STATS 551 - HW3

Zhen Qin

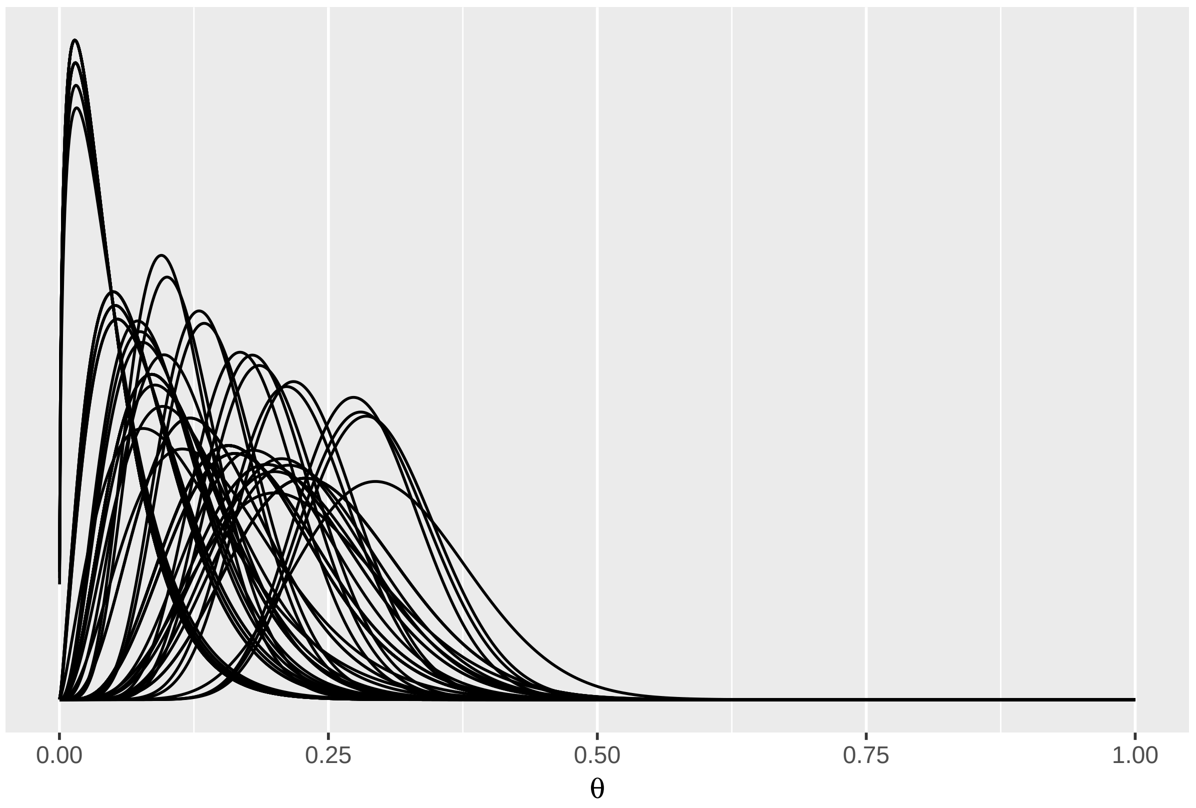
1.

The model is $y_j|\theta_j \sim binomial(n_j, \theta_j)$. n_j is the total number of trials. $\theta_j \sim beta(\alpha, \beta), j = 1, ..., 68$. Suppose that (α, β) is given. The joint posterior distribution is $p(\theta_1, ..., \theta_n|\alpha, \beta, y_1, ..., y_n) \propto \prod_j \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta_j^{\alpha-1}(1-\theta_j)^{\beta-1}\theta_j^{y_j}(1-\theta_j)^{n_j-y_j} = \prod_j \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta_j^{\alpha+y_j-1}(1-\theta_j)^{\beta+n_j-y_j-1}$.

Given $(\alpha, \beta) = (1.4, 8.6)$, θ_j are all independent, so visulization of one parameter is enough. In fact, after integration we know $\theta_i|\alpha, \beta, obs \sim beta(\alpha + y_i, \beta + n_i - y_i)$.

```r
library(ggplot2)
library(gridExtra)
library(tidyr)

# load data
y <- c(0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,2,2,2,2,2,2,2,2,
2,1,5,2,5,3,2,7,7,3,3,2,9,10,4,4,4,4,4,4,4,10,4,4,4,5,
11,12, 5,5,6,5,6,6,6,6,16,15,15,9,4)
n <- c(20,20,20,20,20,19,19,19,19,18,17,20,20,20,20,19,19,18,18,25,24,
23,20,20,20,20,20,20,10,49,19,46,27,17,49,47,20,20,13,48,50,20,
20,20,20,20,20,20,48,19,19,19,22,46,49,20,20,23,19,22,20,20,20,
52,46,47,24,14)

# create the grid
x <- seq(0.0001, 0.9999, length.out = 1000)

# beta distribution
bdens <- function(n, y, x)
  dbeta(x, y+1.4, n-y+8.6)
df_sep <- mapply(bdens, n, y, MoreArgs = list(x = x)) %>%
  as.data.frame() %>% cbind(x) %>% gather(ind, p, -x)

# plot the separate model
plot_sep <- ggplot(data = df_sep) +
  geom_line(aes(x = x, y = p, group = ind)) +
  labs(x = expression(theta), y = '', title = 'Joint Posterior Distribution', color = '') +
  scale_y_continuous(breaks = NULL) +
  theme(legend.background = element_blank(), legend.position = c(0.8,0.9))
# The last one is for emphasize colored red
plot_sep
```

Joint Posterior Distribution



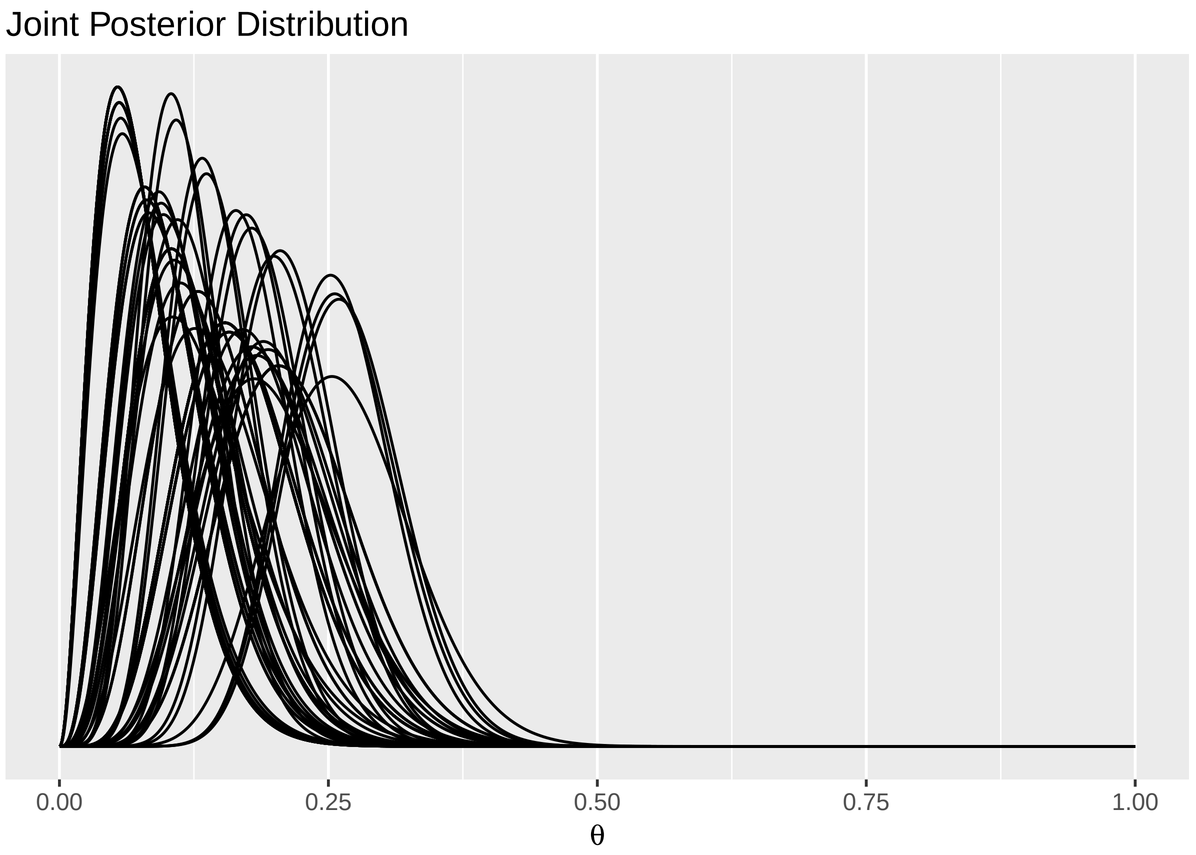
2.

According to the distributions, $E(y_i|\alpha,\beta) = E(E(y_i|\alpha,\beta,\theta_i)) = E(n_i\theta_i) = n_i\frac{\alpha}{\alpha+\beta}$, $E(y_i^2|\alpha,\beta) = E(E(y_i^2|\alpha,\beta,\theta_i)) = E(n_i\theta_i(1-\theta_i) + (n_i\theta_i)^2) = n_iE(\theta_i) + (n_i^2 - n_i)E(\theta^2) = n_i\frac{\alpha}{\alpha+\beta} + (n_i^2 - n_i)(\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} + (\frac{\alpha}{\alpha+\beta})^2)$. Let the moment of population equal the moment of samples, then $1/68 \times \sum_i y_i/n_i = \frac{\hat{\alpha}}{\hat{\alpha}+\hat{\beta}}, (1/68 \times \sum_i(y_i^2 - y_i)/(n_i^2 - n_i)) = \frac{\hat{\alpha}\hat{\beta}}{(\hat{\alpha}+\hat{\beta})^2(\hat{\alpha}+\hat{\beta}+1)} + (\frac{\hat{\alpha}}{\hat{\alpha}+\hat{\beta}})^2$, I solved these two equations and get $\hat{\alpha} = 3.17, \hat{\beta} = 18.92$, so the posterior distribution is $\theta_i \sim beta(3.17 + y_i, 18.92 + n_i - y_i)$.

In question 1, α, β do not contain information from data, so posterior densities differ a lot. In question 2, α, β are the estimators by MOM, so they are close to the true parameters and posterior densities are more stable.

```r
tmp=mean((y^2-y)/(n^2-n))/(mean(y/n))^2-1
tmp=tmp*mean(y/n)/(1-mean(y/n))
tmp=1/tmp-1
alphahat=tmp*mean(y/n)
betahat=tmp-alphahat

# beta distribution
bdens <- function(n, y, x)
  dbeta(x, y+alphahat, n-y+betahat)
df_sep <- mapply(bdens, n, y, MoreArgs = list(x = x)) %>%
  as.data.frame() %>% cbind(x) %>% gather(ind, p, -x)

# plot the separate model
```

```r
plot_sep <- ggplot(data = df_sep) +
  geom_line(aes(x = x, y = p, group = ind)) +
  labs(x = expression(theta), y = '', title = 'Joint Posterior Distribution', color = '') +
  scale_y_continuous(breaks = NULL) +
  theme(legend.background = element_blank(), legend.position = c(0.8,0.9))
# The last one is for emphasize colored red
plot_sep
```



3.

In this question, I use Jacobian determinant to calculate the density of transformations of Random Variables. Suppose $x = \frac{\alpha}{\alpha+\beta}, y = (\alpha+\beta)^{-1/2}$, then $\alpha = x/y^2, \beta = (1-x)/y^2$ and (x, y) obeys uniform distribution. Then $p(\alpha,\beta) \propto 1 \times |J|^{-1} \propto 1/|\frac{2x}{y^5} + \frac{2-2x}{y^5}| \propto y^5 = (\alpha+\beta)^{-5/2}$, where $|J|$ is the Jacobian determinant.

As for the log transformed scale, $log(\alpha/\beta) = log(x/1-x), log(\alpha+\beta) = -2logy$, so $p(log(\alpha/\beta), log(\alpha+\beta)) \propto 1 \times |J|^{-1} \propto 1/|\frac{1}{x(1-x)} \times \frac{-2}{y}| \propto x(1-x)y \propto \alpha\beta(\alpha+\beta)^{-5/2}$, , where $|J|$ is the Jacobian determinant.

4.

The joint posterior distribution is $p(\alpha,\beta,\theta_1,...,\theta_n|y_1,...,y_n) \propto p(\alpha,\beta)\prod_j \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta_j^{\alpha-1}(1-\theta_j)^{\beta-1}\theta_j^{y_j}(1-\theta_j)^{n_j-y_j} = p(\alpha,\beta)\prod_j \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta_j^{\alpha+y_j-1}(1-\theta_j)^{\beta+n_j-y_j-1}$. Suppose that (α,β) obey a noninformative hyperprior distribution, i.e. $p(\alpha,\beta) \propto (\alpha+\beta)^{-5/2}$. The joint posterior distribution is $p(\alpha,\beta,\theta_1,...,\theta_n|y_1,...,y_n) \propto (\alpha+\beta)^{-5/2}\prod_j \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta_j^{\alpha-1}(1-\theta_j)^{\beta-1}\theta_j^{y_j}(1-\theta_j)^{n_j-y_j} = (\alpha+\beta)^{-5/2}\prod_j \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta_j^{\alpha+y_j-1}(1-\theta_j)^{\beta+n_j-y_j-1}$.

On the original scale the marginal distribution is $p(\alpha,\beta|obs) = \int p(\alpha,\beta,\theta_1,...,\theta_{68}|y_1,...,y_{68})d\theta_1...d\theta_{68} \propto (\alpha+\beta)^{-5/2}\prod_j \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\frac{\Gamma(\alpha+y_j)\Gamma(\beta+n_j-y_j)}{\Gamma(\alpha+\beta+n_j)}$.

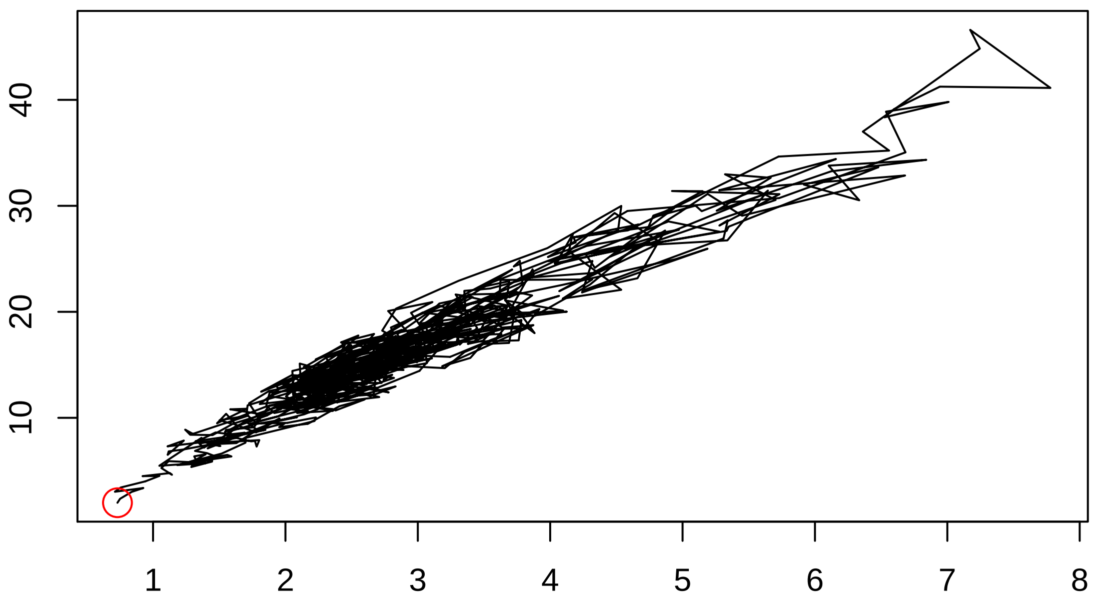
On the log transformed scale the marginal distribution is $p(\log\frac{\alpha}{\beta},\log(\alpha+\beta)|obs) = \int p(\log\frac{\alpha}{\beta},\log(\alpha+\beta),\theta_1,...,\theta_{68}|y_1,...,y_{68})d\theta_1...d\theta_{68} \propto \alpha\beta(\alpha+\beta)^{-5/2}\prod_j \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\frac{\Gamma(\alpha+y_j)\Gamma(\beta+n_j-y_j)}{\Gamma(\alpha+\beta+n_j)}$.

5.

Work with the transformed scale.

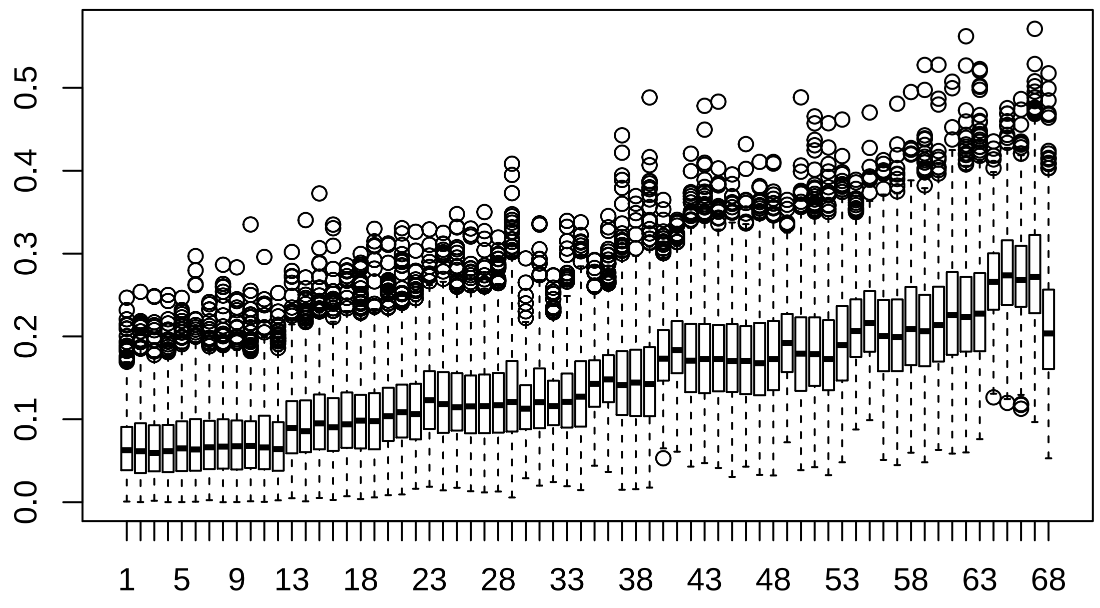
```r
library(mvtnorm)
mcmc_MH <- function(Niter, proposal_sigma, mu_init, LL){
    mu_new <- mu_init
    d <- length(mu_new)
    samples_MH <- mu_new
    accept_MH <- 0
    for(iter in 1: Niter){
        if(d > 1){
            propose_MH_mu <- as.numeric(rmvnorm(1, mean = mu_new,
                                                 sigma = diag(rep(proposal_sigma^2, d))))
        }
        if(d == 1){
            propose_MH_mu <- rnorm(1, mean = mu_new, sd = proposal_sigma)
        }
        rtemp <- LL(propose_MH_mu) - LL(mu_new)
        if(runif(1) < exp(rtemp)){
            mu_new <- propose_MH_mu
            accept_MH <- accept_MH + 1
        }
        samples_MH <- rbind(samples_MH, mu_new)
    }
    return(list(samples = samples_MH, acceptance = accept_MH /Niter))
}

LL <- function(x){
  a = exp(sum(x))/(exp(x[1])+1)
  b = exp(x[2])/(exp(x[1])+1)
  ll1=log(a)+log(b)-2.5*log(a+b)
  ll2=log(gamma(a+y))+log(gamma(b+n-y))-log(gamma(a+b+n))+
    log(gamma(a+b))-log(gamma(a))-log(gamma(b))
  ll1+sum(ll2)
}

trans_inverse = function(x){
  a = exp(sum(x))/(exp(x[1])+1)
  b = exp(x[2])/(exp(x[1])+1)
  c(a,b)
}

Niter = 1000
proposal_sigma = 0.1 # try other values
mu_init = c(-1,1) # try other values
samplesMH = mcmc_MH (Niter, proposal_sigma, mu_init, LL)
```

```r
absamples=data.frame(alpha=exp(rowSums(samplesMH$samples))/(exp(samplesMH$samples[,1])+1),
                     beta=exp(samplesMH$samples[,2])/(exp(samplesMH$samples[,1])+1))
plot(apply(absamples, 2, jitter), type = 'l', xlab = '', ylab = '')
points(t(trans_inverse(mu_init)), cex = 2, col = 'red')
```



6.

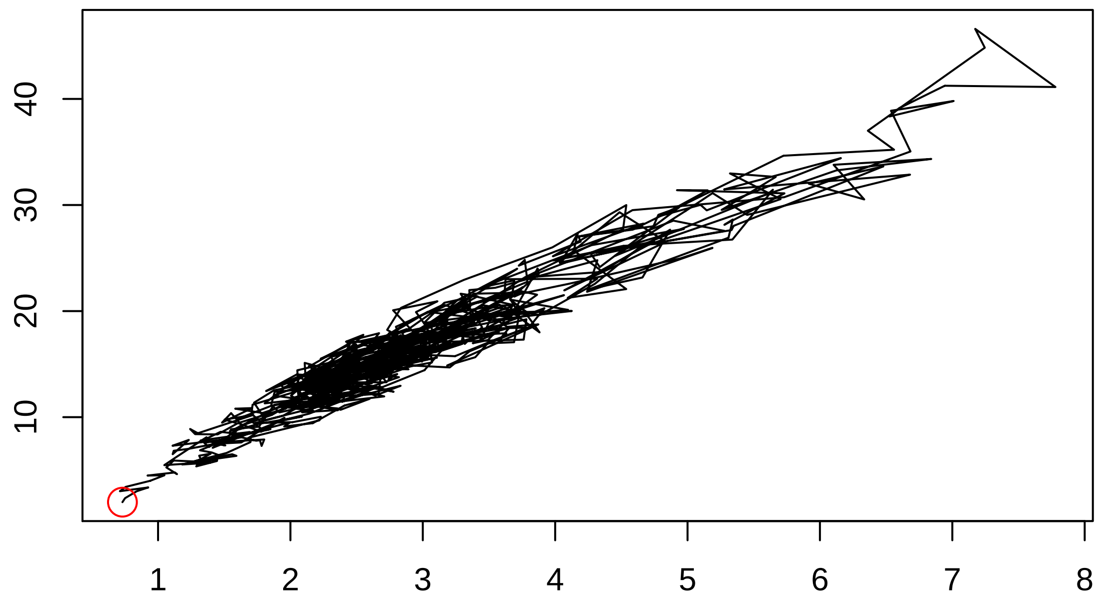
```r
# beta distribution
samplestheta=matrix(rep(0,68*nrow(absamples)),nrow = nrow(absamples))
for(i in 1:nrow(absamples)){
  for(j in 1:68){
    samplestheta[i,j]=rbeta(1,absamples[i,1]+y[j],absamples[i,2]+n[j]-y[j])
  }
}
boxplot(samplestheta)
```



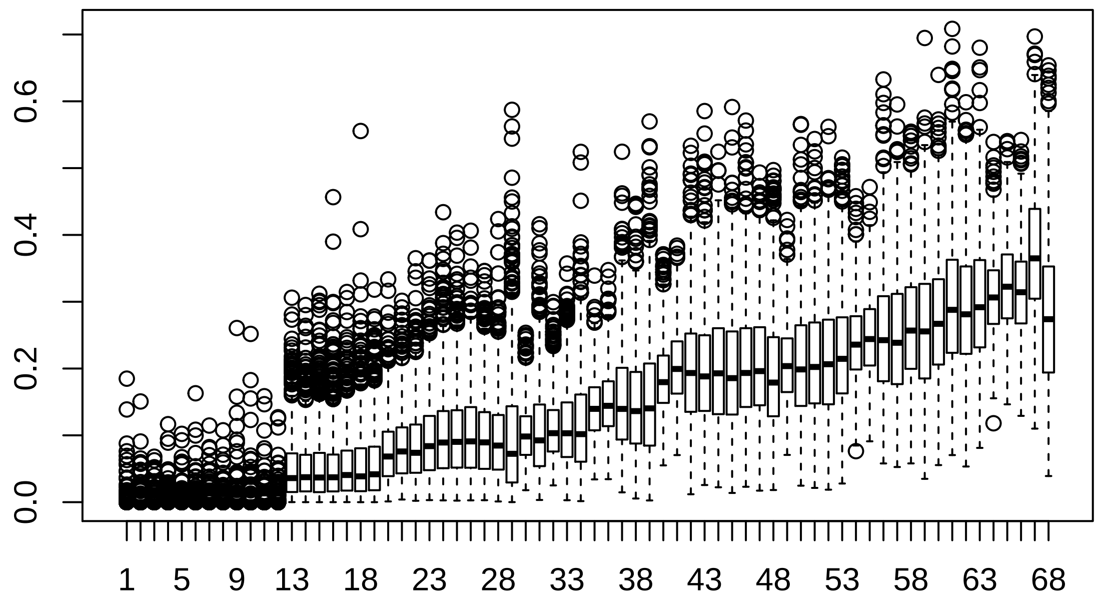
7.

```
library(mvtnorm)
mcmc_MHG <- function(Niter, proposal_sigma, mu_init, LL){
    mu_new <- mu_init
    d <- length(mu_new)
    samples_MH <- mu_new
    thetas=LL(samples_MH)$thetas
    accept_MH <- 0
    for(iter in 1: Niter){
        if(d > 1){
            propose_MH_mu <- as.numeric(rmvnorm(1, mean = mu_new,
                                            sigma = diag(rep(proposal_sigma^2, d))))
        }
        if(d == 1){
            propose_MH_mu <- rnorm(1, mean = mu_new, sd = proposal_sigma)
        }
        rtemp <- LL(propose_MH_mu)$ll - LL(mu_new)$ll
        if(runif(1) < exp(rtemp)){
            mu_new <- propose_MH_mu
            accept_MH <- accept_MH + 1
        }
        samples_MH <- rbind(samples_MH, mu_new)
        thetas=rbind(thetas, LL(propose_MH_mu)$thetas)
```

```r
    }
    absamples=data.frame(alpha=exp(rowSums(samplesMH$samples))/(exp(samplesMH$samples[,1])+1),
                 beta=exp(samplesMH$samples[,2])/(exp(samplesMH$samples[,1])+1))
    return(list(samples = cbind(absamples,thetas), acceptance = accept_MH /Niter))
}

LL = function(x){
  a = exp(sum(x))/(exp(x[1])+1)
  b = exp(x[2])/(exp(x[1])+1)
  samplestheta=1:68
  for(i in 1:68){
    samplestheta[i]=rbeta(1, a+y[i],b+n[i]-y[i])
  }
  ll1=log(a)+log(b)-2.5*log(a+b)
  ll2=log(gamma(a+y))+log(gamma(b+n-y))-log(gamma(a+b+n))+
    (a+y-1)*log(samplestheta)+(b+n-y-1)*log(1-samplestheta)
  ll=ll1+sum(ll2)
  return(list(ll=ll,thetas=samplestheta))
}

Niter = 1000
proposal_sigma = 0.1 # try other values
mu_init = c(-1,1) # try other values
samplesMH = mcmc_MHG (Niter, proposal_sigma, mu_init, LL)
plot(apply(samplesMH$samples[,1:2], 2, jitter), type = 'l', xlab = '', ylab = '')
points(t(trans_inverse(mu_init)), cex = 2, col = 'red')
```



```
boxplot(samplesMH$samples[,3:ncol(samplesMH$samples)])
```



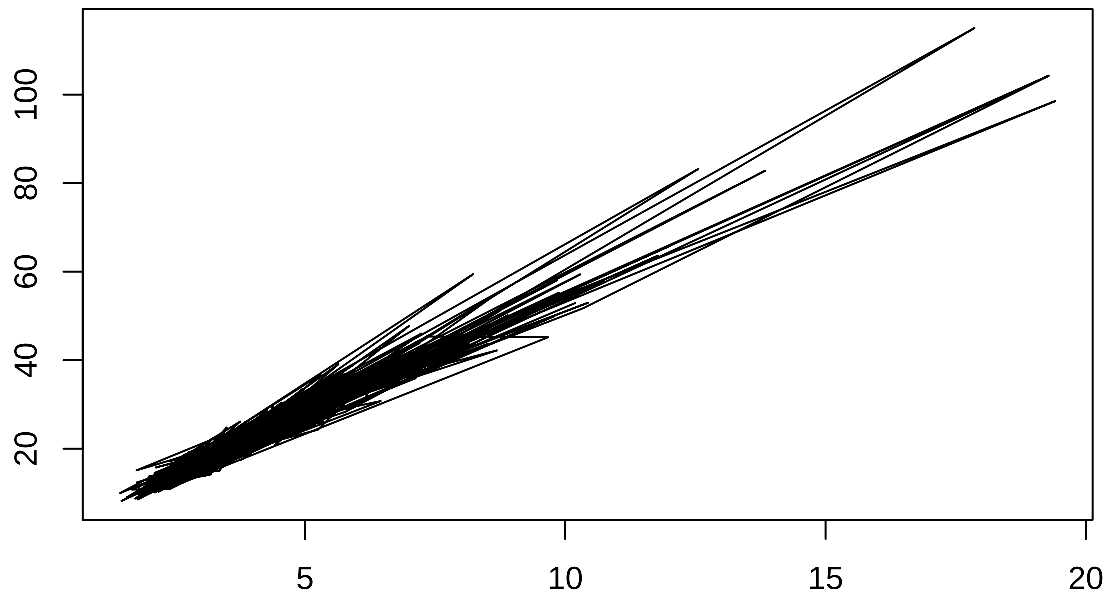
8.

Work with the original scale.

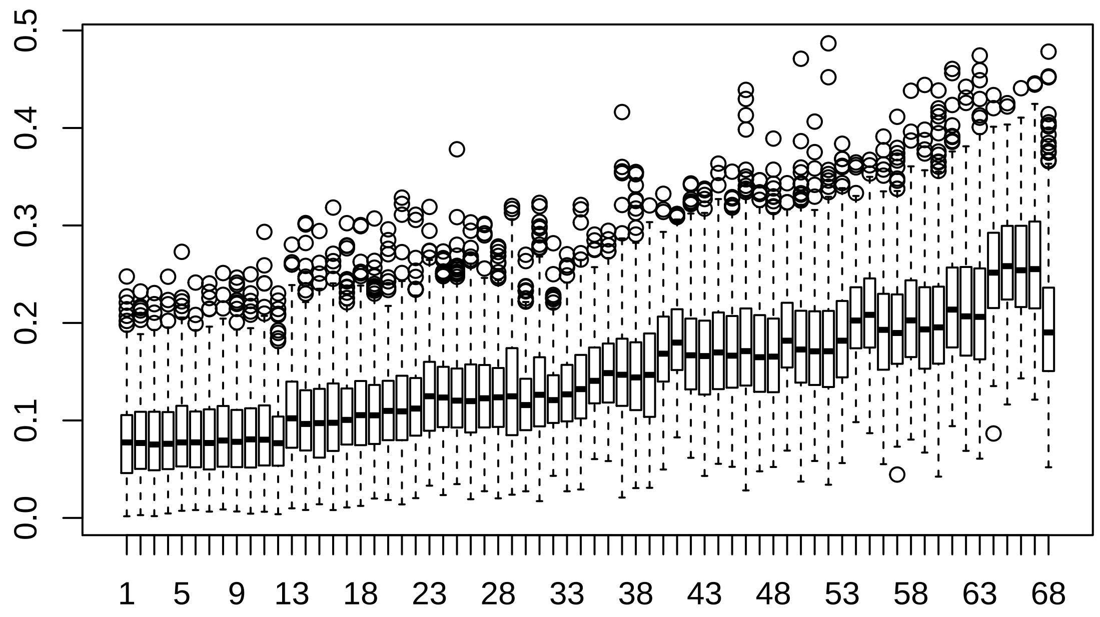
```r
library(rstan)
library(rstudioapi)
rat_data=list(J = 68, y=y,n=n)
fit <- stan(file = 'rat.stan', data = rat_data,
            iter = 1000, chains = 1)
```

```
##
## SAMPLING FOR MODEL 'rat' NOW (CHAIN 1).
##
## Gradient evaluation took 0 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:   1 / 1000 [  0%]  (Warmup)
## Iteration: 100 / 1000 [ 10%]  (Warmup)
## Iteration: 200 / 1000 [ 20%]  (Warmup)
## Iteration: 300 / 1000 [ 30%]  (Warmup)
## Iteration: 400 / 1000 [ 40%]  (Warmup)
## Iteration: 500 / 1000 [ 50%]  (Warmup)
## Iteration: 501 / 1000 [ 50%]  (Sampling)
## Iteration: 600 / 1000 [ 60%]  (Sampling)
## Iteration: 700 / 1000 [ 70%]  (Sampling)
## Iteration: 800 / 1000 [ 80%]  (Sampling)
## Iteration: 900 / 1000 [ 90%]  (Sampling)
## Iteration: 1000 / 1000 [100%]  (Sampling)
##
##  Elapsed Time: 0.384 seconds (Warm-up)
##                0.393 seconds (Sampling)
##                0.777 seconds (Total)
```

```
la <- extract(fit, permuted = TRUE)
plot(apply(cbind(la$alpha,la$beta), 2, jitter), type = 'l', xlab = '', ylab = '')
```



```
boxplot(la$theta)
```



9.

```r
mean(samplesMH$samples[,1])
```

```
## [1] 2.833363
```

```r
mean(samplesMH$samples[,1]<0.2*(samplesMH$samples[,1]+samplesMH$samples[,2]))
```

```
## [1] 0.991009
```

10.

```r
mean(la$alpha)
```

```
## [1] 4.119804
```

```r
mean(la$alpha<0.2*(la$alpha+la$beta))
```

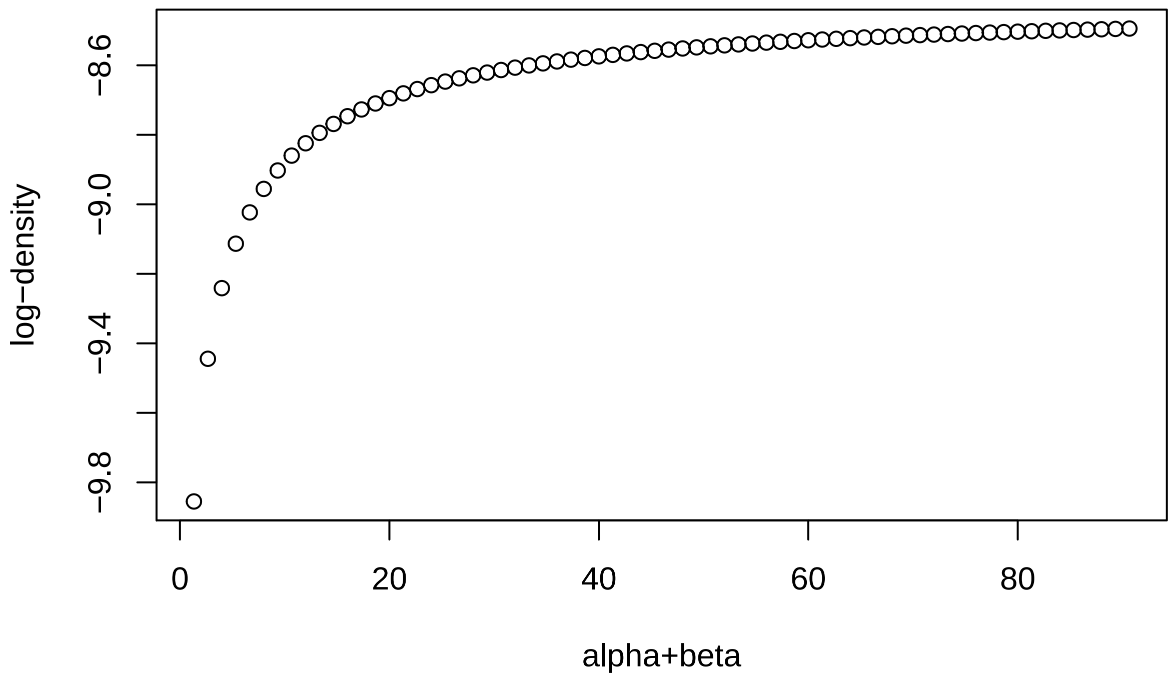
```
## [1] 1
```

11.

The flat priors is not the proper posterior. because if this prior is used, then $\alpha + \beta \to \infty$. In this case the marginal distribution is $p(\alpha, \beta|obs) \propto \prod_j \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \frac{\Gamma(\alpha+y_j)\Gamma(\beta+n_j-y_j)}{\Gamma(\alpha+\beta+n_j)}$. From the plot below we know when α, β

increases by p times ($p > 1$), the log-likelihood increases, thus the integration of the density does not exist. So it is not a proper prior.

```r
ll = function(x){
   a=x[,1]
   b=x[,2]
   y=4
   n=14
   log(gamma(a+y))+log(gamma(b+n-y))-log(gamma(a+b+n))+
    log(gamma(a+b))-log(gamma(a))-log(gamma(b))
}
x=data.frame(a=(1:68)/3,b=1:68)
plot(x$a+x$b,ll(x),xlab = 'alpha+beta',ylab = 'log-density')
```


log−density
alpha+beta