

## BAYESIAN STATISTICS

### ASSIGNMENT 2

---

#### QUESTION 1: PROBIT REGRESSION (HOFF 6.3)

A panel study followed  $n = 25$  married couples over a period of five years. One item of interest is the relationship between divorce rates and the various characteristics of the couples. For example, the researchers would like to model the probability of divorce as a function of age differential, recorded as the man's age minus the woman's age. The data can be found in the file `divorce.RData`. We will model these data with probit regression, in which a binary variable  $Y_i$  is described in terms of an explanatory variable  $x_i$  via the following latent variable model:

$$\begin{aligned} Z_i &= \beta x_i + \varepsilon_i \\ Y_i &= \mathbf{1}_{(c, +\infty)}(Z_i), \end{aligned}$$

where  $\beta$  and  $c$  are unknown coefficients,  $\varepsilon_1, \dots, \varepsilon_n \stackrel{\text{iid}}{\sim} \mathcal{N}(0, 1)$  and  $\mathbf{1}_{(c, +\infty)}(z) = 1$  if  $z > c$  and equals zero otherwise. In the following, since the covariates  $x_i$  are known, they will be treated as constants and so not explicitly written in the conditioning part.

#### Point a.

Assuming  $\beta \sim \mathcal{N}(0, \sigma_\beta^2)$ , obtain the full conditional distribution  $p(\beta | y_{1:n}, z_{1:n}, c)$ .

First of all let us write explicitly the conditional distributions which we can deduce from the text:

–  $\forall i = 1, \dots, n$  we know  $p(z_i | \beta)$ :

$$\begin{aligned} Z_i(\omega) | \beta &= \beta x_i + \varepsilon_i(\omega) \sim \beta x_i + \mathcal{N}(0, 1) \sim \mathcal{N}(\beta x_i, 1) \implies Z_i | \beta \sim \mathcal{N}(\beta x_i, 1) \\ &\Downarrow \\ p(z_i | \beta) &= \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(z_i - \beta x_i)^2}; \end{aligned}$$

–  $\forall i = 1, \dots, n$  we know  $p(y_i | c, z_i)$ :

$$\begin{aligned} Y_i(\omega) &= \mathbf{1}_{(c, +\infty)}(Z_i(\omega)) = \begin{cases} 1 & \text{if } Z_i(\omega) > c \\ 0 & \text{otherwise} \end{cases} \\ &\Downarrow \\ p(y_i) &= \mathbb{P}(Y_i = y_i) = \mathbb{P}(\mathbf{1}_{(c, +\infty)}(Z_i) = y_i) = \\ &= \begin{cases} \mathbb{P}(\mathbf{1}_{(c, +\infty)}(Z_i) = 1) & \text{if } y_i = 1 \\ \mathbb{P}(\mathbf{1}_{(c, +\infty)}(Z_i) = 0) & \text{if } y_i = 0 \\ 0 & \text{otherwise} \end{cases} = \\ &= \begin{cases} \mathbb{P}(\{Z_i > c\}) & \text{if } y_i = 1 \\ \mathbb{P}(\{Z_i > c\}^C) & \text{if } y_i = 0 \\ 0 & \text{otherwise} \end{cases} = \\ &= (y_i \mathbb{P}(\{Z_i > c\}) + (1 - y_i) \mathbb{P}(\{Z_i > c\}^C)) \mathbf{1}_{\{0,1\}}(y_i), \end{aligned}$$

hence  $Y_i \sim \text{Bernoulli}(\mathbb{P}(Z_i > c))$ .

It follows that, conditionally on  $Z_i, c$ , the r.v.  $Y_i$  is no more *random* and it holds<sup>1</sup>

$$p(y_i | c, z_i) = \left( y_i \mathbb{1}_{(-\infty, z_i)}(c) + (1 - y_i) \mathbb{1}_{(-\infty, z_i)^c}(c) \right) \mathbb{1}_{\{0,1\}}(y_i).$$

In order to obtain (and sample) from the full conditionals we assume  $\beta$  and  $c$  a priori independent. The full conditional distribution  $p(\beta | y_{1:n}, z_{1:n}, c)$  can be obtained just from  $p(z_i | \beta)$ , indeed

$$\begin{aligned} p(\beta | y_{1:n}, z_{1:n}, c) &= \frac{p(\beta, y_{1:n}, z_{1:n}, c)}{p(y_{1:n}, z_{1:n}, c)} \frac{p(\beta, z_{1:n}, c)}{p(\beta, z_{1:n}, c)} \frac{p(\beta, c)}{p(\beta, c)} \frac{p(c)}{p(c)} \propto \\ &\propto \frac{p(\beta, y_{1:n}, z_{1:n}, c)}{p(\beta, z_{1:n}, c)} \frac{p(\beta, z_{1:n}, c)}{p(\beta, c)} \frac{p(\beta, c)}{p(c)} = \\ &= p(y_{1:n} | \beta, c, z_{1:n}) p(z_{1:n} | \beta, c) p(\beta | c) \propto \\ &\propto p(z_{1:n} | \beta) p(\beta). \end{aligned}$$

So we can write explicitly

$$\begin{aligned} p(\beta | y_{1:n}, z_{1:n}, c) &\propto p(z_{1:n} | \beta) p(\beta) = \\ &= \prod_{i=1}^n p(z_i | \beta) p(\beta) \propto \\ &\propto \exp \left( -\frac{1}{2} \sum_{i=1}^n (z_i - \beta x_i)^2 \right) \exp \left( -\frac{1}{2} \frac{1}{\sigma_\beta^2} \beta^2 \right) = \\ &= \exp \left( -\frac{1}{2} \left( \beta^2 \sum_{i=1}^n x_i^2 + \sum_{i=1}^n z_i^2 - 2\beta \sum_{i=1}^n x_i z_i + \beta^2 \frac{1}{\sigma_\beta^2} \right) \right) = \\ &= \exp \left( -\underbrace{\left( \sum_{i=1}^n x_i^2 + \frac{1}{\sigma_\beta^2} \right)}_{\stackrel{\text{def}}{=} (\sigma_{\beta,n}^2)^{-1}} \frac{\beta^2}{2} + \underbrace{\left( \sum_{i=1}^n x_i z_i \right)}_{\stackrel{\text{def}}{=} \frac{\mu_{\beta,n}}{\sigma_{\beta,n}^2}} \beta \right), \end{aligned}$$

where from the 1<sup>st</sup> to the 2<sup>nd</sup> line we used  $(Z_i | \beta)_{i=1}^n$  independent, identically distributed r.v.'s. So we can conclude that

$$\begin{aligned} \beta | y_{1:n}, z_{1:n}, c &\sim \mathcal{N} \left( \mu_{\beta,n}, \sigma_{\beta,n}^2 \right) \text{ with } \begin{cases} \sigma_{\beta,n}^2 = \left( \sum_{i=1}^n x_i^2 + \frac{1}{\sigma_\beta^2} \right)^{-1} \\ \mu_{\beta,n} = \sigma_{\beta,n}^2 \left( \sum_{i=1}^n x_i z_i \right) \end{cases} \\ &\Downarrow \\ p(\beta | y_{1:n}, z_{1:n}, c) &= \frac{1}{\sqrt{2\pi\sigma_{\beta,n}^2}} \exp \left( -\frac{1}{2\sigma_{\beta,n}^2} (\beta - \mu_{\beta,n})^2 \right). \end{aligned}$$

□

## Point b.

Assuming  $c \sim \mathcal{N}(0, \sigma_c^2)$ , show that  $p(c | y_{1:n}, z_{1:n}, \beta)$  is a constrained normal density, i.e. proportional to a normal density but constrained to lie in an interval. Similarly, show that  $p(z_i | y_{1:n}, z_{-i}, \beta, c)$  is proportional to a normal density but constrained to be either above  $c$  or below  $c$ , depending on  $y_i$ .

<sup>1</sup>We replace  $\mathbb{P}(\{z_i > c\})$  with  $\mathbb{1}_{(-\infty, z_i)}(c)$  because we will use this characterization afterwards.

**Hint:** A constrained, or truncated, normal random variable  $V$  is obtained by restricting a normally distributed random variable  $\mathcal{N}(\mu, \tau^2)$  to lie in an interval  $(a, b)$ , with possibly  $a = -\infty$  or  $b = +\infty$ . We use the notation  $V \sim \mathcal{TN}_{(a,b)}(\mu, \tau^2)$ . It holds:

- $p(v | \mu, \tau^2, a, b) = \frac{1}{C} \frac{1}{\sqrt{2\pi\tau^2}} \exp\left(-\frac{1}{2\tau^2}(v - \mu)^2\right) \mathbb{1}_{(a,b)}(v)$ , where  $C = \Phi\left(\frac{b-\mu}{\tau}\right) - \Phi\left(\frac{a-\mu}{\tau}\right)$  being  $\Phi(\cdot)$  the cdf of the standard normal distribution. By definition, it holds  $\Phi\left(\frac{b-\mu}{\tau}\right) = 1$  if  $b = \infty$  and  $\Phi\left(\frac{a-\mu}{\tau}\right) = 0$  if  $a = -\infty$ .
- Sampling can be performed thanks to the function `rtruncnorm(n, a, b, mean, sd)` from the package `truncnorm` [<https://cran.r-project.org/web/packages/truncnorm/truncnorm.pdf>]. This function receives in input the number of desired samples ( $n$ ) and the four parameters specifying the distribution of  $V$ :  $a, b, \mu, \tau$ . Pay attention that it takes as last inputs the mean  $\mu$  and the standard deviation  $\tau$  (not the variance  $\tau^2$ ) of the un-truncated normal density.

As before, the full conditional distribution  $p(c | y_{1:n}, z_{1:n}, \beta)$  can be obtained just from  $p(y_i | c, z_i)$ , indeed

$$\begin{aligned} p(c | y_{1:n}, z_{1:n}, \beta) &= \frac{p(c, y_{1:n}, z_{1:n}, \beta)}{p(y_{1:n}, z_{1:n}, \beta)} \frac{p(\beta, c, z_{1:n})}{p(\beta, c, z_{1:n})} \frac{p(c, \beta)}{p(c, \beta)} \frac{p(\beta)}{p(\beta)} \propto \\ &\propto \frac{p(c, y_{1:n}, z_{1:n}, \beta)}{p(\beta, c, z_{1:n})} \frac{p(\beta, c, z_{1:n})}{p(c, \beta)} \frac{p(c, \beta)}{p(\beta)} = \\ &= p(y_{1:n} | \beta, c, z_{1:n}) p(z_{1:n} | \beta, c) p(c | \beta) \propto \\ &\propto p(y_{1:n} | c, z_{1:n}) p(c). \end{aligned}$$

So we can write explicitly

$$\begin{aligned} p(c | y_{1:n}, z_{1:n}, \beta) &\propto p(y_{1:n} | c, z_{1:n}) p(c) = \\ &= \prod_{i=1}^n p(y_i | c, z_i) p(c) \propto \\ &\propto \exp\left(-\frac{1}{2} \frac{1}{\sigma_c^2} c^2\right) \prod_{i=1}^n \left(y_i \mathbb{1}_{(-\infty, z_i)}(c) + (1 - y_i) \mathbb{1}_{(-\infty, z_i)^c}(c)\right) \mathbb{1}_{\{0,1\}}(y_i) = \\ &= \exp\left(-\frac{1}{2} \frac{1}{\sigma_c^2} c^2\right) \prod_{i=1, \dots, n | y_i=1} \mathbb{1}_{(-\infty, z_i)}(c) \cdot \prod_{i=1, \dots, n | y_i=0} \mathbb{1}_{[z_i, +\infty)}(c) = \\ &= \exp\left(-\frac{1}{2} \frac{1}{\sigma_c^2} c^2\right) \mathbb{1}_{(-\infty, \min(z_i | i \in \{1, \dots, n\}, y_i=1))}(c) \mathbb{1}_{[\max(z_i | i \in \{1, \dots, n\}, y_i=0), +\infty)}(c), \end{aligned}$$

where from the 1<sup>st</sup> to the 2<sup>nd</sup> line we used  $(Y_i | c, z_i)_{i=1}^n$  independent, identically distributed r.v.'s. More compactly, defining

$$\begin{aligned} a_n &\stackrel{\text{def}}{=} \max(z_i | i \in \{1, \dots, n\}, y_i = 0) \text{ and} \\ b_n &\stackrel{\text{def}}{=} \min(z_i | i \in \{1, \dots, n\}, y_i = 1), \end{aligned}$$

one has

$$p(c | y_{1:n}, z_{1:n}, \beta) \propto \exp\left(-\frac{1}{2} \frac{1}{\sigma_c^2} c^2\right) \mathbb{1}_{[a_n, b_n)}(c),$$

where, for a good definition, we are using  $a_n < b_n$  which is clearly true because, if  $a_n, b_n$  are finite,  $\forall i, j \in \{1, \dots, n\}$  such that  $y_i = 0, y_j = 1$ ,  $(-\infty, c] \ni z_i < z_j \in (c, +\infty)$ .

First of all we have to observe that the indicator function constrains  $c \in [a_n, b_n)$ , but it is equivalent to  $c \in (a_n, b_n)$  because our  $p(c | y_{1:n}, z_{1:n}, \beta)$  is a density function with respect to the lebesgue measure on  $\mathbb{R}$  so each point has measure 0 (so does  $\{a_n\}$ ).

Then, let us observe that this conditional density is proportional to the kernel of a gaussian (evaluated in  $c$ ) multiplied by an indicator function (also evaluated in  $c$ ), which constrains the domain to an interval (not necessarily limited, possibly  $a_n = -\infty$  or  $b_n = +\infty$ ).

So completing the function  $\exp\left(-\frac{1}{2}\frac{1}{\sigma_c^2}c^2\right)\mathbb{1}_{(a_n,b_n)}(c)$  to a density one obtains

$$\begin{aligned} p(c | y_{1:n}, z_{1:n}, \beta) &= \frac{1}{\Phi\left(\frac{b_n}{\sigma_c}\right) - \Phi\left(\frac{a_n}{\sigma_c}\right)} \frac{1}{\sqrt{2\pi}\sigma_c^2} \exp\left(-\frac{1}{2}\frac{1}{\sigma_c^2}c^2\right) \mathbb{1}_{(a_n,b_n)}(c) \\ &\Downarrow \\ c | y_{1:n}, z_{1:n}, \beta &\sim \mathcal{TN}_{(a_n,b_n)}\left(0, \sigma_c^2\right). \end{aligned}$$

Similarly

$$\begin{aligned} p(z_i | y_{1:n}, z_{-i}, \beta, c) &= \frac{p(z_i, y_{1:n}, z_{-i}, \beta, c)}{p(y_{1:n}, z_{-i}, \beta, c)} \frac{p(z_i, z_{-i}, \beta, c)}{p(z_i, \beta, c)} \frac{p(z_i, \beta, c)}{p(\beta, c)} \propto \\ &\propto \frac{p(z_i, y_{1:n}, z_{-i}, \beta, c)}{p(z_i, z_{-i}, \beta, c)} \frac{p(z_i, z_{-i}, \beta, c)}{p(z_i, \beta, c)} \frac{p(z_i, \beta, c)}{p(\beta, c)} = \\ &= p(y_{1:n} | z_{1:n}, \beta, c) p(z_{-i} | z_{-i}, \beta, c) p(z_i | \beta, c) \propto \\ &\propto p(y_{1:n} | z_{1:n}, c) p(z_i | \beta) \propto \\ &\propto \prod_{j=1}^n p(y_j | z_j, c) p(z_i | \beta) \propto \\ &\propto p(y_i | z_i, c) p(z_i | \beta) \propto \\ &\propto \left( y_i \underbrace{\mathbb{1}_{(-\infty, z_i)}(c)}_{= \mathbb{1}_{(c, +\infty)}(z_i)} + (1 - y_i) \underbrace{\mathbb{1}_{(-\infty, z_i)^c}(c)}_{= \mathbb{1}_{(-\infty, c]}(z_i)} \right) \mathbb{1}_{\{0,1\}}(y_i) \exp\left(-\frac{1}{2}(z_i - \beta x_i)^2\right) = \\ &= \begin{cases} \mathbb{1}_{(c, +\infty)}(z_i) \exp\left(-\frac{1}{2}(z_i - \beta x_i)^2\right) & \text{if } y_i = 1 \\ \mathbb{1}_{(-\infty, c]}(z_i) \exp\left(-\frac{1}{2}(z_i - \beta x_i)^2\right) & \text{if } y_i = 0 \end{cases}. \end{aligned}$$

As before, this conditional density is proportional to the kernel of a gaussian (evaluated in  $z_i$ ) multiplied by an indicator function (also evaluated in  $z_i$ ) which constrains the domain to be  $(c, +\infty)$  or  $(-\infty, c]$  (equivalently  $(-\infty, c)$ , with the same motivation given above) depending on  $y_i$ .

In particular, completing to a density what we found

$$\begin{aligned} p(z_i | y_{1:n}, z_{-i}, \beta, c) &= \begin{cases} \frac{1}{1 - \Phi(c - x_i\beta)} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}(z_i - \beta x_i)^2\right) \mathbb{1}_{(c, +\infty)}(z_i) & \text{if } y_i = 1 \\ \frac{1}{\Phi(c - x_i\beta)} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}(z_i - \beta x_i)^2\right) \mathbb{1}_{(-\infty, c)}(z_i) & \text{if } y_i = 0 \end{cases} \\ &\Downarrow \\ Z_i | y_{1:n}, z_{-i}, \beta, c &\sim \begin{cases} \mathcal{TN}_{(c, +\infty)}(\beta x_i, 1) & \text{if } y_i = 1 \\ \mathcal{TN}_{(-\infty, c)}(\beta x_i, 1) & \text{if } y_i = 0 \end{cases}. \end{aligned}$$

□

### Point c.

Letting  $\sigma_\beta^2 = \sigma_c^2 = 16$ , implement a Gibbs sampling scheme that approximates the joint posterior distribution of  $Z_{1:n}, \beta$  and  $c$ . After a burnin of 1000, run the Gibbs sampler long enough so that the

effective sample sizes of all unknown parameters are greater than 1000 (including the  $Z_i$ 's). Compute the autocorrelation function of the parameters and discuss the mixing of the Markov chain.

The prior distributions of  $\beta$  and  $c$  are

$$\begin{aligned}\beta &\sim \mathcal{N}(0, \sigma_\beta^2), \\ c &\sim \mathcal{N}(0, \sigma_c^2).\end{aligned}$$

The full conditional distributions we found are the following

$$\begin{aligned}Z_i | y_i, \beta, c &\sim \begin{cases} \mathcal{TN}_{(c, +\infty)}(\beta x_i, 1) & \text{if } y_i = 1 \\ \mathcal{TN}_{(-\infty, c)}(\beta x_i, 1) & \text{if } y_i = 0 \end{cases}, \\ \beta | z_{1:n} &\sim \mathcal{N}(\mu_{\beta, n}, \sigma_{\beta, n}^2) \text{ with } \begin{cases} \sigma_{\beta, n}^2 = \left( \sum_{i=1}^n x_i^2 + \frac{1}{\sigma_\beta^2} \right)^{-1} \\ \mu_{\beta, n} = \sigma_{\beta, n}^2 \left( \sum_{i=1}^n x_i z_i \right) \end{cases}, \\ c | y_{1:n}, z_{1:n} &\sim \mathcal{TN}_{(a_n, b_n)}(0, \sigma_c^2) \text{ with } \begin{cases} a_n \stackrel{\text{def}}{=} \max(z_i | i \in \{1, \dots, n\}, y_i = 0) \text{ and} \\ b_n \stackrel{\text{def}}{=} \min(z_i | i \in \{1, \dots, n\}, y_i = 1) \end{cases}.\end{aligned}$$

```
library(coda)
library(truncnorm)
load(file = "divorce.RData")
set.seed(1)

# setting parameters
burnin = 1e3
tmax = burnin + 1e5
n = 25

# build and upload: beta, c, z_1:n, y_1:n
beta = c = matrix(0, tmax, 1)
z = y = matrix(0, tmax, n)
x = matrix(0, 1, n)
x[1, ] = divorce[, "X"]
y[1, ] = divorce[, "Y"]

# parameters for priors and full conditionals distributions
mu_beta = matrix(0, tmax, 1)
mu_beta[1] = 0
mu_c = 0
sigma_sq_beta = sigma_sq_c = 16
sigma_sq_beta_n = (sum(x^2) + (sigma_sq_beta)^(-1))^(-1)
a = matrix(-Inf, tmax, 1)
b = matrix(Inf, tmax, 1)

# prior samples
beta[1] = rnorm(1, mu_beta[1], sqrt(sigma_sq_beta))
c[1] = rnorm(1, mu_c, sqrt(sigma_sq_c))

# gibbs sampler
for (t in 2:(tmax)) {
  # z
  lower_bound = rep(c[t - 1], n)
  lower_bound[y[t - 1, ] == 0] = -Inf
  upper_bound = rep(c[t - 1], n)
  upper_bound[y[t - 1, ] == 1] = +Inf
  z[t, ] = rtruncnorm(n, lower_bound, upper_bound, beta[t - 1] * x, rep(1, n))

  # update y (redundant because they follow the behaviour of z which is sampled given y)
  y[t, ] = 1 * (z[t, ] > c[t - 1])
}
```

```

# beta
mu_beta[t] = sigma_sq_beta_n * sum(x * z[t, ])
beta[t] = dnorm(1, mu_beta[t], sqrt(sigma_sq_beta_n))

# c
a[t] = max(z[t, ][y[t, ] == 0])
b[t] = min(z[t, ][y[t, ] == 1])
c[t] = rtruncnorm(1, a[t], b[t], mu_c, sqrt(sigma_sq_c))

# re-update y (redundant because they follow the behaviour of z which is sampled given y)
y[t, ] = 1 * (z[t, ] > c[t])

# break if eff_size > 1000 (for all params)
if (t > burnin + 1 & (t %% 1000 == 0)) {
  if (effectiveSize(c[c((burnin + 1):t)]) > 1000 &
      # effectiveSize(beta[c((burnin + 1):t)]) > 1000 &
      prod(effectiveSize(z[c((burnin + 1):t), ])) > 1000) == 1)
  {
    c = c[c((burnin + 1):t)]
    beta = beta[c((burnin + 1):t)]
    z = z[c((burnin + 1):t), ]
    break
  }
}
}

# mcmc
params_mcmc = mcmc(cbind(beta, c, z))

```

The effective sample size of the parameters are

- $S_{eff}(c) = 1035.53$ ;
  - $S_{eff}(\beta) = 0$ ;
  - $S_{eff}(Z_{1:25}) = 8607.39, 12521.07, 8668.4, 9521.21, 8780.51, 8608.48, 13599.76, 8770.07, 7791.37$  .
- 13607.26, 12707.45, 11846.7, 12431.12, 11865.7, 11455.85, 12947.48, 8849.05, 12957.28]

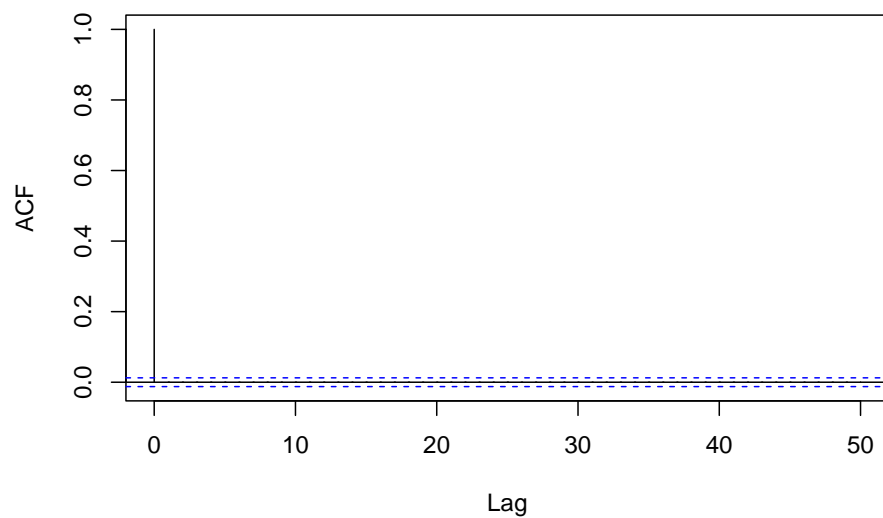
Below are the autocorrelation functions of the parameters.

- $\beta$  :

```

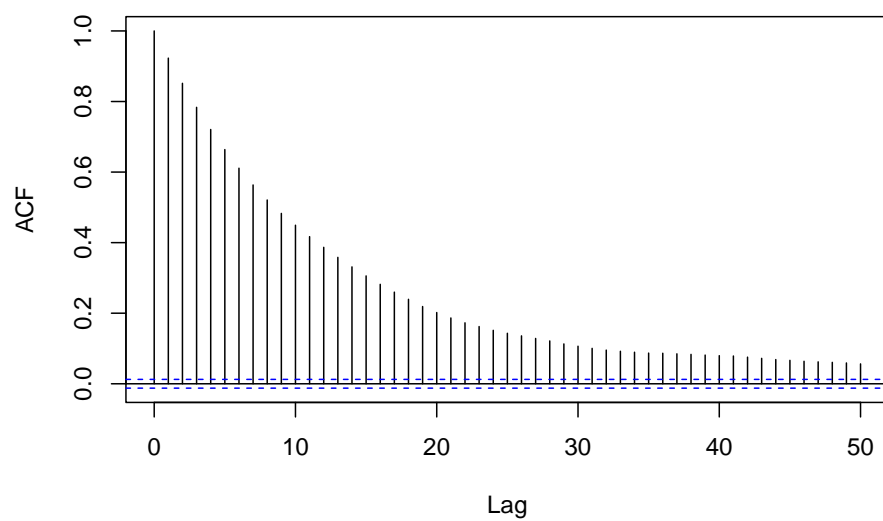
# autocorrelation function of beta
acf_beta = acf(mcmc(beta), lag.max = 50, main = "")

```



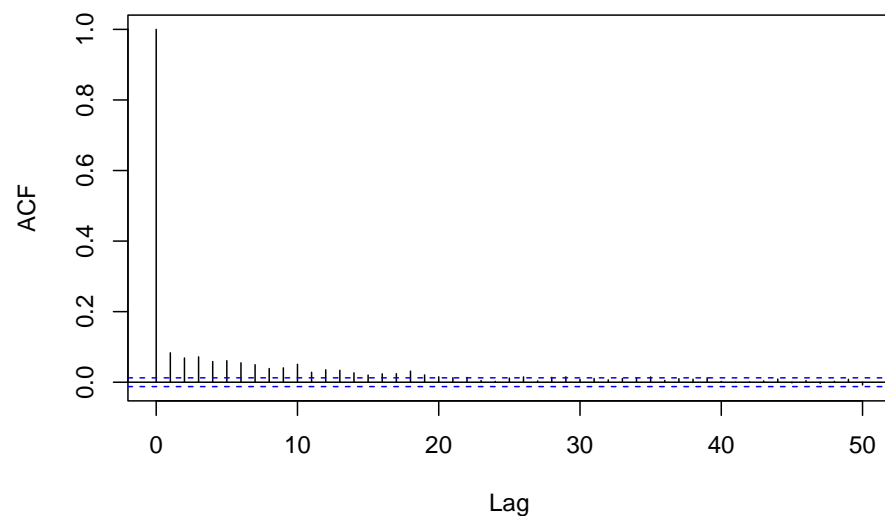
–  $c$  :

```
# autocorrelation function of c
acf_c = acf(mcmc(c), lag.max = 50, main = "")
```



–  $Z_{1:25}$  (taken  $i = 15$  as an example, so the autocorrelation function of  $Z_{15}$ ) :

```
# autocorrelation function of z (i = 15 taken as an example)
acf_z_15 = acf(mcmc(z[, 15]), lag.max = 50, main = "")
```



Discuss the mixing of the markov chain...

#### Point d.

Obtain a 95% posterior credible interval for  $\beta$ , as well as  $\mathbb{P}(\beta > 0 \mid y_{1:n})$ .

## QUESTION 2

#### Point a.

```
set.seed(123)
library(coda)
library(bayesplot)
library(ggplot2)
library(gridExtra)
library(lattice)
library(grid)
library(dplyr)

load(file = "schools.RData")
# Prior parameters
mu0 = 7
g20 = 5
t20 = 10
eta0 = 2
s20 = 15
nu0 = 2

# Number of schools. Y[, 1] are school ids
m = length(unique(Y[, 1]))

# Starting values - use sample mean and variance
sv = double(length = m)
ybar = double(length = m)
n = integer(length = m)
for (j in 1:m) {
  Y_j = Y[Y[, 1] == j, 2]
  ybar[j] = mean(Y_j)
  sv[j] = var(Y_j)
```



```

    n[j] = length(Y_j)
}

# Let initial theta estimates be the sample means
# Similarly, let initial values of sigma2, mu, and tau2 be "sample mean and variance"

theta = ybar
sigma2 = mean(sv)
mu = mean(theta)
tau2 = var(theta)

# MCMC
burnin = 1e3
S = 5e3
theta.mcmc = mcmc(matrix(nrow = S, ncol = m)) # declare them as mcmc objects
# Storing sigma, mu, theta
sigma2.mcmc = mcmc(double(length = S))
mu.mcmc = mcmc(double(length = S))
tau2.mcmc = mcmc(double(length = S))

for (s in 1:(burnin+S)) {
  # Sample thetas
  for (j in 1:m) {
    vtheta = 1 / (n[j] / sigma2 + 1 / tau2)
    etheta = vtheta * (ybar[j] * n[j] / sigma2 + mu / tau2)
    theta[j] = rnorm(1, etheta, sqrt(vtheta))
  }

  # Sample sigma2
  nun = nu0 + sum(n)
  ss = nu0 * s20

  # Pool variance
  for (j in 1:m) {
    ss = ss + sum((Y[, j] - theta[j])^2)
  }
  sigma2 = 1 / rgamma(1, nun / 2, ss / 2)

  # Sample mu
  vmu = 1 / (m / tau2 + 1 / g20)
  emu = vmu * (m * mean(theta) / tau2 + mu0 / g20)
  mu = rnorm(1, emu, sqrt(vmu))

  # Sample tau2
  etam = eta0 + m
  ss = eta0 * t20 + sum((theta - mu)^2)
  tau2 = 1 / rgamma(1, etam / 2, ss / 2)

  # Store params
  if(s > burnin){
    theta.mcmc[s-burnin, ] = theta
    sigma2.mcmc[s-burnin] = sigma2
    mu.mcmc[s-burnin] = mu
    tau2.mcmc[s-burnin] = tau2
  }
}

S_mu = effectiveSize(mu.mcmc)
S_sigma2 = effectiveSize(sigma2.mcmc)
S_tau2 = effectiveSize(tau2.mcmc)

writeLines(paste("The effective sample size for mu is S_eff =", round(S_mu, 0), "\nThe effective sample size for sigma^2 is S_eff ="))

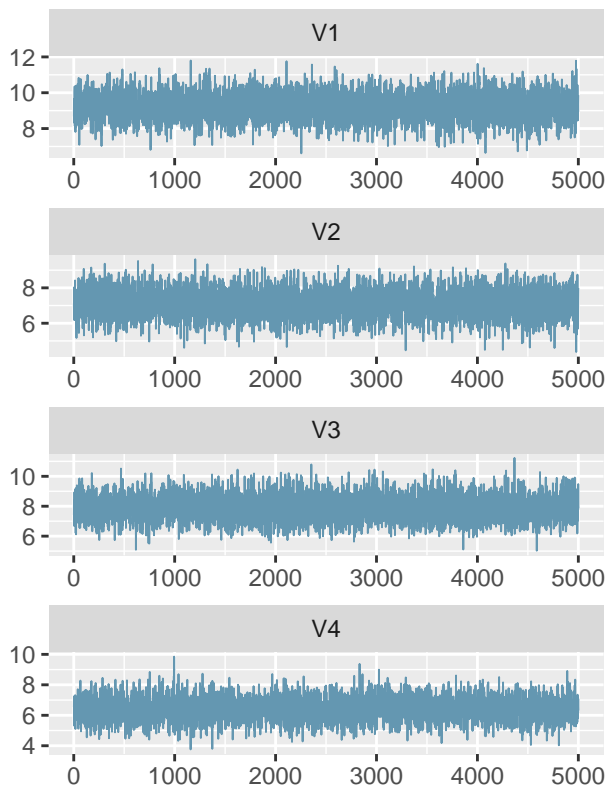
## The effective sample size for mu is S_eff = 4063
## The effective sample size for sigma^2 is S_eff = 4007
## The effective sample size for tau^2 is S_eff = 4164
S_theta = effectiveSize(theta.mcmc)

writeLines(paste("The effective sample size for theta", 1:8, "is S_eff =", round(S_theta, 0), "\n"))

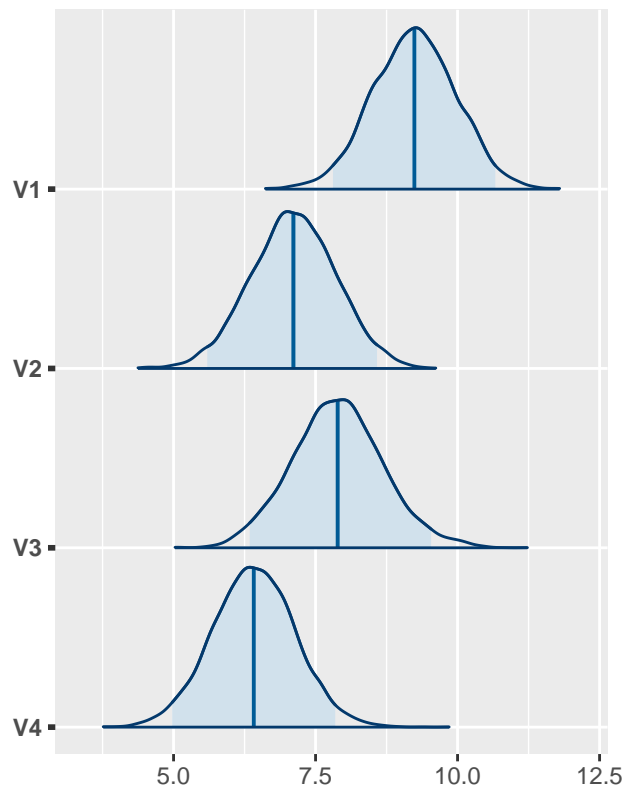
```

```
## The effective sample size for theta 1 is S_eff = 4581
##
## The effective sample size for theta 2 is S_eff = 4739
##
## The effective sample size for theta 3 is S_eff = 6882
##
## The effective sample size for theta 4 is S_eff = 4983
##
## The effective sample size for theta 5 is S_eff = 4448
##
## The effective sample size for theta 6 is S_eff = 5000
##
## The effective sample size for theta 7 is S_eff = 4765
##
## The effective sample size for theta 8 is S_eff = 5000
#theta
color_scheme_set("blue")
p1 <- mcmc_trace(as.data.frame(theta.mcmc), pars = c("V1", "V2", "V3", "V4"), facet_args = list(nrow = 4, labeller = label_parsed)) +
  ggtitle(expression(paste("Traceplots of ", theta[1:4]))) +
  scale_y_continuous(breaks = c(4, 6, 8, 10, 12, 14))
p2 <- mcmc_areas(as.data.frame(theta.mcmc), pars = c("V1", "V2", "V3", "V4"), prob = 0.95,
  prob_outer = 1,
  point_est = "mean") +
  ggtitle(expression(paste("Density plots of the posteriors for ", theta[1:4])))
grid.arrange(p1, p2, nrow = 1)
```

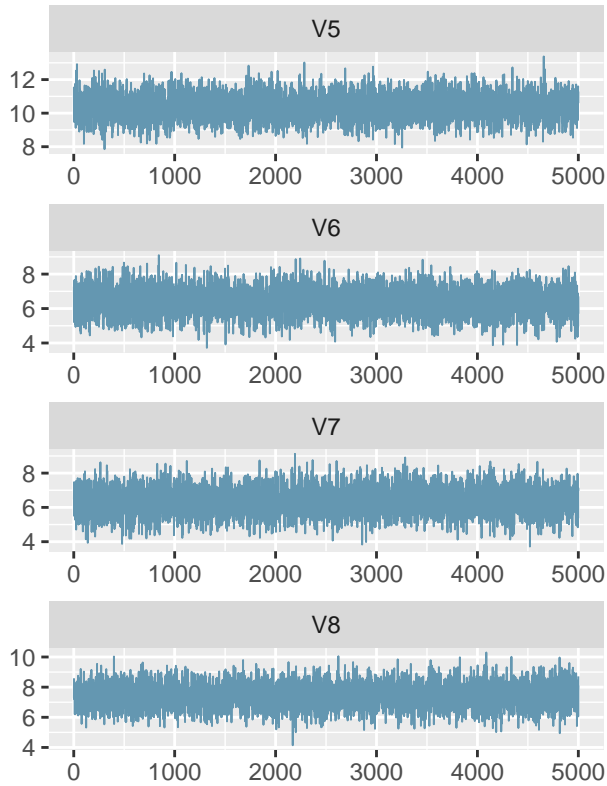
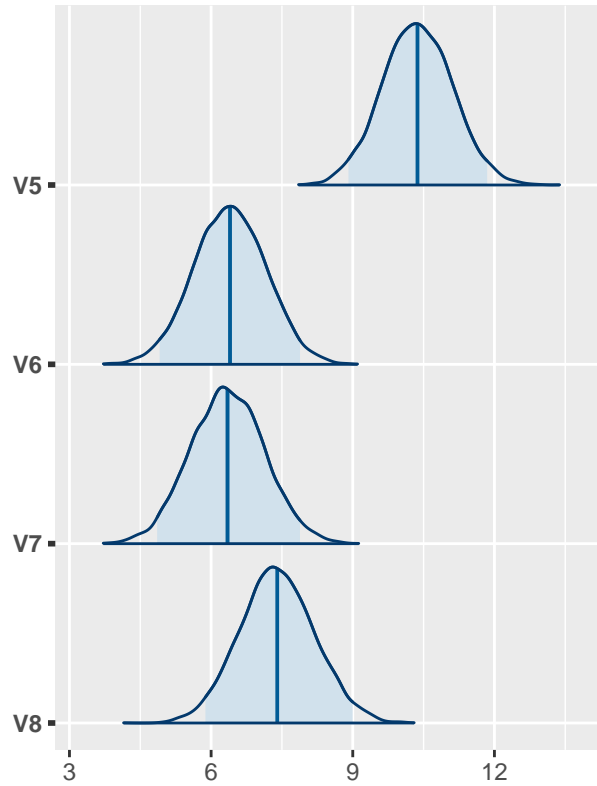
Traceplots of  $\theta_{1:4}$



Density plots of the posteriors for  $\theta_{1:4}$



```
p1 <- mcmc_trace(as.data.frame(theta.mcmc), pars = c("V5", "V6", "V7", "V8"), facet_args = list(nrow = 4, labeller = label_parsed)) +
  ggtitle(expression(paste("Traceplots of ", theta[5:8]))) +
  scale_y_continuous(breaks = c(4, 6, 8, 10, 12, 14))
p2 <- mcmc_areas(as.data.frame(theta.mcmc), pars = c("V5", "V6", "V7", "V8"), prob = 0.95,
  prob_outer = 1,
  point_est = "mean") +
  ggtitle(expression(paste("Density plots of the posteriors for ", theta[5:8])))
grid.arrange(p1, p2, nrow = 1)
```

Traceplots of  $\theta_{5:8}$ Density plots of the posteriors for  $\theta_{5:8}$ 

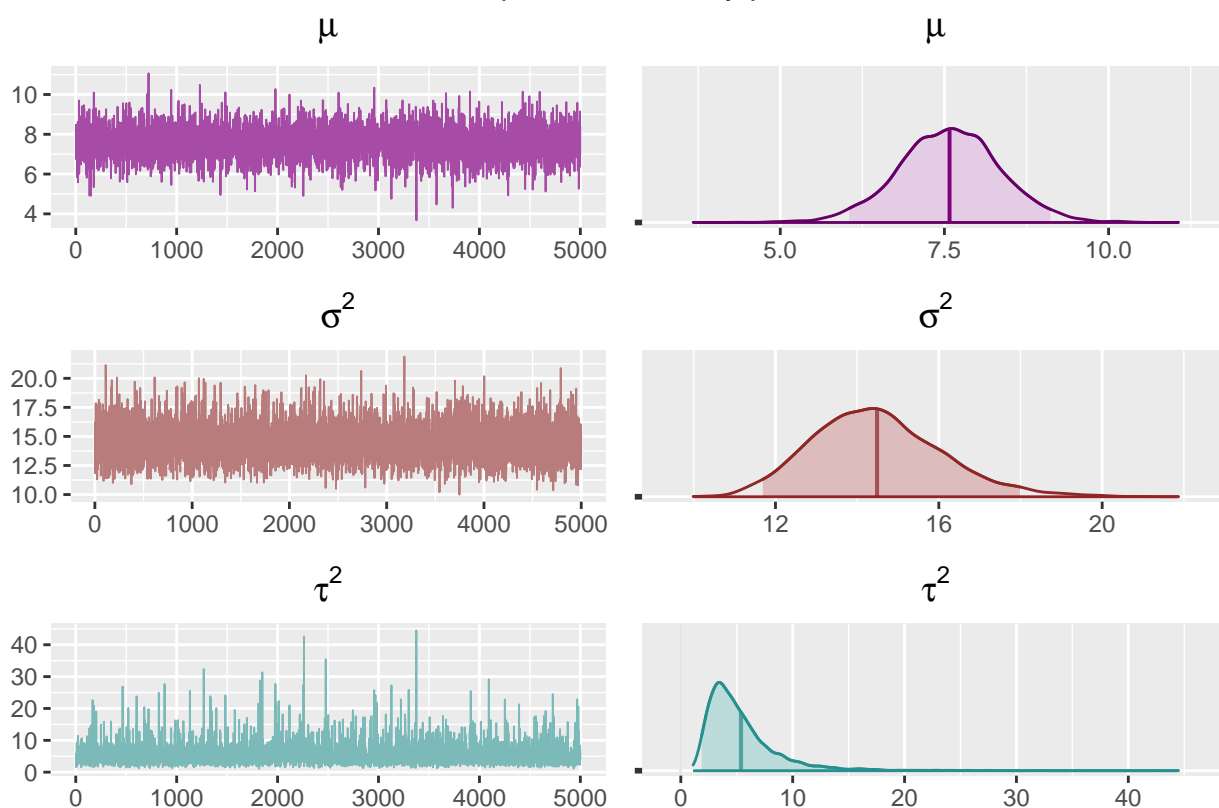
```
#comparison
theme_update(plot.title = element_text(hjust = 0.5))
color_scheme_set("purple")
q1 <- mcmc_trace(as.data.frame(mu.mcmc))+
  ggtitle(expression(mu))+
  yaxis_title(on = FALSE)
q2<- mcmc_areas(
  as.data.frame(mu.mcmc),
  prob = 0.95,
  prob_outer = 1,
  point_est = "mean")+
  ggtitle(expression(mu))+
  yaxis_text(on = FALSE)

color_scheme_set("red")
q3 <- mcmc_trace(as.data.frame(sigma2.mcmc))+
  ggtitle(expression(sigma^2))+
  yaxis_title(on = FALSE)
q4 <- mcmc_areas(
  as.data.frame(sigma2.mcmc),
  prob = 0.95,
  prob_outer = 1,
  point_est = "mean")+
  ggtitle(expression(sigma^2))+
  yaxis_text(on = FALSE)

color_scheme_set("teal")
q5 <- mcmc_trace(as.data.frame(tau2.mcmc))+
  ggtitle(expression(tau^2))+
  yaxis_title(on = FALSE)
q6 <- mcmc_areas(
  as.data.frame(tau2.mcmc),
  prob = 0.95,
  prob_outer = 1,
  point_est = "mean")+
  ggtitle(expression(tau^2))+
```

```
yaxis_text(on = FALSE)
grid.arrange(q1, q2, q3, q4, q5, q6, nrow = 3, top=textGrob("Traceplots and density plots"))
```

Traceplots and density plots



## Point b.

Therefore we can obtain the 95%-quantile confidence regions of the posterior densities for each parameter.

```
library(invgamma)
#quantile-based 95% confidence regions for the MCMC
alpha = 0.05

a_mu = summary(mu.mcmc, quantiles= c(alpha/2, 1-alpha/2))
print(paste("The quantile-based 95%-confidence region for mu is [", a_mu$quantiles[1], ",", a_mu$quantiles[2], "] and the posterior mean is ", a_mu$mean))

## [1] "The quantile-based 95%-confidence region for mu is [ 6.05099577534956 , 9.11350999134755 ] and the posterior mean is 7.578"

a_sigma = summary(sigma2.mcmc, quantiles= c(alpha/2, 1-alpha/2))
print(paste("The quantile-based 95%-confidence region for sigma^2 is [", a_sigma$quantiles[1], ",", a_sigma$quantiles[2], "] and the posterior mean is ", a_sigma$mean))

## [1] "The quantile-based 95%-confidence region for sigma^2 is [ 11.6912071248673 , 17.9929876607772 ] and the posterior mean is 14.842"

a_tau = summary(tau2.mcmc, quantiles= c(alpha/2, 1-alpha/2))
print(paste("The quantile-based 95%-confidence region for tau^2 is [", a_tau$quantiles[1], ",", a_tau$quantiles[2], "] and the posterior mean is ", a_tau$mean))

## [1] "The quantile-based 95%-confidence region for tau^2 is [ 1.86296816063879 , 14.0038507066232 ] and the posterior mean is 5.342"

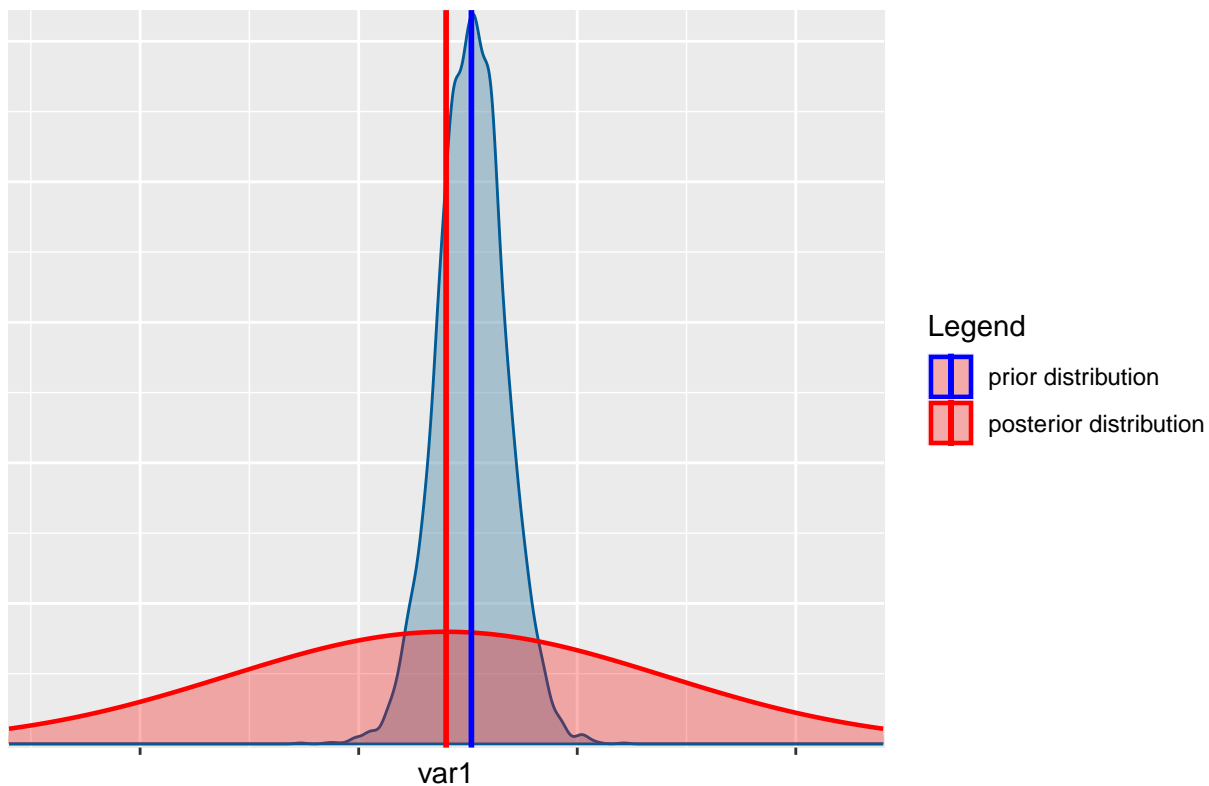
#density plots comparison
x1 = seq(mu0 - 2*g20, mu0 + 2*g20, length = 200)
x2 = seq(0.00001, 60, length = 200)
x3 = seq(a_sigma$quantiles[1] - 10, 50, length = 200)
norm_vals = dnorm(x1, mu0, g20)
invG_tau = dinvgamma(x2, eta0/2, eta0*t20/2)
invG_sigma = dinvgamma(x3, nu0/2, nu0*s20/2)
priors = data.frame(x1, x2, x3, norm_vals, invG_sigma, invG_tau)
```

```

color_scheme_set("blue")
mcmc_dens(as.data.frame(mu.mcmc), alpha = 0.5)+
  axis_text(on = FALSE)+
  ggtitle(expression(paste("Posterior distribution of ", mu))) +
  geom_area(data = priors, aes(x= x1, y=norm_vals, col = "prior"), fill = "red", alpha = 0.3, size = 0.8) +
  geom_vline(aes(xintercept = a_mu$statistics[1], col = "posterior"), size = 1)+
  geom_vline(aes(xintercept = mu0, col = "prior"), size = 1)+
  scale_color_manual(name = "Legend", values = c("prior" = "red", "posterior" = "blue"),
    labels = c("prior distribution", "posterior distribution"))

```

Posterior distribution of  $\mu$

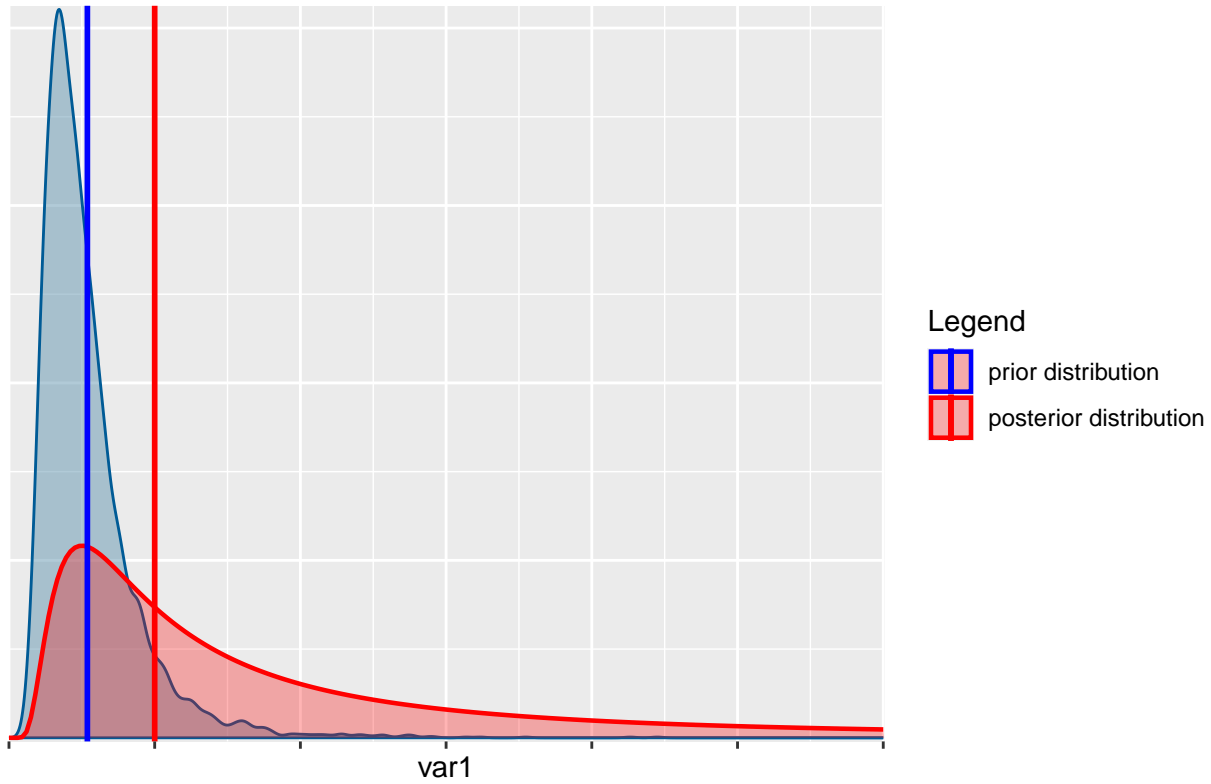


```

mcmc_dens(as.data.frame(tau2.mcmc), alpha = 0.5)+
  axis_text(on = FALSE)+
  ggtitle(expression(paste("Posterior distribution of ", tau^2))) +
  geom_area(data = priors, aes(x= x2, y=invG_tau, col = "prior"), fill = "red", alpha = 0.3, size = 0.8) +
  geom_vline(aes(xintercept = a_tau$statistics[1], col = "posterior"), size = 1)+
  geom_vline(aes(xintercept = t20, col = "prior"), size = 1)+
  scale_color_manual(name = "Legend", values = c("prior" = "red", "posterior" = "blue"),
    labels = c("prior distribution", "posterior distribution"))

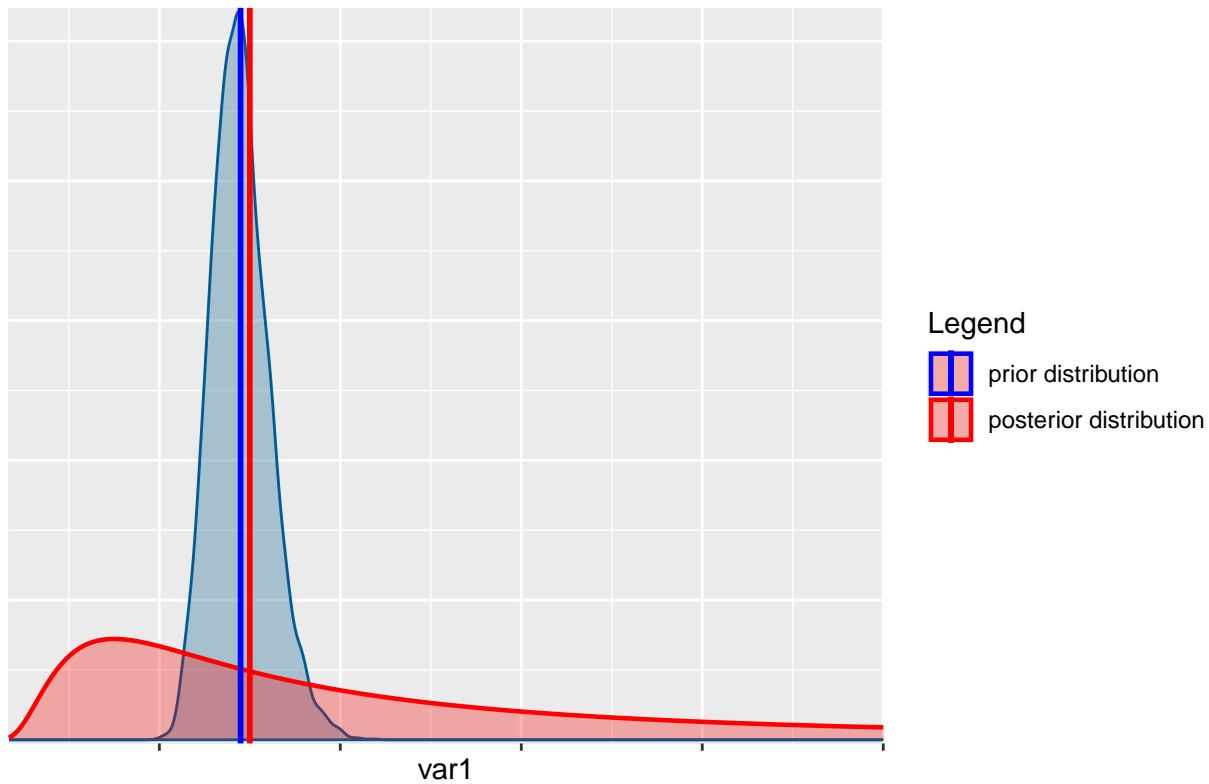
```

## Posterior distribution of $\tau^2$



```
mcmc_dens(as.data.frame(sigma2.mcmc), alpha = 0.5)+
  axis_text(on = FALSE)+
  ggtitle(expression(paste("Posterior distribution of ", sigma^2))) +
  geom_area(data = priors, aes(x= x3, y=invG_sigma, col = "prior"), fill = "red", alpha = 0.3, size = 0.8) +
  geom_vline(aes(xintercept = a_sigma$statistics[1], col = "posterior"), size = 1)+
  geom_vline(aes(xintercept = s20, col = "prior"), size = 1)+
  scale_color_manual(name = "Legend", values = c("prior" = "red", "posterior" = "blue"),
    labels = c("prior distribution", "posterior distribution"))
```

Posterior distribution of  $\sigma^2$



Point c.

Point d.

```
a_theta = summary(theta.mcmc)
post_means = a_theta$statistics[1:8, 1]

sample.theta9 = rep(0, S)
sample.y7 = rep(0, S)
sample.y9 = rep(0, S)

for (i in 1:S){
  sample.theta9[i] = rnorm(1, mu, sqrt(tau2.mcmc[i, drop = TRUE]))
  sample.y7[i] = rnorm(1, theta.mcmc[i, 7, drop = TRUE], sqrt(sigma2.mcmc[i, drop = TRUE]))
  sample.y9[i] = rnorm(1, sample.theta9[i], sqrt(sigma2.mcmc[i, drop = TRUE]))
}

mc_theta = sum(sample.theta9 > theta.mcmc[1:S, 7]) / S
mc_y = sum(sample.y9 > sample.y7)/S
writeLines(paste("The Monte Carlo estimate of the posterior probability of {theta9 > theta7} is:", mc_theta, "\n\nThe Monte Carlo estimate of the posterior predictive probability of {Y9 > Y7} is:", mc_y))

## The Monte Carlo estimate of the posterior probability of {theta9 > theta7} is: 0.8128
## The Monte Carlo estimate of the posterior predictive probability of {Y9 > Y7} is: 0.6484
```

Point e.

```
library(ggplot2)

mean_plt = data.frame(post_means, ybar, names = 1:8)

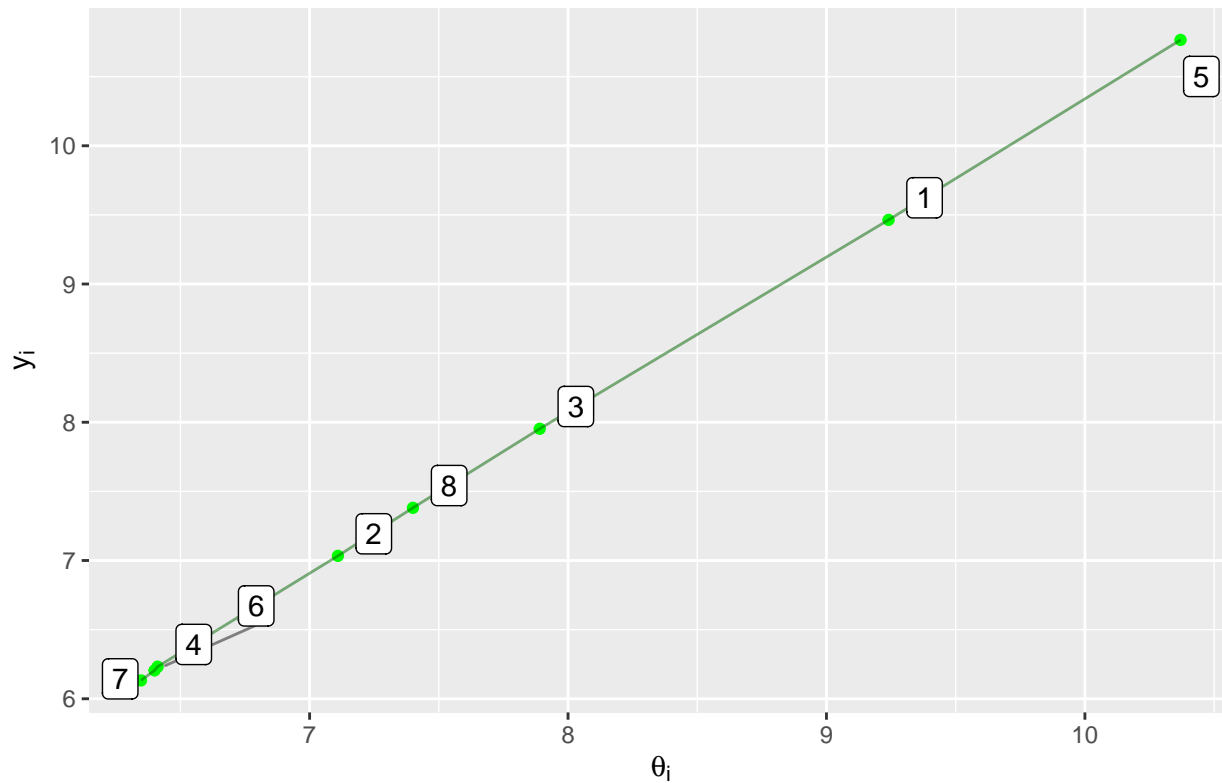
mean_plt %>% ggplot(aes(x=post_means, y = ybar))+
  geom_point(col = "green")+
  geom_line(col = "darkgreen", alpha = 0.5)+
```

```

ylab(expression(y[i])) +
xlab(expression(paste(theta[i]))) +
ggtitle(expression(paste("Posterior mean parameters ", theta[i], " VS sample means ", y[i])))+
scale_color_discrete(guide = "none")+
geom_label_repel(aes(label = names),
                 box.padding = 0.35,
                 point.padding = 0.5,
                 segment.color = 'grey50')

```

Posterior mean parameters  $\theta_i$  VS sample means  $y_i$



```
total.sample.mean = sum(n*ybar)/sum(n)
```

```
writeLines(paste("The total sample mean is: ", round(total.sample.mean, 2), "\nThe posterior expectation of the Grand Mean is: ", r
```

```
## The total sample mean is: 7.69
```

```
## The posterior expectation of the Grand Mean is: 7.58
```

```
## So they only differ by 1.47 % of the former
```