MATH3871 - Assignment

Yue Yu - z5359109

T3 2024

Part I: Frequentist analysis

1).

```r
wine = read.csv("winequality-red.csv")
anyNA(wine)
wine = na.omit(wine)
```

2).

```r
wine$good = ifelse(wine$quality >= 6.5, 1, 0)
```

3).

```r
model = glm(good ~. - quality, data = wine, family = binomial)
mleest = coef(model)
```

Part II: Bayesian analysis

1).

We start from the density function of logistic regression model:

$$P(y_i = 1|\beta, \mathbf{x}_i) = \frac{\exp(\mathbf{x}_i\beta)}{1+\exp(\mathbf{x}_i\beta)}.$$

$$\begin{aligned} P(y_i|\beta, \mathbf{x}_i) &= P(y_i = 1|\beta, \mathbf{x}_i)^{y_i}[1 - P(y_i = 1|\beta, \mathbf{x}_i)]^{(1-y_i)} \\ &= \left[\frac{\exp(\mathbf{x}_i\beta)}{1+\exp(\mathbf{x}_i\beta)}\right]^{y_i}\left[\frac{1}{1+\exp(\mathbf{x}_i\beta)}\right]^{(1-y_i)} \\ &= \frac{\exp(y_i\mathbf{x}_i\beta)}{1+\exp(\mathbf{x}_i\beta)}. \end{aligned}$$

Then we compute the likelihood function $L(y|\beta, \mathbf{x})$

$$L(y|\beta, \mathbf{x}) = \prod_{i=1}^{n} P(y_i|\beta, \mathbf{x}_i).$$

Hence the log-likelihood being

$$\begin{aligned} \log L(y|\beta, \mathbf{x}) &= \log \prod_{i=1}^{n} P(y_i|\beta, \mathbf{x}_i) \\ &= \sum_{i=1}^{n} \log P(y_i|\mathbf{x}_i) \\ &= \sum_{i=1}^{n} \log \frac{\exp(y_i\mathbf{x}_i\beta)}{1+\exp(\mathbf{x}_i\beta)} \\ &= \sum_{i=1}^{n} y_i\mathbf{x}_i\beta - \log\left[1+\exp(\mathbf{x}_i\beta)\right]. \end{aligned}$$

2).

We know that the posterior is

$$\pi(\beta|y, \mathbf{x}) \propto L(y|\beta, \mathbf{x})\pi(\beta)$$

Hence the log-posterior can be obtained by

$$\begin{aligned} \log \pi(\beta|y, \mathbf{x}) &\propto \log[L(y|\beta, \mathbf{x})\pi(\beta)] \\ &= \log L(y|\beta, \mathbf{x}) + \log \pi(\beta), \end{aligned}$$

which is a sum of log-prior and log-likelihood. Since we've derived the log-likelihood in the previous part, we now compute the log-prior. Since $\alpha = 0$, the prior is

$$\pi(\beta) = \frac{1}{(2\pi)^{(k+1)/2}|\Omega|^{1/2}} \exp\left(-\frac{1}{2}\beta^T\Omega^{-1}\beta\right).$$

Hence substitute $\Omega = 100I_{k+1}$ the log-prior is

$$\begin{aligned}\log\pi(\beta) &= -\frac{k+1}{2}\log(2\pi) - \frac{1}{2}\log|\Omega| - \frac{1}{2}\beta^T\Omega^{-1}\beta \\ &= -\frac{k+1}{2}\log(2\pi) - \frac{1}{2}\log|100I_{k+1}| - \frac{1}{2}\beta^T(100I_{k+1})^{-1}\beta \\ &= -\frac{k+1}{2}\log(2\pi) - \frac{1}{2}\log 100 - \frac{1}{200}\beta^T\beta,\end{aligned}$$

and the log-posterior is

$$\begin{aligned}\log\pi(\beta|y,\mathbf{x}) &\propto \log L(y|\beta,\mathbf{x}) + \log\pi(\beta) \\ &= \sum_{i=1}^{n}\{y_i\mathbf{x}_i\beta - \log[1+\exp(\mathbf{x}_i\beta)]\} - \frac{k+1}{2}\log(2\pi) - \frac{1}{2}\log 100 - \frac{1}{200}\beta^T\beta.\end{aligned}$$

```r
lpost.LR <- function(beta,X,y) {

  # log-likelihood
  x_beta = X %*% beta
  log_L = sum(y * x_beta - log(1 + exp(x_beta)))

  # log-prior
  k = length(beta) - 1
  log_prior = - (k + 1) / 2 * log(2 * pi) - 0.5 * (k + 1) * log(100) - 1/200 * t(beta
    ) %*% beta

  # log posterior
  log_post = log_L + log_prior
  return(log_post)
}
```

3).

Note that for numeric stability, we usually compute

$$\log\left(\frac{\pi(\theta^*)q(\theta^*,\theta_t)}{\pi(\theta_t)q(\theta_t,\theta^*)}\right) = \log\pi(\theta^*) + \log q(\theta^*,\theta_t) - \log\pi(\theta_t) - \log q(\theta_t,\theta^*),$$

and compute $\log u$ with $u \sim U(0,1)$. Since in this case our proposal distribution

$$q(\beta|\beta_t) = N_{k+1}(\beta_t,\Sigma)$$

is symmetrical, we can conclude that

$$q(\theta^*,\theta_t) = q(\theta_t,\theta^*).$$

Hence,

$$\log\left(\frac{\pi(\theta^*)q(\theta^*,\theta_t)}{\pi(\theta_t)q(\theta_t,\theta^*)}\right) = \log\pi(\theta^*) - \log\pi(\theta_t),$$

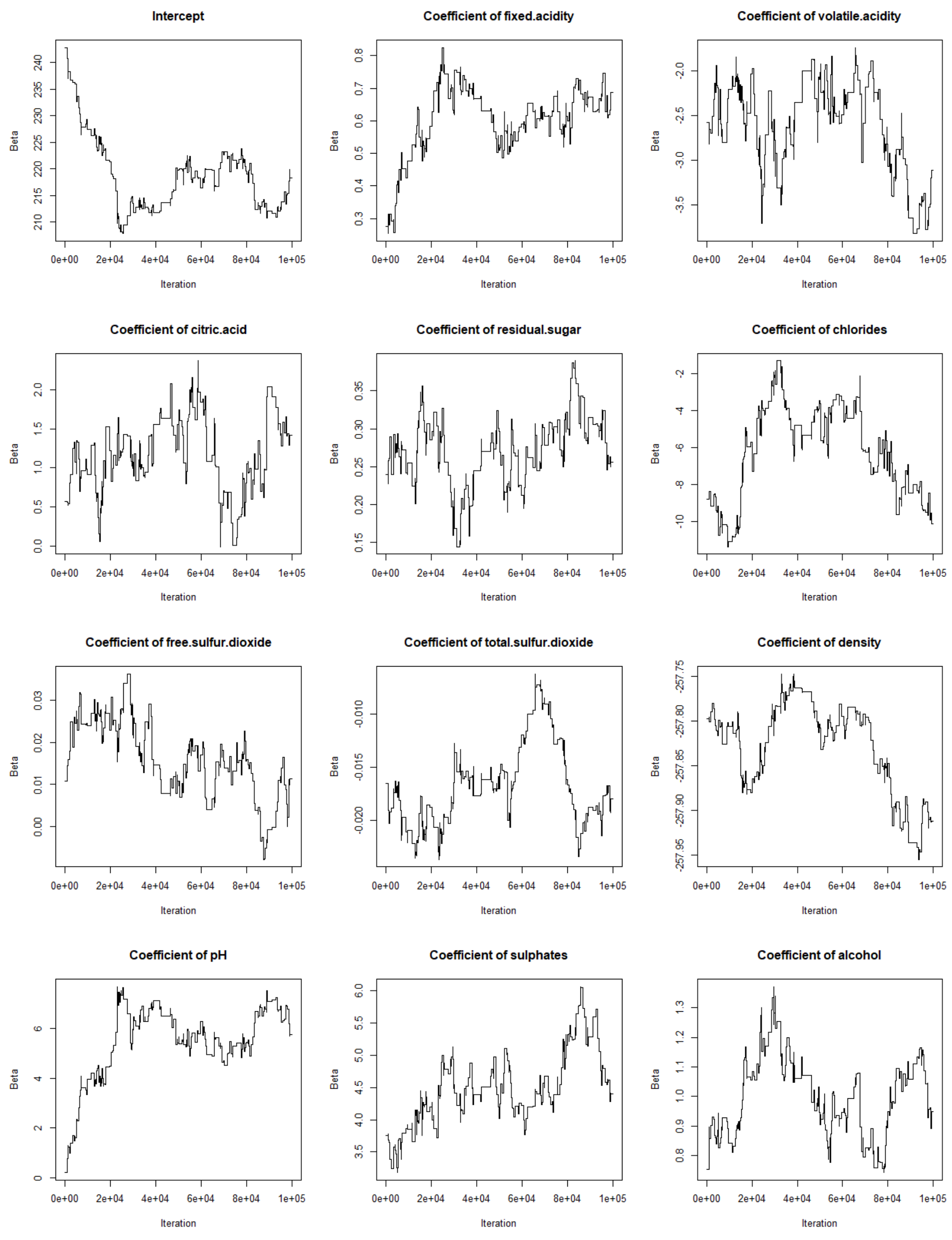
which is used in the code below as the acceptance criteria.

```r
mhmcmc <- function(y, X, B, nsims, Sigma) {

  k = length(B)
  beta_mat = matrix(NA, nrow = nsims, ncol = k)
  beta_mat[1, ] = B
  acc = 0

  for (t in 2:nsims) {

    # Curr and proposed Beta
    beta_curr = beta_mat[t - 1, ]
    beta_star = mvrnorm(1, mu = beta_curr, Sigma = Sigma)

    log_post_curr = lpost.LR(beta_curr, X, y)
    log_post_star = lpost.LR(beta_star, X, y)
    accprob = exp(log_post_star - log_post_curr)

    if (log(runif(1)) < log_post_star - log_post_curr) {
      beta_mat[t, ] = beta_star
      acc = acc + 1
    } else {
      beta_mat[t, ] = beta_curr
    }
  }
  mhout = list("beta_mat"=beta_mat, "accprob"=accprob, "beta_star" = beta_star, "acc"
    =acc)
  return(mhout)
}
```

4).

```r
# Covariance for proposal:
Sigma = diag(c(1100, 0.0015, 0.04, 0.05, 0.0005, 0.4, 0.00001, 0.000001, 0.0001,
    0.09, 0.03, 0.002)) #Do not modify
nsims = 1e5
X = cbind(Intercept = 1, as.matrix(wine[, !names(wine) %in% c("good", "quality")]))
y = wine$good
mhout1 = mhmcmc(y, X, mleest, nsims, Sigma)
```

a).


Intercept
Coefficient of fixed.acidity
Coefficient of volatile.acidity
Coefficient of citric.acid
Coefficient of residual.sugar
Coefficient of chlorides
Coefficient of free.sulfur.dioxide
Coefficient of total.sulfur.dioxide
Coefficient of density
Coefficient of pH
Coefficient of sulphates
Coefficient of alcohol
Beta
Iteration
0e+00
2e+04
4e+04
6e+04
8e+04
1e+05

```r
par(mfrow=c(4,3))
for (j in 1:length(mleest)) {
  name = if (j==1) "Intercept" else paste("Coefficient of", colnames(wine)[j-1])

  plot(mhout1$beta_mat[, j], type = "l", main = name, xlab = "Iteration", ylab = "
    Beta")
}
```

b).

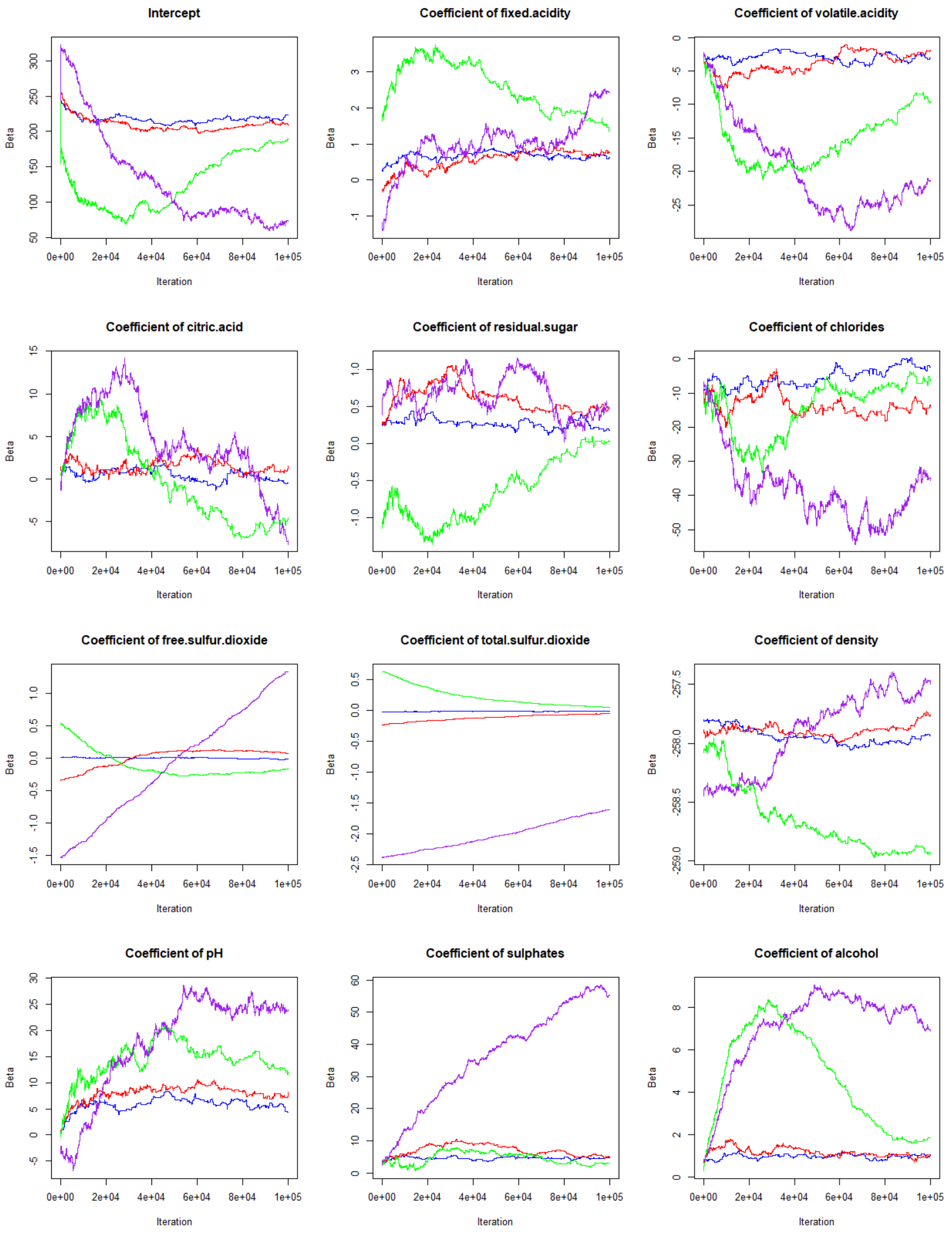
```r
mh4acc = mhout1$acc / nsims
```

$$mh4acc = 0.00232 = 0.2\%.$$

c).

From the graph, we can see there exist no clear sign of convergence for the coefficients, hence the proposed distribution is not very suitable. Also the acceptance is very low at 0.2%, and the initialization is far away from the MLE, hence the initialization is not suitable either. This conclusion aligns with the fact that the rate of acceptance is low.

5).


Intercept
Coefficient of fixed.acidity
Coefficient of volatile.acidity
Coefficient of citric.acid
Coefficient of residual.sugar
Coefficient of chlorides
Coefficient of free.sulfur.dioxide
Coefficient of total.sulfur.dioxide
Coefficient of density
Coefficient of pH
Coefficient of sulphates
Coefficient of alcohol
Beta
Iteration
0e+00
2e+04
4e+04
6e+04
8e+04
1e+05

The lines above have the initialization:

$$\begin{aligned} blue &: mleest, \\ red &: mleest + rnorm(length(mleest), 0, 0.5), \\ green &: mleest + rnorm(length(mleest), 0, 1), \\ purple &: mleest + rnorm(length(mleest), 0, 1.5). \end{aligned}$$

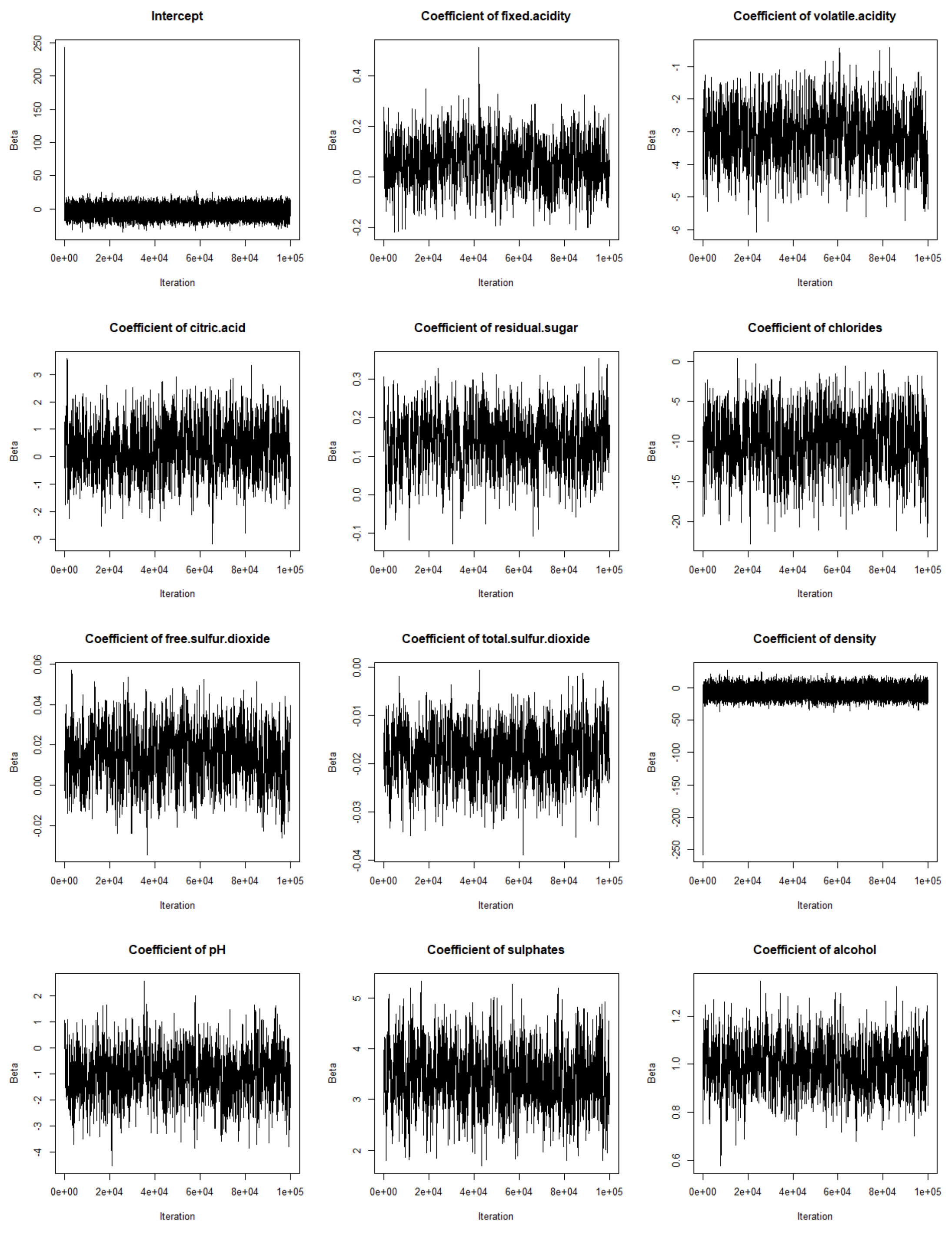
```
B_init2 = mleest
B_init3 = mleest + rnorm(length(mleest), 0, 0.5)
B_init4 = mleest + rnorm(length(mleest), 0, 1)
B_init5 = mleest + rnorm(length(mleest), 0, 1.5)

mhout2 = mhmcmc(y, X, B_init2, nsims, Sigma)
mhout3 = mhmcmc(y, X, B_init3, nsims, Sigma)
mhout4 = mhmcmc(y, X, B_init4, nsims, Sigma)
mhout5 = mhmcmc(y, X, B_init5, nsims, Sigma)

par(mfrow = c(4, 3))

for (j in 1:length(mleest)) {
  name = if (j==1) "Intercept" else paste("Coefficient of", colnames(wine)[j-1])
  plot(mhout2$beta_mat[, j], type = "l", col = "blue", main = name, ylim = range(
    mhout2$beta_mat[, j], mhout3$beta_mat[, j], mhout4$beta_mat[, j], mhout5$beta_mat
    [, j]), xlab = "Iteration", ylab = "Beta")
  lines(mhout3$beta_mat[, j], col = "red")
  lines(mhout4$beta_mat[, j], col = "green")
  lines(mhout5$beta_mat[, j], col = "purple")
}
```

6).

After tuning the the model, the acceptance rate is at 18.5%

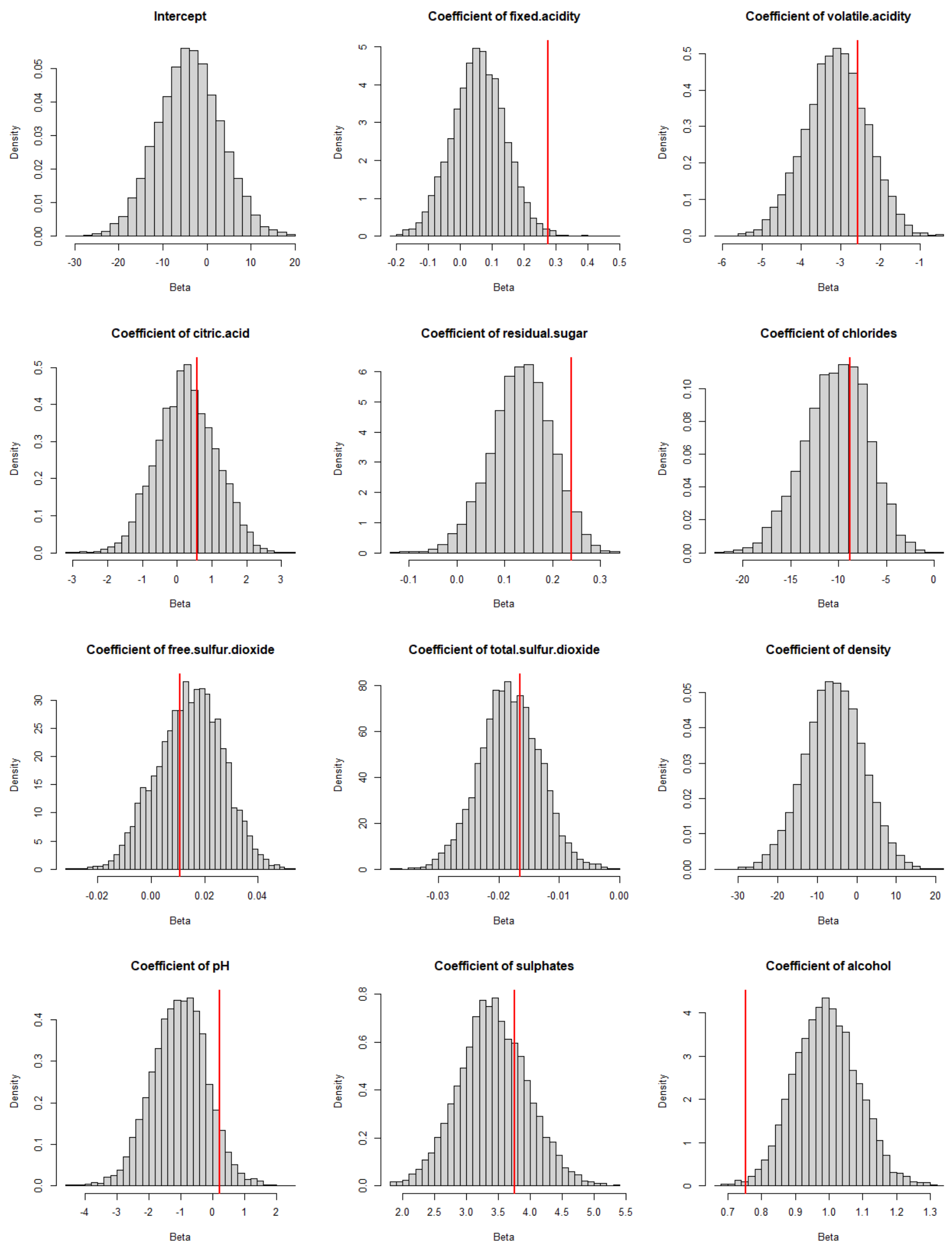
```
cov_mle = vcov(model)
Sigma = cov_mle * 0.1
mhout6 = mhmcmc(y, X, mleest, nsims, Sigma)

#calculate the acceptance rate
mh6acc = mhout6$acc / nsims
```

a).


Intercept
Coefficient of fixed.acidity
Coefficient of volatile.acidity
Coefficient of citric.acid
Coefficient of residual.sugar
Coefficient of chlorides
Coefficient of free.sulfur.dioxide
Coefficient of total.sulfur.dioxide
Coefficient of density
Coefficient of pH
Coefficient of sulphates
Coefficient of alcohol
Beta
Iteration
0e+00
2e+04
4e+04
6e+04
8e+04
1e+05

```r
par(mfrow=c(4,3))
for (j in 1:length(mleest)) {
  name = if (j==1) "Intercept" else paste("Coefficient of", colnames(wine)[j-1])
  plot(mhout6$beta_mat[, j], type = "l", main = name, xlab = "Iteration", ylab = "
    Beta")
}
```

b).


Intercept
Coefficient of fixed.acidity
Coefficient of volatile.acidity
Coefficient of citric.acid
Coefficient of residual.sugar
Coefficient of chlorides
Coefficient of free.sulfur.dioxide
Coefficient of total.sulfur.dioxide
Coefficient of density
Coefficient of pH
Coefficient of sulphates
Coefficient of alcohol
Density
Beta

```r
# First 10% as burn-in and keep every 10th to avoid autocorrection
burn_in = 0.1 * nsims
thin = 10

remaining = mhout6$beta_mat[(burn_in + 1):nsims, ]
thinned = remaining[seq(1, nrow(remaining), by = thin), ]
mleest[1]
par(mfrow = c(4, 3))
for (j in 1:ncol(thinned)) {
  name = if (j==1) "Intercept" else paste("Coefficient of", colnames(wine)[j-1])
  hist(thinned[, j], probability = TRUE, main = name, xlab = "Beta", breaks = 30)
  abline(v = mleest[j], col = "red", lwd = 2)
}
```

c).

From the graph, we can see the tuned graph shows clear sign of convergence, and the rate of acceptance is raised to around 18.5%. Also to solve the burn-in issue, we are discarding the first 10% of the sample. Also to avoid auto-correction, we employed **thinning** by collecting the every 10th sample. Hence this proposal is much better suited for this problem.