BDA - Assignment 5

Anonymous

```
library(mvtnorm)
library(ggplot2)
theme_set(theme_minimal())
library(aaltobda)
```

```
##
## Attaching package: 'aaltobda'
```

```
## The following objects are masked from 'package:mvtnorm':
##
##     dmvnorm, rmvnorm
```

```
library(gridExtra)
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------
```

```
## v tibble  2.1.3     v purrr   0.3.2
## v tidyr   0.8.3     v dplyr   0.8.3
## v readr   1.3.1     v stringr 1.4.0
## v tibble  2.1.3     v forcats 0.4.0
```

```
## -- Conflicts ------------------------------------------------------------------------------------
## x dplyr::combine() masks gridExtra::combine()
## x dplyr::filter()  masks stats::filter()
## x dplyr::lag()     masks stats::lag()
```

```
data("bioassay")
data <- bioassay
```

Problem 1: Bioassay model and importance sampling

The following gaussian prior is used :

$$p(\alpha, \beta) \sim \mathcal{N}\{\mu, \mathbf{\Sigma}\} = (2\pi)^{\frac{-k}{2}} det(\Sigma)^{\frac{-1}{2}} exp\Big(\frac{-1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\Big)$$

Where

$$k = 2$$

and

$$\mu_\alpha = 0$$

,

$$\sigma_\alpha = 2$$

,

$$\mu_\beta = 10$$

,

$$\sigma_\beta = 10$$

and

$$\rho = corr(\alpha, \beta) = 0.5$$

. Therefore,

$$\mu = \begin{pmatrix} \mu_\alpha \\ \mu_\beta \end{pmatrix} = \begin{pmatrix} 0 \\ 10 \end{pmatrix}$$

and

$$\mathbf{\Sigma} = \begin{pmatrix} \sigma_\alpha^2 & \rho\sigma_\alpha\sigma_\beta \\ \rho\sigma_\alpha\sigma_\beta & \sigma_\beta^2 \end{pmatrix} = \begin{pmatrix} 4 & 10 \\ 10 & 100 \end{pmatrix}$$

.

```
mu_alpha <- 0
s_alpha <- 2
mu_beta <- 10
s_beta <- 10
rho <- 0.5
s <-  matrix(c(s_alpha^2, rho*s_alpha*s_beta, rho*s_alpha*s_beta, s_beta^2 ), ncol=2)
mu = c(mu_alpha, mu_beta)

p_log_prior <- function(alpha, beta){
  x = cbind(alpha, beta)
  d <- dmvnorm(x, mean = mu, sigma = s)
  return(log(d))
}

p_log_posterior <- function(alpha, beta, x=bioassay$x, y=bioassay$y, n=bioassay$n){
  p_log_likelihood <- bioassaylp(alpha, beta, x, y, n)
  post <- p_log_prior(alpha, beta) + p_log_likelihood
  return(post)
}
```

a) The implemetation of posterior density ratio function is shown in the following lines of code:

```
density_ratio <- function(alpha_propose, alpha_previous,
                          beta_propose, beta_previous,
                          x = bioassay$x, y = bioassay$y, n = bioassay$n){
  p1 <- p_log_posterior(alpha_propose, beta_propose, x=bioassay$x, y=bioassay$y, n=bioassay$n)
  p0 <- p_log_posterior(alpha_previous, beta_previous, x=bioassay$x, y=bioassay$y, n=bioassay$n)
  ratio <- exp(p1 - p0)
  return(ratio)
}
```

b) The implementation of metropolis algorithm using denstity ratio is as follows:

```
proposal_distribution <- function(param){
  sigma = matrix(c(1, 2, 2, 5), ncol=2)
  return(rnorm(2, mean = param, sd = sigma))
}


metropolis_bioassay <- function(startvalue, iterations){
  chain = array(dim = c(iterations+1,2))
  chain[1,] = startvalue
  for (i in 1:iterations){
    proposal <- proposal_distribution(chain[i,])
    r <- density_ratio(alpha_propose = proposal[1], alpha_previous = chain[i,1],
                       beta_propose = proposal[2], beta_previous = chain[i,2],
                       x = bioassay$x, y = bioassay$y, n = bioassay$n)
    if (runif(1) < r){
      chain[i+1,] = proposal
    }else{
      chain[i+1,] = chain[i,]
    }
  }
  return(chain)
}
```

Problem 2: Metropolis algorithm The proposal distribution that I applied is the same as proposed distributions in the exercise:

$$\alpha^* \sim \mathcal{N}\{\alpha_{t-1}, \sigma = 1\}$$

and

$$\beta^* \sim \mathcal{N}\{\beta_{t-1}, \sigma = 5\}$$

. The number of chanis, the number of draws per chain, the start values and the length of warmup (half of the iterations) are shown as below:

```
n_chains <- 5
iterations <- 100000
startvalues <- matrix(c(-1,1,-5,5,-2,2,4,-5,10,-1),
                      byrow = T, ncol=2)
colnames = c("alpha0", "beta0")
warmUp <- iterations/2

print(paste0('The number of chain is ', n_chains))
```

```
## [1] "The number of chain is 5"
```

```
print(paste0('The number of draws per chain is ', iterations))
```

```
## [1] "The number of draws per chain is 1e+05"
```

```
print('The start values are')
```

```
## [1] "The start values are"
```

```
write.table(startvalues, row.names=F, col.names=T)
```

```
## "V1" "V2"
## -1 1
## -5 5
## -2 2
## 4 -5
## 10 -1
```

```
print(paste0('The warm-up length is ', warmUp))
```

```
## [1] "The warm-up length is 50000"
```

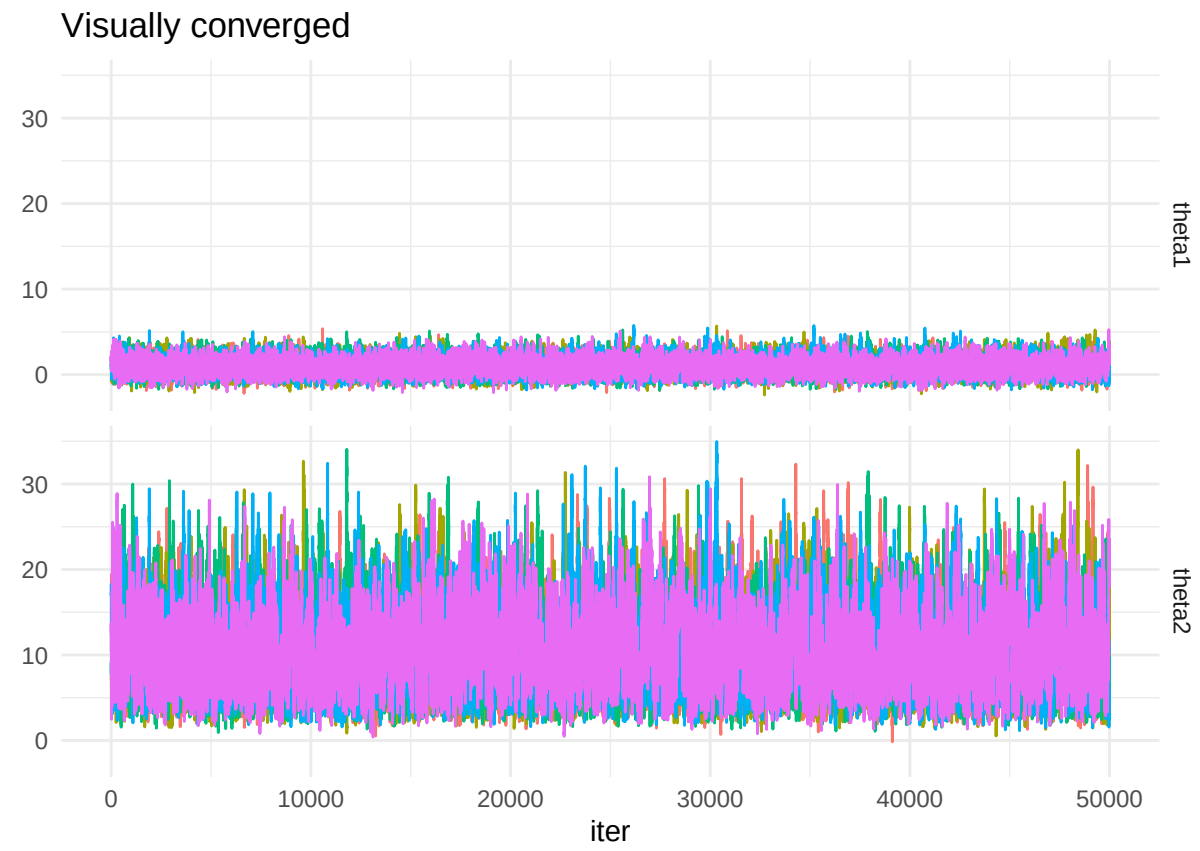
```
run_in_chains <- function(n_chains, iterations, startvalues){
  m <- n_chains*2
  n <- round(iterations/2/2)
  chains <- array(dim = c(2*n,2, n_chains))
  for(j in 1:n_chains){
    chain <- metropolis_bioassay(startvalues[j,], iterations)
    warmUp <- iterations/2
    new_chain <- chain[-(1:warmUp),]
    new_chain <- new_chain[1:warmUp,]
    chains[, ,j] <- new_chain
  }
  return (chains)
}


chains <- run_in_chains(n_chains, iterations, startvalues)
```

Overlapping chains is shown separately for α and β. It is difficult to conclude the convergence of the parameters only by visualization.

```
inds <- 1:warmUp
dfs <- data.frame(iter = inds, chains[inds, 1, ]) %>%
  gather(chain, theta1, -iter) %>%
  within(theta2 <- c(chains[inds, 2, ])) %>%
  gather(var, val, -iter, -chain)


#' Plot trends of the all draws
ggplot(data = dfs) +
  geom_line(aes(iter, val, color = chain)) +
  facet_grid(var~.) +
  labs(title = 'Visually converged', y = '') +
  scale_color_discrete(guide = FALSE)
```



Problem 3: Estimarion of $\hat{R}$ In order to compute $\hat{R}$ I applied the equation 11.4 in BDA3. The

```
R_hat <- function(chains){
  m <- 2*dim(chains)[3]
  n <- round(dim(chains)[1]/2)
  psi_ij <- array(dim = c(n, 2, m))
  for(j in 1:n_chains){
    psi_ij[, , 2*j-1] <- chains[(1:n), , j]
    psi_ij[, , 2*j] <- chains[-(1:n), , j]
  }

  psi_j <- array(dim = c(m, 2))
  s_j_2 <- array(dim = c(m, 2))
  for(j in 1:m){
    psi_j[j,] <- colMeans(psi_ij[, ,j])
    s_j_2[j,] <- rowSums(apply(psi_ij[, ,j], 1, function(x) x-psi_j[j,])^2) / (n-1)
  }
  psi <- colMeans(psi_j)

  B <- ( n/(m-1) ) * sum(apply(psi_j, 1, function(x) x-psi)^2)
  W <- colMeans(s_j_2)
  var_hat <- ((n-1)/n) * W  + (1/n) * B
  R_hat <- sqrt(var_hat/W)
  return(R_hat)
}
```

$\hat{R}$ compares the whithin and between variances of teh chains. The within variance tells hw much each chain has explored and on the other hand between variance is the total variance when we combine all chains. If we run long enough, the average of within variances and total variance will be close together and the $\hat{R}$ would be close to 1.So we will have more and more similar estimate for the mean and variances of the samples.

As it can be seen in the following the $\hat{R}$ in my solution for α and β is very close to 1. Since the starting points for each chain are overdispersed, it seems that chains have converged.

```
R_hat(chains)
```

```
## [1] 1.042262 1.001642
```

Problem 4: Scatter plot of α and β

In the following figure you can see the scatter plot of draws of α and β for one of the chains implemented above.

```
samp_A <- chains[, 1, 1]
samp_B <- chains[, 2, 1]

xl <- c(-5, 10)
yl <- c(-10, 40)

ggplot(data = data.frame(samp_A, samp_B)) +
  geom_point(aes(samp_A, samp_B), color = 'blue', size = 0.3) +
  coord_cartesian(xlim = xl, ylim = yl) +
  labs(x = 'alpha', y = 'beta')
```