BDA - Assignment 5

Contents

Exercise 1.

a)

Using the Gaussian prior from the assignment 4 in which

$$\mu = [0, 10]$$

and

$$\Sigma = \begin{bmatrix} 4 & 10 \\ 10 & 100 \end{bmatrix}$$

.

```
data("bioassay")
density_ratio<-function(alpha_propose, alpha_previous,
beta_propose, beta_previous,x, y, n)
{
  proposed_posterior<-bioassaylp(alpha_propose,beta_propose,x,y,n)+
    log(dmvnorm(c(alpha_propose,beta_propose),c(0,10),matrix(c(4,10,10,100),ncol=2,nrow=2)))
  previous_posterior<-bioassaylp(alpha_previous,beta_previous,x,y,n)+
    log(dmvnorm(c(alpha_previous,beta_previous),c(0,10),matrix(c(4,10,10,100),ncol=2,nrow=2)))
  ratio<-exp(proposed_posterior-previous_posterior)
  return(ratio)
}
#have a test
cat("theoretical value is 1.187524 \n")
```

```
## theoretical value is 1.187524
```

```
cat("calculated value is: ",density_ratio(alpha_propose = 1.89, alpha_previous = 0.374,
beta_propose = 24.76, beta_previous = 20.04,
x = bioassay$x, y = bioassay$y, n = bioassay$n),"\n")
```

```
## calculated value is:  1.187524
```

```
cat("theoretical value is 0.8420882 \n")
```

```
## theoretical value is 0.8420882
```

```
cat("calculated value is: ",density_ratio(alpha_propose = 0.374, alpha_previous = 1.89,
beta_propose = 20.04, beta_previous = 24.76,
x = bioassay$x, y = bioassay$y, n = bioassay$n),"\n")
```

```
## calculated value is:  0.8420882
```

b)

The code for the function that implements metropolis algorithm is shown below. The starting points is generated randomely from the Gaussian prior. And the proposal/jumping distribution are $\alpha_* \sim N(\alpha_{t-1}; \sigma = 1)$ and $\beta_* \sim N(\beta_{t-1}; \sigma = 5)$.

```
metropolis_bioassay<-function(N){
  previous_proposal<-rmvnorm(1,c(0,10),matrix(c(4,10,10,100),ncol=2,nrow=2))
  trial_alpha<-c(previous_proposal[1])
  trial_beta<-c(previous_proposal[2])
  proposals<-c()
  for (i in seq(1,N,length=N)){
    new_proposal_alpha<-rnorm(1,mean=previous_proposal[1],sd=1)
    new_proposal_beta<-rnorm(1,mean=previous_proposal[2],sd=5)
    ratio<-density_ratio(new_proposal_alpha,previous_proposal[1],
                         new_proposal_beta,previous_proposal[2],
                         x = bioassay$x, y= bioassay$y, n = bioassay$n)
    p <- runif(1, 0, 1)
    if (p<ratio){
      trial_alpha<-c(trial_alpha,new_proposal_alpha)
      trial_beta<-c(trial_beta,new_proposal_beta)
      previous_proposal<-c(new_proposal_alpha,new_proposal_beta)
      }
    else{
      trial_alpha<-c(trial_alpha,new_proposal_alpha)
      trial_beta<-c(trial_beta,new_proposal_beta)
      }}
  return (list(alphas=trial_alpha,betas=trial_beta))
}
```

I run the metropolis algorithm for 10 times to generate 10 chains. Each chain consists of 1 starting point and 3000 following draws. The warm-up (burn-in) length is set to 600.

```
# generate 10 chains
n=3000
trial_alphas<-matrix(0, nrow = 10, ncol = n-600)
trial_betas<-matrix(0, nrow = 10, ncol = n-600)
for (i in seq(1,10,length=10)){
  trial_list<-metropolis_bioassay(n)
  trial_alpha<-unlist(trial_list["alphas"],use.names=FALSE)
  trial_beta<-unlist(trial_list["betas"],use.names=FALSE)
  trial_alphas[i,]<-trial_alpha[601:n+1]
  trial_betas[i,]<-trial_beta[601:n+1]
}
```

We can use the value of $\hat{R}$ to check if we gain convergence in these chains.

```
cat("the r hat value for alpha is: ",Rhat(t(trial_alphas)),'\n')
```

```
## the r hat value for alpha is:  1.002441
```

```
cat("the r hat value for beta is: ",Rhat(t(trial_betas)))
```

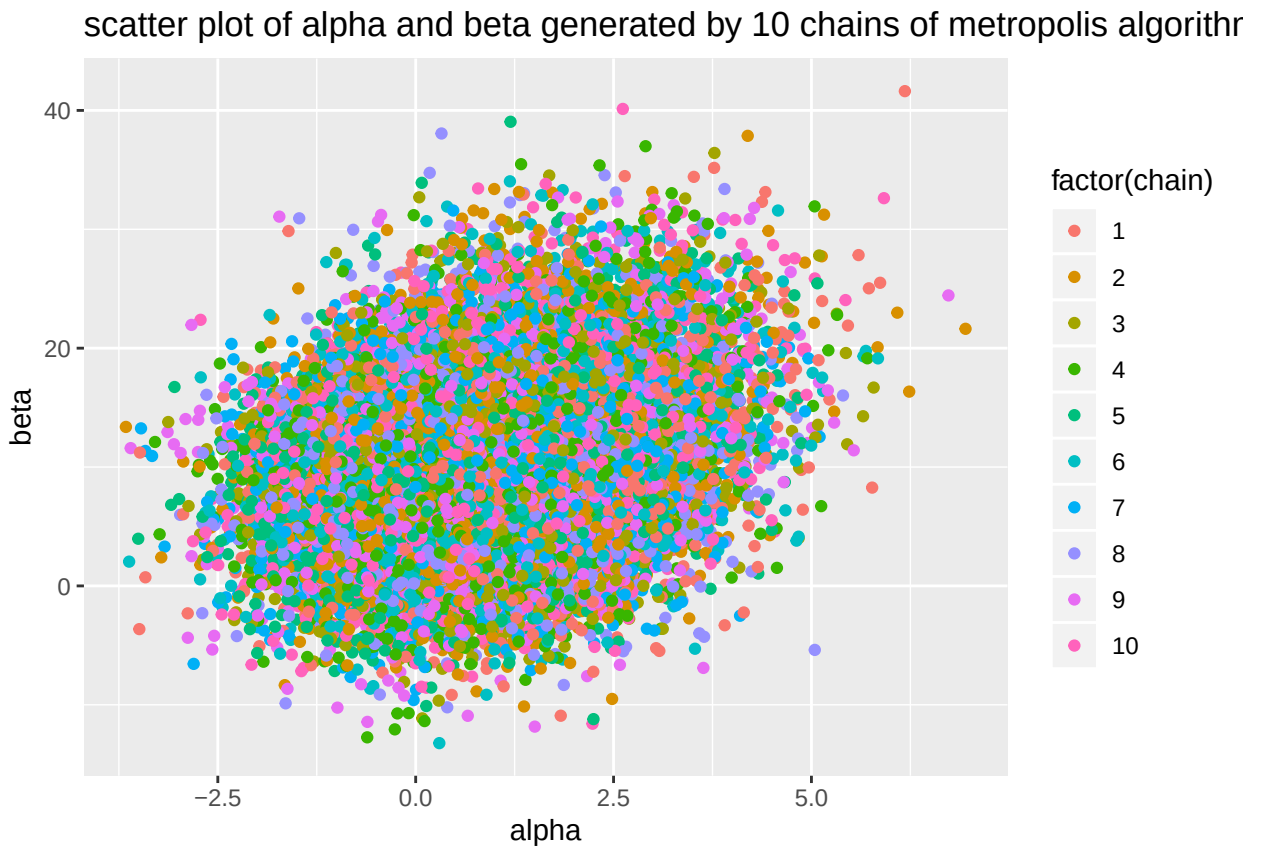
```
## the r hat value for beta is:  1.002702
```

According to the documentation of the function *Rhat*, "the Rhat function produces R-hat convergence diagnostic, which compares the between- and within-chain estimates for model parameters and other univariate quantities of interest. If chains have not mixed well (ie, the between- and within-chain estimates

don't agree), R-hat is larger than 1. We recommend running at least four chains by default and only using the sample if R-hat is less than 1.05."
As the ouput of the Rhat function shows, R-hat is far less than 1.05 and close to 1 so these generated samples are really fine to use.

Also scatter plots are generated to see if the convergence to the posterior is achieved.

```
sim <- data.frame("alphas" = as.vector(trial_alphas),
                  "betas" = as.vector(trial_betas),"chain"=rep(1:10, 2400))
ggplot(sim,aes(x=alphas,y=betas)) +
  geom_point(aes(color = factor(chain)))+
  labs(title="scatter plot of alpha and beta generated by 10 chains of metropolis algorithm",
       x="alpha", y = "beta")
```



We can see that both the shape, the variance and the location of the generated samples of all 10 chains are very close to the posterior distribution, which means that we've gained the convergence to the posterior.