(c)

```
y <- c(5,1,5,14,3,19,1,1,4,22)
n <- c(94,16,63,126,5,31,1,1,2,10)

n.iter <- 5000  #the number of itertaions
m <- 4          #the number of chains
```

1. Declare the objects to save samples.

```
theta <- array(NA, dim=c(10, n.iter, m))
al <- array(NA, dim=c(n.iter, m))
be <- array(NA, dim=c(n.iter, m))
a <- array(0, dim=c(1,m))   # for the number of acceptance.
```

2. Declare the function for the target distribution, p(x).

```
Px <- function(x){
  (prod(theta[,i,h]))^(al[i-1,h]-1) * (be[i-1,h])^(10*al[i-1,h]) / (factorial(al[i-1,h]-1))^10
}
```

3. Set the starting values.

```
for(h in 1:m){
  theta[,1,h] <- sample(y, 10, replace=T)
  al[1,h] <- 1
  be[1,h] <- runif(1,0,15)
}
```

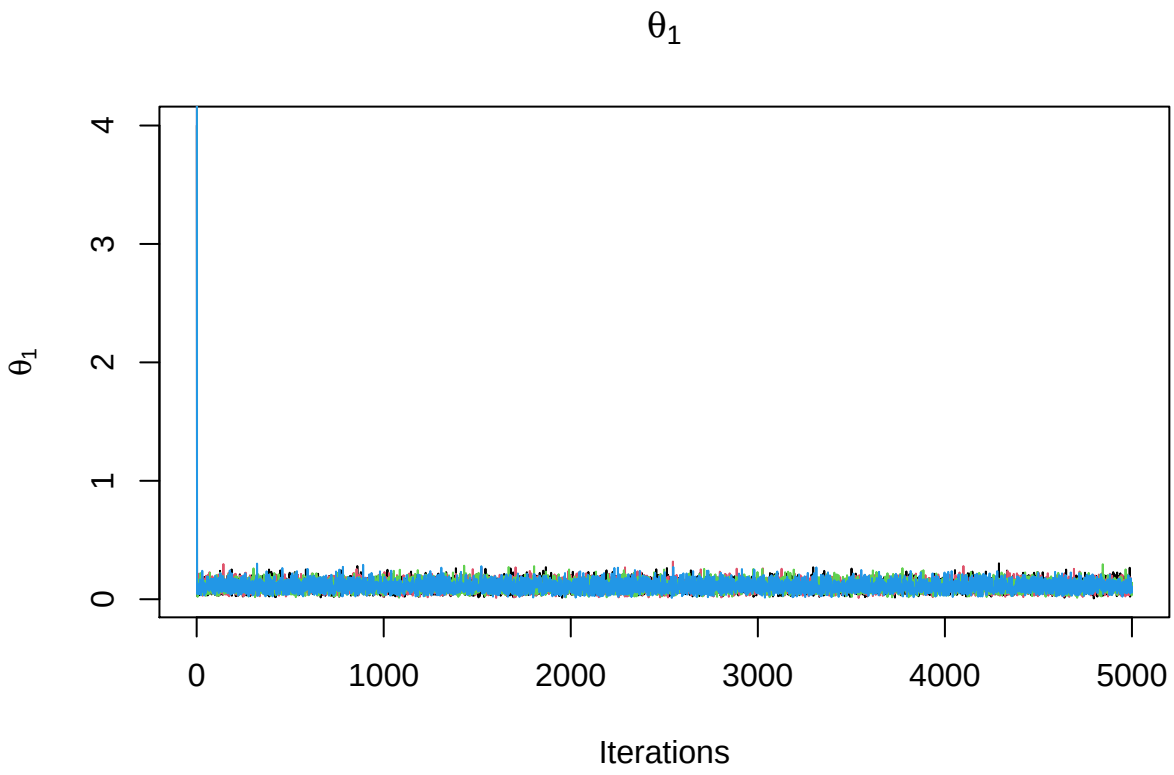
4. Run 4 MCMC chains (Metropolis-within-gibbs).

```
for(h in 1:m){
  for(i in 2:n.iter){
    for(j in 1:10){
      theta[j,i,h] <- rgamma(1, y[j]+al[i-1,h], rate=n[j]+be[i-1,h])
    }

    al_cur <- al[i-1,h]
    al_prop <- runif(1,0,15)

    num <- Px(al_prop) / dunif(al_prop, 0, 15)
    dem <- Px(al_cur) / dunif(al_cur, 0, 15)

    r <- num / dem

    if(runif(1) > min(r,1)){
      al[i,h] <- al_cur
    }else{
      al[i,h] <- al_prop; a[,h] <- a[,h] + 1
    }
```

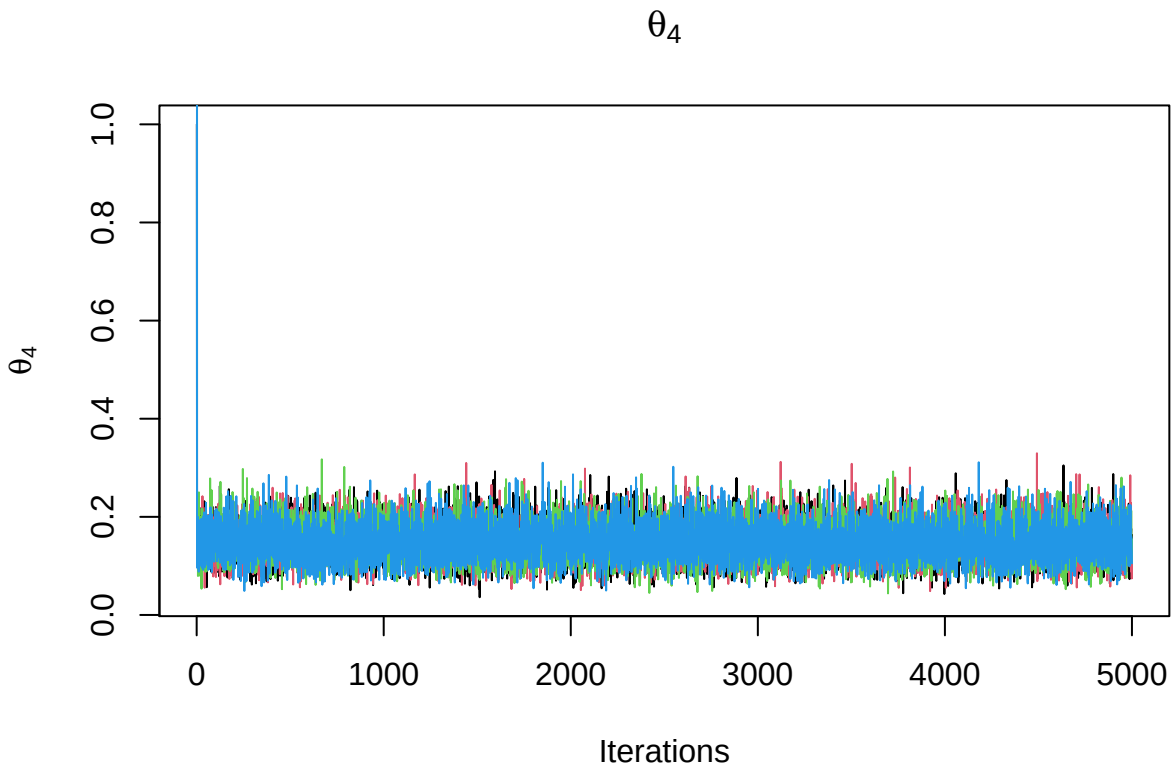
```
    be[i,h] <- rgamma(1, 10*al[i,h]+1, rate=sum(theta[,i,h]))
  }
}
```

5. Report trace plots.

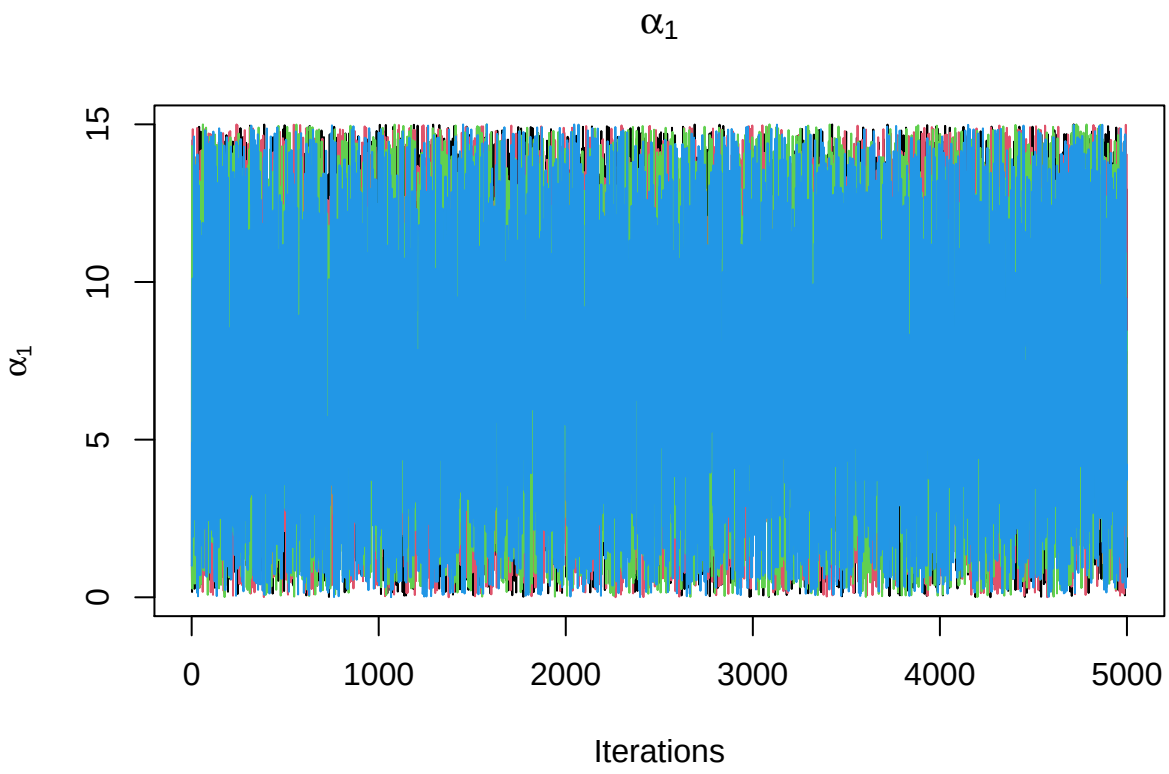
```
plot(theta[1, ,1], type="l", main=expression(theta[1]), ylab=expression(theta[1]), xlab="Iterations")
for(t in 2:m){
  points(theta[1,,t], type="l", col=t)
  Sys.sleep(1)
}
```



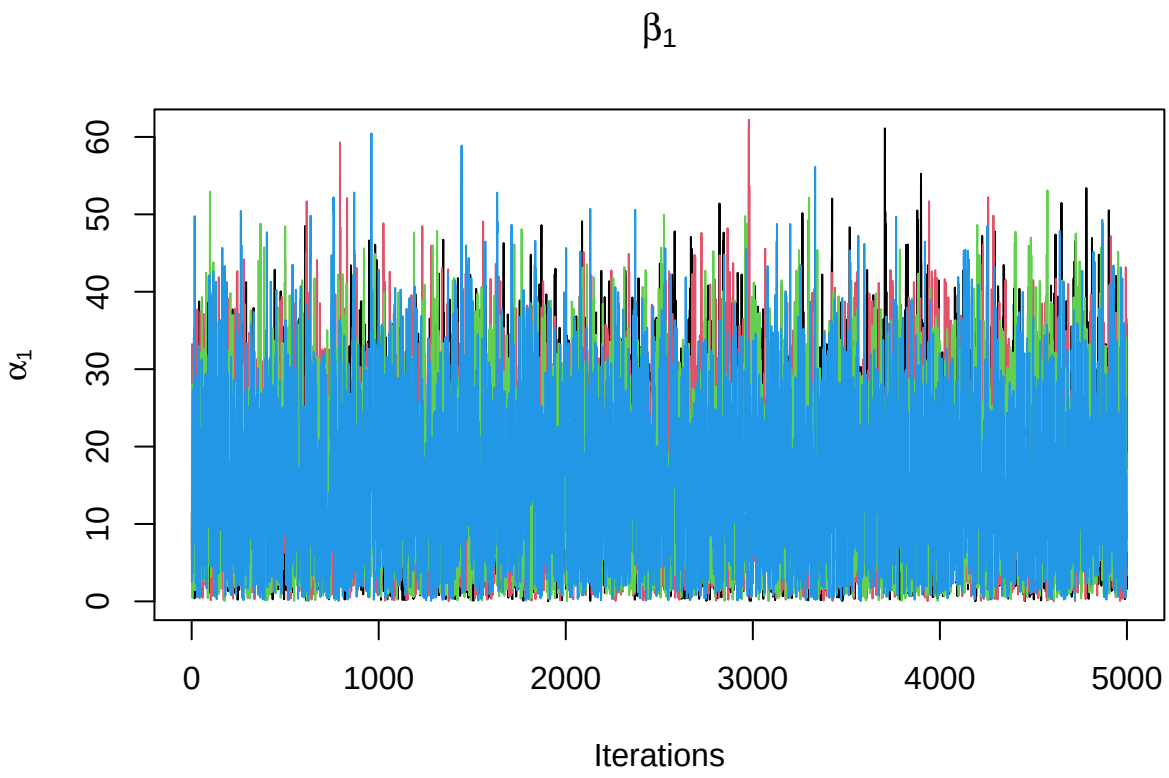

```
plot(theta[4, ,1], type="l", main=expression(theta[4]), ylab=expression(theta[4]), xlab="Iterations")
for(t in 2:m){
  points(theta[4,,t], type="l", col=t)
  Sys.sleep(1)
}
```




```
plot(al[,1], type="l", main=expression(alpha[1]), ylab=expression(alpha[1]), xlab="Iterations")
for(t in 2:m){
  points(al[,t], type="l", col=t)
  Sys.sleep(1)
}
```




```
plot(be[,1], type="l", main=expression(beta[1]), ylab=expression(alpha[1]), xlab="Iterations")
for(t in 2:m){
  points(be[,t], type="l", col=t)
  Sys.sleep(1)
}
```



Commentary:

θ's converge to values close to zero.

α and β are mixed well going up and down.

(d)

1. Gelman-Rubin diagnostics.

```
theta_new <- theta[ ,(n.iter/2+1):n.iter, ] # burn-in; discard the first half
theta_new1 <- theta_new[,1:(n.iter/10),]
theta_new2 <- theta_new[,(n.iter/10+1):(n.iter/10),]

THETA <- array(NA, dim=c(10, n.iter/10, m*2))
THETA[,,1:m] <- theta_new1
THETA[,,(m+1):(2*m)] <- theta_new2

M <- 2*m
N <- n.iter/10

mean_dot <- apply(THETA[1,,], 2, mean)
mean_dot_dot <- mean(mean_dot)

B <- N/(M-1)*sum((mean_dot-mean_dot_dot)^2)
W <- 1/((N-1)*M)*sum((THETA[1,,]-matrix(mean_dot, nrow=N, ncol=M, byrow=TRUE))^2)
```

```
R <- sqrt(((N-1)/N*W+1/N*B)/W)
R
```

```
## [1] 1.012874
```

Comentary: Gelman/Rubin shrink factor is bigger than 1, so the parameter θ didn't converge well.

2. Posterior mean and 95% credible intervals.

```
post_mean <- NULL
post_ci <- array(NA, dim=c(10,1))

for(j in 1:10){
  post_mean[j] <- mean(THETA[j,,])
  post_ci[j,1] <- paste(qgamma(0.025,y[j]+mean(al),rate=(n[j]+mean(be))),',',qgamma(0.975,y[j]+mean(al)
}

post_mean
```

```
##  [1] 0.1170091 0.2177774 0.1455836 0.1606214 0.5098156 0.5465985 0.4922597
##  [8] 0.5322564 0.6851901 1.2327533
```

```
post_ci
```

```
##       [,1]
##  [1,] "0.0590261718866869 , 0.183183748555652"
##  [2,] "0.114958295206296 , 0.460278519039316"
##  [3,] "0.0819876282053704 , 0.254443081599101"
##  [4,] "0.0936575212652005 , 0.220414723586638"
##  [5,] "0.235870386502203 , 0.815652896948592"
##  [6,] "0.36398715892648 , 0.785430658223271"
##  [7,] "0.212432561708893 , 0.850553191691226"
##  [8,] "0.212432561708893 , 0.850553191691226"
##  [9,] "0.311306413956929 , 1.01606631756499"
## [10,] "0.741878487441021 , 1.53671425016748"
```