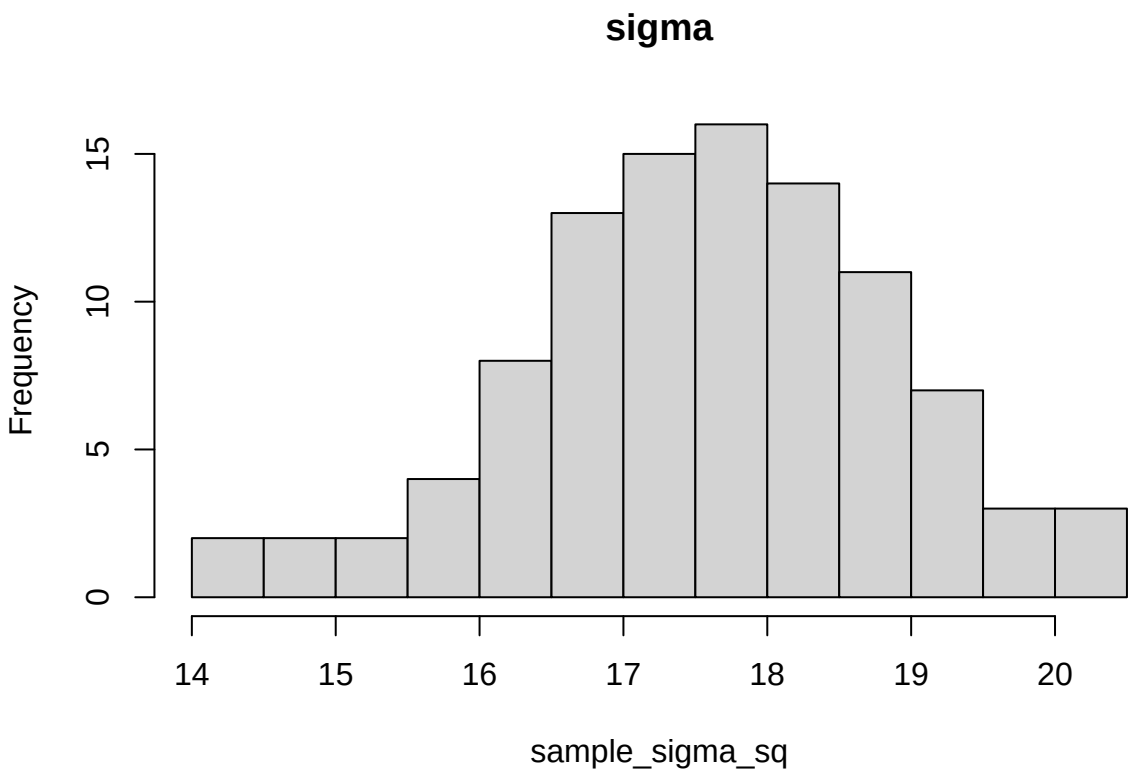
Write a function that simulate draws from the joint posterior distribution of $\beta_0, \beta_1, \beta_2$ *and* σ^2 i. Plot a histogram for each marginal posterior of the parameters. ii. Make a scatter plot of the temperature data and overlay a curve for theposterior median of the regression function $f(time) = E[temp|time] = \beta_0 + \beta_1 \cdot time + \beta_2 \cdot time^2$,i.e. the median of $f(time)$ is computed for every value of time.In addition, overlay curves for the 90% equal tail posterior probability intervals of f(time), i.e. the 5 and 95 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?

```r
library(mvtnorm)
library(ggplot2)
library(reshape2) # for melt function
set.seed(1234)

#Reading the excel data
data= read_excel("Linkoping2022.xlsx")
data$time <- (1:365)/ 365

Y=data$temp
X=cbind("intercept"=1,"time_1"=data$time, "time_2"=data$time^2)

# Initial Parameters
mu_not = c(0, 100, -100)
omega_not = 0.01 * diag(1,3,3)
nu_not = 1
sigma_sq_not = 1
n=365 # number of observations
n_sample=100

# draw x form chi square posterior parameters
beta_hat=solve(t(X)%*%X)%*% t(X)%*%Y

mu_n=solve((t(X)%*% X)+omega_not)%*%((t(X)%*% X %*% beta_hat)+omega_not%*%mu_not)
omega_n=(t(X) %*% X)+omega_not
nu_n=nu_not+n
sigma_sq_n=(nu_not*sigma_sq_not+(t(Y)%*%Y+t(mu_not)%*%omega_not%*%mu_not)-
              t(mu_n)%*%omega_n%*%mu_n)/(nu_n)

# draws (samples) for beta and sigma square posteriors:

# sigma posterior
# darwn the x from chi square
x = rchisq(n = 100, df = nu_n)
sample_sigma_sq= suppressWarnings((nu_n*sigma_sq_n)/x)

# beta posterior
sample_beta=c()
for (draw in sample_sigma_sq) {
  temp=rmvnorm(1, mean = mu_n, sigma = (draw * solve(omega_n)))
  sample_beta=c(sample_beta,temp)
}

sample_beta=matrix(sample_beta, 3,100)
y_hat=X%*%sample_beta

# melt for plot
y_hat_1=melt(y_hat)

# plot of the regression curve for each draw
ggplot()+
  geom_line(aes(x=rep(data$time, n_sample),color="Sampled y",
                y=y_hat_1$value,
```

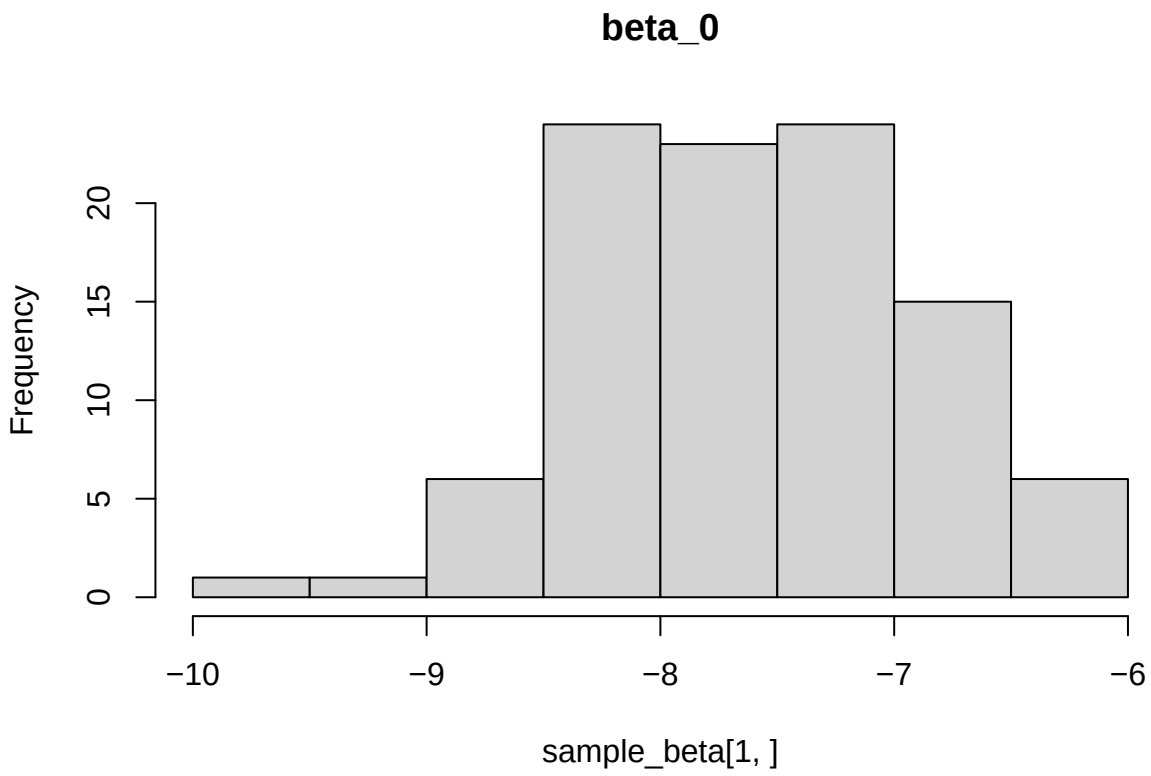
```
                group=y_hat_1$Var2))+
  geom_line(aes(x=data$time, y=data$temp, color="Actual y"))+
  labs(title="Temperature vs time", x="time", y="Temperature")
```



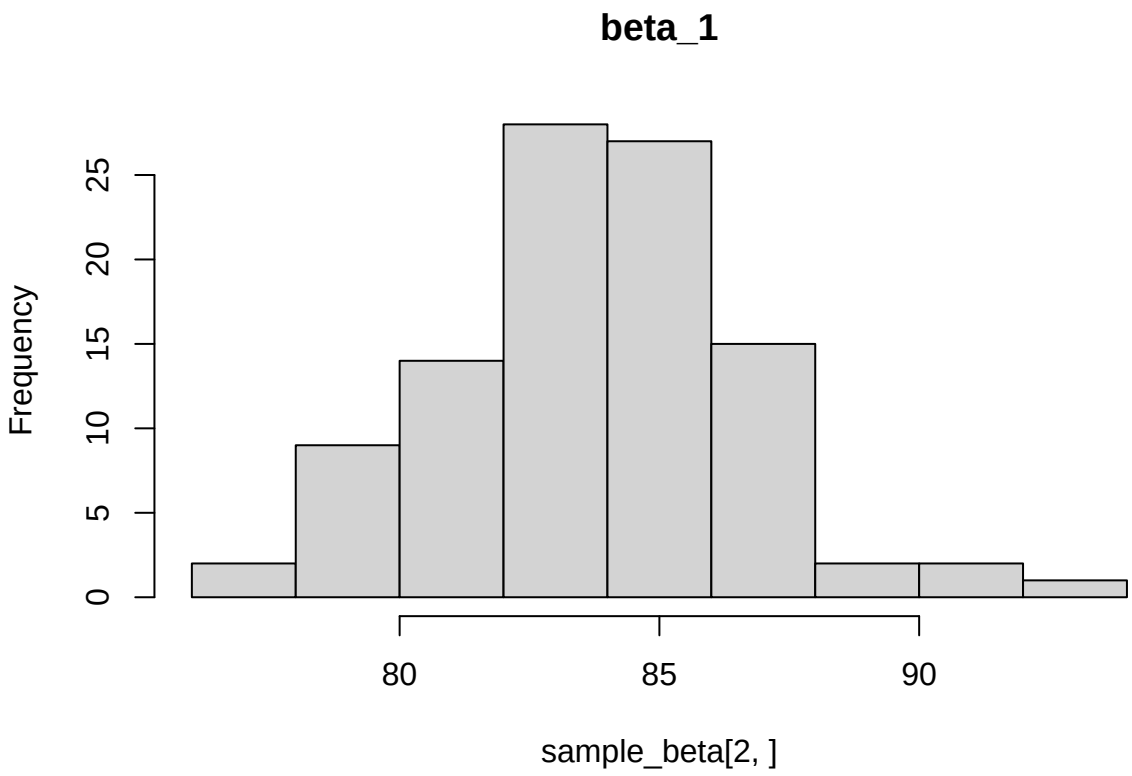

```
# Marginal posteriors
hist(sample_sigma_sq, breaks = 10, main = "sigma")
```



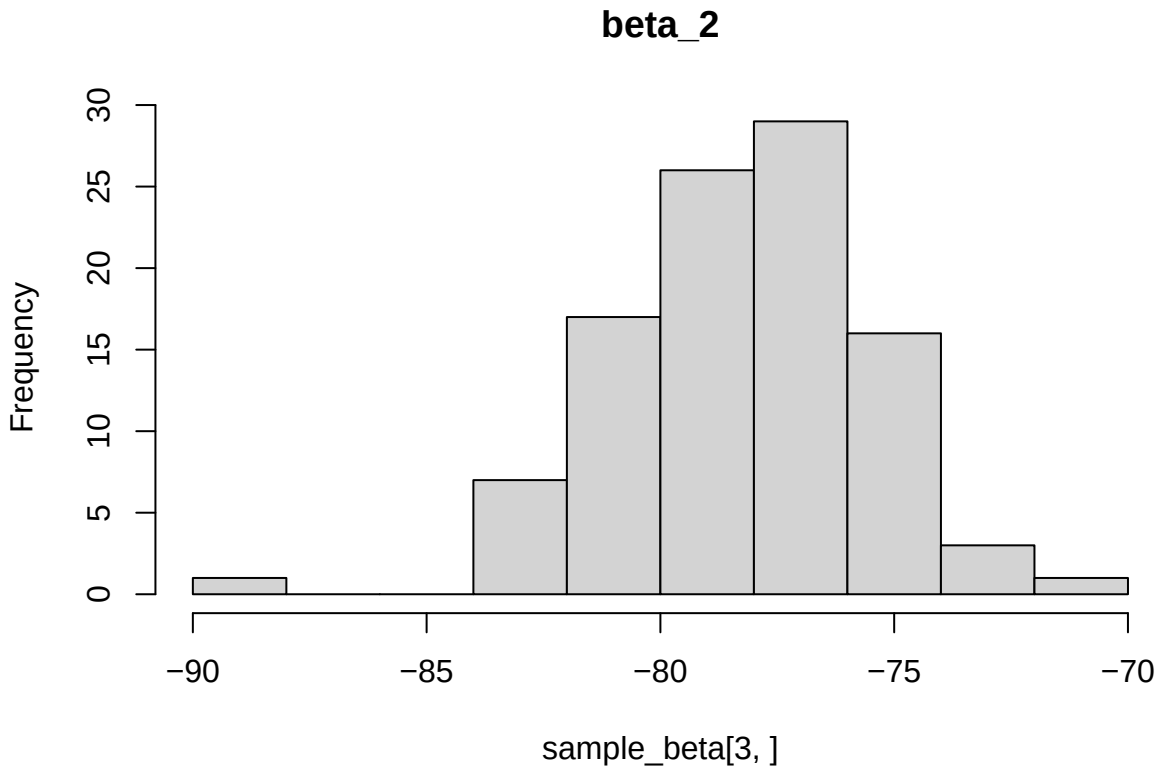

```r
hist(sample_beta[1,], breaks = 10, main = "beta_0")
```



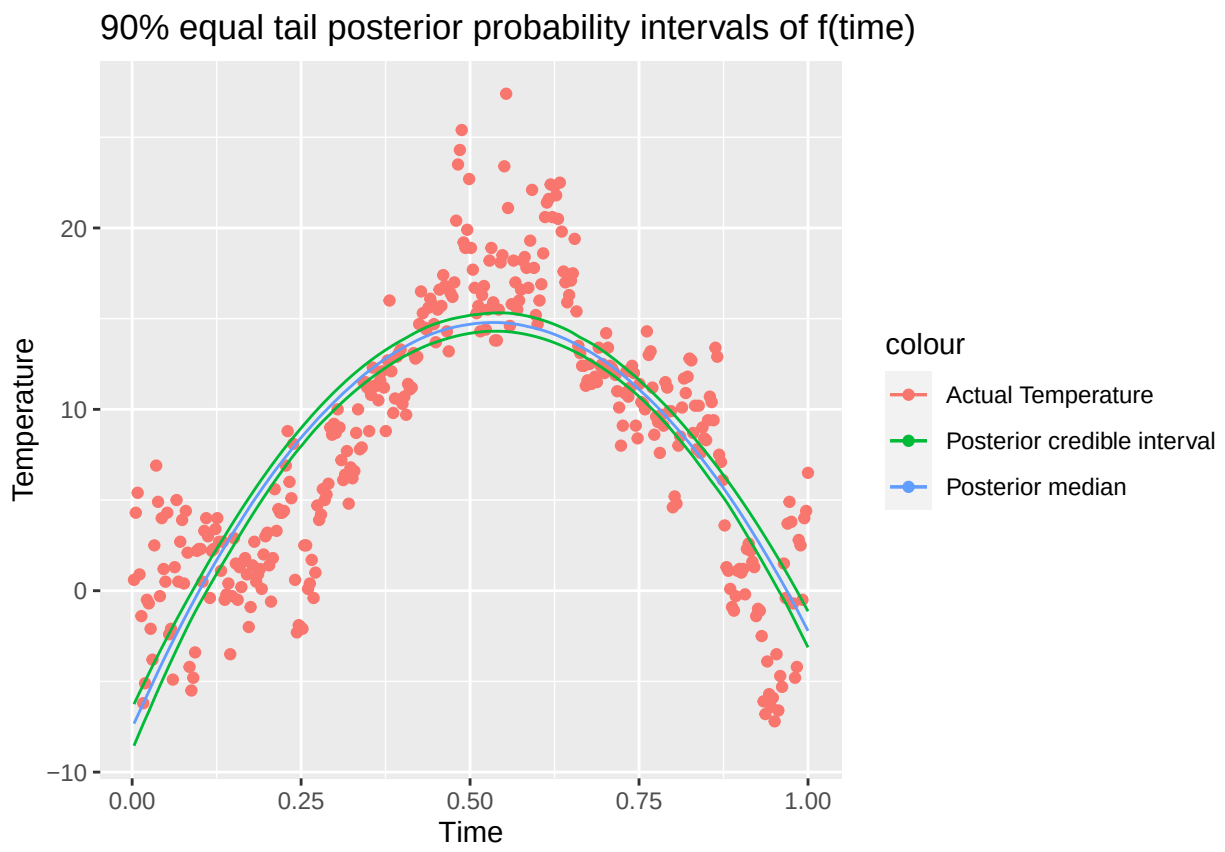

```r
hist(sample_beta[2,], breaks = 10, main = "beta_1")
```




```r
hist(sample_beta[3,], breaks = 10, main = "beta_2")
```




```
# requierments for 90% confidence interval

# y hat returns 365 rows and 100 column
# median for the predicted temperature (posterior median) for each time(row)
y_hat_median = apply(y_hat, 1, median)

# Upper and lower limit for the predicted temperature (y hat)
U_yhat =c()
L_yhat = c()
for (i in 1:365)
{
  xyz<-y_hat[i,]
  U_yhat = c(U_yhat, quantile(x= xyz, probs = 0.950))
  L_yhat = c(L_yhat, quantile(x= xyz, probs = 0.050))
}
# combining data for plot ( Time, actual temperature, median of the predicted temperature(y_hat))

df = data.frame("Time"=data$time, "Temperature"= data$temp, "Pred_Temp"= y_hat_median,
                "U_limit"= U_yhat, "L_limit"= L_yhat)


ggplot(data= df, aes(x = Time)) + geom_point(aes(y = Temperature, color = "Actual Temperature")) +
  geom_line(aes(y = Pred_Temp, color = "Posterior median")) +
  geom_line(aes(y = U_limit, color = "Posterior credible interval")) +
  geom_line(aes(y = L_limit, color = "Posterior credible interval")) +
  ggtitle("90% equal tail posterior probability intervals of f(time)")
```



The scatter plot displayed above represents the 90% posterior median's of the regression credible interval for f(time) and not a prediction band for the regression. Therefore, not all data points are contained within the interval.

task c)

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 $\widetilde{x}$. Use the simulated draws in (b) to simulate from the posterior distribution of $\widetilde{x}$. [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 $\widetilde{x}$.]

Using the quadratic regression, determine the time at which the highest expected temperature occurs.

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

By solving for time, we have:

$$\frac{df(time)}{dtime} = \beta_1 + 2\beta_2 \cdot time$$

Setting to zero, time will equal to:

$$0 = \beta_1 + 2\beta_2 \cdot time$$

$$time = -\frac{\beta_1}{2 \cdot \beta_2}$$

```r
cat(paste("The highest expected temperature is:", - (1/2) * mean(sample_beta[2,]/sample_beta[3,])))
```

```
## The highest expected temperature is: 0.534452925126825
```

Based on the observation of both the posterior probability interval plot above 90% and the calculated formula, it appears that the highest expected temperature is approximately 0.534452.

task d)

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. Just write down your prior. [Hint: the task is to specify μ_0 and Ω_0 in a suitable way.]

To prevent overfitting in higher order polynomial models, we recommend incorporating a regularization prior. This can be achieved by considering the following options:

Simple Ridge regression:

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

Or Lasso regression:

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

Question 2: Posterior approximation for classification with logistic regression

The dataset WomenAtWork.dat contains n = 132 observations on the following eight variables related to women:

Variable	Data type	Meaning	Role
Work	Binary	Whether or not the woman works	Response y
Constant	1	Constant to the intercept	Feature
HusbandInc	Numeric	Husband's income	Feature
EducYears	Counts	Years of education	Feature
ExpYears	Counts	Years of experience	Feature
Age	Counts	Age	Feature
NSmallChild	Counts	Number of child ≤ 6 years in household	Feature
NBigChild	Counts	Number of child > 6 years in household	Feature

task a)

Consider the logistic regression model:

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

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

$$\beta|y, X \sim N\left(\widetilde{\beta}, J_y^{-1}(\widetilde{\beta})\right)$$

where $\widetilde{\beta}$ is the posterior mode and$J(\widetilde{\beta}) = -\frac{\partial^2 lnp(\beta|y)}{\partial\beta\partial\beta^T}|_{\beta=\widetilde{\beta}}$ is the negative of the observed Hessian evaluated at the posterior mode. Note that $\frac{\partial^2 lnp(\beta|y)}{\partial\beta\partial\beta^T}$ is a 7×7 matrix with second derivatives on the diagonal and cross-derivatives $\frac{\partial^2 lnp(\beta|y)}{\partial\beta_i\partial\beta_j}$ on the offdiagonal.You can compute this derivative by hand, but we will let the computer do it numerically for you. Calculate both $\widetilde{\beta}$ and $J(\widetilde{\beta})$ by using the optim function in R.

-Prior is given as

$$\beta \sim N(0, \tau^2 I)$$

-Likelihood is calculated as (slide 4 in lecture 6):

$$p(y|X, \beta) = \prod_{i=1}^{n} y \cdot (x * \beta) - log(1 + exp(x * \beta))$$

-Posterior is given as

$$\beta|y, X \sim N\left(\widetilde{\beta}, J_y^{-1}(\widetilde{\beta})\right)$$

```
data_women <- read.table("WomenAtWork.dat" , header = T)
head(data_women)
```

```
##   Work Constant HusbandInc EducYears ExpYears Age NSmallChild NBigChild
## 1    1        1  22.394940        12        7  43           0         3
## 2    0        1   7.232000         8       10  34           0         7
## 3    1        1  18.271990        12        4  41           1         5
## 4    0        1  28.069000        14        2  43           0         2
## 5    1        1   7.799889        12       10  31           0         1
## 6    0        1  28.630000        16        6  37           0         3
```

```
glmModel <- glm(Work ~ 0 + ., data = data_women, family = binomial) # fit model
summary(glmModel)
```

```
##
## Call:
## glm(formula = Work ~ 0 + ., family = binomial, data = data_women)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3210  -0.9799   0.4423   0.9707   1.9131
##
## 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
```

```r
# Required inputs
tav <- 2 # Prior scaling factor such that Prior Covariance = (tav^2)*I
y <- as.matrix(data_women["Work"])
X <- as.matrix(data_women[,2:8])

nPara <- ncol(X)
# Setting up for the prior
mu_0 <- as.vector(rep(0,nPara)) # Prior mean vector
Sigma_0 <- tav^2*diag(nPara) # tav^2*I # Prior sigma
# compute the posterior
library("mvtnorm") # for dmvnorm function


posterior_logistic <- function(beta_Vector,y,X,mu,Sigma){
  nPara <- length(beta_Vector);
  line_prediction <- X%*%beta_Vector;
  # evaluating the log-likelihood
  log_Likeli <- sum( y * line_prediction -log(1 + exp(line_prediction)));
  if (abs(log_Likeli) == Inf) log_Likeli = -100000; # Likelihood is not finite,
  # stear the optimizer away from here!
  # evaluating the prior
  log_prior <- dmvnorm(beta_Vector, matrix(0,nPara,1), Sigma, log=TRUE);
  # add the log prior and log-likelihood together to get log posterior
  return(log_Likeli + log_prior)
}

# Optimization
initial_val <- as.vector(rep(0,nPara))
optim_val <- optim(par=initial_val,fn=posterior_logistic,gr=NULL,y,X=X,mu=mu_0,Sigma_0,
                   method="BFGS",control=list(fnscale=-1),hessian=TRUE)
# Results
posterior_Mode <- optim_val$par
posterior_Cov <- -solve(optim_val$hessian) # Posterior covariance matrix is -inv(Hessian)
posterior_SD = sqrt(diag(posterior_Cov)) # standard deviation
```

Result

$$\widetilde{\beta}$$

```
## Posterior mode:
```

```
## [1] -0.04036943 -0.03730689  0.17868950  0.12073637 -0.04618995 -1.47248930
## [7] -0.02014458
```

Result

$$J_y^{-1}(\widetilde{\beta})$$

```
## Inverse hessian:
```

```
##              [,1]          [,2]          [,3]          [,4]          [,5]
## [1,]  1.909882159  4.032517e-03 -6.280726e-02  1.041874e-03 -0.0257559994
## [2,]  0.004032517  4.833287e-04 -9.147892e-04 -2.666479e-05 -0.0000642848
## [3,] -0.062807260 -9.147892e-04  7.958354e-03  5.508998e-05 -0.0003181372
## [4,]  0.001041874 -2.666479e-05  5.508998e-05  1.112877e-03 -0.0002845111
## [5,] -0.025755999 -6.428480e-05 -3.181372e-04 -2.845111e-04  0.0007547741
## [6,] -0.137712005  1.585545e-03 -1.438778e-02 -1.336628e-03  0.0055481315
## [7,] -0.088876440  4.986972e-06  1.133513e-04  7.206537e-04  0.0010449347
##              [,6]          [,7]
## [1,] -0.137712005 -8.887644e-02
## [2,]  0.001585545  4.986972e-06
## [3,] -0.014387780  1.133513e-04
## [4,] -0.001336628  7.206537e-04
## [5,]  0.005548132  1.044935e-03
## [6,]  0.227975343  1.122711e-02
## [7,]  0.011227114  2.690243e-02
```

```
## Standard deviation:
```

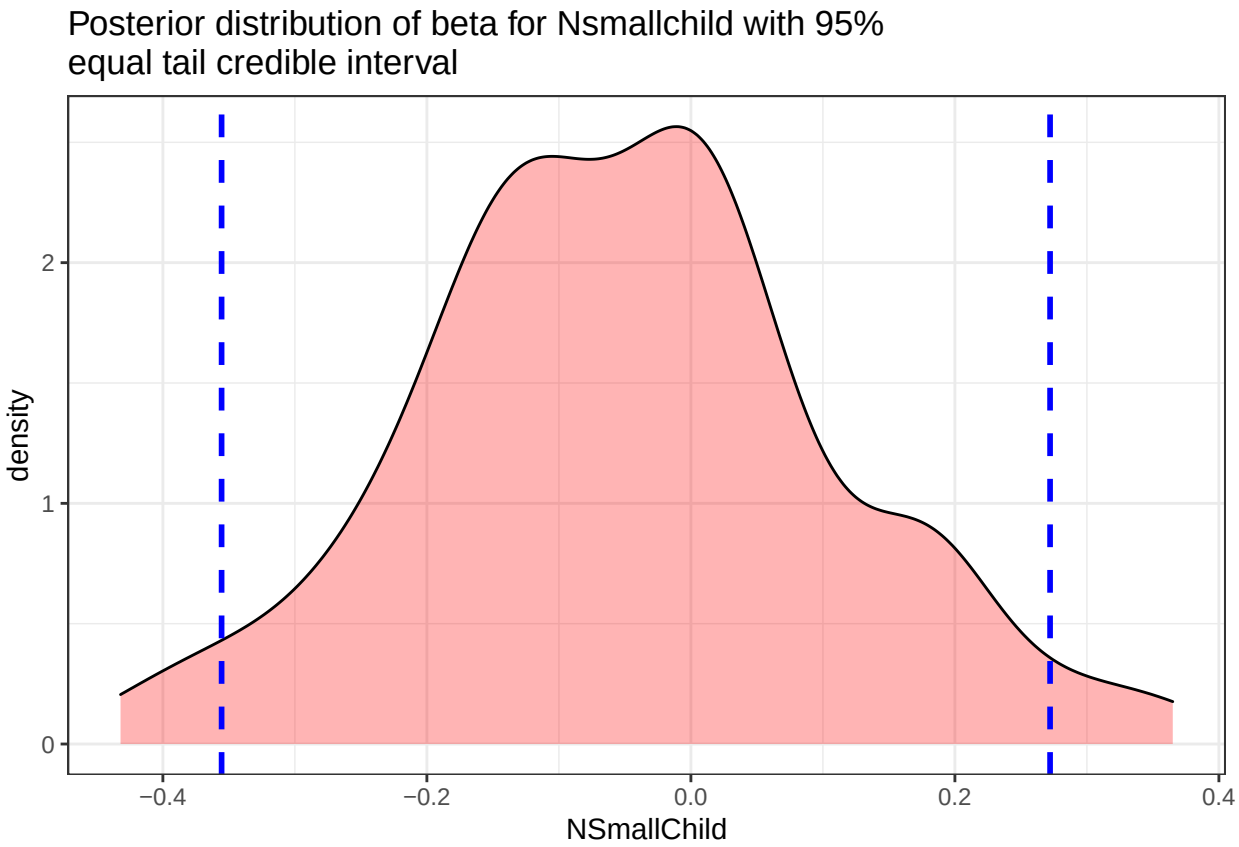
```
## [1] 1.38198486 0.02198474 0.08920960 0.03335982 0.02747315 0.47746764 0.16401959
```

```r
# 95% credible interval for NSmallChild variable
# simulate beta
beta_Vect <- rmvnorm(100, mean = posterior_Mode, sigma = posterior_Cov)
# Select NsmallChild (7th col)
NSmallChild <- beta_Vect[,7]
credible_interval <- quantile(NSmallChild, c(0.025, 0.975)) # # for a 95% credible interval
```

95% credible interval for the variable NSmallChild:

```
## 95% credible interval:
```

```
##       2.5%      97.5%
## -0.3555244  0.2721025
```



Based on the plot, we can infer that the feature NSmallChild significantly influences the likelihood of a woman being employed. The intervals appear to be negative and positive, and in our opinion, women with several small children are unlikely to work as they need to attend to their care.

Reference

1. https://stats.stackexchange.com/questions/162158/how-do-i-compute-the-posterior-predictive-distribution-of-a-logit-model

Appendix

```
knitr::opts_chunk$set(echo = TRUE)
set.seed(1234)
library(mvtnorm)
library(ggplot2)
library(reshape2) # melt function
library("readxl")

#Reading the excel data
data= read_excel("Linkoping2022.xlsx")
data$time <- (1:365)/ 365
```

```r
Y=data$temp
X=cbind("intercept"=1,"time_1"=data$time, "time_2"=data$time^2)

# Initial Parameters
mu_not = c(0, 100, -100)
omega_not = 0.01 * diag(1,3,3)
nu_not = 15
sigma_sq_not = 1
n=365 # number of observations
n_sample=10

# prior
x_prior = rchisq(10, nu_not)
sigma_prior = (nu_not * sigma_sq_not)/x_prior
sample_beta_prior=c()
for (draw in sigma_prior) {
  temp=rmvnorm(1, mean = mu_not, sigma = (draw * solve(omega_not)))
  sample_beta_prior=c(sample_beta_prior,temp)
}

# By transforming the beta samples into a 3x10 matrix, it becomes possible to utilize it for the purpos
sample_beta_prior=matrix(sample_beta_prior, 3,10)
y_hat_prior=X%*%sample_beta_prior

yhat_pr=melt(y_hat_prior)

# For every draw, a regression curve can be plotted.
ggplot()+
  geom_line(aes(x=rep(data$time, n_sample),color="Sampled y",
                y=yhat_pr$value,
                group=yhat_pr$Var2))+
  geom_line(aes(x=data$time, y=data$temp, color="Actual y"))+
  labs(title="TEMPERTURE vs TIME", x="TIME", y="TEMPERTURE")

library(mvtnorm)
library(ggplot2)
library(reshape2) # for melt function
set.seed(1234)

#Reading the excel data
data= read_excel("Linkoping2022.xlsx")
data$time <- (1:365)/ 365

Y=data$temp
X=cbind("intercept"=1,"time_1"=data$time, "time_2"=data$time^2)

# Initial Parameters
mu_not = c(0, 100, -100)
omega_not = 0.01 * diag(1,3,3)
nu_not = 1
sigma_sq_not = 1
n=365 # number of observations
n_sample=100
```

```r
# draw x form chi square posterior parameters
beta_hat=solve(t(X)%*%X)%*% t(X)%*%Y

mu_n=solve((t(X)%*% X)+omega_not)%*%((t(X)%*% X %*% beta_hat)+omega_not%*%mu_not)
omega_n=(t(X) %*% X)+omega_not
nu_n=nu_not+n
sigma_sq_n=(nu_not*sigma_sq_not+(t(Y)%*%Y+t(mu_not)%*%omega_not%*%mu_not)-
              t(mu_n)%*%omega_n%*%mu_n)/(nu_n)

# draws (samples) for beta and sigma square posteriors:

# sigma posterior
# darwn the x from chi square
x = rchisq(n = 100, df = nu_n)
sample_sigma_sq= suppressWarnings((nu_n*sigma_sq_n)/x)

# beta posterior
sample_beta=c()
for (draw in sample_sigma_sq) {
  temp=rmvnorm(1, mean = mu_n, sigma = (draw * solve(omega_n)))
  sample_beta=c(sample_beta,temp)
}

sample_beta=matrix(sample_beta, 3,100)
y_hat=X%*%sample_beta

# melt for plot
y_hat_1=melt(y_hat)

# plot of the regression curve for each draw
ggplot()+
  geom_line(aes(x=rep(data$time, n_sample),color="Sampled y",
                y=y_hat_1$value,
                group=y_hat_1$Var2))+
  geom_line(aes(x=data$time, y=data$temp, color="Actual y"))+
  labs(title="Temperature vs time", x="time", y="Temperature")


# Marginal posteriors
hist(sample_sigma_sq, breaks = 10, main = "sigma")
hist(sample_beta[1,], breaks = 10, main = "beta_0")
hist(sample_beta[2,], breaks = 10, main = "beta_1")
hist(sample_beta[3,], breaks = 10, main = "beta_2")


# requierments for 90% confidence interval

# y hat returns 365 rows and 100 column
# median for the predicted temperature (posterior median) for each time(row)
y_hat_median = apply(y_hat, 1, median)

# Upper and lower limit for the predicted temperature (y hat)
U_yhat =c()
```

```r
L_yhat = c()
for (i in 1:365)
{
  xyz<-y_hat[i,]
  U_yhat = c(U_yhat, quantile(x= xyz, probs = 0.950))
  L_yhat = c(L_yhat, quantile(x= xyz, probs = 0.050))
}
# combining data for plot ( Time, actual temperature, median of the predicted temperature(y_hat))

df = data.frame("Time"=data$time, "Temperature"= data$temp, "Pred_Temp"= y_hat_median,
                "U_limit"= U_yhat, "L_limit"= L_yhat)


ggplot(data= df, aes(x = Time)) + geom_point(aes(y = Temperature, color = "Actual Temperature")) +
  geom_line(aes(y = Pred_Temp, color = "Posterior median")) +
  geom_line(aes(y = U_limit, color = "Posterior credible interval")) +
  geom_line(aes(y = L_limit, color = "Posterior credible interval")) +
  ggtitle("90% equal tail posterior probability intervals of f(time)")

cat(paste("The highest expected temperature is:", - (1/2) * mean(sample_beta[2,]/sample_beta[3,])))
knitr::include_graphics("1.PNG")
data_women <- read.table("WomenAtWork.dat" , header = T)
head(data_women)

glmModel <- glm(Work ~ 0 + ., data = data_women, family = binomial) # fit model
summary(glmModel)

# Required inputs
tav <- 2 # Prior scaling factor such that Prior Covariance = (tav^2)*I
y <- as.matrix(data_women["Work"])
X <- as.matrix(data_women[,2:8])

nPara <- ncol(X)
# Setting up for the prior
mu_0 <- as.vector(rep(0,nPara)) # Prior mean vector
Sigma_0 <- tav^2*diag(nPara) # tav^2*I # Prior sigma
# compute the posterior
library("mvtnorm") # for dmvnorm function


posterior_logistic <- function(beta_Vector,y,X,mu,Sigma){
  nPara <- length(beta_Vector);
  line_prediction <- X%*%beta_Vector;
  # evaluating the log-likelihood
  log_Likeli <- sum( y * line_prediction -log(1 + exp(line_prediction)));
  if (abs(log_Likeli) == Inf) log_Likeli = -100000; # Likelihood is not finite,
  # stear the optimizer away from here!
  # evaluating the prior
  log_prior <- dmvnorm(beta_Vector, matrix(0,nPara,1), Sigma, log=TRUE);
  # add the log prior and log-likelihood together to get log posterior
  return(log_Likeli + log_prior)
}
```

```r
# Optimization
initial_val <- as.vector(rep(0,nPara))
optim_val <- optim(par=initial_val,fn=posterior_logistic,gr=NULL,y,X=X,mu=mu_0,Sigma_0,
                   method="BFGS",control=list(fnscale=-1),hessian=TRUE)
# Results
posterior_Mode <- optim_val$par
posterior_Cov <- -solve(optim_val$hessian) # Posterior covariance matrix is -inv(Hessian)
posterior_SD = sqrt(diag(posterior_Cov)) # standard deviation

cat("Posterior mode:","\n")
posterior_Mode

cat("Inverse hessian:","\n")
posterior_Cov

cat("Standard deviation:","\n")
posterior_SD
# 95% credible interval for NSmallChild variable
# simulate beta
beta_Vect <- rmvnorm(100, mean = posterior_Mode, sigma = posterior_Cov)
# Select NsmallChild (7th col)
NSmallChild <- beta_Vect[,7]
credible_interval <- quantile(NSmallChild, c(0.025, 0.975)) # # for a 95% credible interval
cat("95% credible interval:","\n")
credible_interval
# plot intervals

ggplot()+geom_density(aes(NSmallChild), fill="red", alpha=0.3)+
  geom_vline(aes(xintercept=credible_interval[1]),colour="blue", linetype="dashed",size=1)+
  geom_vline(aes(xintercept=credible_interval[2]),colour="blue", linetype="dashed",size=1)+
  labs(title="Posterior distribution of beta for Nsmallchild with 95%
equal tail credible interval")+theme_bw()
```