Lab Report:Lab1-Group8

Udaya Shanker Mohanan Nair(udamo524), Uday Jain(udaja983)

2025-05-05

Assignment 1 - Linear and polynomial regression

The dataset `temp_linkoping.csv` contains daily average temperatures (in degree Celsius) in Linköping between July 2023 and June 2024. Import the dataset in R. The response variable is *temp* and the covariate *time*, which is defined as:

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

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

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

Part A

Use the conjugate prior for the linear regression model. The prior hyperparameters μ_0, Ω_0, ν_0 and σ_0^2 shall be set to sensible values. Start with:

- $\mu_0 = (0, -100, 100)^T$
- $\Omega_0 = 0.01 \cdot I_3$
- $\nu_0 = 1$
- $\sigma_0^2 = 1$

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 $Scaled - Inv - \chi^2$ simulator of random draws from Lab 1.

Firstly, we imported the dataset and calculated the time for each date given in the dataset.

```
library(mvtnorm)
data <- read.csv("temp_linkoping.csv")
data$time <- as.numeric(as.Date(data$date) - as.numeric(as.Date("2023-07-01"))) / 365
time_values <- seq(min(data$time), max(data$time), length=100)
```

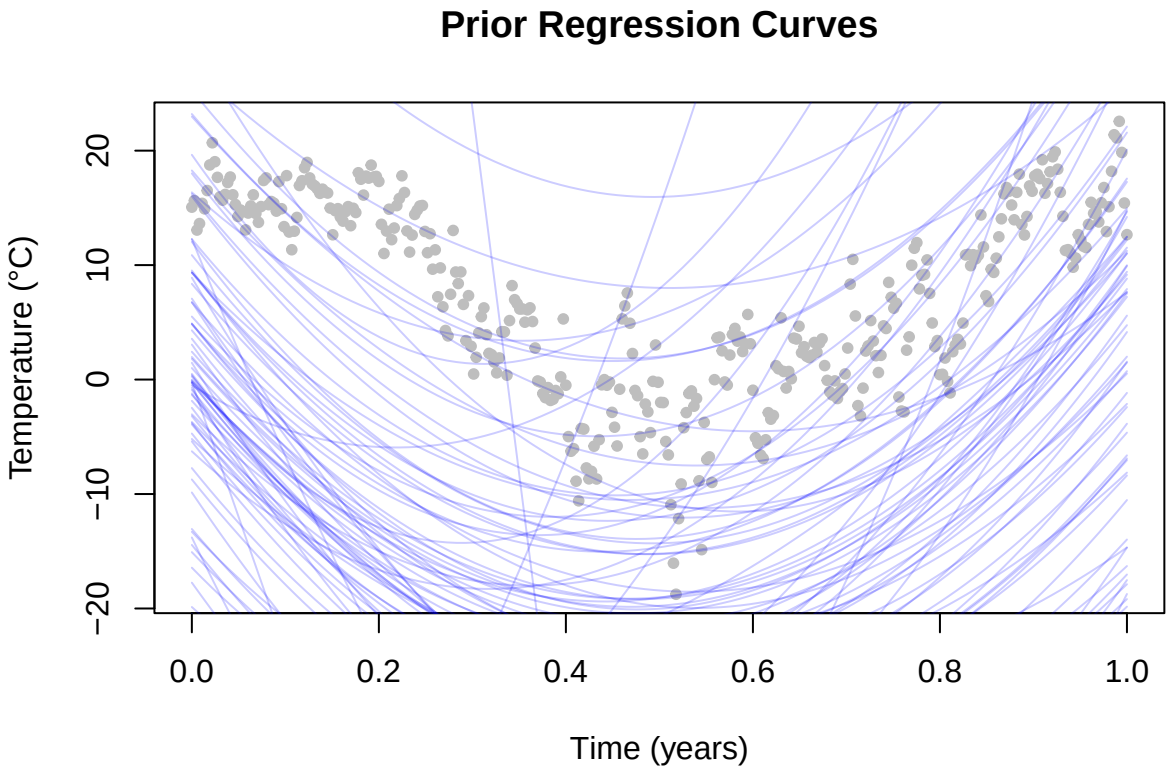
Now, we created a function to calculate conjugate prior

```r
conjugate_prior <- function(mu, omega, vu, sigma) {
  sigma_square <- sigma * vu / rchisq(100, df=vu)
  beta <- matrix(NA, nrow=100, ncol=3)
  return(list(res_beta=beta, res_sigma=sigma_square))
}
```

Initial values of prior hyper parameters are the following:

$$\mu_0 = (0, -100, 100)$$
$$\Omega_0 = 0.01.I_3$$
$$\nu_0 = 1$$
$$\sigma_0^2 = 1$$

```r
mu <- c(0, -100, 100)
omega <- 0.01 * diag(3)
vu <- 1
sigma <- 1
prior <- conjugate_prior(mu, omega, vu, sigma)
plot(data$time, data$temp, main="Prior Regression Curves",
       xlab="Time (years)", ylab="Temperature (°C)", pch=20, col="gray")
for (value in 1:length(prior$res_sigma)) {
  prior$res_beta[value,] <- rmvnorm(1, mean=mu, sigma=prior$res_sigma[value] * solve(omega))
  y_values <- prior$res_beta[value,1] + prior$res_beta[value,2]*time_values + prior$res_beta[value,3]*t
  lines(time_values, y_values, col=rgb(0,0,1,0.2))
}
```

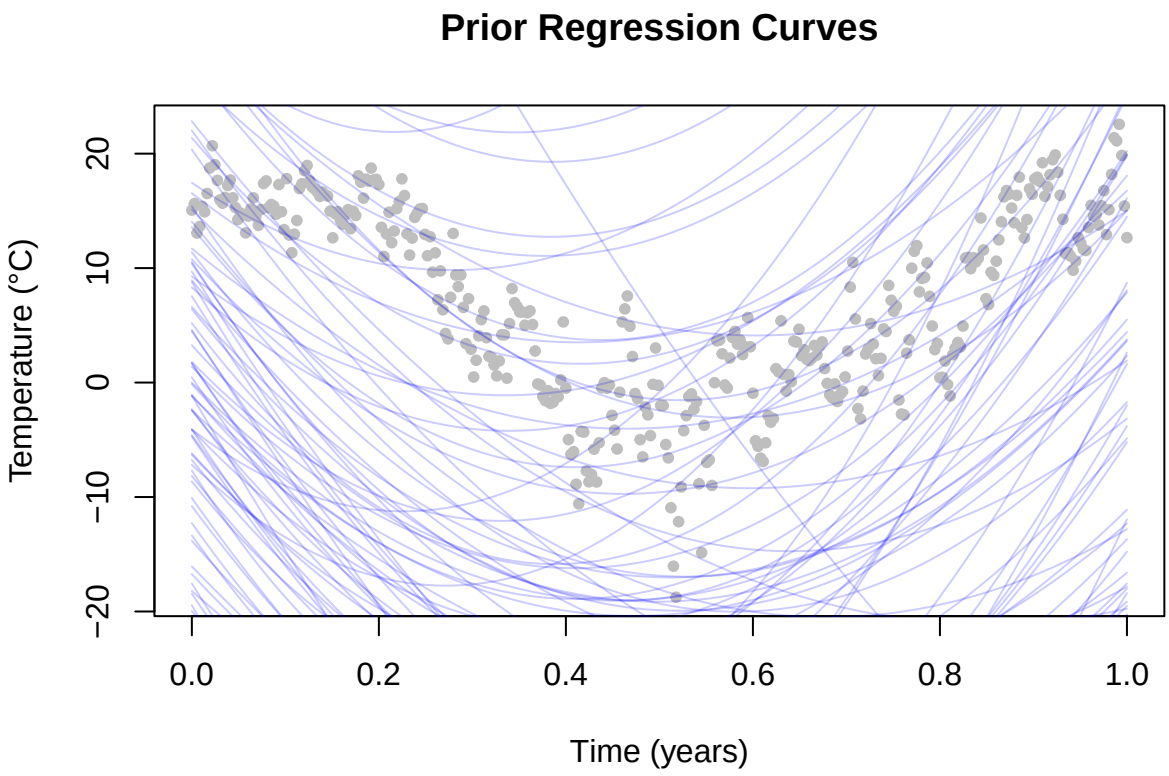


Here the prior regression curves for the initial parameters are unrealistic for the given temperature data as the curve shows an extreme and unconvincing behavior with temperature swinging between -100°C and +100°C because of large coefficients and weak prior precision.

Now adjusting the prior hyper parameters to the following values:

$$\mu_0 = (0, -100, 100)$$
$$\Omega_0 = 0.01.I_3$$
$$\nu_0 = 1$$
$$\sigma_0^2 = 1$$

```
new_mu <- c(10, 0, -10)
new_omega <- 0.001 * diag(3)
new_vu <- 5
new_sigma <- 5
new_prior <- conjugate_prior(new_mu, new_omega, new_vu, new_sigma)
plot(data$time, data$temp, main="Prior Regression Curves",
       xlab="Time (years)", ylab="Temperature (°C)", pch=20, col="gray")
for (value in 1:length(new_prior$res_sigma)) {
  new_prior$res_beta[value,] <- rmvnorm(1, mean=mu, sigma=new_prior$res_sigma[value] * solve(omega))
  y_values <- new_prior$res_beta[value,1] + new_prior$res_beta[value,2]*time_values + new_prior$res_beta
  lines(time_values, y_values, col=rgb(0,0,1,0.2))
}
```



Part B

Write a function that simulates draws from the joint posterior distribution of β_0, β_1, β_2 and σ^2.

In this part we implemented a function that simulate draws from the joint posterior distribution of β_0, β_1, β_2 and σ^2.

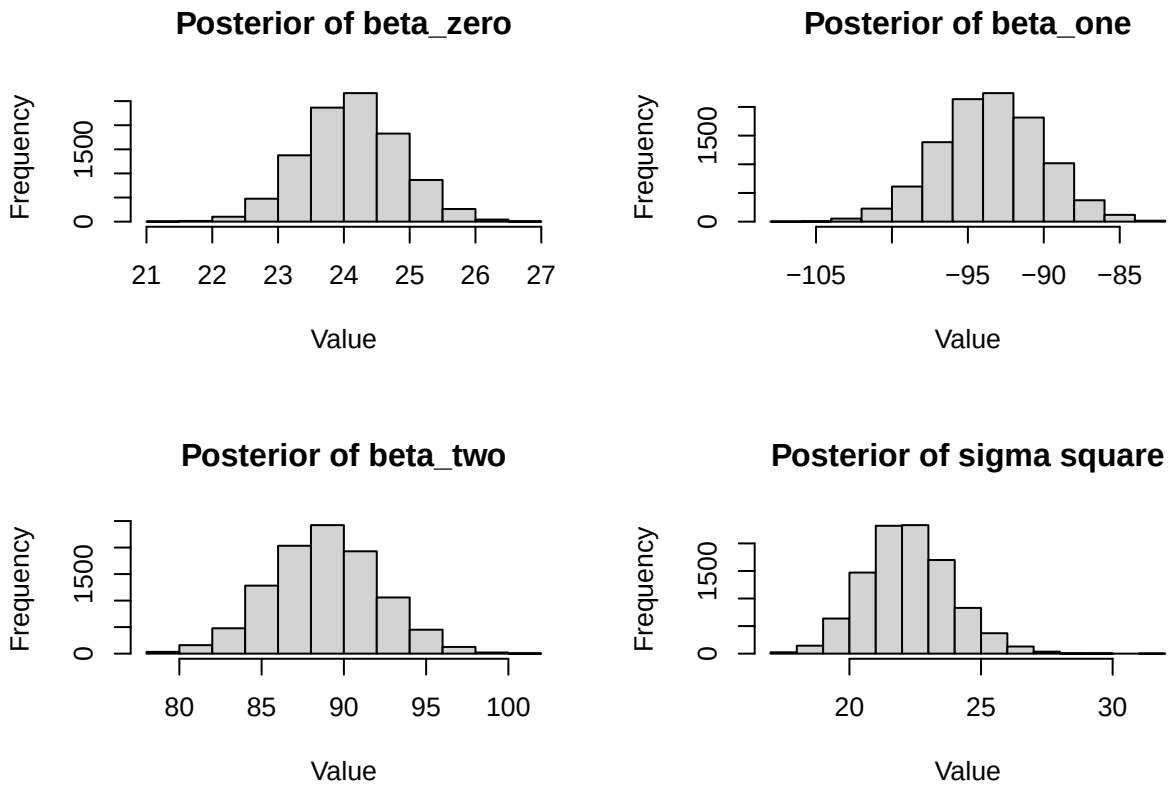
```
simulate_posterior <- function(y, x, mu, omega, vu, sigma) {
  new_omega_value <- t(x) %*% x + omega
  new_mu_value <- solve(new_omega_value) %*% (t(x) %*% y + omega %*% mu)
  new_vu_value <- vu + length(y)
  new_sigma_value <- (vu * sigma + sum(y^2) +
                        t(mu) %*% omega %*% mu -
                        t(new_mu_value) %*% new_omega_value %*% new_mu_value) / new_vu_value

  new_sigma_square <- as.numeric(new_sigma_value) * new_vu_value / rchisq(10000, df=new_vu_value)
  posterior_beta <- matrix(NA, nrow=10000, ncol=3)
  for (i in 1:10000) {
    posterior_beta[i, ] <- rmvnorm(1, mean=as.vector(new_mu_value),
                                   sigma=new_sigma_square[i] * solve(new_omega_value))
  }
  list(sim_beta = posterior_beta,sim_sigma = new_sigma_square)
}
```

Here for the execution of the function we are using the same values that we initially used for prior hyper parameters.

```
x <- cbind(1, data$time, data$time^2)
y <- data$temp

posterior <- simulate_posterior(y,x,mu,omega,vu,sigma)
```

Point(i) Plot a histogram for each marginal posterior of the parameters.

```
par(mfrow = c(2, 2))
hist(posterior$sim_beta[,1], main="Posterior of beta_zero", xlab="Value")
hist(posterior$sim_beta[,2], main="Posterior of beta_one", xlab="Value")
hist(posterior$sim_beta[,3], main="Posterior of beta_two", xlab="Value")
hist(posterior$sim_sigma, main="Posterior of sigma square", xlab="Value")
```

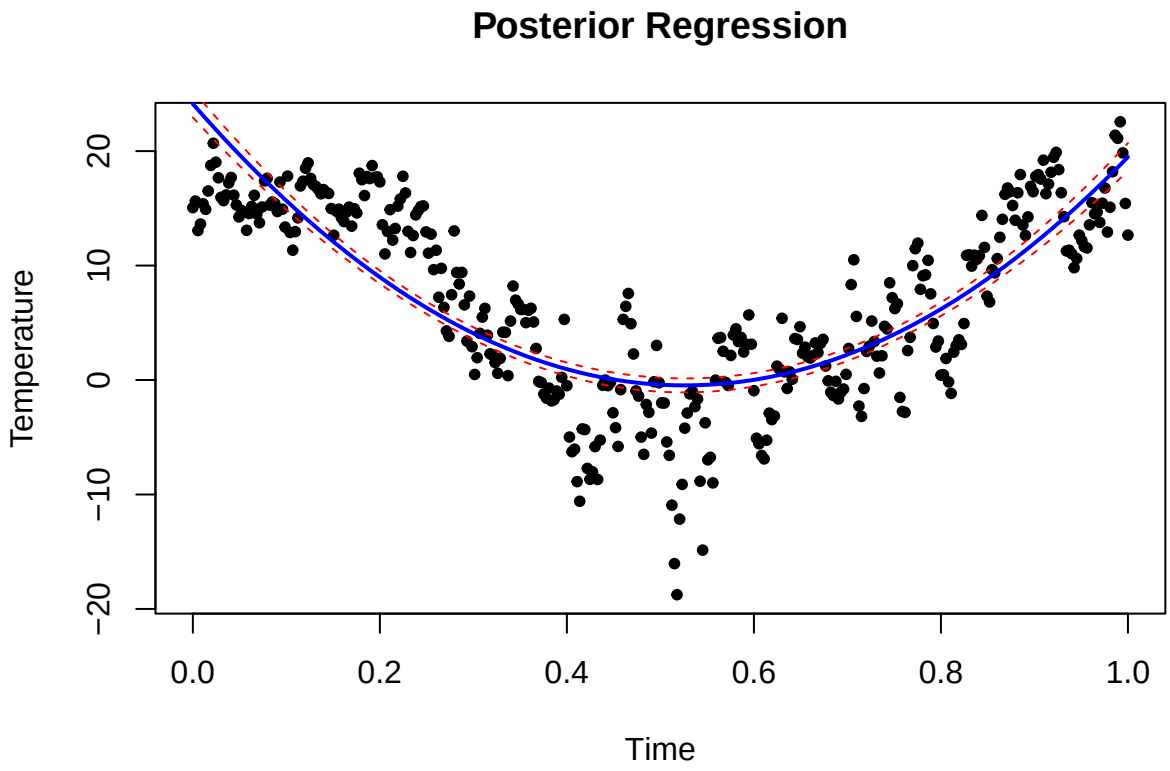



```
par(mfrow = c(1, 1))
```

Point(ii) Make a scatter plot of the temperature data and overlay: - A curve for the posterior median of the regression function $f(time) = E[temp|time] = \beta_0 + \beta_1 \cdot time + \beta_2 \cdot time^2$ - Curves for the 90% equal tail posterior probability intervals of $f(time)$ (the 5th and 95th posterior percentiles) Does the posterior probability interval contain most of the data points? Should they?

```
posterior_predictions <- matrix(NA, nrow=10000, ncol=length(time_values))
for (i in 1:10000) {
  posterior_predictions[i,] <- posterior$sim_beta[i,1] + posterior$sim_beta[i,2]*time_values +
```

```
    posterior$sim_beta[i,3]*time_values^2
}

prediction_median <- apply(posterior_predictions, 2, median)
prediction_lower <- apply(posterior_predictions, 2, quantile, 0.05)
prediction_upper <- apply(posterior_predictions, 2, quantile, 0.95)

plot(data$time, data$temp, main="Posterior Regression",
     xlab="Time", ylab="Temperature", pch=20)
lines(time_values, prediction_median, col="blue", lwd=2)
lines(time_values, prediction_lower, col="red", lty=2)
lines(time_values, prediction_upper, col="red", lty=2)
```



From the plot, the 90% posterior probability interval contains most of the temperature data points. However there are very few points that fell outside the bands especially at the extremes of the time range.

Part C

It is of interest to locate the time with the lowest expected temperature (i.e., the time where $f(time)$ is minimal). Let's call this value $\tilde{x}$. Use the simulated draws from (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 $\tilde{x}$.

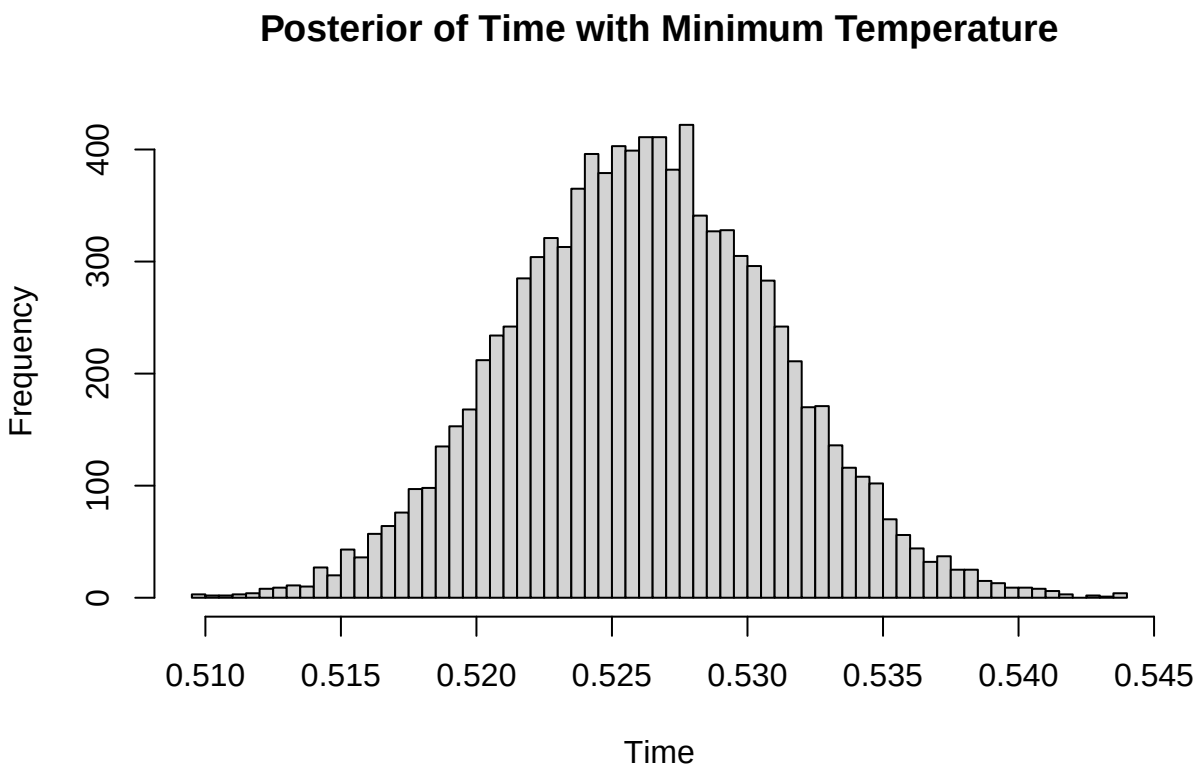
For the given quadratic regression model

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

minimum occurs at $\bar{x}$.

$$\bar{x} = -\beta_1/(2 * \beta_2)$$

```
min_time <- -posterior$sim_beta[,2] / (2 * posterior$sim_beta[,3])
hist(min_time, main="Posterior of Time with Minimum Temperature",
     xlab="Time", breaks=50)
```



Part D

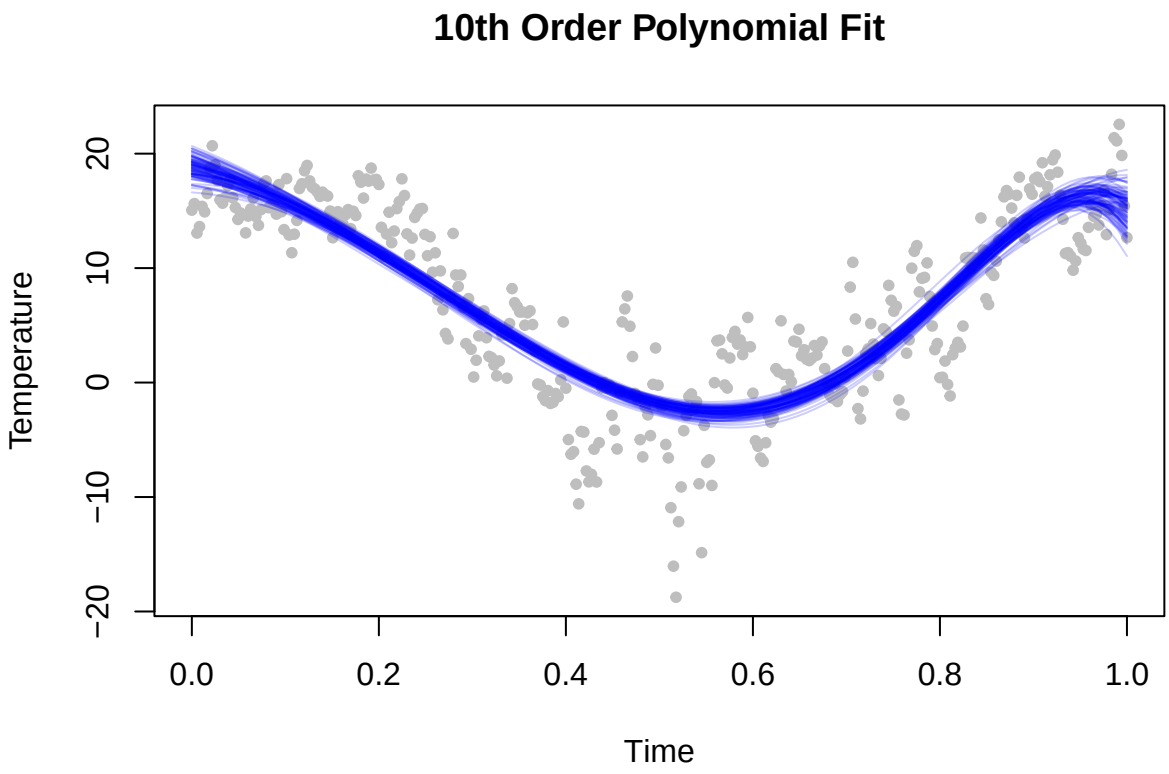
Now consider estimating a polynomial regression of order 10, but you suspect that higher order terms may not be needed, and you worry about overfitting the data.

1. Suggest a suitable prior that mitigates this potential problem and motivate your choice.
2. Repeat task (a) for the selected prior and the higher order polynomial to see if it behaves as expected.

Hint: The task is to specify μ_0 and Ω_0 in a suitable way.

In this part, we extended the quadratic model to a 10th-order polynomial regression and then proposed a hierarchical prior to shrink higher-order coefficients and then compared the posterior behavior against the quadratic model.

```r
order <- 10
mu_poly <- rep(0, order+1)
omega_poly <- diag(order+1) * 0.01

# Create design matrix
x_poly <- matrix(1, nrow=length(y), ncol=order+1)
for (j in 1:order) {
  x_poly[,j+1] <- data$time^j
}

# Posterior calculations
omega_poly_post <- t(x_poly) %*% x_poly + omega_poly
mu_poly_post <- solve(omega_poly_post) %*% (t(x_poly) %*% y + omega_poly %*% mu_poly)
vu_poly_post <- new_vu + length(y)
sigma_poly_post <- (new_vu * new_sigma + sum(y^2) +
                      t(mu_poly) %*% omega_poly %*% mu_poly -
                      t(mu_poly_post) %*% omega_poly_post %*% mu_poly_post) / vu_poly_post

# Simulate from posterior
sigma_poly_sim <- as.numeric(sigma_poly_post) * vu_poly_post / rchisq(100, df=vu_poly_post)
beta_poly <- matrix(NA, nrow=100, ncol=order+1)
for (i in 1:100) {
  beta_poly[i,] <- rmvnorm(1, mean=as.vector(mu_poly_post),
                           sigma=sigma_poly_sim[i] * solve(omega_poly_post))
}

# Plot prior curves
x_poly_new <- matrix(1, nrow=length(time_values), ncol=order+1)
for (j in 1:order) {
  x_poly_new[,j+1] <- time_values^j
}

plot(data$time, data$temp, main="10th Order Polynomial Fit",
     xlab="Time", ylab="Temperature", pch=20, col="gray")
for (i in 1:100) {
  y_poly <- x_poly_new %*% beta_poly[i,]
  lines(time_values, y_poly, col=rgb(0,0,1,0.2))
}
```



Assignment 2 - Posterior approximation for classification with logistic regression

The dataset `Disease.csv` contains $n = 313$ observations on the following six variables related to a certain disease:

Variable	Data type	Meaning	Role
`Class_of_diagnosis`	Binary	Disease (1) or not (0)	Response y
`Gender`	Binary	Woman (1) or Male (0)	Feature
`Age`	Counts	Age	Feature
`Duration_of_symptoms`	Numeric	Duration of symptoms in days	Feature
`Dyspnoea`	Binary	1 if person has laboured breathing	Feature
`White_blood`	Counts	White blood cells per microliter	Feature

Part (a) Logistic Regression Model and Posterior Approximation

Consider the logistic regression model:

$$\Pr(y = 1|\mathbf{x}, \beta) = \frac{\exp\left(\mathbf{x}^T\beta\right)}{1 + \exp\left(\mathbf{x}^T\beta\right)},$$

where y equals 1 if the person has a disease and 0 if not. $\mathbf{x}$ is a 6-dimensional vector containing five features and a column of 1's to include an intercept in the model. The values of the variables `Age`, `Duration_of_symptoms`, and `White_blood` in $\mathbf{x}$ need to be standardized to mean 0 and variance 1. For each of these variables x_j, calculate the standardized value for each observation i by using the formula:

$$\frac{x_{ij} - \bar{x}_j}{s_{x_j}},$$

where $\bar{x}_j$ and s_{x_j} are the sample mean and standard deviation of x_j, respectively.

The goal is to approximate the posterior distribution of the parameter vector β with a multivariate normal distribution:

$$\beta|\mathbf{y}, \mathbf{x} \sim N\left(\hat{\beta}, J_{\mathbf{y}}^{-1}(\hat{\beta})\right),$$

where $\hat{\beta}$ is the posterior mode and $J(\hat{\beta}) = -\frac{\partial^2 \ln j(\hat{\beta})\mathcal{O}}{\partial\beta\partial\hat{\beta}}\big|_{\beta=\hat{\beta}}$ is the negative of the observed Hessian evaluated at the posterior mode. Note that $\frac{\partial^2 \ln j(\hat{\beta})\mathcal{O}}{\partial\beta\partial\hat{\beta}}$ is a 6×6 matrix with second derivatives on the diagonal and cross-derivatives $\frac{\partial^2 \ln J(\hat{\beta})\mathcal{O}}{\partial\beta\partial\hat{\beta}}$ on the off-diagonal. You can compute this derivative by hand, but we will let the computer do it numerically for you. Calculate both $\hat{\beta}$ and $J(\hat{\beta})$ by using the `optim` function in R.

Hint: You may use code snippets from the demo of logistic regression in Lecture 6. Use the prior $\beta \sim \mathcal{N}(0, \tau^2 I)$, where $\tau = 2$.

Tasks:

1. Present the numerical values of $\hat{\beta}$ and $J_{\chi}^{-1}(\hat{\beta})$ for the Disease data.
2. Compute an approximate 95% equal tail posterior probability interval for the regression coefficient corresponding to the variable `Age`.
3. Would you say that this feature is of importance for the probability that a person has the disease?

Hint: You can verify that your estimation results are reasonable by comparing the posterior means to the maximum likelihood estimates, given by:

```
glmModel <- glm(Class_of_diagnosis ~ 0 + ., data = Disease, family = binomial)
```

First we standardize the attributes age, duration_of_symptoms and white blood and combined with the original data.

```
library(mvtnorm)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
# Part A
disease_data <- read.csv("Disease.csv")

x <- cbind(
  Intercept = 1,
  age_new = (disease_data$age - mean(disease_data$age))/sd(disease_data$age),
  gender = disease_data$gender,
  duration_new = (disease_data$duration_of_symptoms - mean(disease_data$duration_of_symptoms))/sd(diseas
  dyspnoea = disease_data$dyspnoea,
  white_Blood_new = (disease_data$white_blood - mean(disease_data$white_blood))/sd(disease_data$white_bl
)

y <- disease_data$class_of_diagnosis

posterior_function <- function(beta, x, y) {
  linear_predictor <- x %*% beta
  log_likelihood <- sum(y * linear_predictor - log(1 + exp(linear_predictor)))
  log_prior <- sum(dnorm(beta, mean = 0, sd = 2, log = TRUE))
  return(log_likelihood + log_prior)
}

optim_res <- optim(par = rep(0, ncol(x)),
                   fn = posterior_function,
                   x = x,
                   y = y,
                   control = list(fnscale = -1),
                   method = "BFGS",
                   hessian = TRUE)

posterior_mode <- optim_res$par
negative_hessian <- -optim_res$hessian
covariance_matrix <- solve(negative_hessian)

coefficient_age <- posterior_mode[2]
standard_error_age <- sqrt(covariance_matrix[2, 2])
ci_age <- coefficient_age + c(-1.96, 1.96) * standard_error_age
```

```r
cat("\n95% Posterior interval for coefficient of age:", ci_age)
```

```
##
## 95% Posterior interval for coefficient of age: -0.5170715 -0.01522428
```

```r
disease_data <- disease_data %>%
  mutate(age_new = (age - mean(age))/sd(age),
         duration_new = (duration_of_symptoms - mean(duration_of_symptoms))/sd(duration_of_symptoms),
         white_blood_new = (white_blood - mean(white_blood))/sd(white_blood))

glm_model <- glm(class_of_diagnosis ~ age_new + gender + duration_new + dyspnoea + white_blood_new,
                 data = disease_data, family = binomial)

print("MLE from glm:")
```

```
## [1] "MLE from glm:"
```

```r
print(coef(glm_model))
```

```
##     (Intercept)         age_new          gender    duration_new        dyspnoea
##     -0.38960953     -0.26762162     -0.64938471      0.19367028     -0.13347816
## white_blood_new
##     -0.04504174
```

```r
print("Posterior mode Values:")
```

```
## [1] "Posterior mode Values:"
```

```r
print(posterior_mode)
```

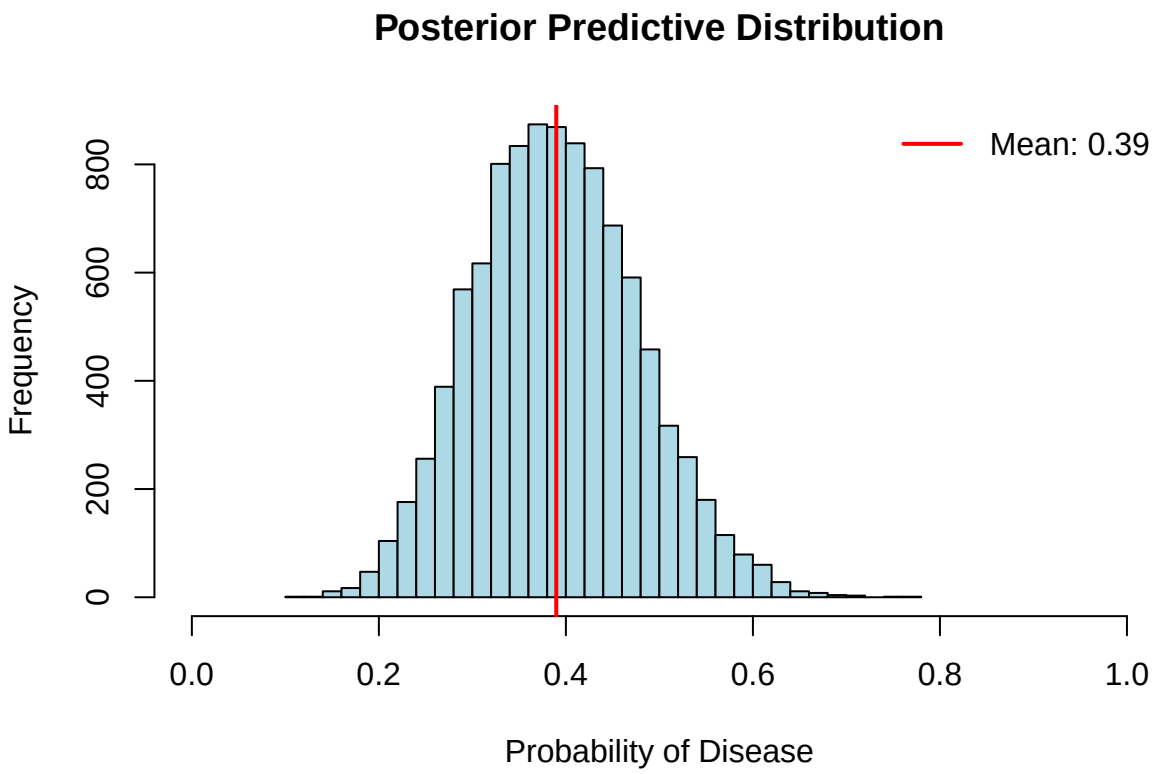
```
## [1] -0.38824219 -0.26614789 -0.64238957  0.19312998 -0.13752407 -0.04498936
```

The values are nearly identical, confirming the Bayesian approximation is reasonable

Part B

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 certain diagnosis method was used and where the values of x corresponds to a 38-year-old woman, with 10 days of symptoms, no laboured breathing and 11000 white blood cells per microliter. Note that the corresponding standardized values need to be calculated for the variables Age, Duration_of_symptoms, and White_blood in x by using the formula in (a). Plot the posterior predictive distribution of Pr(y = 1|x) for this person. [Hints: The R package mvtnorm will be useful. Remember that Pr(y = 1|x) can be calculated for each posterior draw of β.]

```r
posterior_predictive_distribution <- function(x_values) {
  beta_values <- rmvnorm(10000, mean = posterior_mode, sigma = covariance_matrix)
  probabilities <- as.vector(plogis(x_values %*% t(beta_values)))
  return(probabilities)
}

new_age <- (38 - mean(disease_data$age)) / sd(disease_data$age)
new_duration <- (10 - mean(disease_data$duration_of_symptoms)) / sd(disease_data$duration_of_symptoms)
new_white <- (11000 - mean(disease_data$white_blood)) / sd(disease_data$white_blood)

x_values <- c(1, new_age, 1, new_duration, 0, new_white)

posterior_predictive_values <- posterior_predictive_distribution(x_values)

hist(posterior_predictive_values, breaks = 30, col = "lightblue",
     main = "Posterior Predictive Distribution",
     xlab = "Probability of Disease",
     xlim = c(0, 1))
abline(v = mean(posterior_predictive_values), col = "red", lwd = 2)
legend("topright", legend = paste("Mean:", round(mean(posterior_predictive_values), 3)),
       col = "red", lwd = 2, bty = "n")
```




```
cat("\nSummary of posterior predictive distribution:\n")
```

```
##
## Summary of posterior predictive distribution:
```

```
print(summary(posterior_predictive_values))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.1142  0.3277  0.3874  0.3897  0.4479  0.7784
```

```
cat("Mean:", mean(posterior_predictive_values))
```

```
## Mean: 0.3897308
```

```
cat("95% interval")
```

```
## 95% interval
```

```
print(quantile(posterior_predictive_values, c(0.025, 0.975)))
```

```
##      2.5%     97.5%
## 0.2284295 0.5681827
```