Portfolio 9 Solutions

Rachel Wood

2023-05-03

We load the data and scale it as in the last portfolio for ease of comparison:

```
library(mlbench)
data("PimaIndiansDiabetes")
head(PimaIndiansDiabetes)
```

```
##   pregnant glucose pressure triceps insulin mass pedigree age diabetes
## 1        6     148       72      35       0 33.6    0.627  50      pos
## 2        1      85       66      29       0 26.6    0.351  31      neg
## 3        8     183       64       0       0 23.3    0.672  32      pos
## 4        1      89       66      23      94 28.1    0.167  21      neg
## 5        0     137       40      35     168 43.1    2.288  33      pos
## 6        5     116       74       0       0 25.6    0.201  30      neg
```

```
data <- PimaIndiansDiabetes
y <- as.numeric((data$diabetes))-1
X <- scale(data[,1:8])
```

Choosing Proposal Distributions

For this we modify the joint proposal of the Metropolis Hastings algorithm:

$$Q(z, dz') = \mathcal{N}_{p+1}(z, c\mathbf{\Sigma}_n)$$

for tuning parameter $c > 0$ and

$$\mu_n = \underset{(\alpha,\beta)\in\mathbb{R}^p}{\arg\max} \log \pi(\alpha, \beta | y), \qquad \mathbf{\Sigma}_n = -(\mathbf{H}_n(\mu_q))^{-1}$$

In this, we instead want a proposal Q_i for each parameter for which we use the conditional distribution under the multivariate normal:

$$Q_i(z_t^{(i)} | z', dz') = \mathcal{N}_1\left(z_{t-1}^{(i)}, \tilde{\Sigma}_n^{(i,i)} - \tilde{\Sigma}_n^{(i)} \tilde{\Sigma}_n'^{-1} \left(\tilde{\Sigma}_n^{(i)}\right)^T\right)$$

where $z' = (z_t^{1:(i-1)}, z_{t-1}^{(i+1):p})$ and $\tilde{\Sigma}_n = c\Sigma_n$ We get μ_n and Σ_n as before:

```
fit <- glm(y ~ X, family = "binomial")
z_0 <- fit$coefficients
Sigma_n <- summary(fit)$cov.scaled
```

and write the function to obtain generate proposals:

```
proposal <- function(z, i, sigma){
  var <- sigma[i,i] - sigma[i,-i] %*% solve(sigma[-i,-i]) %*% sigma[-i,i]
  z_i <- rnorm(1, z[i], sqrt(var))
  return(z_i)
}
```

The Metropolis-within-Gibbs Algorithm

For this we generate proposals of each element of z and accept it with probability:

$$\alpha = \frac{\pi\left(z_t^{1:(i-1)}, \tilde{z}_t^i, z_{t-1}^{(i+1):p}|y\right) Q_i\left(z_{t-1}^{(i)}|\tilde{z}_t^{(i)}\right)}{\pi\left(z_t^{1:(i-1)}, z_{t-1}^i, z_{t-1}^{(i+1):p}|y\right) Q_i\left(\tilde{z}_t^{(i)}|z_{t-1}^{(i)}\right)} = \frac{\pi\left(z_t^{1:(i-1)}, \tilde{z}_t^i, z_{t-1}^{(i+1):p}|y\right)}{\pi\left(z_t^{1:(i-1)}, z_{t-1}^i, z_{t-1}^{(i+1):p}|y\right)}$$

since all Q_j are symmetric.

Thus we need a function to compute $\pi(z|y) = L_n(z)\pi(z) = L_n(z)$ where $\pi(z)$ is uniformly distributed on $\mathcal{Z}$:

```
library(mvtnorm)
post_likelihood <- function(par, X, y){

  alpha <- par[1]
  beta <- par[2:9]
  p <- 1 - 1 / (1 + exp(alpha + beta%*%t(X)))
  l_likelihood <- sum(dbinom(y, size=1, prob=p, log=TRUE))

  return(l_likelihood)
}
```

We now implement the MwG algorithm

```
logistic_MwG <- function(z_0, c, sigma_n, tmax, X, y){
  zs <- matrix(NA, nrow =tmax, ncol = length(z_0))
  z_current <- z_0
  z_current_ll <- post_likelihood(z_current, X,y)
  sigma_prime <- c * sigma_n
  accepted <- numeric(length(z_0))


  for (t in 1:tmax){
    for (i in 1:9){
      z_new <- z_current
      z_new[i] <- proposal(z_current, i, sigma_prime)
      z_new_ll <- post_likelihood(z_new, X, y)
      alpha <- exp(z_new_ll - z_current_ll)
      if (runif(1) < min(1, alpha)){
        z_current[i] <- z_new[i]
        z_current_ll <- z_new_ll
        accepted[i] <- accepted[i] +1
      }
      zs[t,] <- z_current
    }

  }
  return(list(zs = zs, acceptance_rate = accepted/tmax))
}
```

Convergence

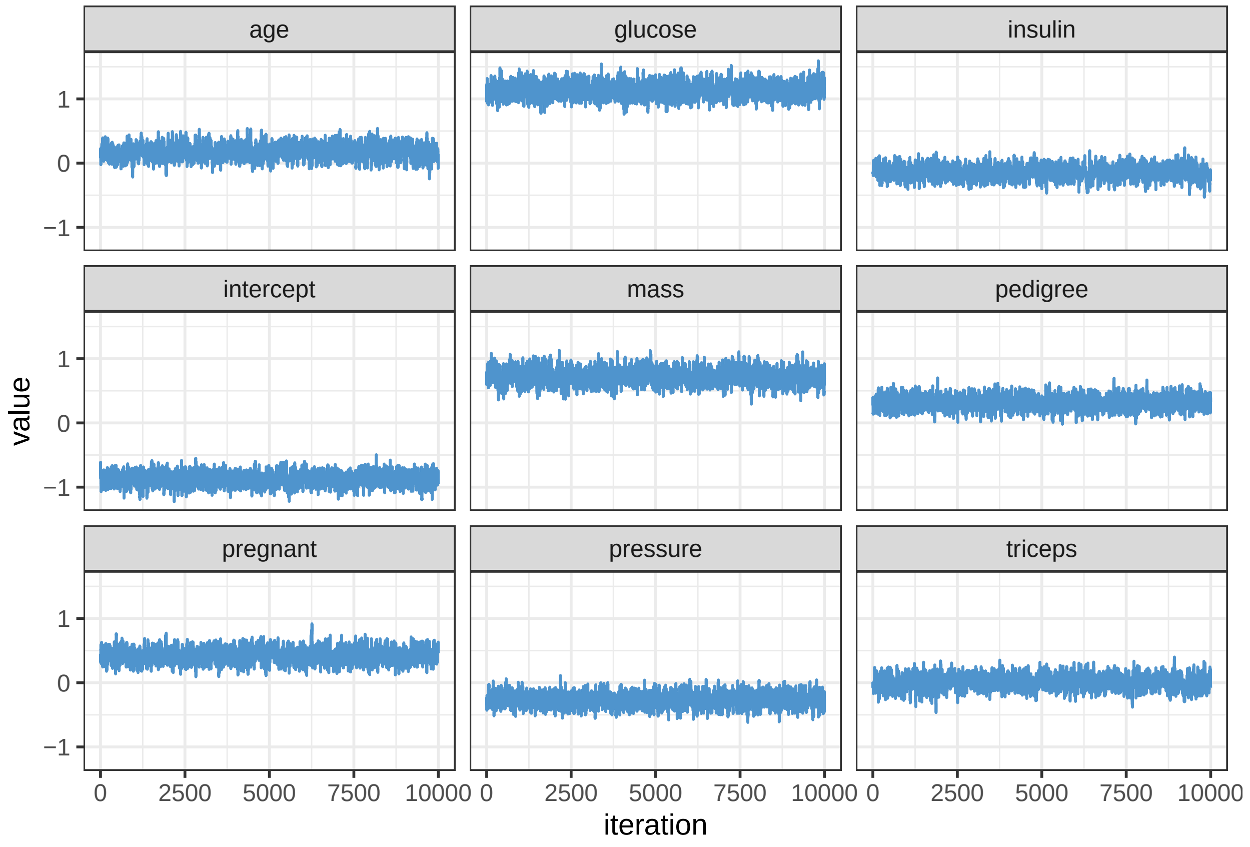
We now run the function on our data and obtain the initial acceptance rates for each variable:

```
mwg<- logistic_MwG(z_0, 1, Sigma_n, 10000, X, y)
mwg$acceptance_rate
```

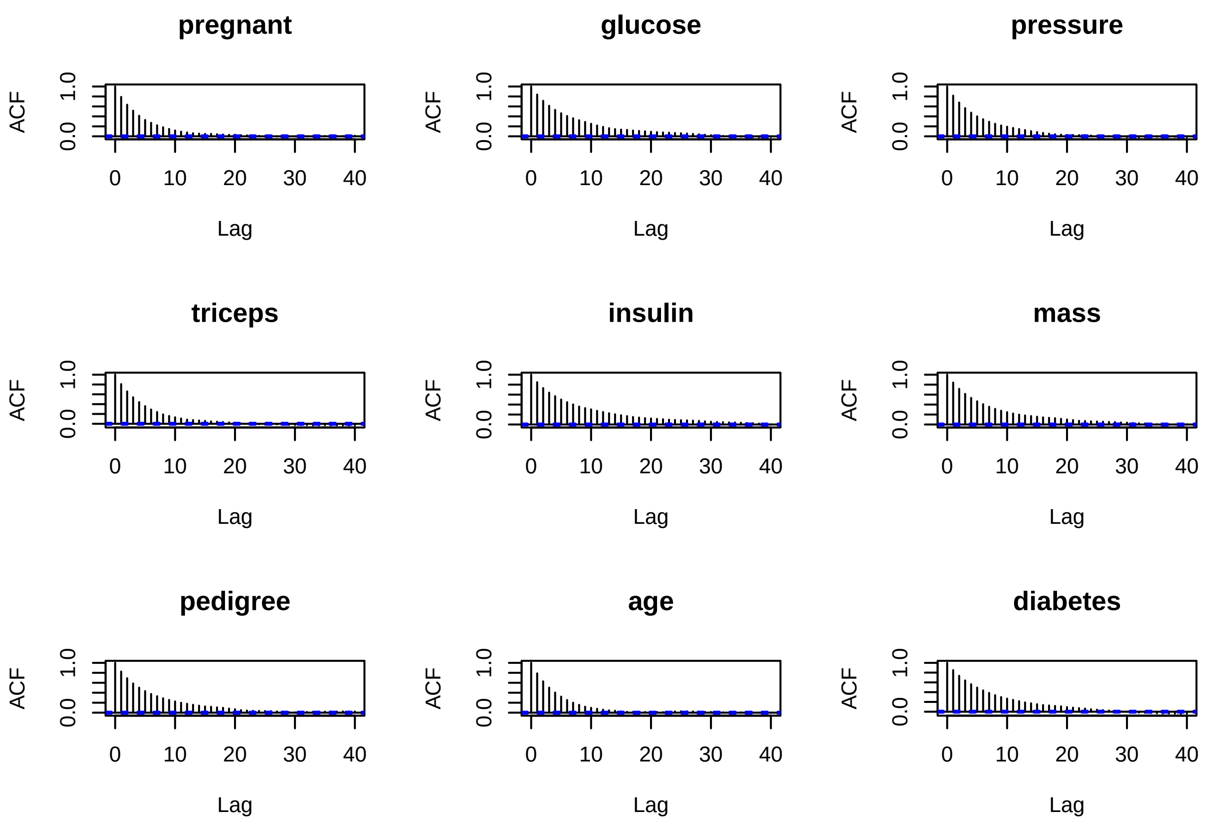
```
## [1] 0.7125 0.7061 0.7038 0.7100 0.7077 0.7072 0.7076 0.6979 0.7097
```

We then produce trace plots and autocorrelation plots:

```
library(dplyr)
library(tidyr)
library(ggplot2)

colnames(mwg$zs) <- c("intercept", colnames(data)[1:8])
zs_plot <- mwg$zs %>%
  as_tibble() %>%
  mutate(iteration = 1:10000) %>%
  pivot_longer(cols = 1:9)

ggplot(zs_plot, aes(iteration, value)) +
  geom_line(colour = "steelblue3") +
  facet_wrap(vars(name))
```



```r
par(mfrow = c(3,3))
for (i in 1:9){
  acf(mwg$zs[,i], main = colnames(data)[i])
}
```



The trace plots seem to show good mixing and the acf decreases quickly but the acceptance rate in very high.

Modifying the proposal distributions

We now try a larger c in the proposal distribution to lower the acceptance rate:

```
mwg <- logistic_MwG(z_0, 10, Sigma_n, 10000, X, y)
mwg$acceptance_rate
```

```
## [1] 0.3670 0.3585 0.3595 0.3593 0.3603 0.3522 0.3641 0.3689 0.3582
```

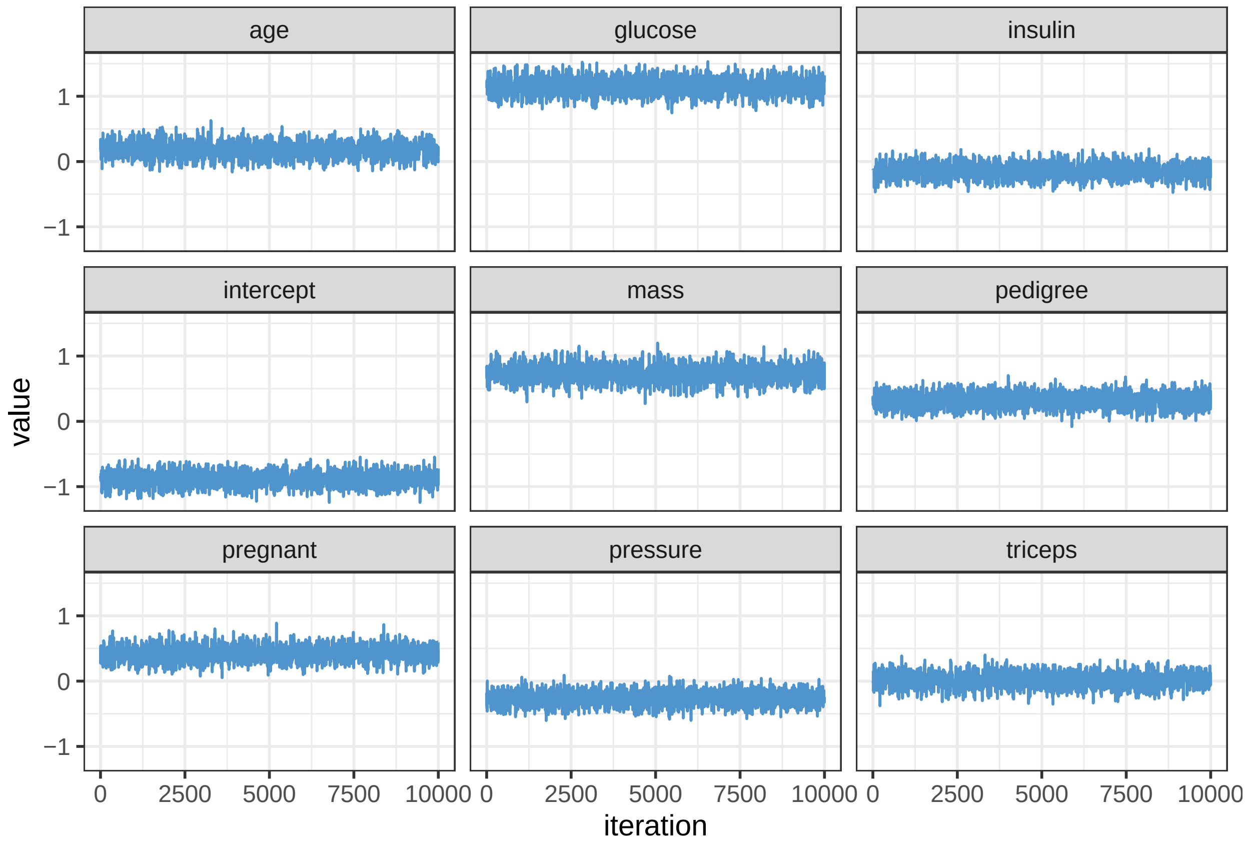
This is much better, but we are looking for an acceptance rate of approximately 0.234 so we increase c again to get:

```
mwg <- logistic_MwG(z_0, 20, Sigma_n, 10000, X, y)
mwg$acceptance_rate
```

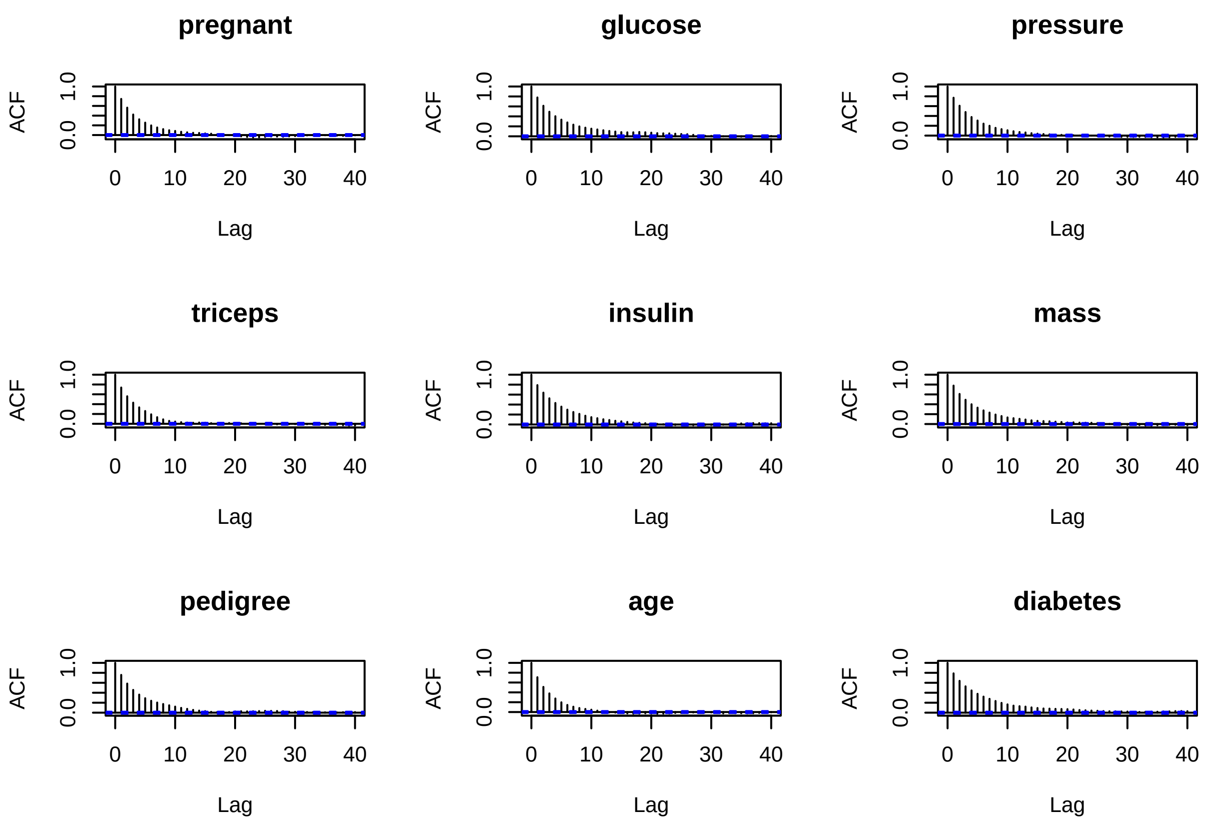
```
## [1] 0.2665 0.2789 0.2707 0.2704 0.2722 0.2725 0.2630 0.2744 0.2707
```

This is much closer to what we are looking for and we now check the same plots again:

```
colnames(mwg$zs) <- c("intercept", colnames(data)[1:8])
zs_plot <- mwg$zs %>%
  as_tibble() %>%
  mutate(iteration = 1:10000) %>%
  pivot_longer(cols = 1:9)

ggplot(zs_plot, aes(iteration, value)) +
  geom_line(colour = "steelblue3") +
  facet_wrap(vars(name))
```



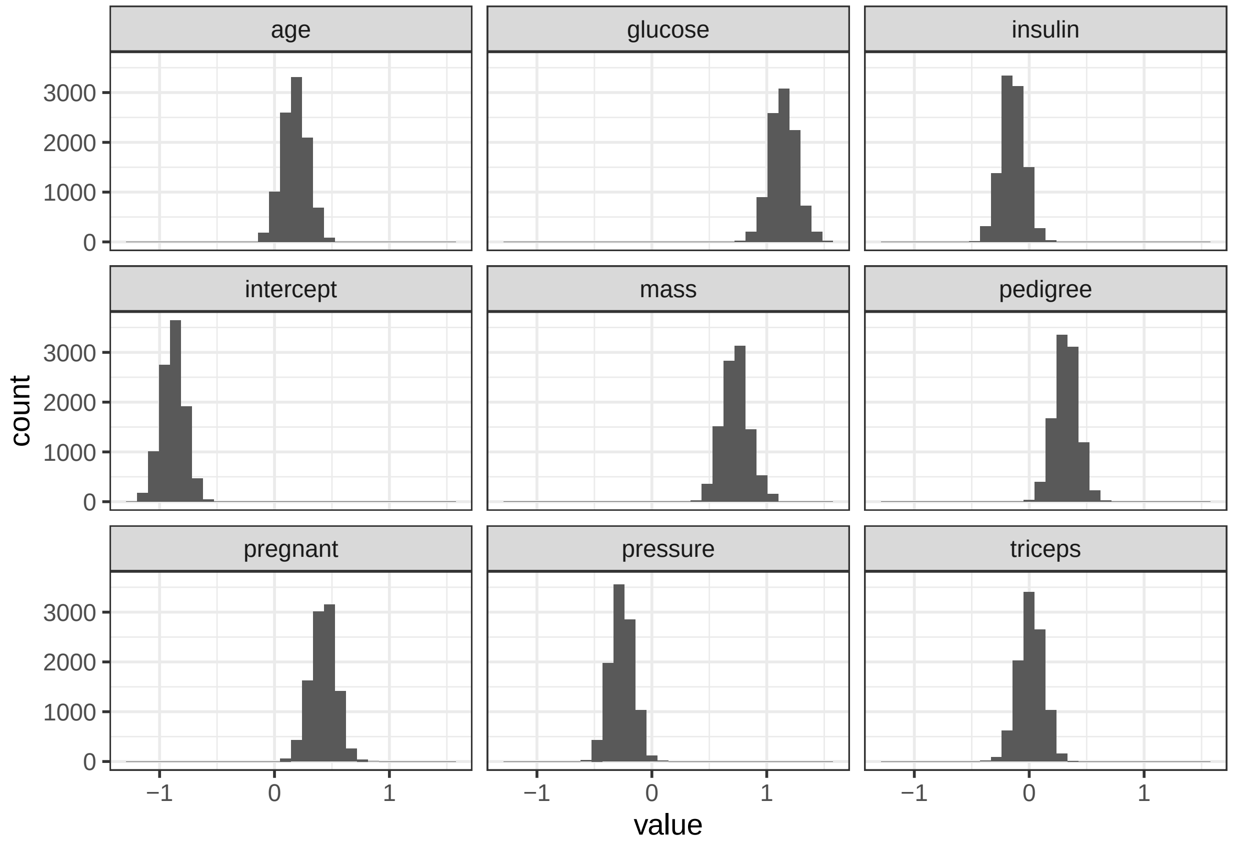
```
par(mfrow = c(3,3))
for (i in 1:9){
  acf(mwg$zs[,i], main = colnames(data)[i])
}
```



These all seem reasonable and so we move form # Marginal Posterior Distribution

```
ggplot(zs_plot, aes(value)) +
  geom_histogram() +
  facet_wrap(vars(name))
```

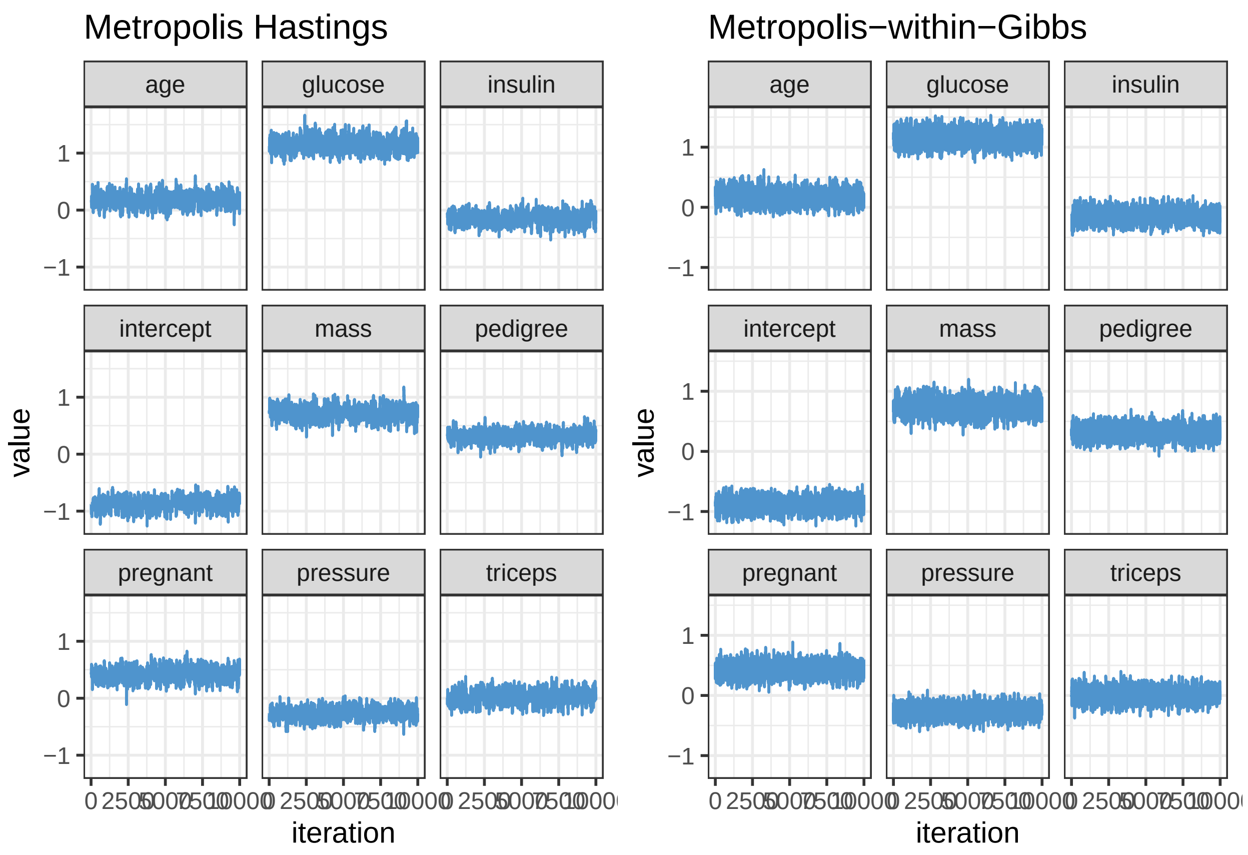
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



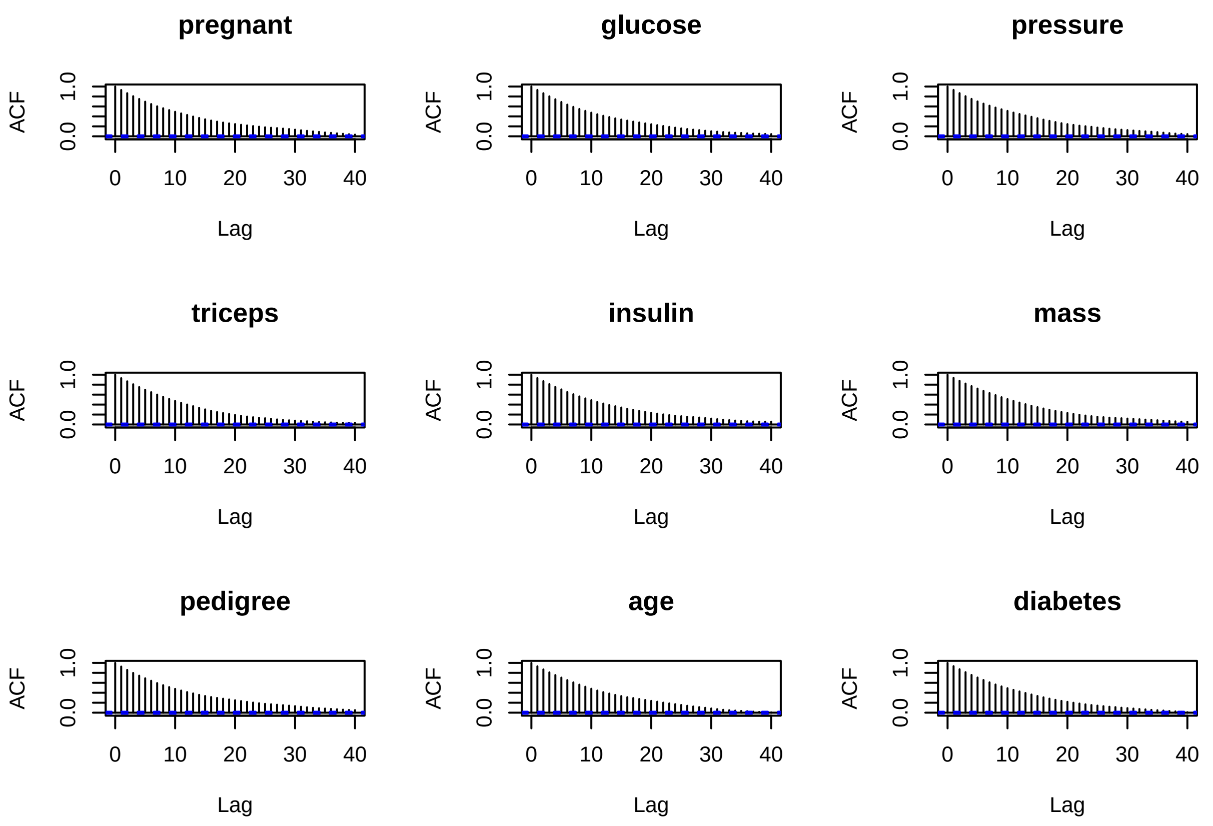
Comparing with the Metropolis Hastings Results

Comparing the acf plots, we can see the Metropolis within Gibbs has much lower correlation between samples and seems to have better mixing:

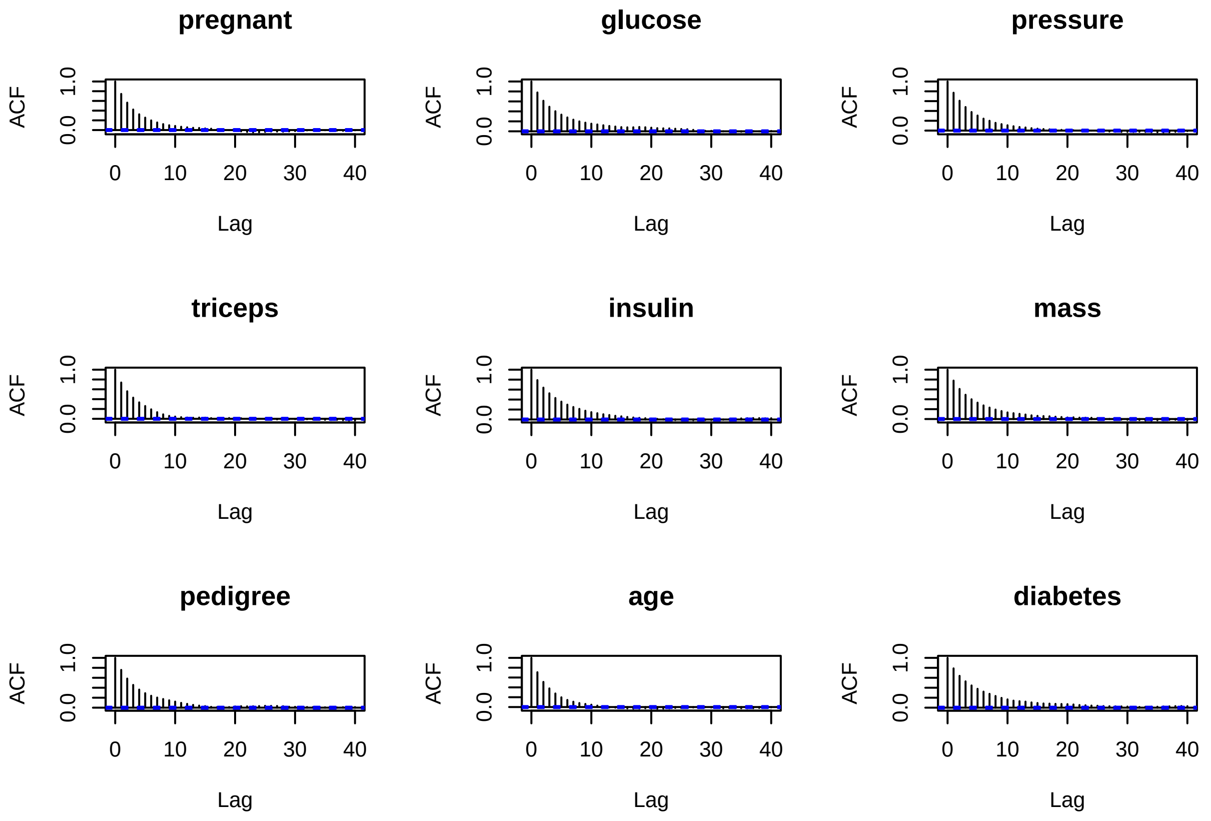
```
library(gridExtra)
mh<- logistic_MH(z_0, 0.75, Sigma_n, 10000, X, y)

colnames(mh$zs) <- c("intercept", colnames(data)[1:8])
mh_zs_plot <- mh$zs %>%
  as_tibble() %>%
  mutate(iteration = 1:10000) %>%
  pivot_longer(cols = 1:9)

mh_mix <- ggplot(mh_zs_plot, aes(iteration, value)) +
  geom_line(colour = "steelblue3") +
  facet_wrap(vars(name)) +
  labs(title = "Metropolis Hastings")

mwg_mix <- ggplot(zs_plot, aes(iteration, value)) +
  geom_line(colour = "steelblue3") +
  facet_wrap(vars(name)) +
  labs(title = "Metropolis-within-Gibbs")

grid.arrange(mh_mix, mwg_mix, ncol = 2)
```



```r
par(mfrow = c(3,3))
for (i in 1:9){
  acf(mh$zs[,i], main = colnames(data)[i])
}
```



```r
par(mfrow = c(3,3))
for (i in 1:9){
  acf(mwg$zs[,i], main = colnames(data)[i])
}
```



From these we conclude that the Metropolis-within-Gibbs sampler works better for the data.