Bayesian Learning - Computer Lab 2

Stefano Toffol (steto820) and Nahid Farazmand (nahfa911)

01 May, 2019

Question 1 - Linear and polynomial regression

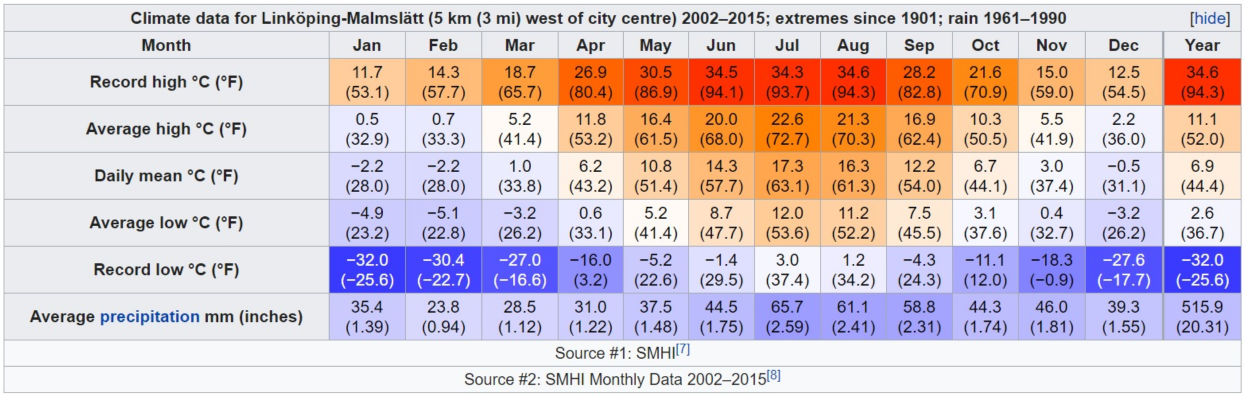
We are asked to analyze the dataset of daily temperatures (in Celsius degrees) in the city of Linköping during the year 2016. The total amount of observations are 366 (one per each day of the year) and there are two available variables:

- *temp*, the response variable itself;
- *time*, a value within the interval $(0, 1]$ and given by the formula $time = \frac{\text{\# days since the beginning of the year}}{366}$

We are asked to perform a Bayesian analysis for the following quadratic regression:

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

To have an idea of what to expect as results from our analysis, we made a brief research and from the Wikipedia page we found a table containing the various average monthly temperatures recorded in a location close to the city center over the years 2002-2015 (Figure 1). If we look at the row of the daily mean temperature we can see how these range between $-2.2\ °C$ (January/February) and $+17.3\ °C$ (July). Considering that our model describe the average behaviour during the year, we would like our prior to describe a similar trend.



Climate data for Linköping-Malmslätt (5 km (3 mi) west of city centre) 2002–2015; extremes since 1901; rain 1961–1990 [hide]

Month	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Year
Record high °C (°F)	11.7 (53.1)	14.3 (57.7)	18.7 (65.7)	26.9 (80.4)	30.5 (86.9)	34.5 (94.1)	34.3 (93.7)	34.6 (94.3)	28.2 (82.8)	21.6 (70.9)	15.0 (59.0)	12.5 (54.5)	34.6 (94.3)
Average high °C (°F)	0.5 (32.9)	0.7 (33.3)	5.2 (41.4)	11.8 (53.2)	16.4 (61.5)	20.0 (68.0)	22.6 (72.7)	21.3 (70.3)	16.9 (62.4)	10.3 (50.5)	5.5 (41.9)	2.2 (36.0)	11.1 (52.0)
Daily mean °C (°F)	−2.2 (28.0)	−2.2 (28.0)	1.0 (33.8)	6.2 (43.2)	10.8 (51.4)	14.3 (57.7)	17.3 (63.1)	16.3 (61.3)	12.2 (54.0)	6.7 (44.1)	3.0 (37.4)	−0.5 (31.1)	6.9 (44.4)
Average low °C (°F)	−4.9 (23.2)	−5.1 (22.8)	−3.2 (26.2)	0.6 (33.1)	5.2 (41.4)	8.7 (47.7)	12.0 (53.6)	11.2 (52.2)	7.5 (45.5)	3.1 (37.6)	0.4 (32.7)	−3.2 (26.2)	2.6 (36.7)
Record low °C (°F)	−32.0 (−25.6)	−30.4 (−22.7)	−27.0 (−16.6)	−16.0 (3.2)	−5.2 (22.6)	−1.4 (29.5)	3.0 (37.4)	1.2 (34.2)	−4.3 (24.3)	−11.1 (12.0)	−18.3 (−0.9)	−27.6 (−17.7)	−32.0 (−25.6)
Average precipitation mm (inches)	35.4 (1.39)	23.8 (0.94)	28.5 (1.12)	31.0 (1.22)	37.5 (1.48)	44.5 (1.75)	65.7 (2.59)	61.1 (2.41)	58.8 (2.31)	44.3 (1.74)	46.0 (1.81)	39.3 (1.55)	515.9 (20.31)

Source #1: SMHI[7]

Source #2: SMHI Monthly Data 2002–2015[8]

Figure 1: Table of the average monthly temperatures during the period 2002-2015 in Linköping.

a) Determining the prior distribution of the model parameters

We are asked to use the conjugate prior of the linear regression model. The model is the following:

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

As starting parameters we are going to use:

$$\mu_0 = (-10, 100, -100)^T; \quad \Omega_0 = 0.01 \cdot I_3; \quad \nu_0 = 4; \quad \sigma_0^2 = 1\,.$$

These however are not granted to return plausible values for the β of our problem. In fact we would expect the temperatures in Sweden to be included at maximum between $-30\ °C$ and $+30\ °C$. Therefore linear combinations generating values totally outside this range are, to our beliefe, extremely unlikely, if not impossible.
We should then simulates draws from the joint prior of all the parameters and then draw the regression curve. To help us generating multivariate normal observation, we will use the package `mvtnorm` of `R`. To instead generate random numbers from $Inv-\chi^2$ we will use the same generator created in the previous lab (included in the appendix).
There was no specified number of values to be tested, so we decided to set the number of draws to 100. For each iteration i we will draw the various random numbers following this order:

- We will first generate a value, that we will call σ_i^2, for σ_0^2 from the $Inv-\chi^2$ distribution;
- We will use the previously generated value for the variance to generate the three regression coefficients β_i, from the multivariate normal distribution of the prior β_0;
- To an existing plot, we will add a line corresponding to the regression curve generated from the randomly drawn β_i.

The results for the starting values defined above are summarized in Figure 2. As we can see, this 100 draws result quite messy and sparse accross the graph. The regressions sometimes even predict temperature higher than $+50\ °C$ (or lower than $-50\ °C$) and in some cases one of the terms (either the quadratic or the linear one) is approximately zero, which leads to unrealistic results. Moreover some of the generated trends are close to linearity, which is not plausible for the problem we are asked to solve.

```r
library(mvtnorm)
set.seed(9876)

# Setting the starting parameters
nu_0 <- 4
mu_0 <- c(-10,100,-100)
omega_0 <- 0.01*diag(3)
omega_0_inv <- solve(omega_0)
sigma_0 <- 1


# --------------------------------------------------------------------------------
# Generate from the joint prior:
# --------------------------------------------------------------------------------

# Number of draws
NumDraws <- 100
# Allocate the space for the different beta
matrix_beta_0 <- matrix(NA, NumDraws, 3)

# Crate function for the plot:
poly_plot <- function(x, beta) {
  return( beta[1] + x*beta[2] + x^2*beta[3])
}
# Empty plot where to draw the various lines
p1 <- ggplot(data = data.frame(0)) + xlim(c(0, 1)) + theme_light()

for(i in 1:NumDraws) {

  # First pick a value for the variance (sigma)
  temporary_sigma <- rinvchisq(1, nu_0, sigma_0)
  # Then generate samples for beta
  matrix_beta_0[i,] <- rmvnorm(1, mu_0, omega_0_inv*temporary_sigma)
  # Draw the line
  p1 <- p1 + stat_function(fun = poly_plot, args = list(beta = matrix_beta_0[i,]),
                           col = "black", alpha = 0.5)
}

# Add to the plot the line corresponding to the average of the NumDraws regression lines
p1 <- p1 + stat_function(fun = poly_plot, args = list(beta = mu_0),
                         col = "red", size = 1)

print(p1)
```

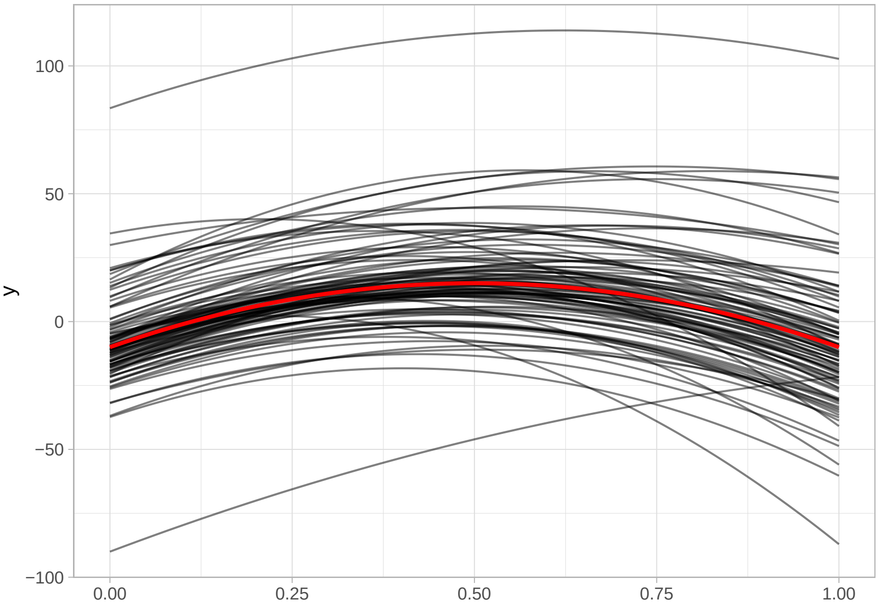


Figure 2: 100 possible regression curves drawn from our priors of the parameters σ_0^2 and β_0; the red line corresponds to the mean of the β_0 distribution.

In order to correct the observed behaviour and make it match our beliefes, we decided to reduce the impact of the variance Ω_0 and to slightly "pull up" the regression curve. In order achieve the desired results, we set $\Omega_0 = 0.1 \cdot I_3$ and $\beta_0 = (-5, 100, -100)$ (we only increase of $+5\ °C$ the intercept value). We also decided to modify the degrees of freedom of the $Inv - \chi^2$, which represent the degree of certanty of our prior: even though the natural variability of the climate leaves us quite unsure of what we will actually observe in the data, the information we got are quite reliable and resulting from various years of observations. We therefore modified ν_0 to 30, which means the prior will weight less than 10% in the distribution of the posterior.
The outcome of these new set of parameters is displayed in Figure 3. In this case, even though the regression lines are still fairly variable, the span of the generated lines seems plausible and we consider ourselves satisfied with the result.

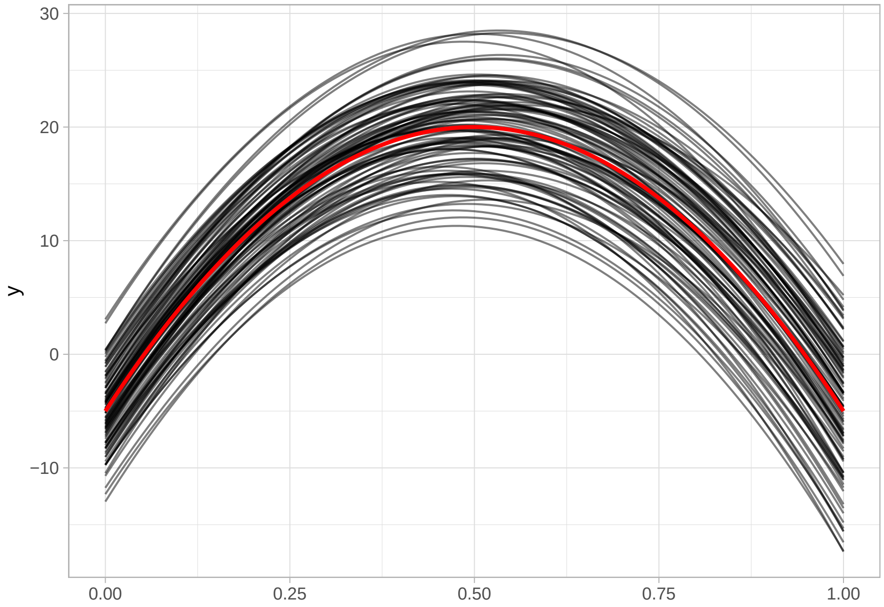


Figure 3: 100 possible regression curves drawn from our priors of the updated parameters σ_0^2 and β_0; the red line corresponds to the mean of the β_0 distribution.

b) Simulate from the joint posterior and the marginal posterior of the distribution

We will now simulate from joint posterior distribution of $\beta_0, \beta_1, \beta_2$ and σ^2. From the theory saw during the lectures we know that:

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

In other words the posteriors obtained depend on new parameters, derived by updating the ones of the prior (which are of the same family, so they actually are *conjugate priors*) with the information obtained from the data. These new parameters are:

$$\begin{aligned}
\Omega_n &= X^T X + \Omega_0 \\
\mu_n &= (X^T X + \Omega_0)^{-1}(X^T X \hat{\beta} + \Omega_0 \mu_0) \\
\nu_n &= \nu_0 + n \\
\sigma_n^2 &= \frac{1}{\nu_n}[\nu_0 \sigma_0^2 + (y^T y + \mu_0^T \Omega_0 \mu_0 - \mu_n^T \Omega_n \mu_n)]
\end{aligned}$$

As before, we will first generate a possible value for the parameter σ^2 and then use that to obtain a random sample of the 3-dimensional vector of β, which corresponds (approximately) to randomly generate from the marginal posterior distribution of the β distribution. The code to complete the request is the following:

```r
# ------------------------------------------------------------------------------
# Q1 - b)
# ------------------------------------------------------------------------------

# Set the number of parameters
k <- 3
# Set the number of samples
NumDraws <- 1000

# Create the new, updated parameters for the posterior
nu_n <- nu_0 + n - k
X <- matrix(c(rep(1, n), data$time, data$time^2), n, 3)
omega_n <- t(X)%*%X + omega_0
y <- data$temp
beta_hat <- solve(t(X)%*%X) %*% (t(X)%*%y)
mu_n <- solve(t(X)%*%X + omega_0) %*% (t(X)%*%X %*% beta_hat + omega_0 %*% mu_0)
sigma_n <- (1/nu_n)*(nu_0*sigma_0 + (t(y)%*%y + t(mu_0)%*%omega_0%*%mu_0
                                    - t(mu_n)%*%omega_n%*%mu_n))

# Allocate the space for the betas and sigmas (useful for the histogram afterwards)
matrix_beta_n <- matrix(NA, NumDraws, 3)
vector_sigma_n <- rep(NA, NumDraws)

for(i in 1:NumDraws) {

  # First pick a value for the variance (sigma)
  vector_sigma_n[i] <- as.numeric(rinvchisq(1, nu_n, sigma_n))
  # Then generate samples for beta
  matrix_beta_n[i,] <- rmvnorm(1, mu_n, solve(omega_n)*vector_sigma_n[i])

}
```

We are then asked to plot the marginal posteriors for each parameter as a histogram. We do not have an explicit solution to this distribution, and will require to integrate over σ the probability density of $\beta \mid y, \sigma$. We can get around this problem by first generating a big sample from the posterior distribution of σ and then used those generated numbers to generate the β coefficients: in this way the β will be simulated following the distribution of σ, which practically is an approximation for the computation of the integral we were supposed to solve.
So we will then simply plot the individual columns of the matrix containing the generated β (Figure 4). As we can observe, the distributions of the marginal are symmetric and relatively concentrated around their mode. The mean of their distribution is μ_n, which is equal to the vector $[-10.8255,\ 94.4863 \text{ and } -86.7086]$ respectively. As we can see, the mean value of the parameters changed from the prior we set. In particular, the posterior intercept actually got back to almost -10, the value that we originally started with: the changed we made for the prior of β_0 was therefore in the wrong direction.
On the other hand, both β_2 and β_3 got shrinked and the biggest change was observed for the quadratic term, which is decreased by almost 15 in respect to our prior belief. The observed data is therefore showing a more restrained trend. XMoreover the two coefficients are not equal anymore as our prior, and since $\beta_2 < \beta_1$ we expect the temperatures of January to be lower to the ones of December.
Finally we observe how much the value of σ^2 increased compared to our prior, which was $\sigma_0^2 = 1$.

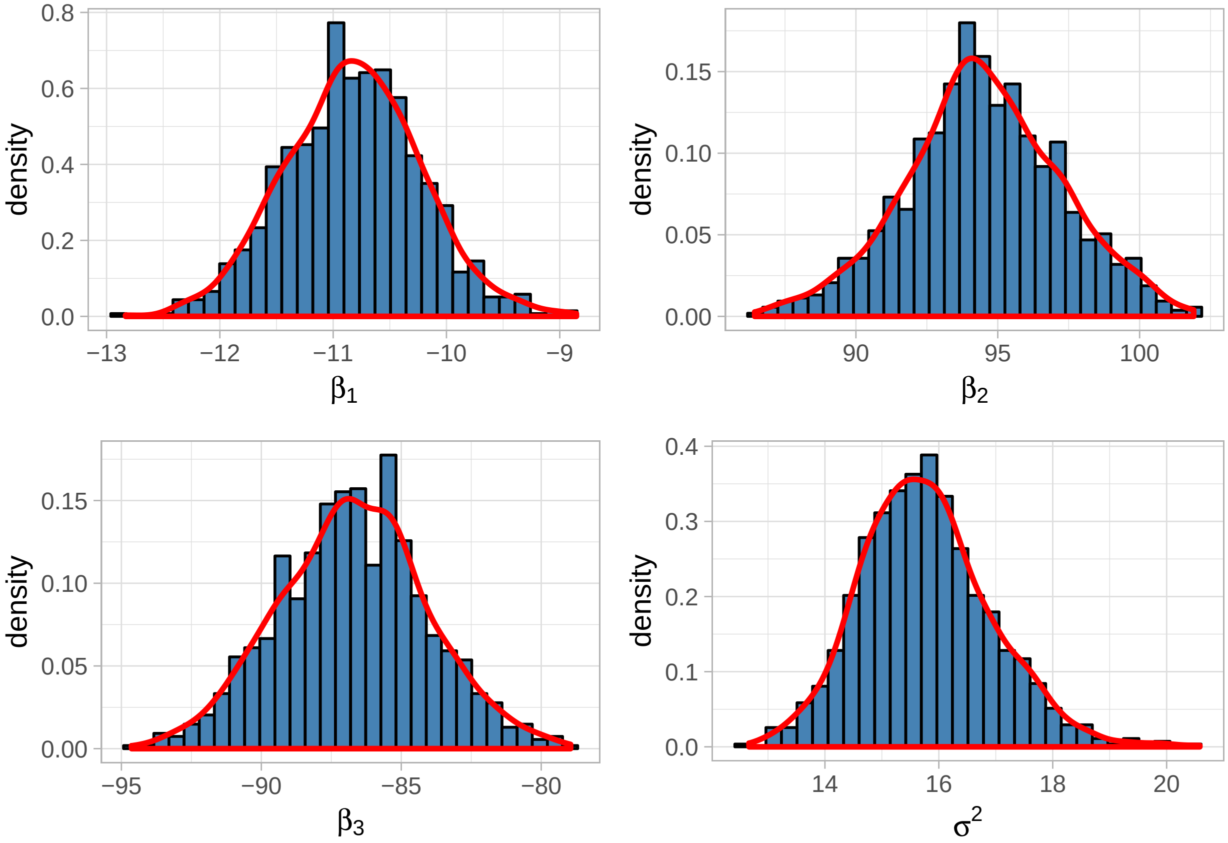


Figure 4: Histograms of the marginal posterior distribution of the β and σ^2 parameters

Finally, we were asked to compute the 95% equal tail posterior probability intervals for every value of *time* estimated using the different sets of coefficients we generated before. The middle brown line in Figure 5 is the median of all the β generated from our joint distribution. The dashed red lines representing the 95% credible interval were drawn taking the quantiles (0.025; 0.975) of the predicted $\tilde{y}$ at each point in time. We also included the 95% prediction interval, obtained by adding $\pm 1.96\sigma_n$ to the extremes of the mean of the joint posterior distribution (given by μ_n).
The credible interval band is quite narrow and barely contains 10% of the observed data points. This finding does not surprise us: the confidence band in fact is related to the regression curve, which is a summary of the average behaviour of the response for each moment in time. The uncertanty we are plotting is therefore the one related to the trend of the data, not to the actually observed data. In other words, we are not taking into account the random error ϵ, which is normally distributed with variance $\sigma_n^2 = 15.7204$. If we want an interval which really contains (on average) 95% of the data points, we have to refer to the prediction interval (green dotted line), which in fact almost contains that proportion of observations.

```r
# Median of the marginal posterior
beta_median_posterior <- apply(matrix_beta_n, 2, median)

# Estimation of the whole dataset with 1000 different beta parameters
estimated_y <- matrix(0, n, NumDraws)
for(j in 1:NumDraws) {
  estimated_y[,j] <- poly_plot(data$time, matrix_beta_n[j,])
}
# Computing the 95% credible interval
```

```
extremes_interval_curve <- apply(estimated_y, 1, function(x)
  return( c(quantile(x, 0.025), quantile(x, 0.975)) ))

# Mean of the marginal posterior and related prediction interval
beta_mean_posterior <- apply(matrix_beta_n, 2, mean)
pred_inter_up <- c(mu_n[1,1] + qnorm(0.025)*sqrt(sigma_n[1,1]), mu_n[2:3,1])
pred_inter_down <- c(mu_n[1,1] + qnorm(0.975)*sqrt(sigma_n[1,1]), mu_n[2:3,1])
```

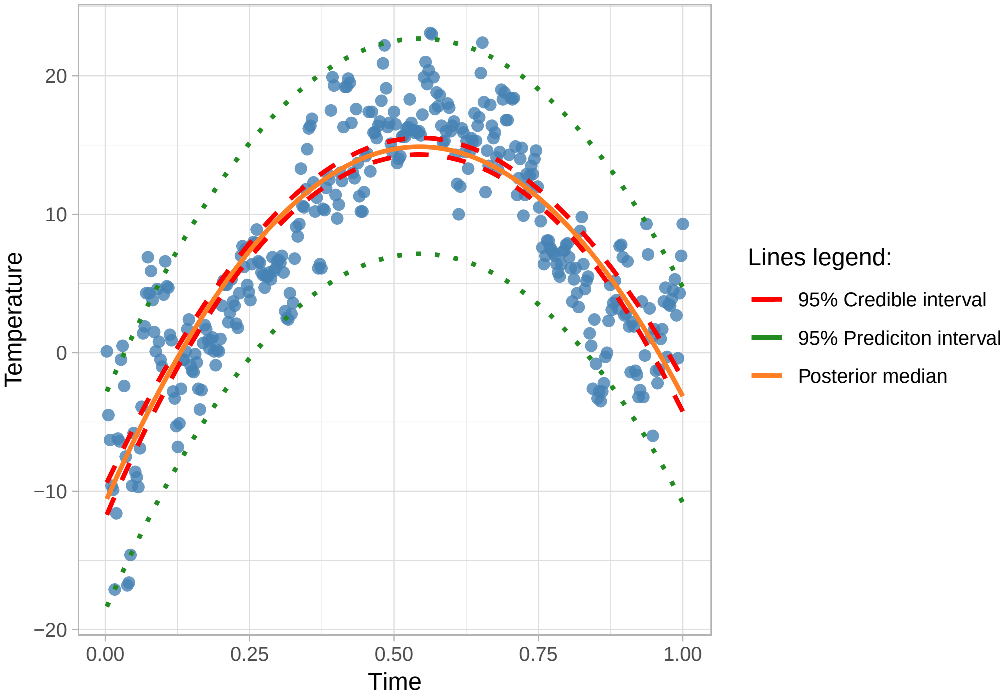


Figure 5: Joint posterior median of the β parameters. The 95% credible interval, together with the 95% prediction interval, have been plotted over the observed data points.

c) Posterior distribution of the highest temperature

In order to get the time with the highest expected temperature (that we will call $\tilde{x}$) we will reuse the β generated in the previous step. Again, the posterior variance σ_n^2 will not appear in our computations, since we are interested in the mean value of the model $X\beta$.
Since the distribution we have is unimodal, we can find $\tilde{x}$ for each generated set of coefficients taking the derivative of $f(time)$:

$$f(time) = \beta_0 + \beta_1 \cdot time + \beta_2 \cdot time^2 \quad \rightarrow \quad \frac{\partial f(time)}{\partial time} = \beta_1 + 2\beta_2 \cdot time = 0 \quad \rightarrow \quad \tilde{x} = \frac{-\beta_1}{2\beta_2}$$

From the plot in Figure 6, we can see that from our previously generated 1000 samples of β, the values of $\tilde{x}$ span between 0.53 and 0.56, which corresponds to almost 11 days. If we consider its 95% credible interval what we get is $[0.536, 0.5534]$, which corresponds to a range of 6.4 days, which is quite precise. The mean value of the sample distribution is 199.415, which correspond to the date 2016-07-18. The value found is totally plausible since July is the month that also according to our prior information has the highest daily mean during the year.

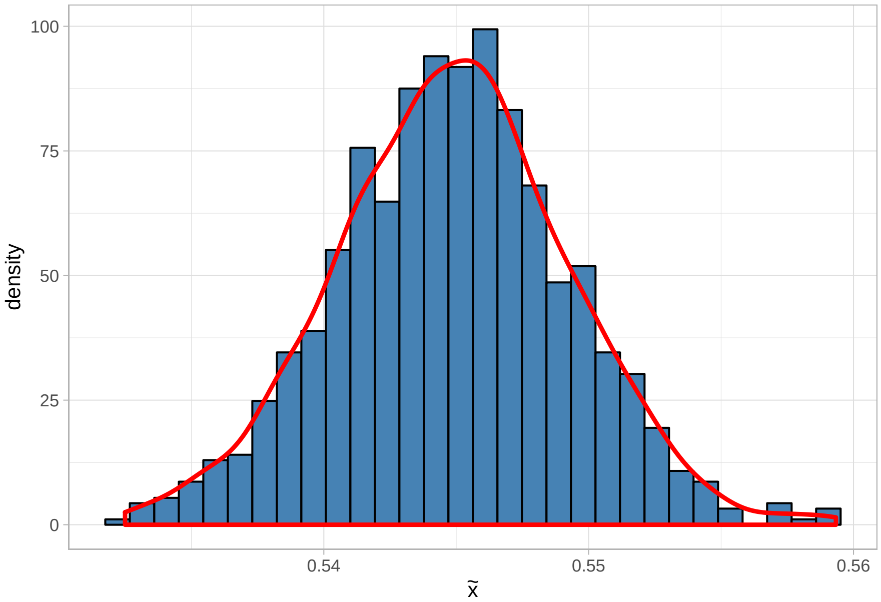


Figure 6: Histogram of $\tilde{x}$, the values of *time* corresponding to the highest predicted average temperature.

d) Priors for the estimation of a polynomial model of order 7

In order to avoid overfitting when trying to estimate a higher order polynom we will try to implement some sort of regularization. We think that the *lasso* is probably the most suitable solution. From the theory we know, in the Bayesian world the *lasso* corresponds to setting the prior of the marginal β distribution to the *Laplace* distribution:

$$\beta \mid \sigma^2 \sim \mathit{Laplace}\left(0, \frac{\sigma^2}{\lambda}\right)$$

The distribution for σ^2 remains unchanged. In other words, together with changing the distribution, we are setting the paramter $\mu_0 = (0, 0, 0)$ and $\Omega_0 = \lambda \cdot I_3$, where λ represents the penalizing factor. We can imagine to set it equal to a value between 10 and 100 (depending on the problem and the informations we have), or using *cross-validation* to find out which value is the best. Sticking to the Bayesian approach, we could set another prior for the parameter λ itself. We could hypothesize $\lambda \sim \mathit{Gamma}\left(\frac{\eta_0}{2}, \frac{\eta_0}{2\lambda_0}\right)$, setting a relatively low η_0 (since we are not sure about our prior beliefes) and a high λ_0 (because we want to avoid overfitting). One could also have used the *Ridge* regression, which has an equivalent prior but with the β being distributed as a Gaussian. However, the *Ridge* regression does not set to exactly zero the parameters and, since we are expecting a 7-degree polynom to overfit our data, it may not be the best solution.

Question 2 - Posterior approximation for classification with logistic regression

In this dataset we are presented with $n = 200$ observations regarding the status and consitions of a women. We are interested into understanding whether or not the woman is currently working in respect to the other variables. A brief description of the dataset can be found in Table 1.

Table 1: Description of the dataset.

Variable	**Data type**	**Meaning**	**Role**
Work	Binary	Whether or not the woman works	Response
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
ExpYears2	Numeric	(Years of experience/10)^2	Feature
Age	Counts	Age	Feature
NSmallChild	Counts	Number of child $\leq$ 6 years in household	Feature
NBigChild	Counts	Number of child $>$ 6 years in household	Feature

a) Fitting the logistic regression

We consider the following model for our data:

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

where y is our response ($y = 1$ in case the woman in working, $y = 0$ otherwise); x is the 8-dimensional vector of the features.

We implement the logistic regression as requested, using the `glm(·)` function of `R`. The estimates of the coefficients is reported in Figure 7 and the significant coefficients are the ones related to the variables: *Age, EducYears, ExpYears, NSmallChild.* Of the four of them, the number of small children has both the highest significance and the most impact on the response. In other words, according to our model a woman is likely to be unemployed if she has a son or daughter younger than 6 years. If the children is older instead it does not seem to effect the carrier of the mother.

The level of expertise of the subjects is linearly linked to the response, as well as the number of years spent into education. However, a quadratic trend with the job experience seems non-relevant for the prediction of the response.

The age of the woman does effect her job condition, with tendentially less and less chanches of being employed as she gets older. The income of the husband does not highlight any significant link with the response.

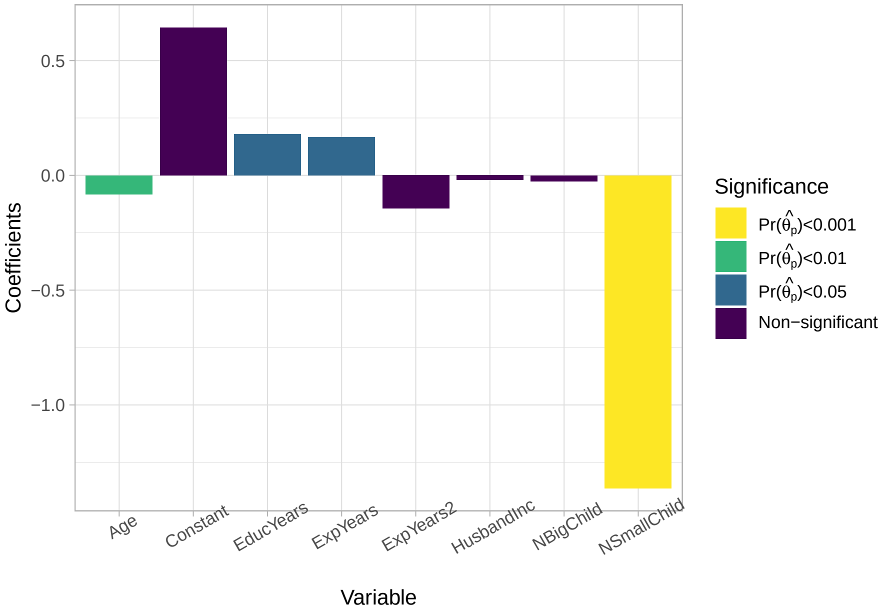


Figure 7: Barplot of the estimated coefficients of the logistic regression, colored by their significance level.

b) Posterior approximation

We will now try to approximate the posterior distribution of the 8-dim parameter vector β with a multivariate normal distribution. We are supposing:

$$\beta|y, X \sim \mathcal{N}\left(\tilde{\beta}, J_y^{-1}(\tilde{\beta})\right) \quad \text{where :} \begin{cases} \tilde{\beta} & \text{is the posterior mode} \\ \\ J(\tilde{\beta}) = -\left.\frac{\partial^2 \ln(p(\beta|y))}{\partial\beta\partial\beta^T}\right|_{\beta=\tilde{\beta}} & \text{is the observed Hessian evaluated at } \tilde{\beta} \end{cases}$$

Note that $\frac{\partial^2 \ln(p(\beta|y))}{\partial\beta\partial\beta^T}$ is an 8×8 matrix with second derivatives on the diagonal and cross-derivatives $\frac{\partial^2 \ln(p(\beta|y))}{\partial\beta_i\partial\beta_j}$ on the off-diagonal. We will approximate it numerically through the function `optim(·)` of `R`. As a prior, we will use $\beta \sim \mathcal{N}(0, \tau^2 I)$ with $\tau = 10$ ($\tau^2 = 100$).
Using this sort of distribution and model, the formula for the log-likelihood is the following:

$$\ln\Big(p(\boldsymbol{y} \mid \boldsymbol{X}, \beta)\Big) = \ln\left(\prod_{i=1}^{n} \frac{[\exp(x_i^T\beta)]^{y_i}}{\exp(1 + x_i^T\beta)}\right) = \sum_{i=1}^{n} y_i(x_i\beta) - \sum_{i=1}^{n} \ln\Big(1 + \exp(x_i\beta)\Big)$$

We solved the task using the following code:

```r
# Set the parameters of the prior distribution
beta_0 <- as.vector(rep(0, p))
tau_2 <- 100
variance_0 <- tau_2 * diag(p)
```

```r
# Create a function to compute the posterior distribution
logistic_post <- function(starting_beta, mu_prior, var_prior, x, y) {

  # For some reason we were not able to make a matrix product even using Mattias' code
  loglik <- apply(x, 1, function(row) sum(row*starting_beta))
  loglik <- sum(y*loglik) - sum(log(1+exp(loglik)))
  # Compute the density according to the normal prior
  logprior <- dmvnorm(starting_beta, mu_prior, var_prior, log = T)

  return( loglik + logprior )

}

# Numerical optimization
optimal_res <- optim(beta_0, logistic_post, gr = NULL, beta_0, variance_0, x, y,
                     method = "BFGS", control = list(fnscale = -1), hessian = T)

# Extract the results
post_beta_mode <- optimal_res$par
post_cov <- solve(-optimal_res$hessian)
colnames(post_cov) <- colnames(x)
rownames(post_cov) <- colnames(x)
approx_post_sd <- sqrt(diag(post_cov))
```

Using this procedure, the estimated $\tilde{\beta}$ are: 0.6267, -0.0198, 0.1802, 0.1676, -0.1446, -0.0821, -1.3591, -0.0247 for the variables *Constant, HusbandInc, EducYears, ExpYears, ExpYears2, Age, NSmallChild, NBigChild* respectively. Just for curiosity, we checked how much these estimates differ from the ones obtained using `glm(·)`. It turned out that they were almost identical, with the biggest difference related to *NBigChild*: the two $\tilde{\beta}$ values differ of only 2.93%.
Regarding the inverse observed information $J_y^{-1}(\tilde{\beta})$, the estimation of the matrix is equal to:

	Constant	HusbandInc	EducYears	ExpYears	ExpYears2	Age	NSmallChild	NBigChild
Constant	2.266023	0.003339	-0.065451	-0.011791	0.045781	-0.030293	-0.188748	-0.098024
HusbandInc	0.003339	0.000253	-0.000561	-0.000031	0.000141	-0.000036	0.000507	-0.000144
EducYears	-0.065451	-0.000561	0.006218	-0.000356	0.001896	-0.000003	-0.006135	0.001753
ExpYears	-0.011791	-0.000031	-0.000356	0.004352	-0.014249	-0.000134	-0.001469	0.000544
ExpYears2	0.045781	0.000141	0.001896	-0.014249	0.055579	-0.000330	0.003208	0.000512
Age	-0.030293	-0.000036	-0.000003	-0.000134	-0.000330	0.000718	0.005184	0.001095
NSmallChild	-0.188748	0.000507	-0.006135	-0.001469	0.003208	0.005184	0.151262	0.006769
NBigChild	-0.098024	-0.000144	0.001753	0.000544	0.000512	0.001095	0.006769	0.019972

Table 2: Approximate posterior covariance matrix of $\tilde{\beta}$. It corresponds to the observed information $J_y^{-1}(\tilde{\beta})$.

We are then asked to compute a 95% credible interval for the coefficient $\tilde{\beta}$ for the variable *NSmallChild*. In order to get it, we can exploit the multivariate-normality of the coefficients, which simplifies to a univariate normal distribution when only one specific $\tilde{\beta}_j$ is considered. In fact, $\tilde{\beta}_j \sim \mathcal{N}\Big(\tilde{\beta}_j\ ,\ J_y^{-1}(\tilde{\beta})_{jj}\Big)$ and its 95% credible interval is $\tilde{\beta}_j \pm 1.96\sqrt{J_y^{-1}(\tilde{\beta})_{jj}}$. The values found are: $[-2.121411,\ -0.596855]$.

c) Simulation and prediction

We are asked to estimate the predictive distribution for the response for a 40 year old woman, with two children (3 and 9 years old), 8 years of education, 10 years of experience and a husband with an income of 10. In order to generate predictions for the model we will use the approximate distribution of the parameters obtained in step 2b).

We already know that the parameters follow the posterior distribution $\mathcal{N}\left(\tilde{\beta}, J_y^{-1}(\tilde{\beta})\right)$ and we have numerically estimated the posterior mode and variance of the distribution. Using these optimal value we can generate various $\tilde{\beta}$ and use them to compute the probability of `Work` through the logistic function. Once the probabilities are obtained, we can then simulate the outcome of the associated *Bernoulli* distribution using a second random number generator. The code to solve this task is the following:

```r
# X-values for the target prediction
data_pred <- c(1, 10, 8, 10, 1, 40, 1, 1)

# Time to generate! We will use the approximate distribution obtained before
NumDraws <- 1000
# We generate many betas from the posterior of the parameters
many_beta_pred <- rmvnorm(1000, post_beta_mode, post_cov)
# We create a function to compute the logistic regression AND
# draw from a Bernoulli distribution using the computed probability
logistic_fun <- function(beta, x) {
  pi <- exp(as.numeric(x%*%beta)) / (1+exp(as.numeric(x%*%beta)))
  return( rbinom(1, 1, pi) )
}
# We know estimate the response for each draw of beta
predictions_y <- apply(many_beta_pred, 1, logistic_fun, data_pred)
```

In Figure 8 we can observe the outcome of 1000 simulated predictions. According to our model, for this specific subject the probability of being working is quite low, 22.8 %. We are therefore pretty confident that this typology of subject will be unemployed.

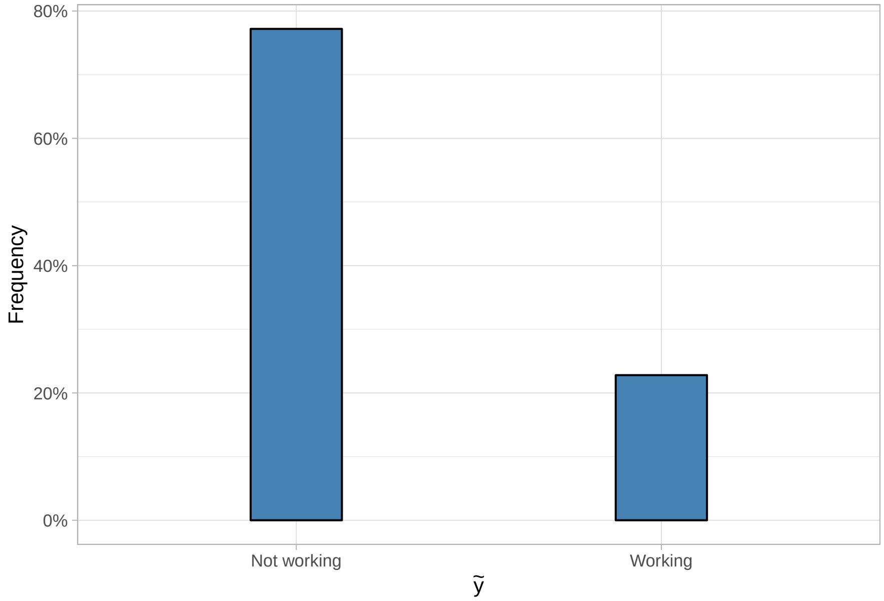


Figure 8: Barplot of the predicted working status (sample dimension of 1000) for a 40-years-old woman, with one small and one big child, 8 years of education, 10 years of experience and a husband with an income of 10.

Appendix

```
knitr::opts_chunk$set(echo = F, message = F, error = F, warning = F,
                      fig.align='center', out.width="70%")


# -------------------------------------------------------------------------------
# Q1 - Intro
# -------------------------------------------------------------------------------

# Read the data
data <- read.table("TempLinkoping.txt", header = T)
n <- nrow(data)
p <- ncol(data)


library(ggplot2)
# knitr::include_graphics("Linköping_temperatures.jpeg")
library('jpeg')
myjpeg <- readJPEG(source = "Linköping_temperatures.jpeg", native = T)

res <- dim(myjpeg)[2:1]
plot(1, 1, xlim=c(1,res[1]), ylim=c(1,res[2]), asp=1, type='n', xaxs='i',
     yaxs='i', xaxt='n', yaxt='n', xlab='', ylab='', bty='n')
rasterImage(myjpeg, 1, 1, res[1], res[2])


old_text = "plot the given time series in Figure ???. As we can see, the temperature follows indeed a q
Tendentially, we would expect our model to perform well on the provided data, managing to capture the g


ggplot(data, aes(x = time*366, y = temp)) +
  geom_line(size = 1, col = "steelblue") +
  labs(x = "Day of the year",
       y = expression(paste("Temperature (", degree, "C)", sep = ""))) +
  theme_light()


# -------------------------------------------------------------------------------
# Q1 - a)
# -------------------------------------------------------------------------------

# Generator of random numbers from inverse chisquare:
rinvchisq <- function(draws, n, tau) {
  chi_square <- rchisq(draws, n-1)
  return( tau*(n-1)/chi_square )
}

# Number of draws
NumDraws <- 100


library(mvtnorm)
```

```
set.seed(9876)

# Setting the starting parameters
nu_0 <- 4
mu_0 <- c(-10,100,-100)
omega_0 <- 0.01*diag(3)
omega_0_inv <- solve(omega_0)
sigma_0 <- 1


# ------------------------------------------------------------------------------
# Generate from the joint prior:
# ------------------------------------------------------------------------------

# Number of draws
NumDraws <- 100
# Allocate the space for the different beta
matrix_beta_0 <- matrix(NA, NumDraws, 3)

# Crate function for the plot:
poly_plot <- function(x, beta) {
  return( beta[1] + x*beta[2] + x^2*beta[3])
}
# Empty plot where to draw the various lines
p1 <- ggplot(data = data.frame(0)) + xlim(c(0, 1)) + theme_light()

for(i in 1:NumDraws) {

  # First pick a value for the variance (sigma)
  temporary_sigma <- rinvchisq(1, nu_0, sigma_0)
  # Then generate samples for beta
  matrix_beta_0[i,] <- rmvnorm(1, mu_0, omega_0_inv*temporary_sigma)
  # Draw the line
  p1 <- p1 + stat_function(fun = poly_plot, args = list(beta = matrix_beta_0[i,]),
                           col = "black", alpha = 0.5)
}

# Add to the plot the line corresponding to the average of the NumDraws regression lines
p1 <- p1 + stat_function(fun = poly_plot, args = list(beta = mu_0),
                         col = "red", size = 1)

print(p1)


# Change the starting parameters
# We want less variation and to increase slightly the intercept,
# the average trends behaviour seem fine instead

omega_0 <- 0.1*diag(3)
omega_0_inv <- solve(omega_0)
mu_0[1] <- -5
nu_0 <- 30
```

```r
# Empty plot where to draw the various lines
p2 <- ggplot(data = data.frame(0)) + xlim(c(0, 1)) + theme_light()

for(i in 1:NumDraws) {

  # First pick a value for the variance (sigma)
  temporary_sigma <- rinvchisq(1, nu_0, sigma_0)
  # Then generate samples for beta
  matrix_beta_0[i,] <- rmvnorm(1, mu_0, omega_0_inv*temporary_sigma)
  # Draw the line
  p2 <- p2 + stat_function(fun = poly_plot, args = list(beta = matrix_beta_0[i,]),
                           col = "black", alpha = 0.5)
}

# Add to the plot the line corresponding to the average of the NumDraws regression line
p2 <- p2 + stat_function(fun = poly_plot, args = list(beta = mu_0),
                         col = "red", size = 1)
# p2 <- p2 + geom_line(aes(x = time, y = temp), data = data,
#                       size = 1, col = viridis::viridis(2)[2])

print(p2)


# ------------------------------------------------------------------------------
# Q1 - b)
# ------------------------------------------------------------------------------

# Set the number of parameters
k <- 3
# Set the number of samples
NumDraws <- 1000

# Create the new, updated parameters for the posterior
nu_n <- nu_0 + n - k
X <- matrix(c(rep(1, n), data$time, data$time^2), n, 3)
omega_n <- t(X)%*%X + omega_0
y <- data$temp
beta_hat <- solve(t(X)%*%X) %*% (t(X)%*%y)
mu_n <- solve(t(X)%*%X + omega_0) %*% (t(X)%*%X %*% beta_hat + omega_0 %*% mu_0)
sigma_n <- (1/nu_n)*(nu_0*sigma_0 + (t(y)%*%y + t(mu_0)%*%omega_0%*%mu_0
                                    - t(mu_n)%*%omega_n%*%mu_n))

# Allocate the space for the betas and sigmas (useful for the histogram afterwards)
matrix_beta_n <- matrix(NA, NumDraws, 3)
vector_sigma_n <- rep(NA, NumDraws)

for(i in 1:NumDraws) {

  # First pick a value for the variance (sigma)
  vector_sigma_n[i] <- as.numeric(rinvchisq(1, nu_n, sigma_n))
  # Then generate samples for beta
  matrix_beta_n[i,] <- rmvnorm(1, mu_n, solve(omega_n)*vector_sigma_n[i])
```

```r
}


# Generate from the marginal posterior (T-distribution)
# Beta_posterior <- rmvt(n = 1000, delta = mu_n,
#                          sigma = as.numeric(sigma_n)*solve(omega_n), df = nu_n)

# Histogram of Beta1
p1 <- ggplot(data = data.frame(x = matrix_beta_n[,1]), aes(x = x ))+
  geom_histogram(aes(y = ..density..), bins = 30,
                 col = "black", fill = "steelblue") +
  geom_density(col = "red", size = 1) +
  labs(x = expression(beta[1])) +
  theme_light()

# Histogram of Beta2
p2 <- ggplot(data = data.frame(x = matrix_beta_n[,2]), aes(x = x ))+
  geom_histogram(aes(y = ..density..), bins = 30,
                 col = "black", fill = "steelblue") +
  geom_density(col = "red", size = 1) +
  labs(x = expression(beta[2])) +
  theme_light()

# Histogram of Beta3
p3 <- ggplot(data = data.frame(x = matrix_beta_n[,3]), aes(x = x ))+
  geom_histogram(aes(y = ..density..), bins = 30,
                 col = "black", fill = "steelblue") +
  geom_density(col = "red", size = 1) +
  labs(x = expression(beta[3])) +
  theme_light()

# Histogram of Sigma
p4 <- ggplot(data = data.frame(x = vector_sigma_n), aes(x = x ))+
  geom_histogram(aes(y = ..density..), bins = 30,
                 col = "black", fill = "steelblue") +
  geom_density(col = "red", size = 1) +
  labs(x = expression(sigma^2)) +
  theme_light()

# Put the plots together
gridExtra::grid.arrange(p1, p2, p3, p4, ncol = 2)


# Median of the marginal posterior
beta_median_posterior <- apply(matrix_beta_n, 2, median)

# Estimation of the whole dataset with 1000 different beta parameters
estimated_y <- matrix(0, n, NumDraws)
for(j in 1:NumDraws) {
  estimated_y[,j] <- poly_plot(data$time, matrix_beta_n[j,])
}
# Computing the 95% credible interval
extremes_interval_curve <- apply(estimated_y, 1, function(x)
  return( c(quantile(x, 0.025), quantile(x, 0.975)) ))

# Mean of the marginal posterior and related prediction interval
beta_mean_posterior <- apply(matrix_beta_n, 2, mean)
pred_inter_up <- c(mu_n[1,1] + qnorm(0.025)*sqrt(sigma_n[1,1]), mu_n[2:3,1])
pred_inter_down <- c(mu_n[1,1] + qnorm(0.975)*sqrt(sigma_n[1,1]), mu_n[2:3,1])


p1 <- ggplot(data, aes(x = time)) +
  geom_point(aes(x = time, y = temp), data = data,
             col = "steelblue", alpha = 0.8, cex = 2) +
  # stat_function(aes(x = time), data = data, fun = poly_plot,
  #               args = list(beta = beta_median_marginal)) +
  stat_function(aes(col = "Posterior median"), fun = poly_plot,
                args = list(beta = beta_median_posterior), size = 1) +
  stat_function(aes(col = "95% Prediciton interval"), fun = poly_plot,
                args = list(beta = pred_inter_up), size = 1, lty = 3) +
  stat_function(aes(col = "95% Prediciton interval"), fun = poly_plot,
                args = list(beta = pred_inter_down), size = 1, lty = 3) +
  geom_line(aes(y = extremes_interval_curve[1,], col = "95% Credible interval"),
            lty = 2, size = 1) +
  geom_line(aes(y = extremes_interval_curve[2,], col = "95% Credible interval"),
            lty = 2, size = 1) +
  scale_color_manual(values = c("red", "forestgreen", "chocolate1"),
                     name = "Lines legend:") +
  labs(x = "Time", y = "Temperature") +
  theme_light() #+
  #theme(legend.position = "bottom")

p1


# ------------------------------------------------------------------------------
# Q1 - c)
# ------------------------------------------------------------------------------

beta_posterior <- as.data.frame(matrix_beta_n)
beta_posterior$X_tilda <- (-1*beta_posterior[,2])/(2*beta_posterior[,3])


beta_posterior <- as.data.frame(matrix_beta_n)
beta_posterior$X_tilda <- (-1*beta_posterior[,2])/(2*beta_posterior[,3])

###------- Histogram of X tilda
ggplot(data = data.frame(x = beta_posterior$X_tilda), aes(x = x,  y = ..density..))+
  geom_histogram(aes(y = ..density..), bins = 30,
                 col = "black", fill = "steelblue") +
  geom_density(col = "red", size = 1.1) +
  labs(x = expression(tilde(x)))+
  theme_light()


# ------------------------------------------------------------------------------
# Q1 - d)
# -------------------------------------------------------------------------------


# -------------------------------------------------------------------------------
# Q2 - Intro
# -------------------------------------------------------------------------------

data <- read.delim("WomenWork.dat", header = T, sep="")
y <- data$Work
x <- data[,-1]
n <- nrow(data)
p <- ncol(x)


library(kableExtra)

df_table <- data.frame(a = colnames(data),
                       b = c("Binary", "1", "Numeric", "Counts", "Counts",
                             "Numeric", "Counts", "Counts", "Counts"),
                       c = c("Whether or not the woman works",
                             "Constant to the intercept", "Husband's income",
                             "Years of education", "Years of experience",
                             "$(\\text{Years of experience}/10)^2$", "Age",
                             "Number of child $\\leq$ 6 years in household",
                             "Number of child > 6 years in household"),
                       d = c("Response", rep("Feature", 8)))
names_table <- c("Variable", "Data type", "Meaning", "Role")

kable(df_table, "latex", booktabs = T, align = "c", col.names = names_table,
      linesep = "", escape = F, caption = "Description of the dataset.") %>%
  column_spec(c(1), border_right = T) %>%
  row_spec(0, bold = T) %>%
  kable_styling(latex_options = "hold_position", font_size = 8)


# -------------------------------------------------------------------------------
# Q2 - a)
# -------------------------------------------------------------------------------

glmModel <- glm(Work ~ 0 + ., data = data, family = binomial)


# Plot of the coefficients

# For labels in the legend
library(latex2exp)

s <- as.data.frame(summary(glmModel)$coef)
features_names <- rownames(s)
pval <- s$`Pr(>|z|)`
df_coef <- data.frame(Variable = features_names, Coefficients = s$Estimate,
                      Significance = as.factor(
                        ifelse(pval>0.05, "Non-significant",
                               ifelse(pval>0.01, "alpha = 0.05",
                                      ifelse(pval>0.001, "alpha = 0.01",
                                             "alpha = 0.001")))))

ggplot(df_coef, aes(x = Variable, y = Coefficients, fill = Significance)) +
  geom_bar(stat = "identity") +
  scale_fill_manual(values = rev(viridis::viridis(4)),
                    labels = list(TeX("$\\Pr(\\hat{\\theta_p})<0.001$"),
                                  TeX("$\\Pr(\\hat{\\theta_p})<0.01$"),
                                  TeX("$\\Pr(\\hat{\\theta_p})<0.05$"),
                                  TeX("Non-significant"))) +
  theme_light() +
  theme(axis.text.x = element_text(angle = 30))


# Some measures of fitness of our model
library(pROC)
library(cvAUC)

plot(roc(data$Work, predict(glmModel, newdata = data, type = "response")))
cvAUC::AUC(predict(glmModel, newdata = data, type = "response"), data$Work)


# ------------------------------------------------------------------------------
# Q2 - b)
# ------------------------------------------------------------------------------


# Set the parameters of the prior distribution
beta_0 <- as.vector(rep(0, p))
tau_2 <- 100
variance_0 <- tau_2 * diag(p)

# Create a function to compute the posterior distribution
logistic_post <- function(starting_beta, mu_prior, var_prior, x, y) {

  # For some reason we were not able to make a matrix product even using Mattias' code
  loglik <- apply(x, 1, function(row) sum(row*starting_beta))
  loglik <- sum(y*loglik) - sum(log(1+exp(loglik)))
  # Compute the density according to the normal prior
  logprior <- dmvnorm(starting_beta, mu_prior, var_prior, log = T)

  return( loglik + logprior )

}

# Numerical optimization
optimal_res <- optim(beta_0, logistic_post, gr = NULL, beta_0, variance_0, x, y,
                     method = "BFGS", control = list(fnscale = -1), hessian = T)

# Extract the results
post_beta_mode <- optimal_res$par
```

```
post_cov <- solve(-optimal_res$hessian)
colnames(post_cov) <- colnames(x)
rownames(post_cov) <- colnames(x)
approx_post_sd <- sqrt(diag(post_cov))


library(xtable)
print(xtable(post_cov, digits = 6, caption = "Approximate posterior covariance matrix of
             $\\tilde \\beta$. It corresponds to the observed information
             $J^{-1}_y(\\tilde \\beta)$."), scalebox='0.85', comment = F)


# ------------------------------------------------------------------------------
# Q2 - c)
# ------------------------------------------------------------------------------


# X-values for the target prediction
data_pred <- c(1, 10, 8, 10, 1, 40, 1, 1)

# Time to generate! We will use the approximate distribution obtained before
NumDraws <- 1000
# We generate many betas from the posterior of the parameters
many_beta_pred <- rmvnorm(1000, post_beta_mode, post_cov)
# We create a function to compute the logistic regression AND
# draw from a Bernoulli distribution using the computed probability
logistic_fun <- function(beta, x) {
  pi <- exp(as.numeric(x%*%beta)) / (1+exp(as.numeric(x%*%beta)))
  return( rbinom(1, 1, pi) )
}
# We know estimate the response for each draw of beta
predictions_y <- apply(many_beta_pred, 1, logistic_fun, data_pred)


pred_plot <- as.factor(predictions_y)
levels(pred_plot) <- c("Not working", "Working")

ggplot() +
  geom_bar(aes(x = pred_plot), col = "black", fill = "steelblue",
           width = 0.25) +
  labs(x = expression(tilde(y)), y = "Frequency") +
  scale_y_continuous(labels = scales::percent_format(accuracy = 4, scale = 0.1)) +
  theme_light()
```