STAT425_Midterm2_Hongyu_Mao

Hongyu Mao

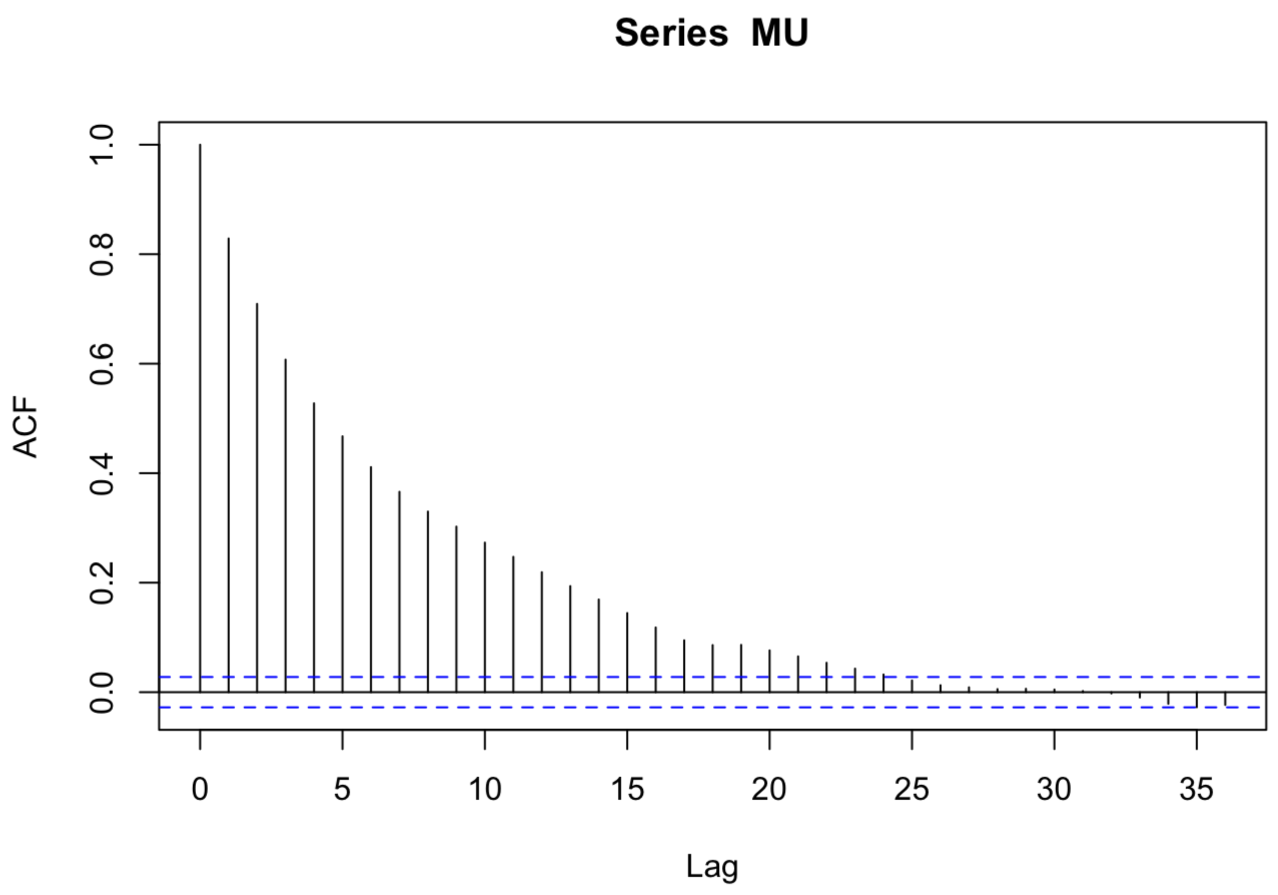
11/29/2018

Problem 1 (b)

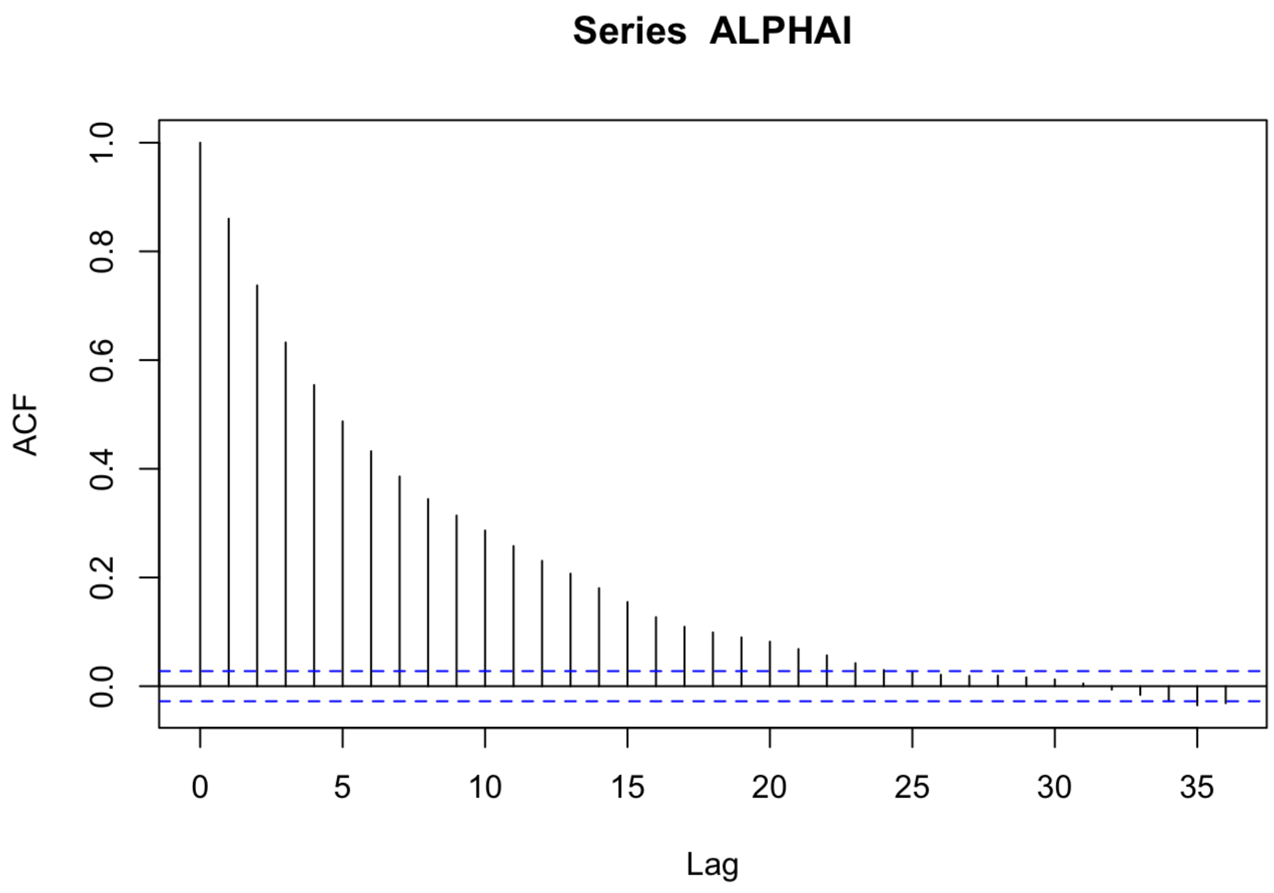
"Convergence diagnosis" means methods to determine if the chain has converged. We need to make sure to check if the converges because otherwise the estimates won't be as nearly precise. Some tools we can use are autocorrelation plots and trace plots. To improve slow convergence, we can parametrize the model again to see if new parametrization can make each step "bigger" in the chain so that the chain can explore the whole space as quickly as possible. We can also combine some correlated variables together as one variable, do integrals to eliminate some variables, and so on.

Problem 1 (c)

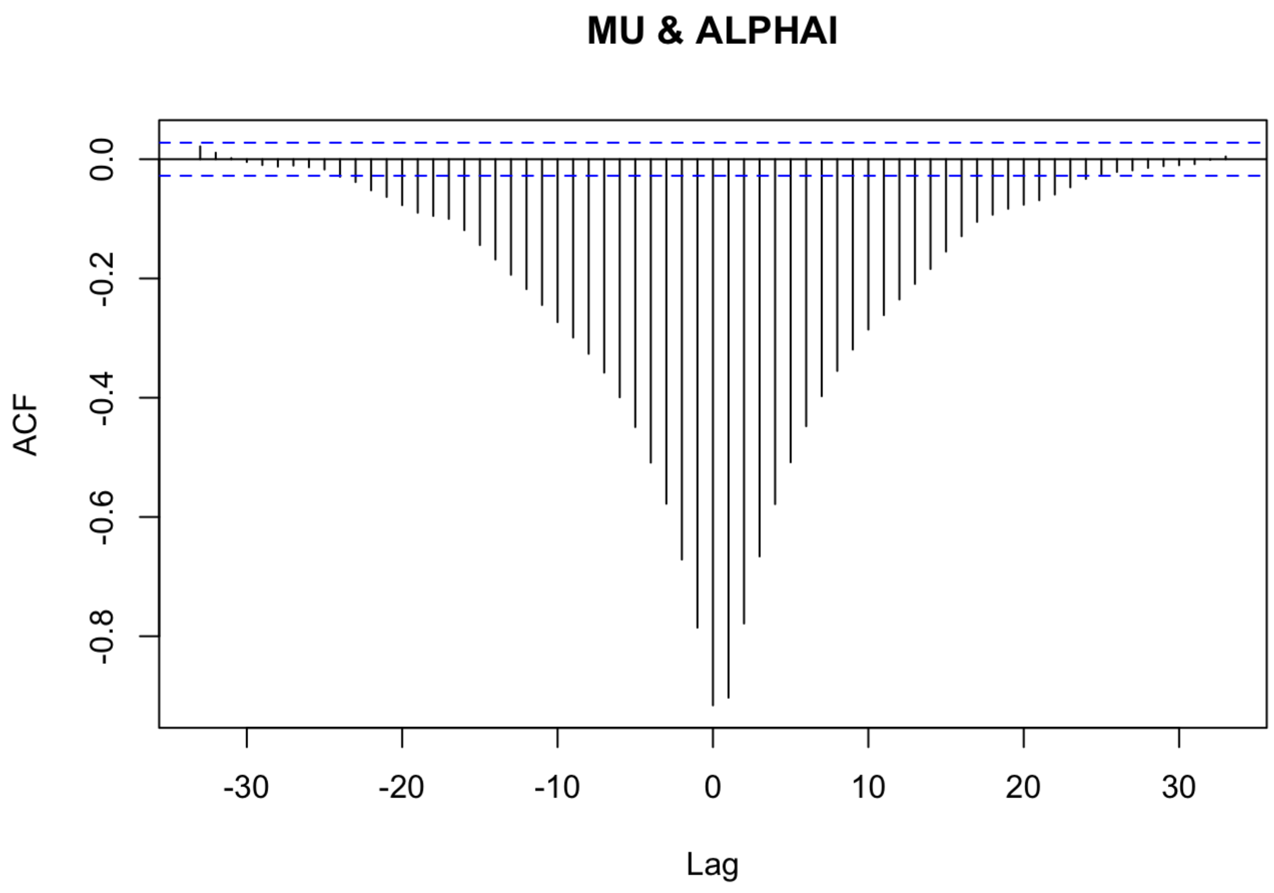
```
## (i) parametrization of (mu, alpha)
# set a flat prior for mu
sigma_mu <- 1
# set values about sigma's
sigma_e <- 1
sigma_alpha <- 1
# set values about others
y <- rnorm(5, 0, sigma_mu) + rnorm(5, 0, sigma_alpha)
alphai <- rnorm(1,0,sigma_alpha)
IJ <- n <- 5

set.seed(1)
MU <- c()
ALPHAI <- c()
for (s in 1: 5000){
  # sample mu
  var_mu <- 1/(1/sigma_mu + n/sigma_e)
  mean_mu <- var_mu*(sum(y-alphai) / sigma_e)
  mu <- rnorm(1, mean_mu, sqrt(var_mu))
  # sample alphai
  var_alphai <- 1/(1/sigma_alpha+n/sigma_e)
  mean_alphai <- var_alphai*(alphai/sigma_alpha+sum(y-mu)/sigma_e)
  alphai <- rnorm(1, mean_alphai, sqrt(var_alphai))
  # store values
  MU <- c(MU, mu)
  ALPHAI <- c(ALPHAI, alphai)
}
acf(MU)
```



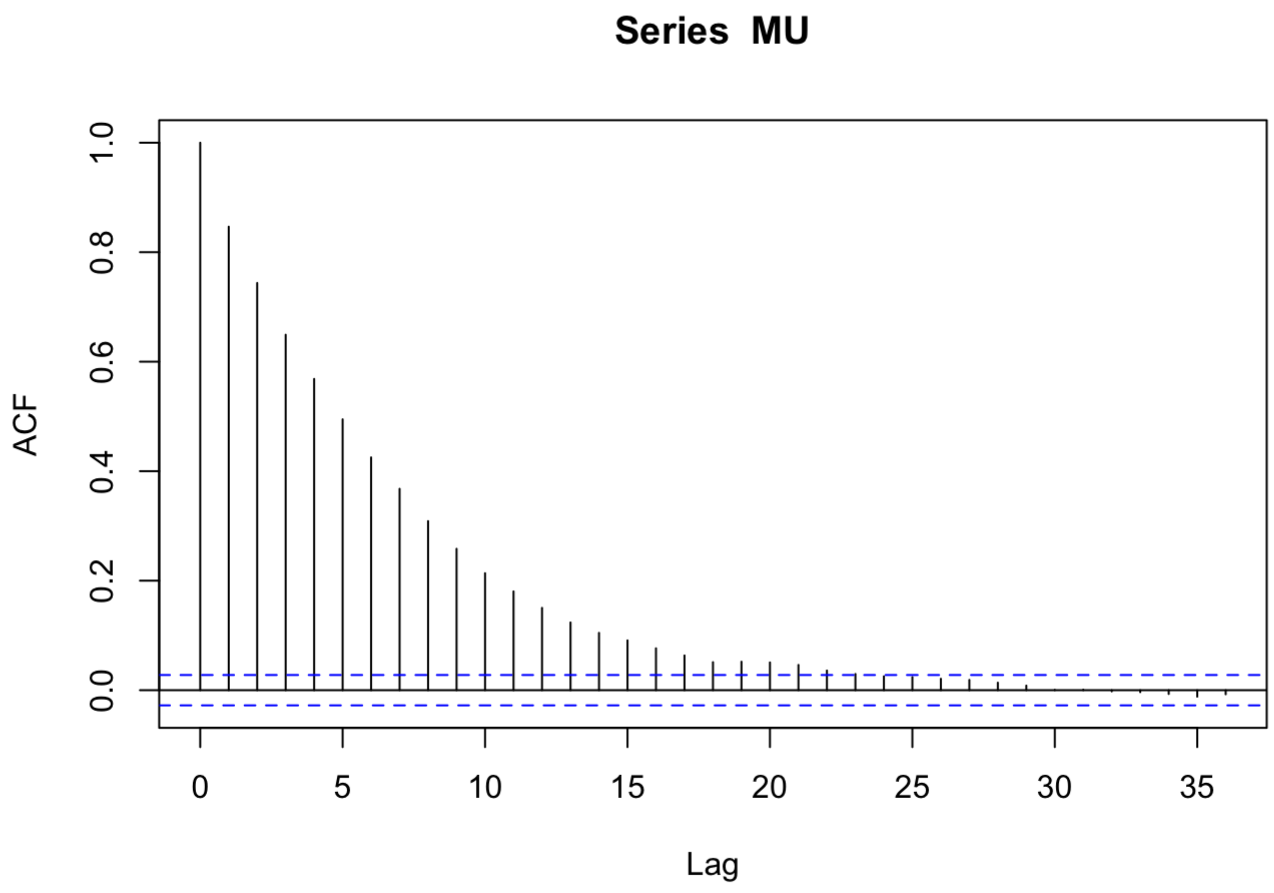
```
acf(ALPHAI)
```



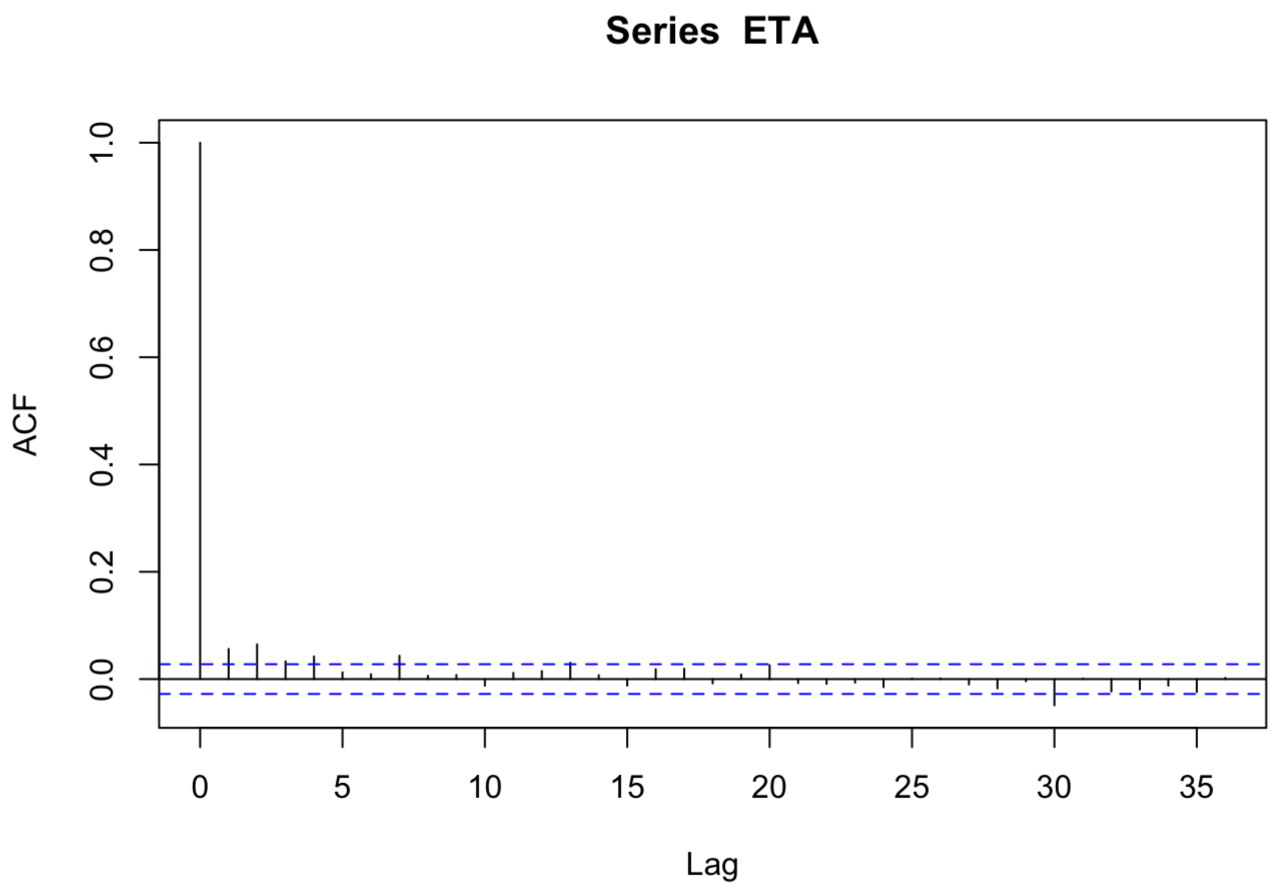
```
ccf(MU, ALPHAI)
```



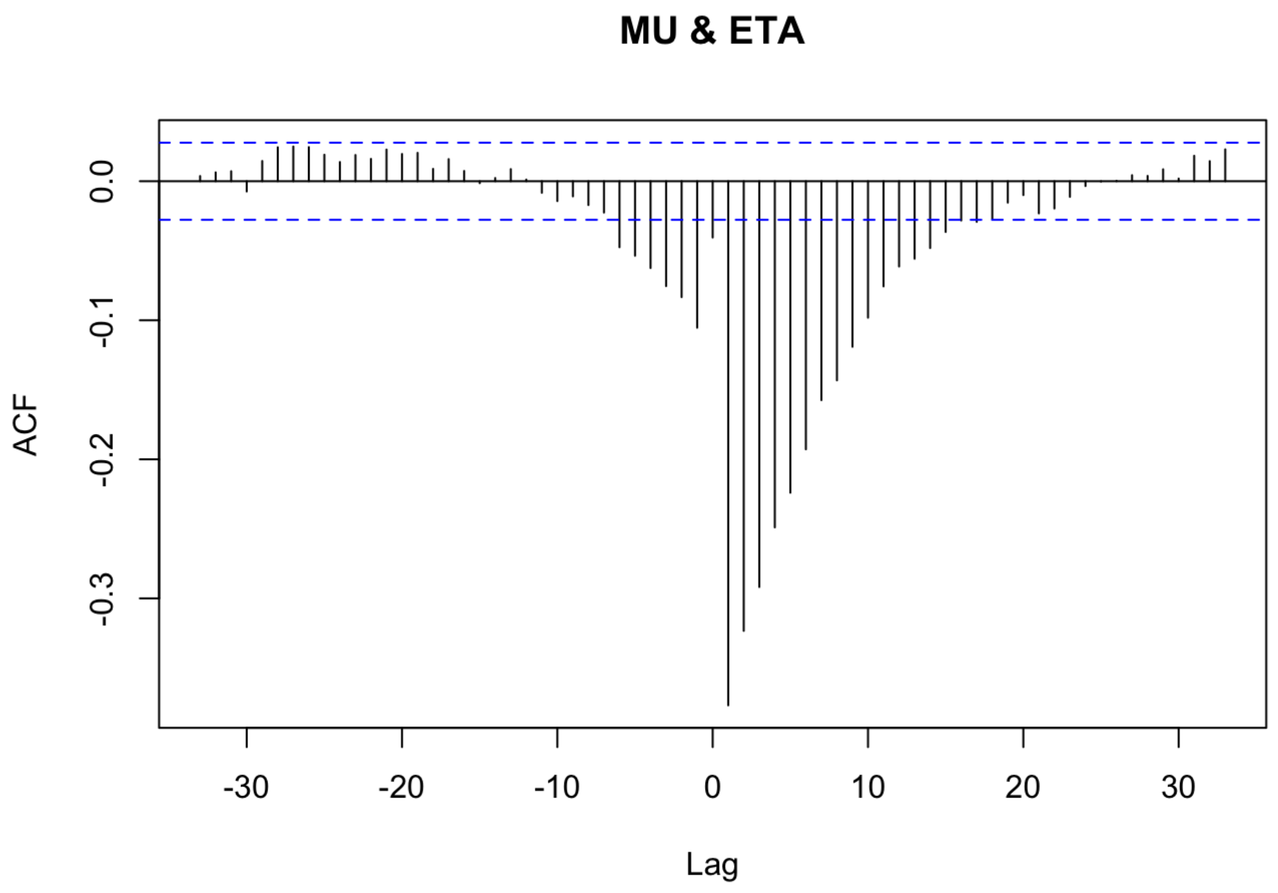
```
## (ii) parametrization of (mu, eta)
alphai <- rnorm(1,0,sigma_alpha)
eta <- alphai

set.seed(2)
MU <- c()
ETA <- c()
for (s in 1: 5000){
  ## sample mu
  var_mu <- 1/(1/sigma_mu+n/sigma_e)
  mean_mu <- var_mu*(sum(y-alphai)/sigma_e)
  mu <- rnorm(1, mean_mu, sqrt(var_mu))
  ## sample alphai
  var_alphai <- 1/(1/sigma_alpha+n/sigma_e)
  mean_alphai <- var_alphai*(alphai/sigma_alpha+sum(y-mu)/sigma_e)
  alphai <- rnorm(1, mean_alphai, sqrt(var_alphai))
  ## calculate eta
  eta <- mu+alphai
  # store values
  ETA <- c(ETA, eta)
  MU <- c(MU, mu)
}
acf(MU)
```



```
acf(ETA)
```



```
ccf(MU, ETA)
```

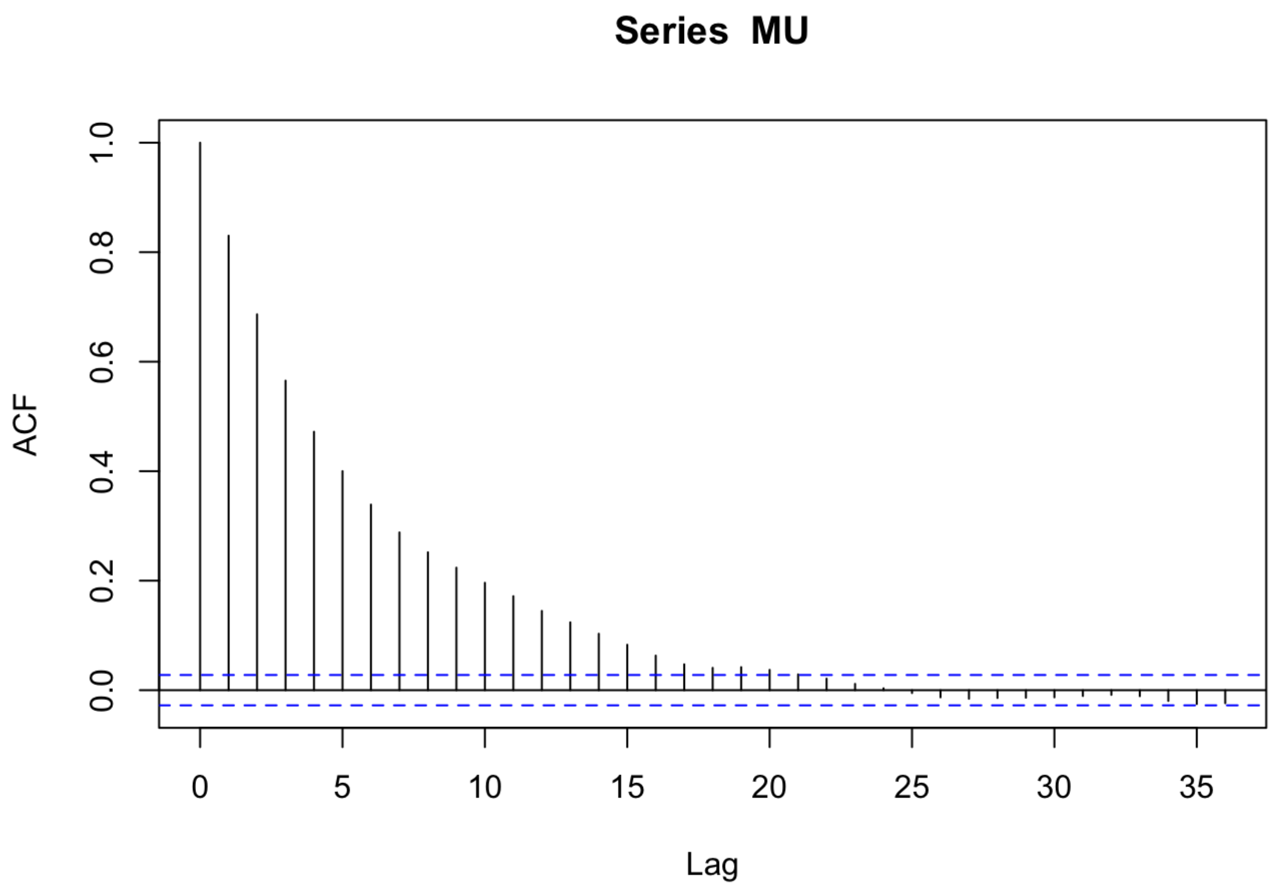


```
cat("\n")
```

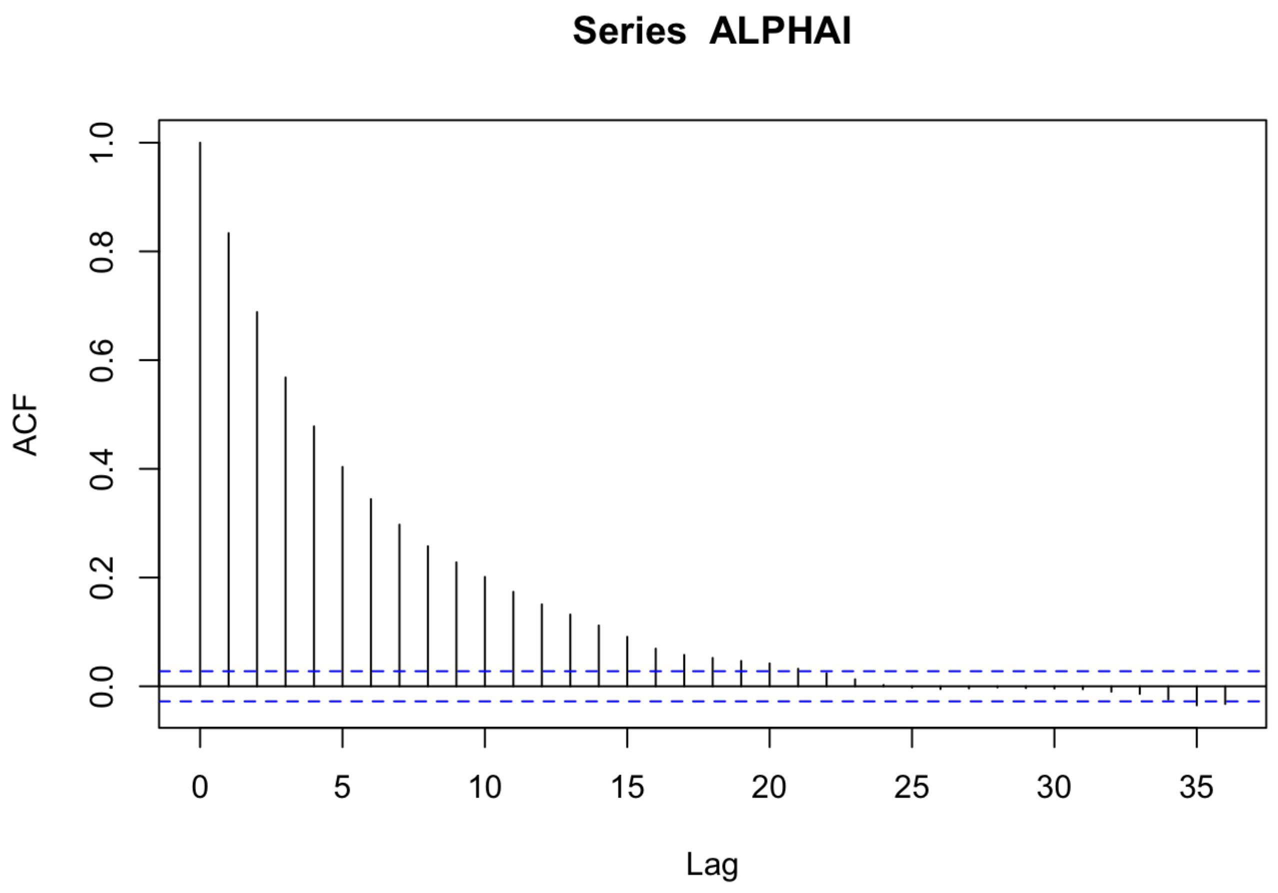
The (ii) parametrization is better than (i) according to the graphs because both autocorrelations and crosscorrelations in (ii) converge quicker than the ones in (i).

Problem 1 (d)

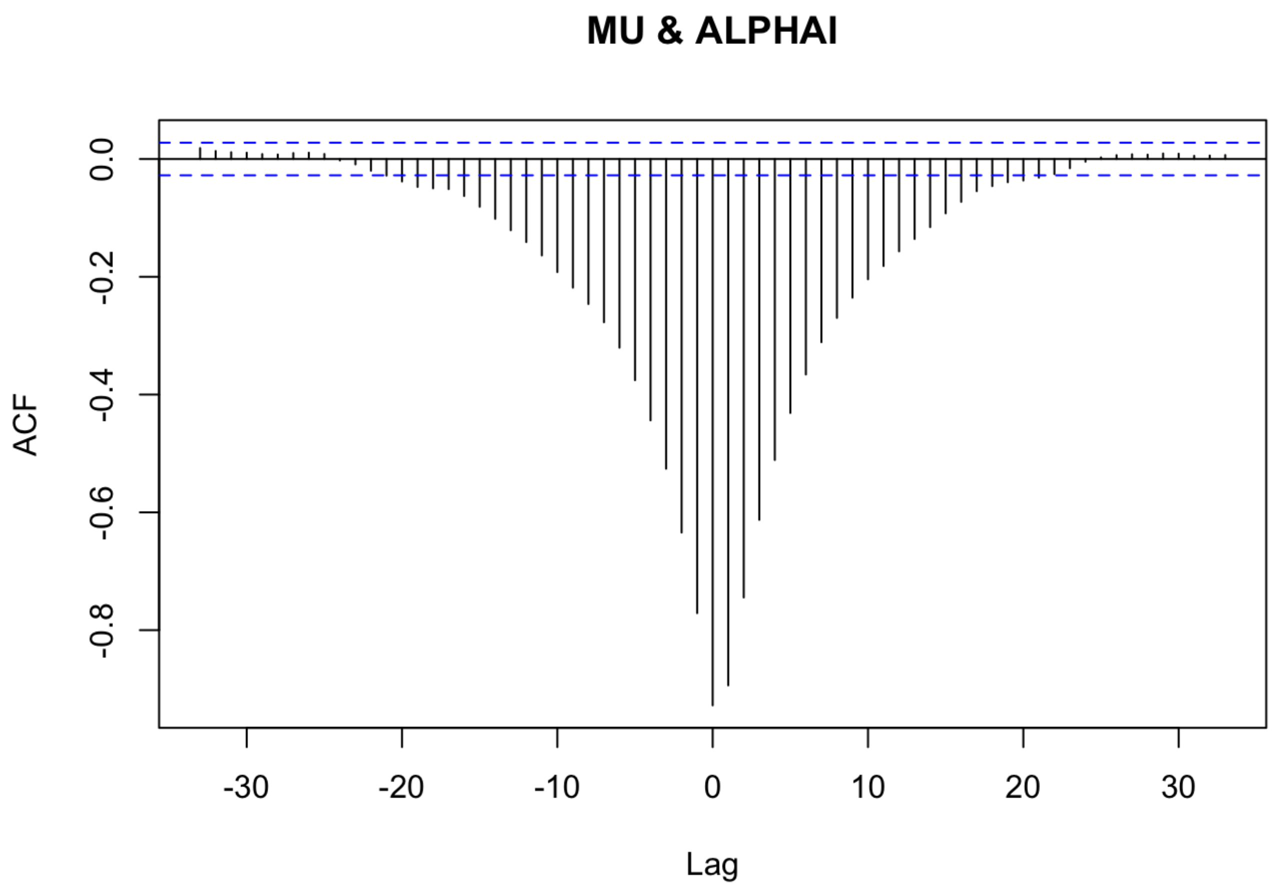
```
## (i) parametrization of (mu, alpha)
# set a flat prior for mu
sigma_mu <- 1
# set values about sigma's
sigma_e <- 1
sigma_alpha <- 10
# set values about others
y <- rnorm(5, 0, sigma_mu) + rnorm(5, 0, sigma_alpha)
alphai <- rnorm(1,0,sigma_alpha)
IJ <- n <- 5

set.seed(1)
MU <- c()
ALPHAI <- c()
for (s in 1: 5000){
  # sample mu
  var_mu <- 1/(1/sigma_mu + n/sigma_e)
  mean_mu <- var_mu*(sum(y-alphai) / sigma_e)
  mu <- rnorm(1, mean_mu, sqrt(var_mu))
  # sample alphai
  var_alphai <- 1/(1/sigma_alpha+n/sigma_e)
  mean_alphai <- var_alphai*(alphai/sigma_alpha+sum(y-mu)/sigma_e)
  alphai <- rnorm(1, mean_alphai, sqrt(var_alphai))
  # store values
  MU <- c(MU, mu)
  ALPHAI <- c(ALPHAI, alphai)
}
acf(MU)
```



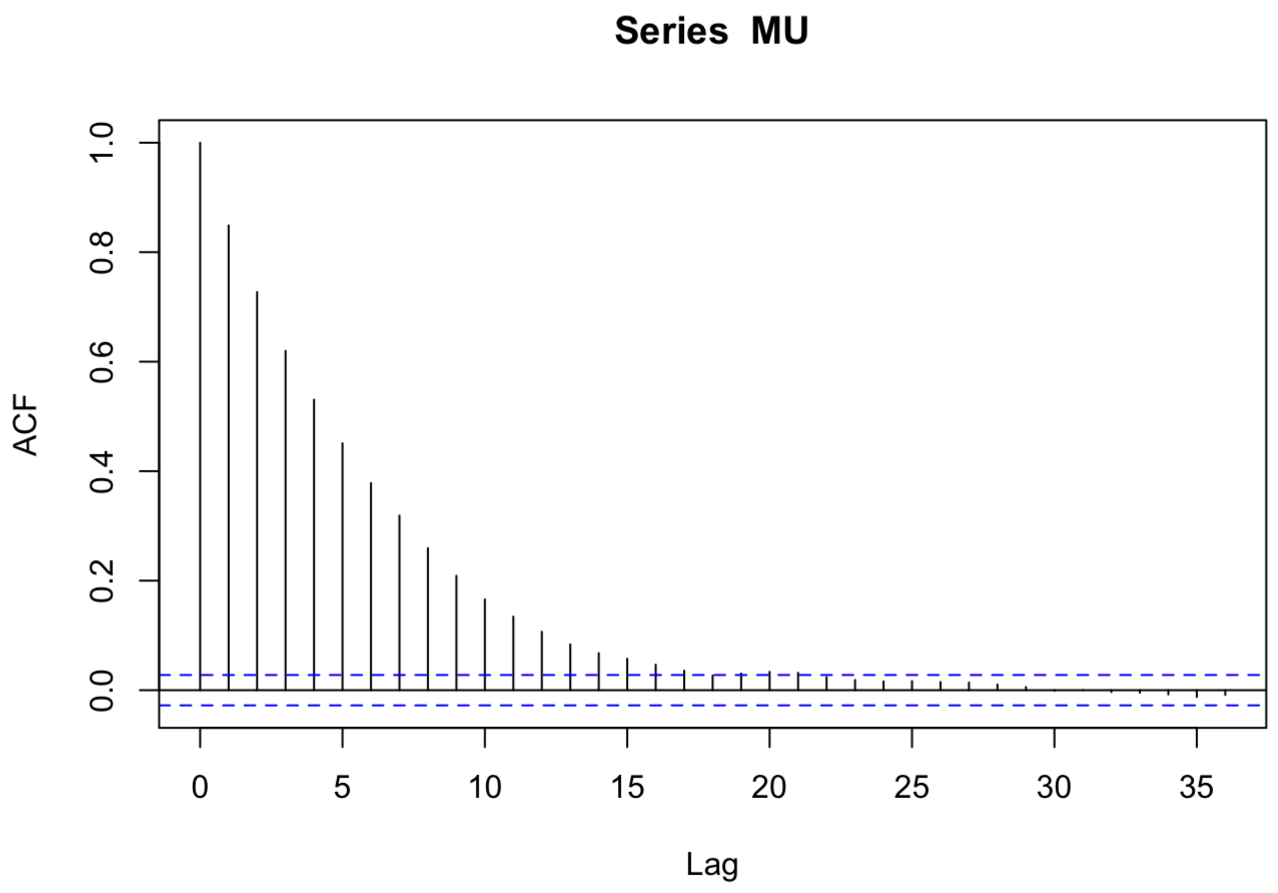
```
acf(ALPHAI)
```



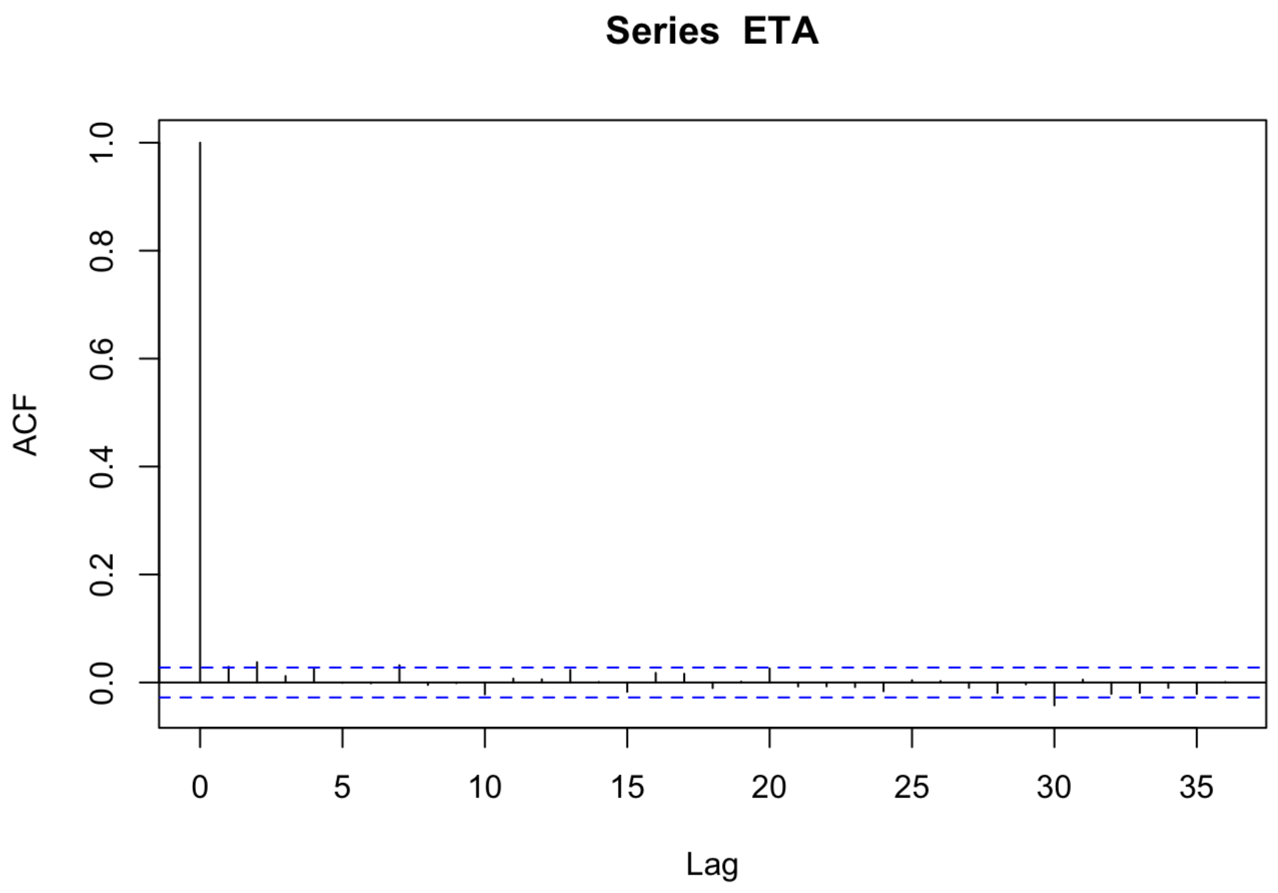
```
ccf(MU, ALPHAI)
```



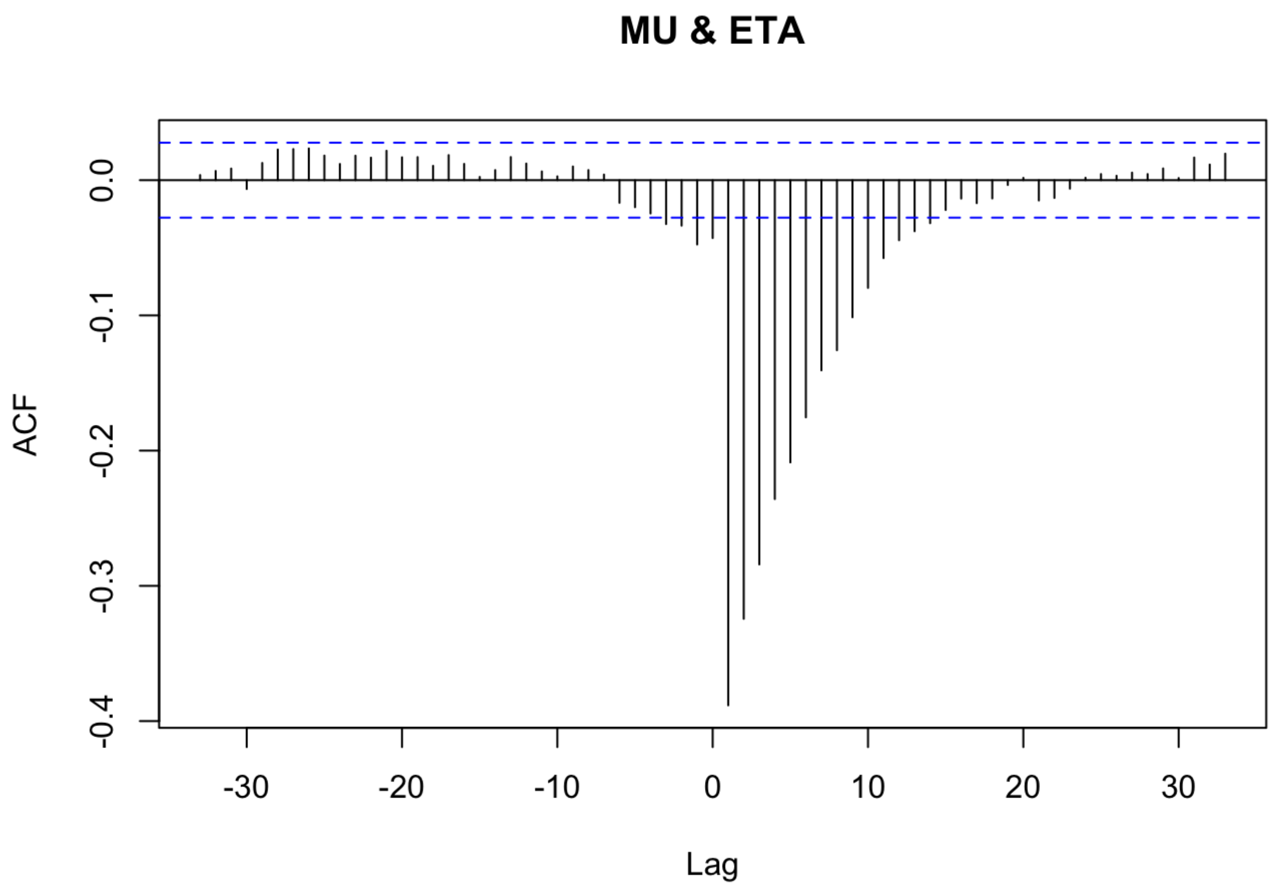
```
## (ii) parametrization of (mu, eta)
alphai <- rnorm(1,0,sigma_alpha)
eta <- alphai

set.seed(2)
MU <- c()
ETA <- c()
for (s in 1: 5000){
  ## sample mu
  var_mu <- 1/(1/sigma_mu+n/sigma_e)
  mean_mu <- var_mu*(sum(y-alphai)/sigma_e)
  mu <- rnorm(1, mean_mu, sqrt(var_mu))
  ## sample alphai
  var_alphai <- 1/(1/sigma_alpha+n/sigma_e)
  mean_alphai <- var_alphai*(alphai/sigma_alpha+sum(y-mu)/sigma_e)
  alphai <- rnorm(1, mean_alphai, sqrt(var_alphai))
  ## calculate eta
  eta <- mu+alphai
  # store values
  ETA <- c(ETA, eta)
  MU <- c(MU, mu)
}
acf(MU)
```



```
acf(ETA)
```



```
ccf(MU, ETA)
```



```
cat("\n")
```

The (ii) parametrization is still better than (i) according to the graph. And in this case, the autocreelation and crosscorrelation in (ii) drop even quicker than the ones in question (c). Especially, ETA's autocorrelation drops almost at once. This phenomenon suggests that hierarchical centering parametrizations can make the approximation process converge quicker and perform better.

Problem 2 (a)

An independent prior that is a normal distribution with mean 0 and variance b is an informative prior. A g-prior involves both information from the sample and the prior in it. The g-prior has an advantage in terms of interpretability and flexibility because we can incorporate how strongly our prior guess is into the g-prior by changing the value of g. The constant g is a measure of amount of information relative to the sample versus the information relative to the prior. For example, if we believe strongly in the prior guess and want the sample to have a lower weight in our analysis, then we would choose a small value for g. The disadvantage of a g-prior is that it is complicated to calcualte and inference on. On the other hand, an independent prior is simple to get a closed form of posterior because it's usually conjugate. But an independent prior means super confidence about the prior, so it's not felxible to reflect a statistician's confidence level about the prior guess.

Problem 2 (b)

In this problem, we standardized the BMI, age predictors.

```r
## Clean data
olddat <- read.csv('midterm2_hersdata_1.csv')

newdat <- olddat # make a copy of the data
newdat <- newdat[, -1] # delete id column
newdat$BMI <- scale(newdat$BMI) #standardize BMI
newdat$age <- scale(newdat$age) # standardize age

for (i in c(3,4,5,6,7)){           # change yes/no precitors to binary values
  newdat[,i] <- ifelse(newdat[,i]=="yes",1,0)
}
# change HT precitors to binary values.
# HT values are not characters, so convert them to numbers first,
# and R automatically codes hermone therapy as 1 and placebo as 2,
# then change placebo from 2 to 0.
newdat[,1] <- as.numeric(newdat[,1])
for (i in 1:nrow(newdat)) {
  if (newdat[i, 1] == 2) {
    newdat[i, 1] <- 0
  }
}
# View(newdat)
# View(olddat)

y <- newdat$LDL1
n <- length(y)
## Do g-prior inference
# data: y, X
# prior parameters : g , nu0 , s20
# number of independent samples to generate : S

# set g as the amount of data entries
# set a sigma's prior as inv-gamma(1,1) with nu0 = 2, s20 = 1
g <- n ; nu0 <- 2 ; s20 <-1
X <- newdat[,-9]
X <- as.matrix(cbind(rep(1, n), X))
# X <- as.matrix(X)

S <-10000
p <- dim(X)[2]
Hg <- (g/(g+1)) * X%*%solve(t(X)%*%X)%*%t(X)
SSRg <- t(y)%*%( diag(1,nrow=n) - Hg ) %*%y
s2 <-1/rgamma(S, (nu0+n)/2, (nu0*s20+SSRg)/2 )
Vb <- g*solve(t(X)%*%X)/(g+1)
Eb <- Vb%*%t(X)%*%y
E <- matrix(rnorm(S*p, 0, sqrt(s2)), S, p)
beta <- t(t(E%*%chol(Vb)) + c(Eb))

cat("means for beta using g-prior: \n")
```

```
## means for beta using g-prior:
```

```
(beta_mean <- apply(beta, 2, mean))
```

```
##   rep(1, n)          HT         age     smoking    drinkany    exercise 
## 145.2102471 -15.5658627  -1.6171967   3.0229630  -0.8765865  -0.6525549 
##     statins    diabetes         BMI 
## -10.3747950  -4.2579751   1.6729612
```

```
cat("\n 95% CI's for beta using g-prior: \n")
```

```
## 
##  95% CI's for beta using g-prior:
```

```
(beta_CI <- apply(beta, 2, function(beta){quantile(beta, c(0.025, 0.975))}))
```

```
##       rep(1, n)        HT        age   smoking  drinkany  exercise
## 2.5%   142.0093 -18.45295 -3.1270190 -1.580529 -3.863526 -3.724418
## 97.5%  148.3689 -12.68930 -0.1009636  7.487385  2.126217  2.389105
##          statins  diabetes        BMI
## 2.5%  -13.410410 -7.767049 0.08862957
## 97.5%  -7.341745 -0.757345 3.25118514
```

Discussion: As we can see from the beta coefficients, the beta for HT (around -15) has a very high absolute value, which means HT have big negative impact on LDL. So a practical importance of this analysis for the physicians is that the use of HT has a large negative relationship with LDL, and using HT is likely to decrease the LDL for a patient. The beta for the use of statins (around -10) also has a large absolute value, which means the use of statins is nagatively associated with LDL in a significant extent. So statins are helpful to physicians to try to lower LDL for pattients. The beta for smoking status is around 3, which means smoking has somehow positive relationship with LDL value. So a patient who smokes is likely to have a higher LDL than a patient who doesn’t. & nbsp;

Problem 2 (c)

```
# add a "1" at the beginning of X vector for each case for the intercept
case1 <- c(1,
           1,
           (60 - mean(olddat$age))/sd(olddat$age),
           0,
           0,
           1,
           0,
           0,
           (25.8 - mean(olddat$BMI))/sd(olddat$BMI))
pred1 <- beta%*%case1
cat("Under case1, the median of the LDL predictions is:", median(pred1), "\n",
    "the range of values of the LDL predictions is:", range(pred1), "\n")
```

```
## Under case1, the median of the LDL predictions is: 129.7532
##  the range of values of the LDL predictions is: 122.687 136.9123
```

```
case2 <- c(1,
           1,
           (60 - mean(olddat$age))/sd(olddat$age),
           0,
           0,
           1,
           1,
           0,
           (25.8 - mean(olddat$BMI))/sd(olddat$BMI))
pred2 <- beta%*%case2
cat("Under case2, the median of the LDL predictions is:", median(pred2), "\n",
    "the range of values of the LDL predictions is:", range(pred2), "\n")
```

```
## Under case2, the median of the LDL predictions is: 119.358
##  the range of values of the LDL predictions is: 112.229 127.9678
```

```
case3 <- c(1,
           0,
           (60 - mean(olddat$age))/sd(olddat$age),
           0,
           0,
           1,
           0,
           0,
           (25.8 - mean(olddat$BMI))/sd(olddat$BMI))
pred3 <- beta%*%case3
cat("Under case3, the median of the LDL predictions is:", median(pred3), "\n",
    "the range of values of the LDL predictions is:", range(pred3), "\n")
```

```
## Under case3, the median of the LDL predictions is: 145.336
##  the range of values of the LDL predictions is: 137.452 152.2801
```

Problem 2 (d)

A g value that puts equal weight to prior and likelihood is g = 1. A g value that has the equivalent weight of one observation means 1/g = 1/n, so g = n. A g value that Fernandez et al suggested is g = max(n, r^2) where r is the total number of covariates. I would prefer g = n, which is the g values that has the equivalent weight of one observation, because we don’t have a clear and very informative prior in our example, so we shouldn’t put too much weight on the prior. Then we would better to choose a g that puts more weight on the data in order to do a more reasonable posterior inference. In our example, since n = 2604 and r can only be up to 8, i.e. r^2 can only be up to 64, so max(n, r^2) will always be n. Then g = n is the same as g = max(n, r^2). This is what happens when we have way more observations than covariates. When we have way more covariates than observations, then max(n, r^2) might be r^2.

```
## g = 1 --------------------------------------------
g <- 1 ; nu0 <- 2 ; s20 <-1
X <- newdat[,-9]
# X <- as.matrix(cbind(rep(1, n), X))
X <- as.matrix(X)

S <-10000
p <- dim(X)[2]
Hg <- (g/(g+1)) * X%*%solve(t(X)%*%X)%*%t(X)
SSRg <- t(y)%*%( diag(1,nrow=n) - Hg ) %*%y
s2 <-1/rgamma(S, (nu0+n)/2, (nu0*s20+SSRg)/2 )
Vb <- g*solve(t(X)%*%X)/(g+1)
Eb <- Vb%*%t(X)%*%y
E <- matrix(rnorm(S*p, 0, sqrt(s2)), S, p)
beta <- t(t(E%*%chol(Vb)) + c(Eb))

cat("g = 1: \n")
```

```
## g = 1:
```

```
cat("means for beta: \n")
```

```
## means for beta:
```

```
(beta_mean <- apply(beta, 2, mean))
```

```
##        HT       age   smoking  drinkany  exercise   statins  diabetes
## 22.112101  2.853469 32.441843 27.481204 28.205835 19.077010 27.658109
##       BMI
##  2.298939
```

```
cat("\n 95% CI's for beta: \n")
```

```
##
##  95% CI's for beta:
```

```
(beta_CI <- apply(beta, 2, function(beta){quantile(beta, c(0.025, 0.975))}))
```

```
##             HT        age  smoking drinkany exercise  statins diabetes
## 2.5%  16.62266 -0.2253128 23.36037 21.72391 22.40493 13.12254 20.80588
## 97.5% 27.63947  6.0467569 41.59447 33.19132 34.05823 25.06094 34.47496
##              BMI
## 2.5%  -0.9322584
## 97.5%  5.5380629
## g = n ---------------------------------------------
g <- n ; nu0 <- 2 ; s20 <-1
X <- newdat[,-9]
# X <- as.matrix(cbind(rep(1, n), X))
X <- as.matrix(X)

S <-10000
p <- dim(X)[2]
Hg <- (g/(g+1)) * X%*%solve(t(X)%*%X)%*%t(X)
SSRg <- t(y)%*%( diag(1,nrow=n) - Hg ) %*%y
s2 <-1/rgamma(S, (nu0+n)/2, (nu0*s20+SSRg)/2 )
Vb <- g*solve(t(X)%*%X)/(g+1)
Eb <- Vb%*%t(X)%*%y
E <- matrix(rnorm(S*p, 0, sqrt(s2)), S, p)
beta <- t(t(E%*%chol(Vb)) + c(Eb))

cat("\n g = n: \n")
```

```
##
##  g = n:
```

```
cat("means for beta: \n")
```

```
## means for beta:
```

```
(beta_mean <- apply(beta, 2, mean))
```

```
##        HT       age   smoking  drinkany  exercise   statins  diabetes
## 44.124607  5.692573 64.846718 54.951698 56.367396 38.243241 55.371120
##       BMI
##  4.603053
```

```
cat("\n 95% CI's for beta: \n")
```

```
##
##  95% CI's for beta:
```

```
(beta_CI <- apply(beta, 2, function(beta){quantile(beta, c(0.025, 0.975))}))
```

```
##             HT      age  smoking drinkany exercise  statins diabetes
## 2.5%  38.95047 2.676418 56.08876 49.30018 50.78240 32.62638 48.85741
## 97.5% 49.23367 8.707995 73.60719 60.37132 62.01153 43.85257 61.94379
##            BMI
## 2.5%  1.493605
## 97.5% 7.694586
## g = max(n, r^2) ---------------------------------------------
g <- max(n, p^2) ; nu0 <- 2 ; s20 <-1
X <- newdat[,-9]
# X <- as.matrix(cbind(rep(1, n), X))
X <- as.matrix(X)

S <-10000
p <- dim(X)[2]
Hg <- (g/(g+1)) * X%*%solve(t(X)%*%X)%*%t(X)
SSRg <- t(y)%*%( diag(1,nrow=n) - Hg ) %*%y
s2 <-1/rgamma(S, (nu0+n)/2, (nu0*s20+SSRg)/2 )
Vb <- g*solve(t(X)%*%X)/(g+1)
Eb <- Vb%*%t(X)%*%y
E <- matrix(rnorm(S*p, 0, sqrt(s2)), S, p)
beta <- t(t(E%*%chol(Vb)) + c(Eb))

cat("\n g = max(n, r^2): \n")
```

```
##
##  g = max(n, r^2):
```

```
cat("means for beta: \n")
```

```
## means for beta:
```

```
(beta_mean <- apply(beta, 2, mean))
```

```
##        HT       age   smoking  drinkany  exercise   statins  diabetes
## 44.084769  5.711813 64.881970 54.976439 56.344700 38.233803 55.334147
##       BMI
##  4.617778
```

```
cat("\n 95% CI's for beta: \n")
```

```
##
##  95% CI's for beta:
```

```
(beta_CI <- apply(beta, 2, function(beta){quantile(beta, c(0.025, 0.975))}))
```

```
##             HT      age  smoking drinkany exercise  statins diabetes
## 2.5%  38.82332 2.723567 56.31056 49.52704 50.77578 32.67979 48.80932
## 97.5% 49.37322 8.719374 73.54293 60.51695 62.01854 43.81615 61.98550
##            BMI
## 2.5%  1.517203
## 97.5% 7.797879
```

Problem 2 (e)

```
lm_with = lm(LDL1~. + BMI*statins, data = newdat)
lm_with
```

```
##
## Call:
## lm(formula = LDL1 ~ . + BMI * statins, data = newdat)
##
## Coefficients:
## (Intercept)           HT          age      smoking     drinkany
##    145.2188     -15.5056      -1.5859       3.1059      -0.8531
##    exercise      statins     diabetes          BMI  statins:BMI
##     -0.6751     -10.4203      -4.2752       2.9589      -3.6427
```

```
lm_without= lm(LDL1~., data = newdat)
lm_without
```

```
##
## Call:
## lm(formula = LDL1 ~ ., data = newdat)
##
## Coefficients:
## (Intercept)           HT          age      smoking     drinkany
##    145.2567     -15.5692      -1.6254       3.0160      -0.8578
##    exercise      statins     diabetes          BMI
##     -0.6513     -10.3978      -4.2329       1.6687
```

Discussion: In both cases when the model has interaction term and when the model doesn't have interaction term, BMI is assocaited with LDL to a small extent with an coefficient ranging from 1 to 3, but this association obviously increases when the interaction term is added. The interaction has a coefficient around -3.5, which has a fairly large absolute value, so the interaction is negatively associated with LDL. In terms of statistical relevance, the significance of interaction increases the complicacy of the model, so it's harder to predict the consequence if we change either BMI or statins. In terms of practical meaning, the significance of interaction tells physicians that BMI and statins have a mixed effect together.

Problem 2 (f)

For two competing models M1 and M2, Bayes Factor assesses the evidence in favor of M1. The strength of evidence is evaluated by the following scale: when 2LBF is in (0, 2), then the evidence is not really worth considering; when it is in (2, 6), then the evidence is positive; One advantage if BF is that if the likelihood of a model is not available, we can do Bayesian approximation to obtain the posterior and then calculate BF to compare models. Another advantage of BF is its transitivity. If M1 outperforms M2 by 5 times, and M2 outperforms M3 by 6 times, and M1 will outperform M3 by 5*6=30 times in terms of BF. One disadvantage of BF is that it doesn't work for improper priors. Another disadvantage of BF is that if we have very complicated models, then it will take so much time and energy to do Bayesian approximation to calculate the BF.

Problem 2 (g)

We are comparing the following linear regression models: one model has “HT + statins + smoking + diabetes” as predictors, another model has the same predictors except diabetes, which is “HT + statins + smoking”. So the first model is with diabetes, and the second model is without diabetes. When we compare the first model with the second model, according to the 15th line of the output, the BF is 0.28, so 2LBF = 2*log(0.28) = -2.5, which is very weak. Such a BF tells us that diabetes is not significant for LDL in our model.

```
library(BayesFactor)
```

```
## Warning: package 'BayesFactor' was built under R version 3.4.4
```

```
## Loading required package: coda
```

```
## Loading required package: Matrix
```

```
## Warning: package 'Matrix' was built under R version 3.4.4
```

```
## ************
## Welcome to BayesFactor 0.9.12-4.2. If you have questions, please contact Richard More
y (richarddmorey@gmail.com).
##
## Type BFManual() to open the manual.
## ************
```

```
b <- regressionBF(LDL1 ~ HT + statins + smoking + diabetes, data = newdat)
b2 <- b / b["HT + statins + smoking"]
b2
```

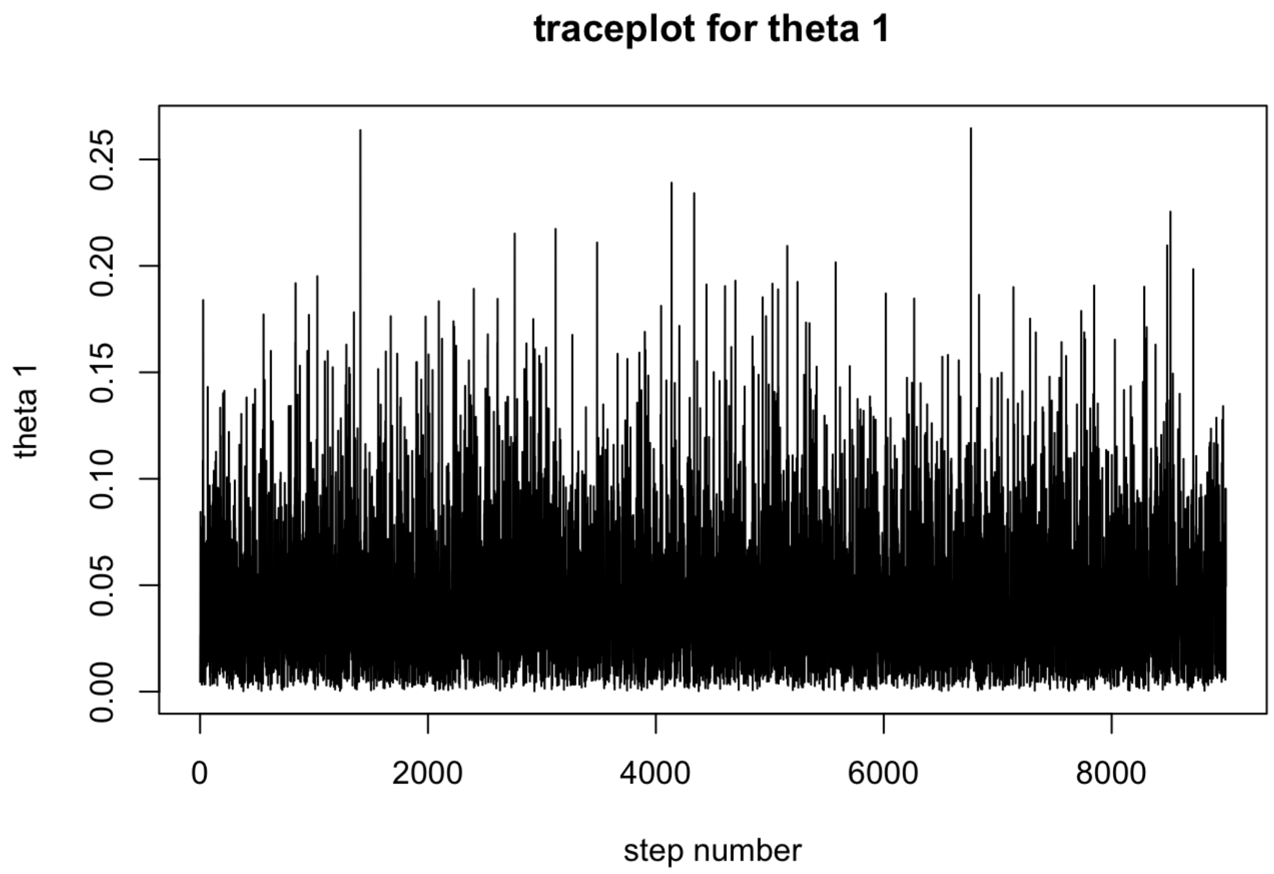
```
## Bayes factor analysis
## --------------
## [1] HT                                  : 2.781099e-09 ±0.01%
## [2] statins                             : 3.947439e-22 ±0.01%
## [3] smoking                             : 2.556693e-30 ±0.01%
## [4] diabetes                            : 8.444309e-31 ±0.01%
## [5] HT + statins                        : 2.775581     ±0.01%
## [6] HT + smoking                        : 2.215325e-09 ±0.01%
## [7] HT + diabetes                       : 5.348578e-10 ±0.01%
## [8] statins + smoking                   : 1.266841e-22 ±0.01%
## [9] statins + diabetes                  : 1.455027e-22 ±0.01%
## [10] smoking + diabetes                 : 5.377106e-31 ±0.01%
## [11] HT + statins + smoking             : 1            ±0%
## [12] HT + statins + diabetes            : 0.866791     ±0.01%
## [13] HT + smoking + diabetes            : 4.107224e-10 ±0.01%
## [14] statins + smoking + diabetes       : 4.508676e-23 ±0.01%
## [15] HT + statins + smoking + diabetes : 0.2860213    ±0.01%
##
## Against denominator:
##   LDL1 ~ HT + statins + smoking
## ---
## Bayes factor type: BFlinearModel, JZS
```

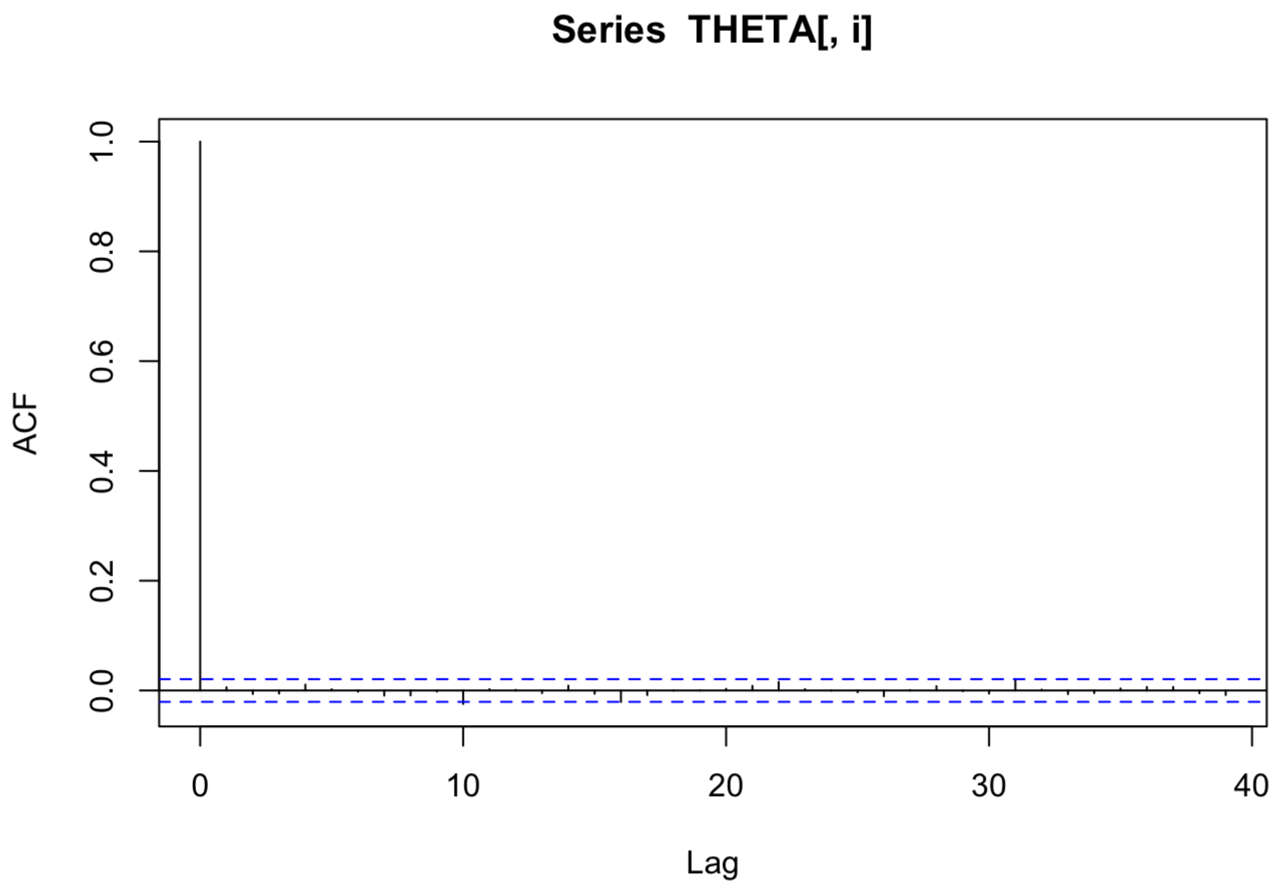
Problem 3 (b)

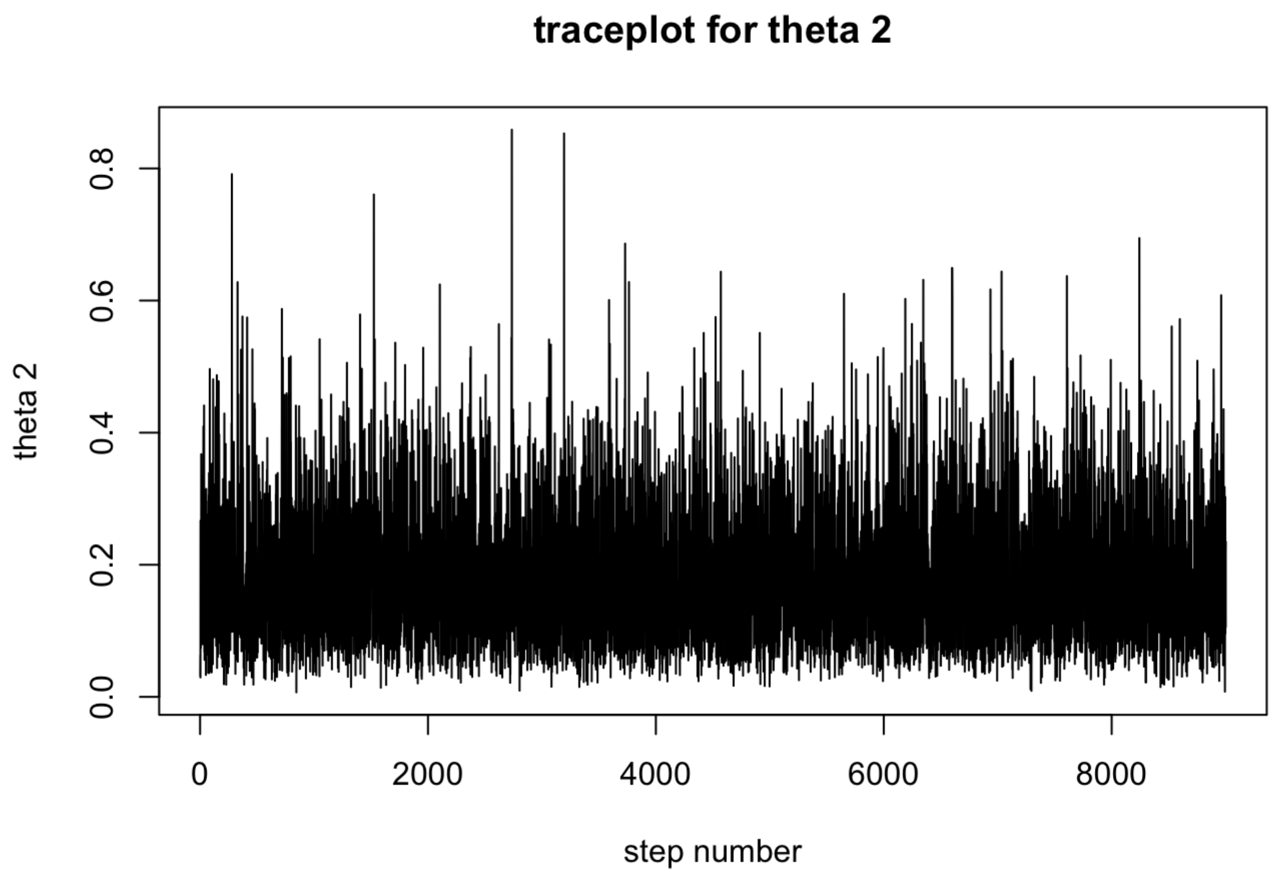
```
y <- c(1, 3, 2, 12, 1, 1)
x <- c(33, 14, 27, 90, 12, 17)

a <- rgamma(1, 1, 1)
b <- rgamma(1, 10, 1)

S <-  10000
accept <- 0
ALPHA <- rep(NA, S)
BETA <- rep(NA, S)
THETA <- array(NA, dim=c(S,6))

starting_a <- 0.001
starting_b <- 7

for(i in 1:S)
  {
    THETA[i,] <- rgamma(6, y+a, x+b)
    a_star <- abs( runif(1, min=a-starting_a, max=a+starting_a) )
    b_star <- abs( runif(1, min=b-starting_b, max=b+starting_b) )
    logr <- (a_star - a) * (sum(log(THETA[i,])) - 1) - (b_star - b) * (sum(THETA[i,]) +
1) + 9 * (log(b_star) - log(b))
    u <- runif(1)
    if (log(u) < logr) {
      a <- a_star
      b <- b_star
      accept <-  accept + 1
      }
    ALPHA[i] <- a
    BETA[i] <- b
  }

## burn in
burn <- 1000
THETA <- THETA[burn:S, ]
ALPHA <- ALPHA[burn:S]
BETA <- BETA[burn:S]

cat("acceptance rate:", accept/S)
```

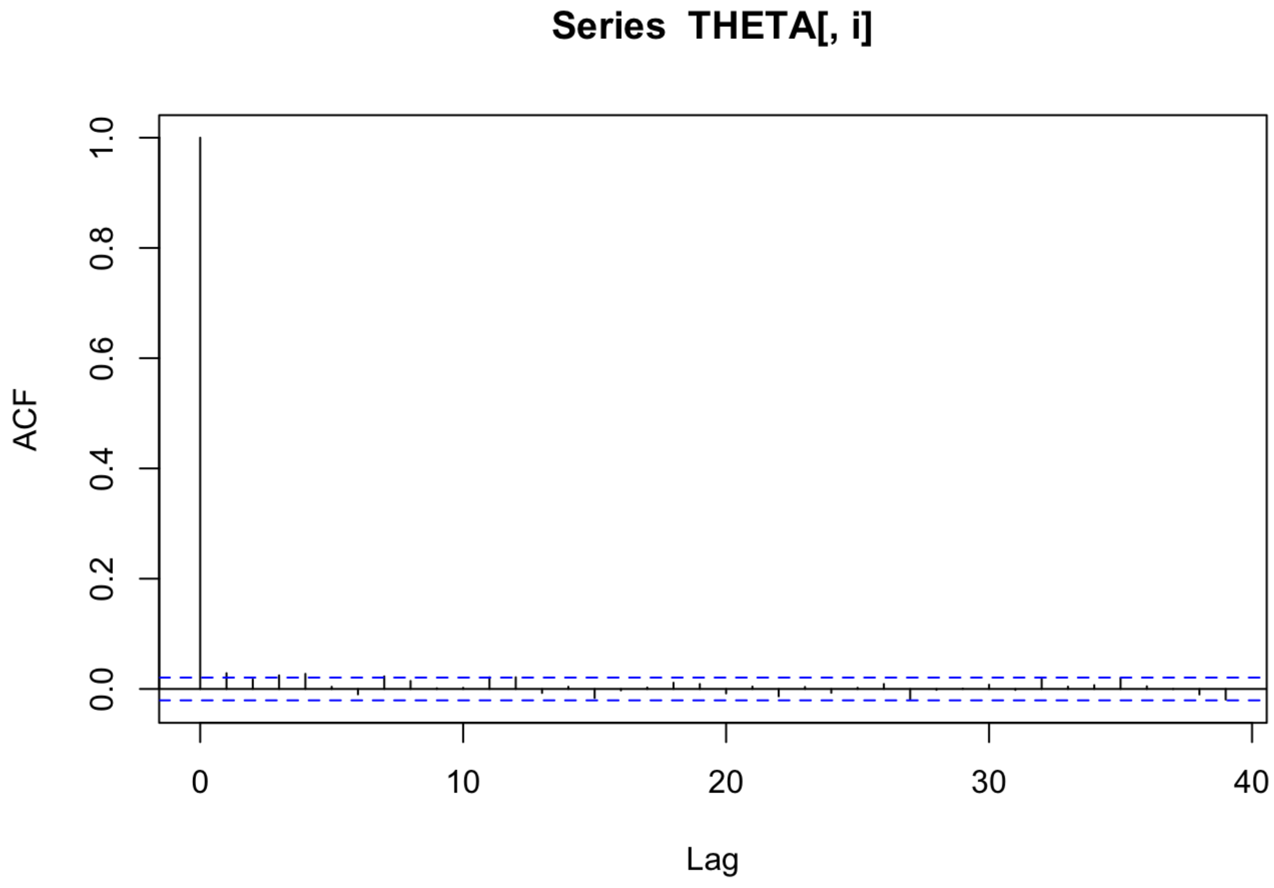
```
## acceptance rate: 0.4272
```

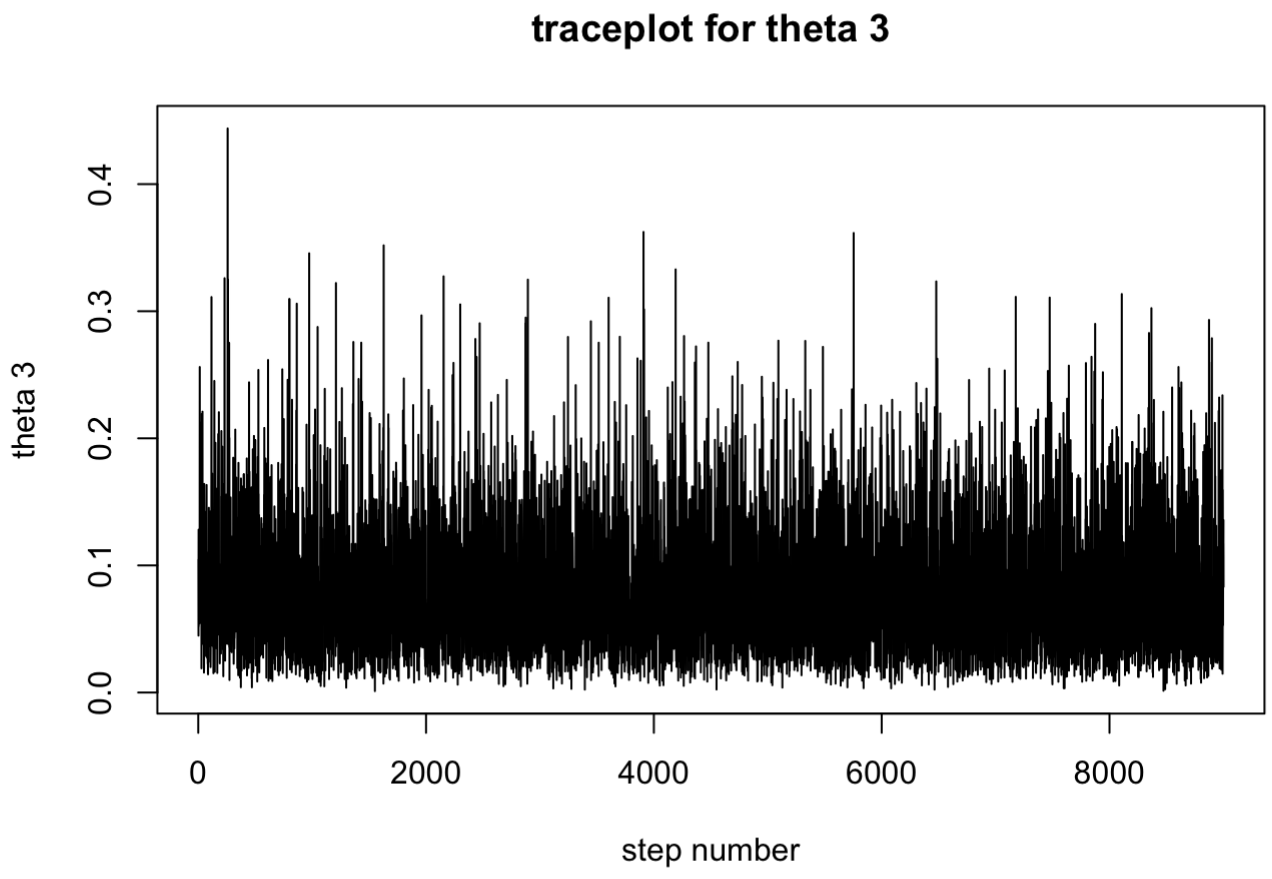
```
## diagnosis
for (i in 1:6) {
  plot(THETA[,i], type="l", xlab="step number", ylab=paste("theta", i), main = paste("tr
aceplot for theta", i))
  acf(THETA[,i])
}
```

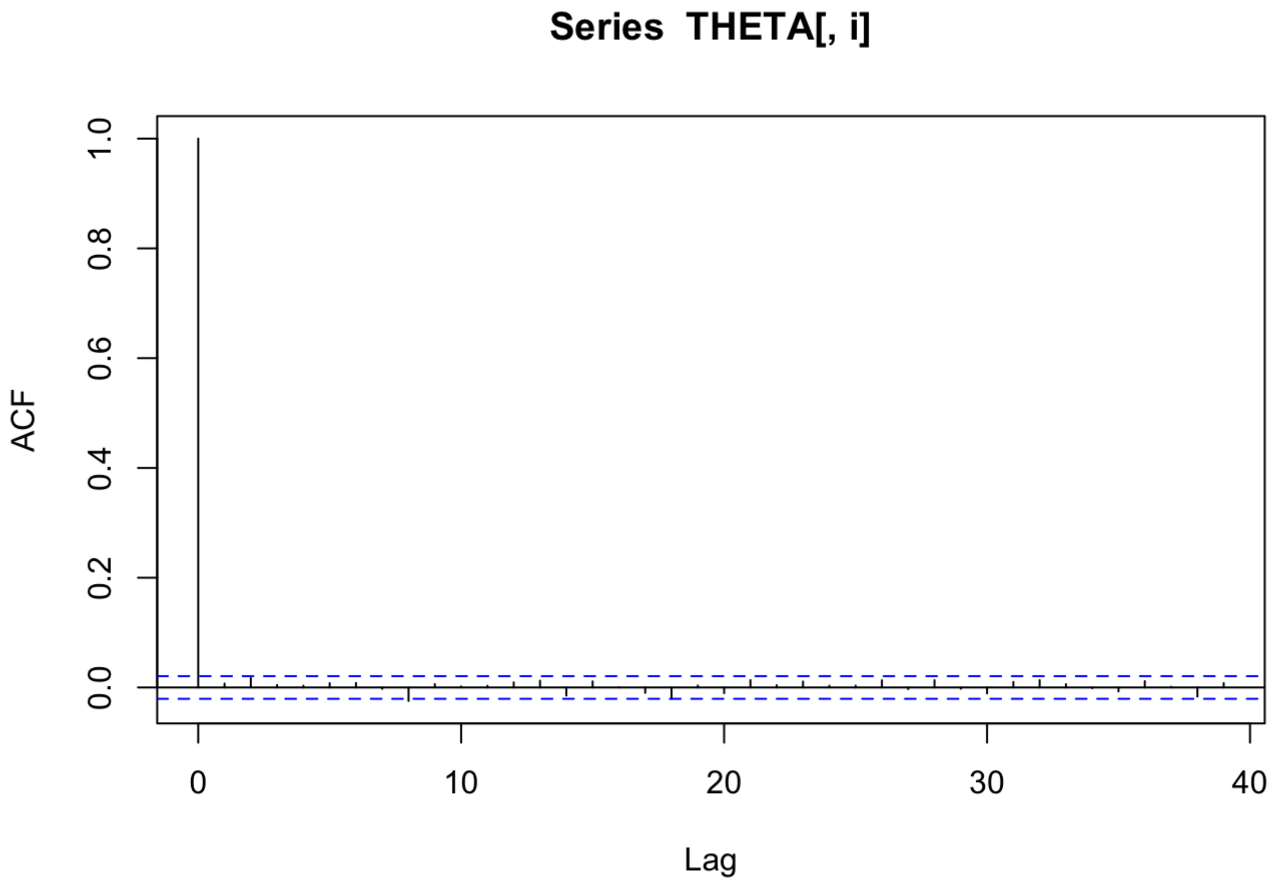

traceplot for theta 1
theta 1
0.00
0.05
0.10
0.15
0.20
0.25
0
2000
4000
6000
8000
step number

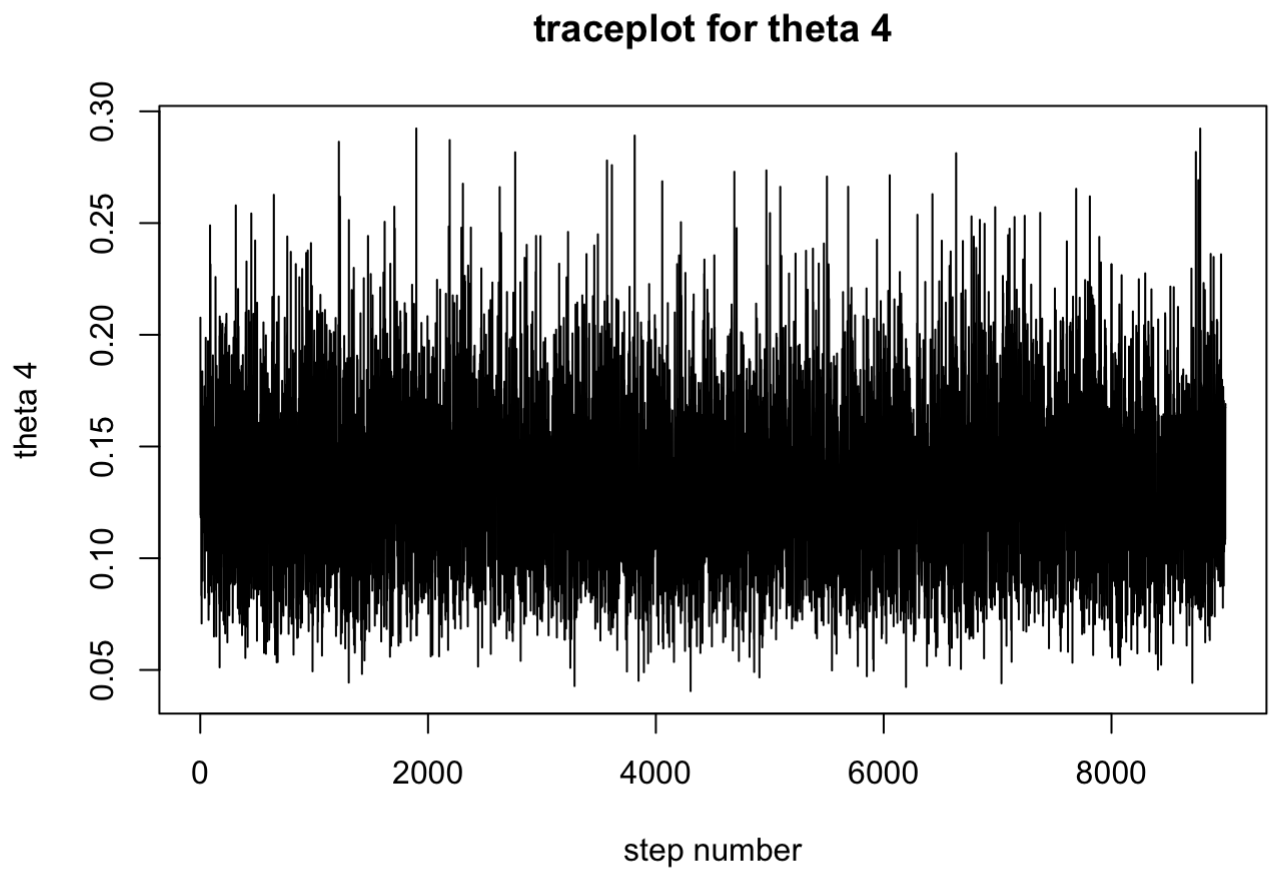

Series THETA[, i]
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40
Lag

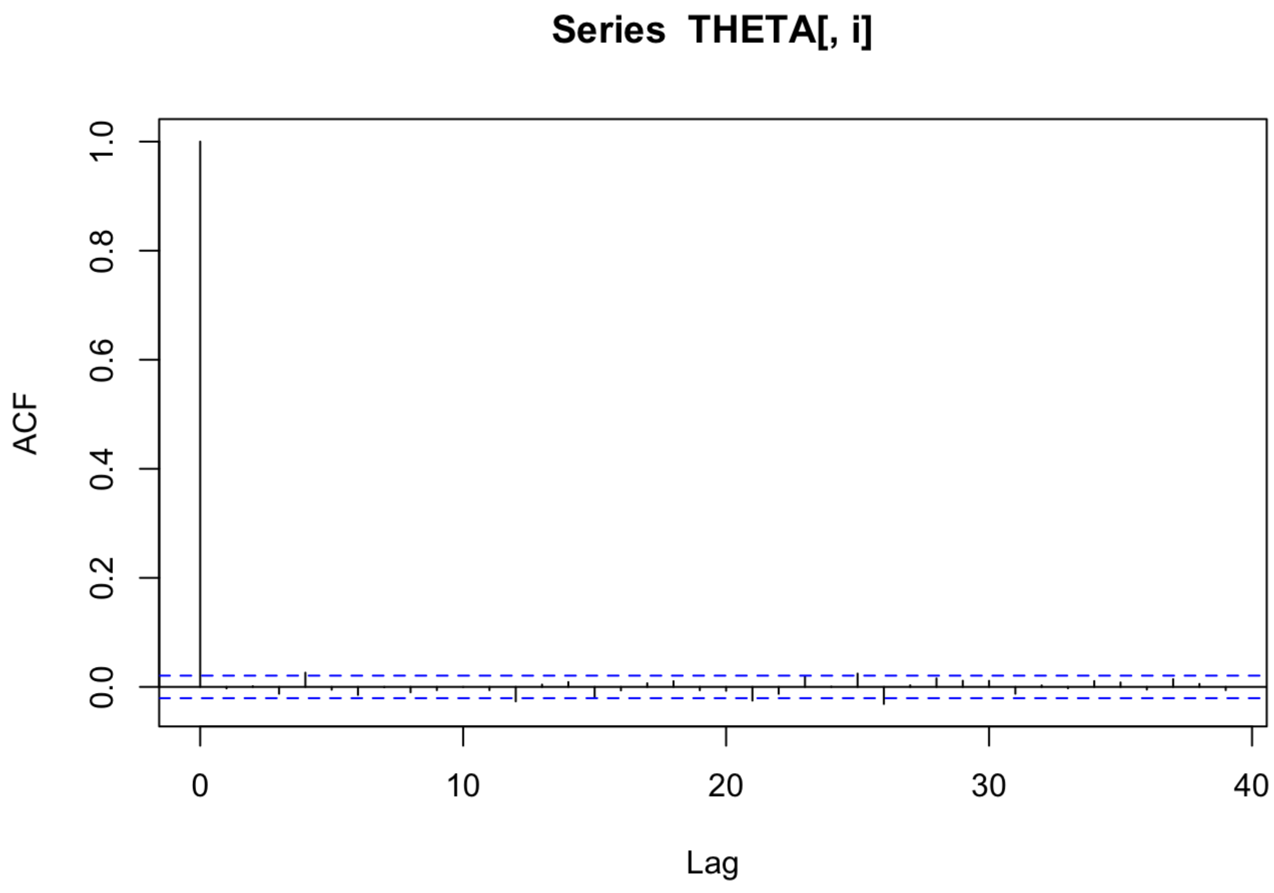

traceplot for theta 2
theta 2
0.0
0.2
0.4
0.6
0.8
0
2000
4000
6000
8000
step number

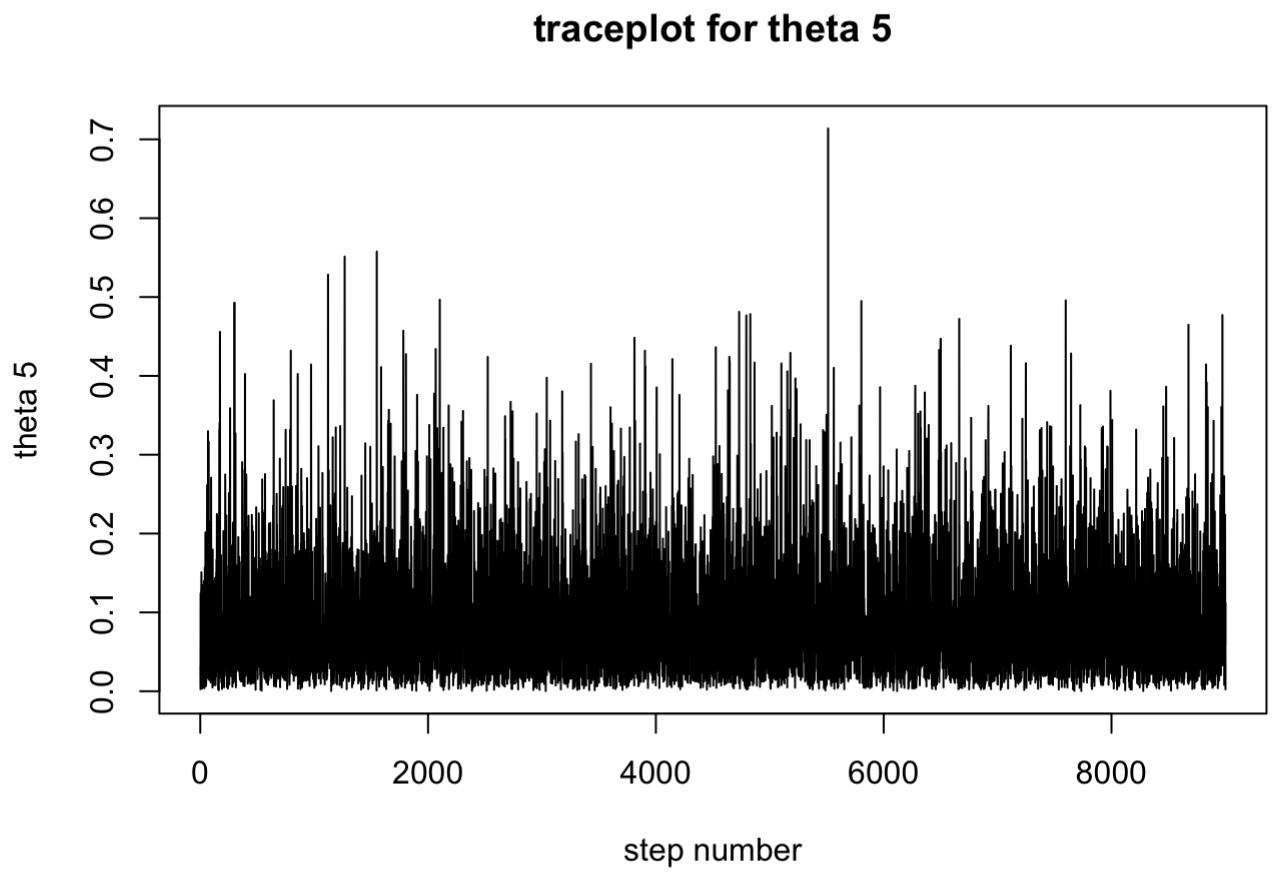

Series THETA[, i]
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40
Lag

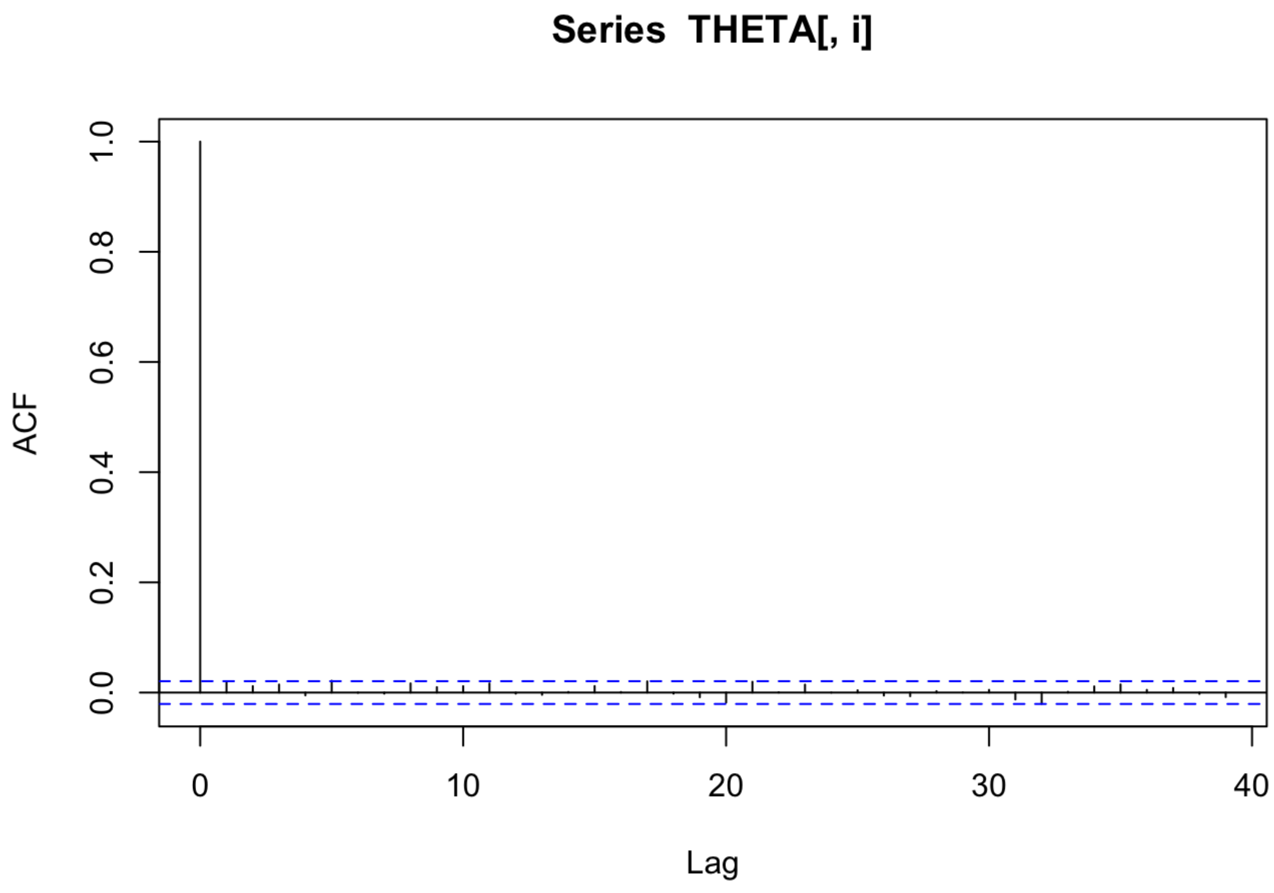

traceplot for theta 3
theta 3
step number
0.0
0.1
0.2
0.3
0.4
0
2000
4000
6000
8000

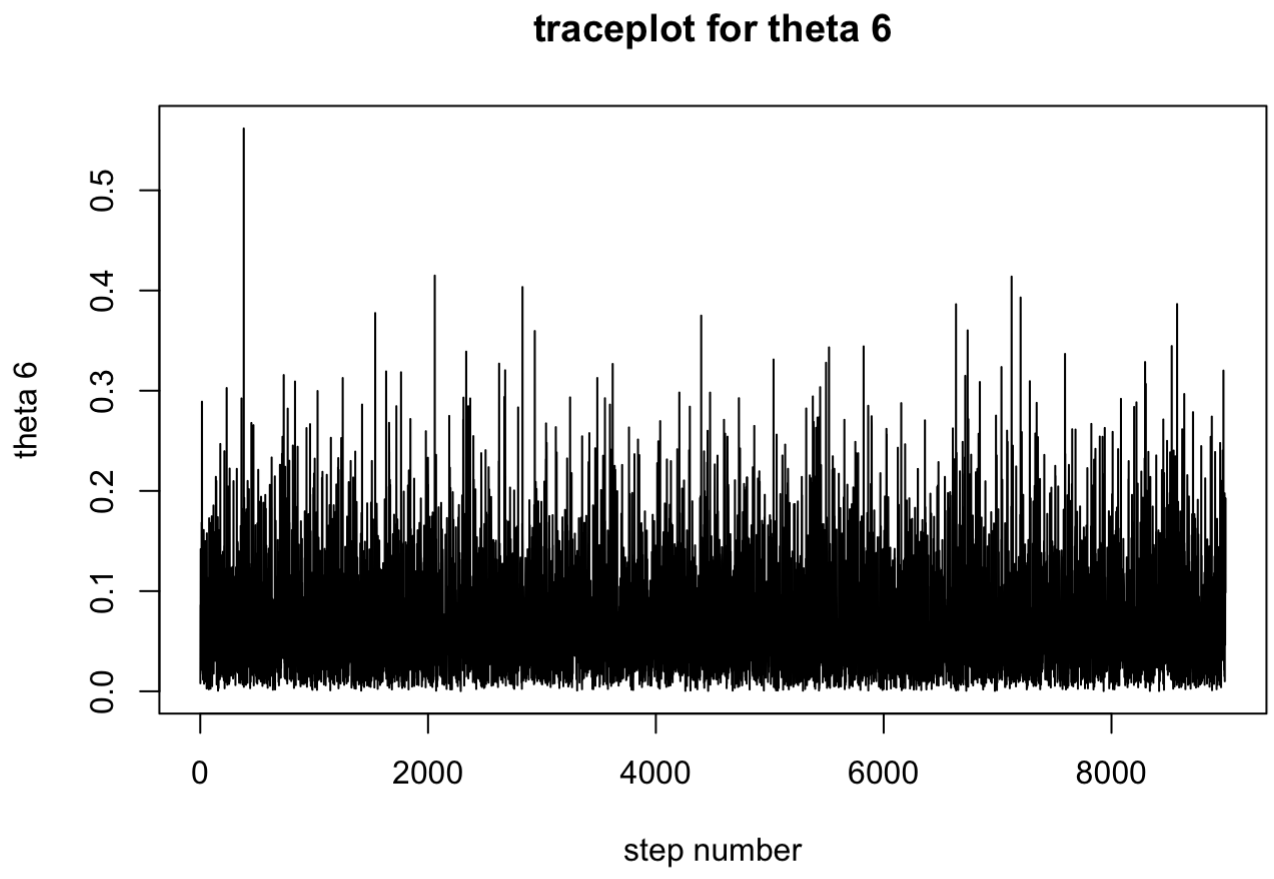

Series THETA[, i]
ACF
Lag
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40

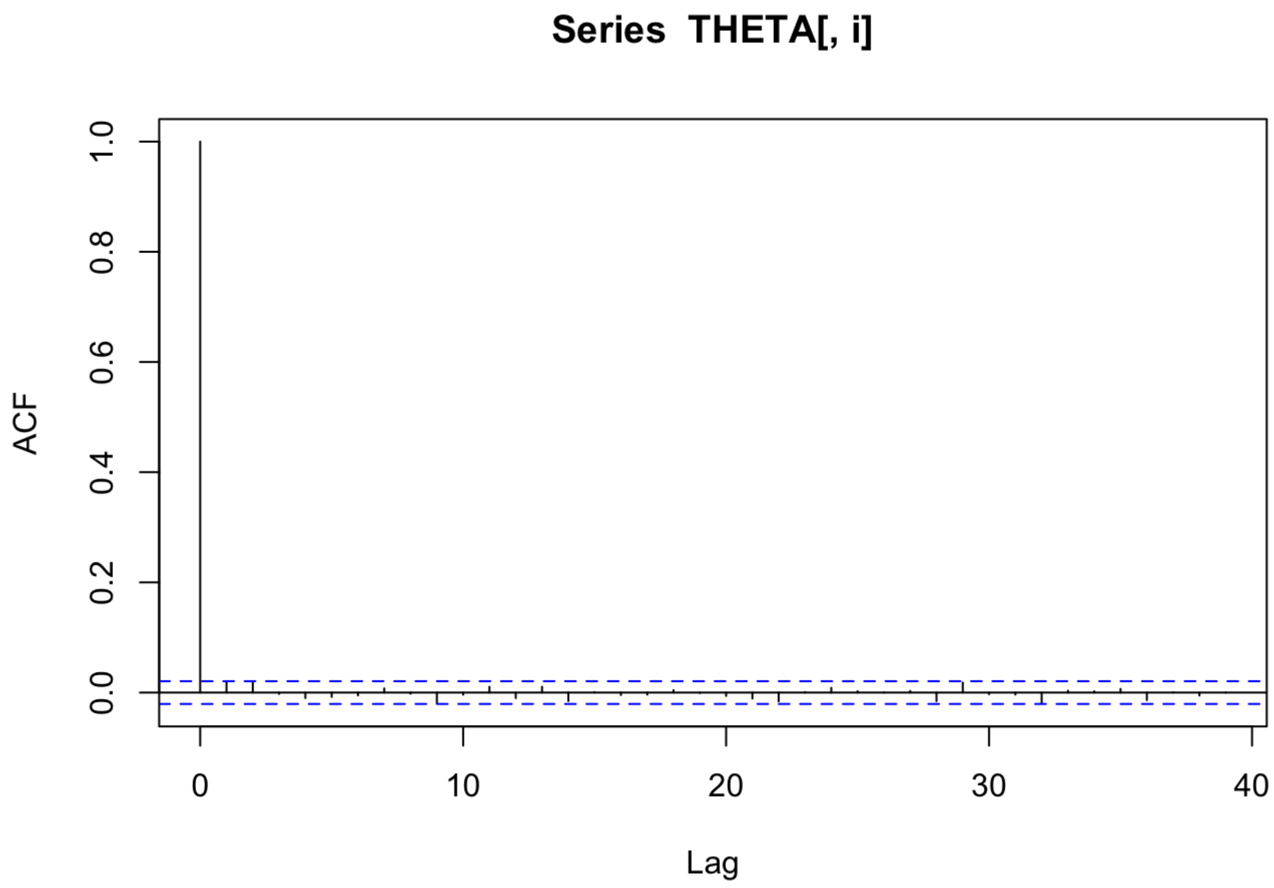

traceplot for theta 4
theta 4
0.05
0.10
0.15
0.20
0.25
0.30
0
2000
4000
6000
8000
step number


Series THETA[, i]
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40
Lag


traceplot for theta 5
theta 5
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0
2000
4000
6000
8000
step number


Series THETA[, i]
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40
Lag

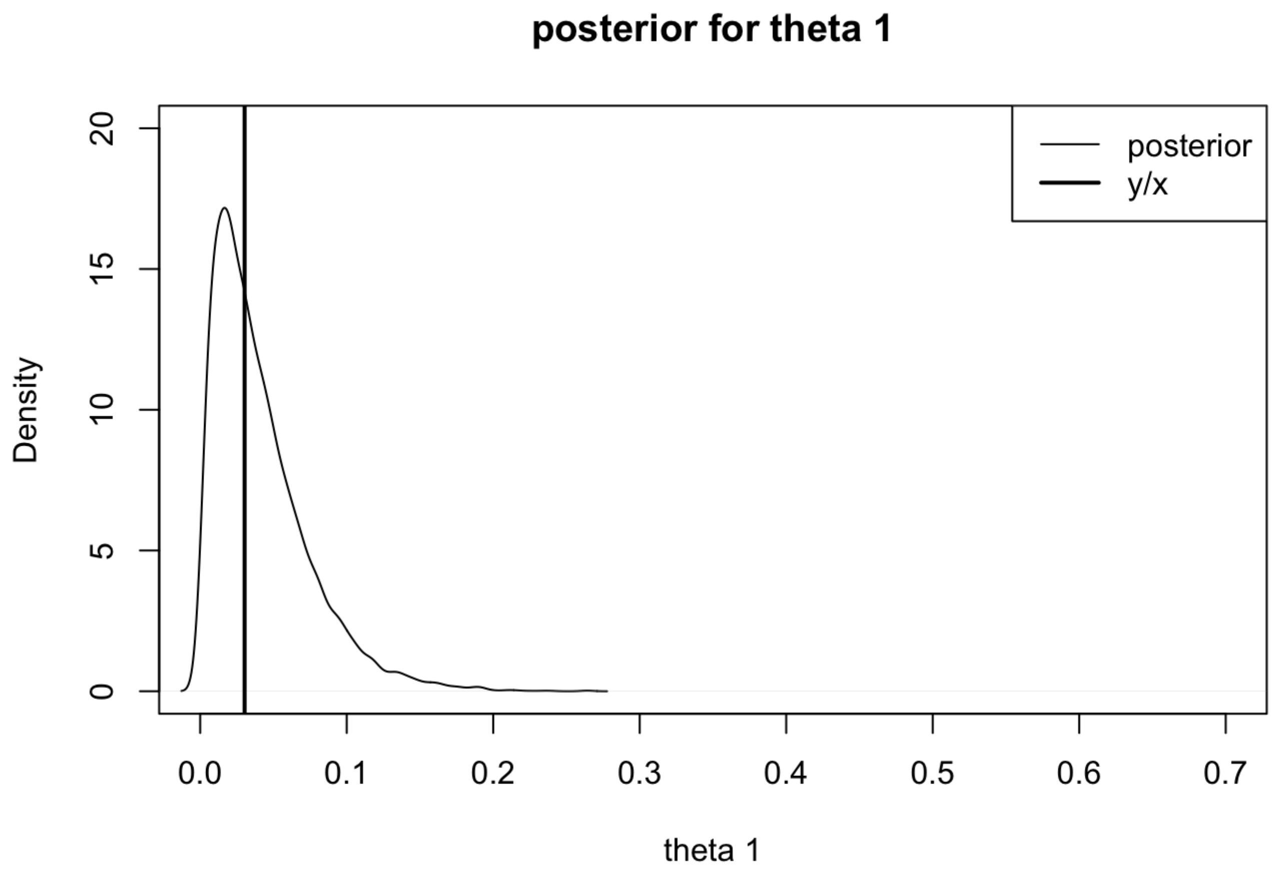

traceplot for theta 6
theta 6
0.0
0.1
0.2
0.3
0.4
0.5
0
2000
4000
6000
8000
step number

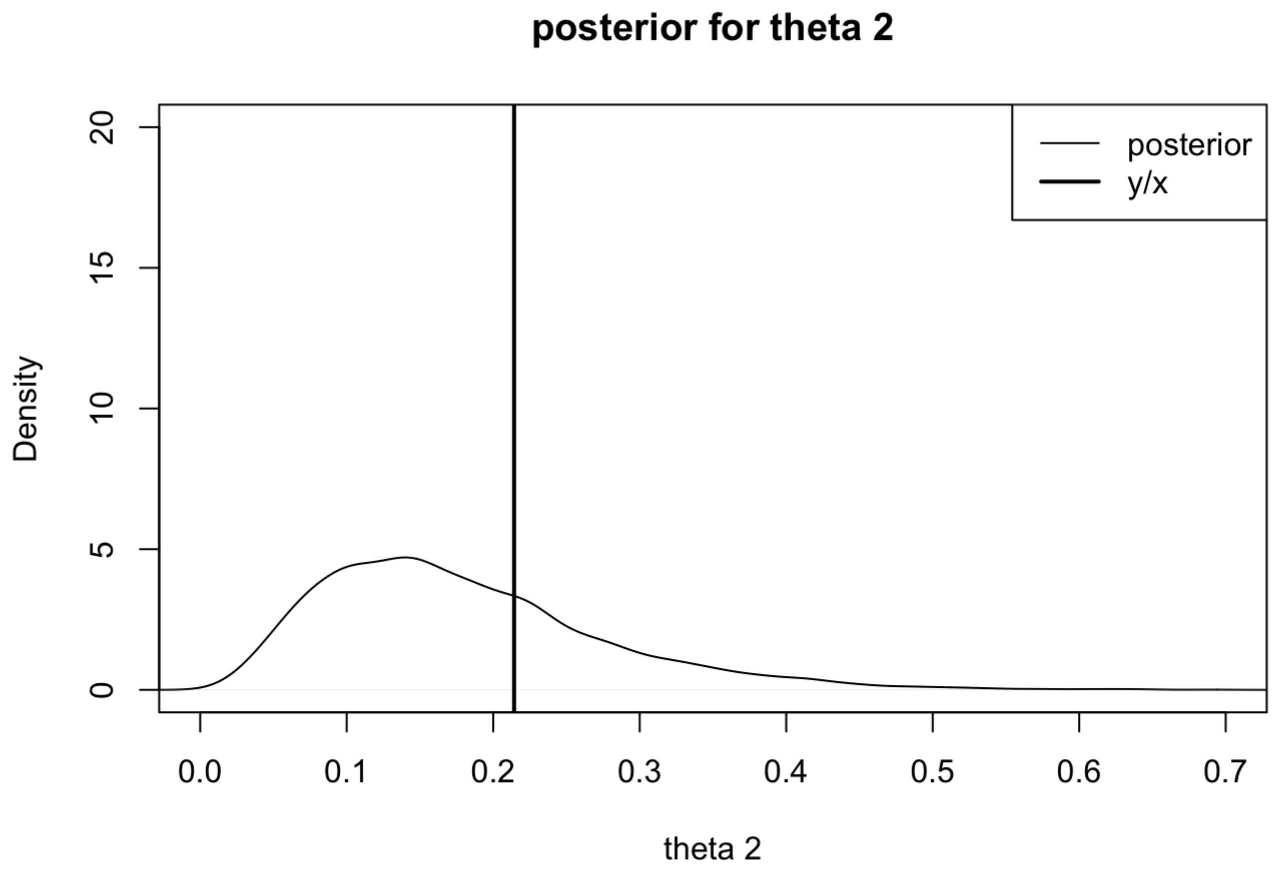

Series THETA[, i]
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40
Lag

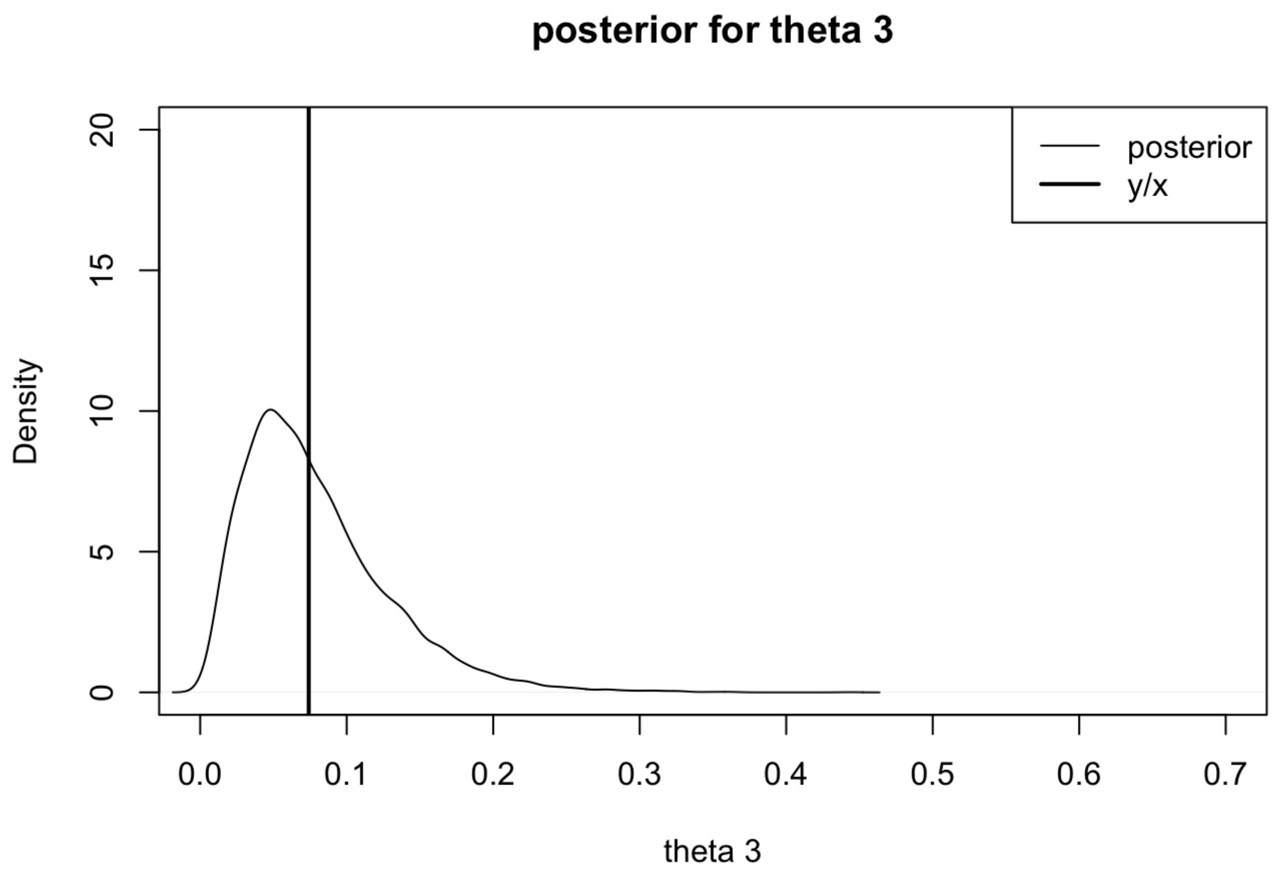
##

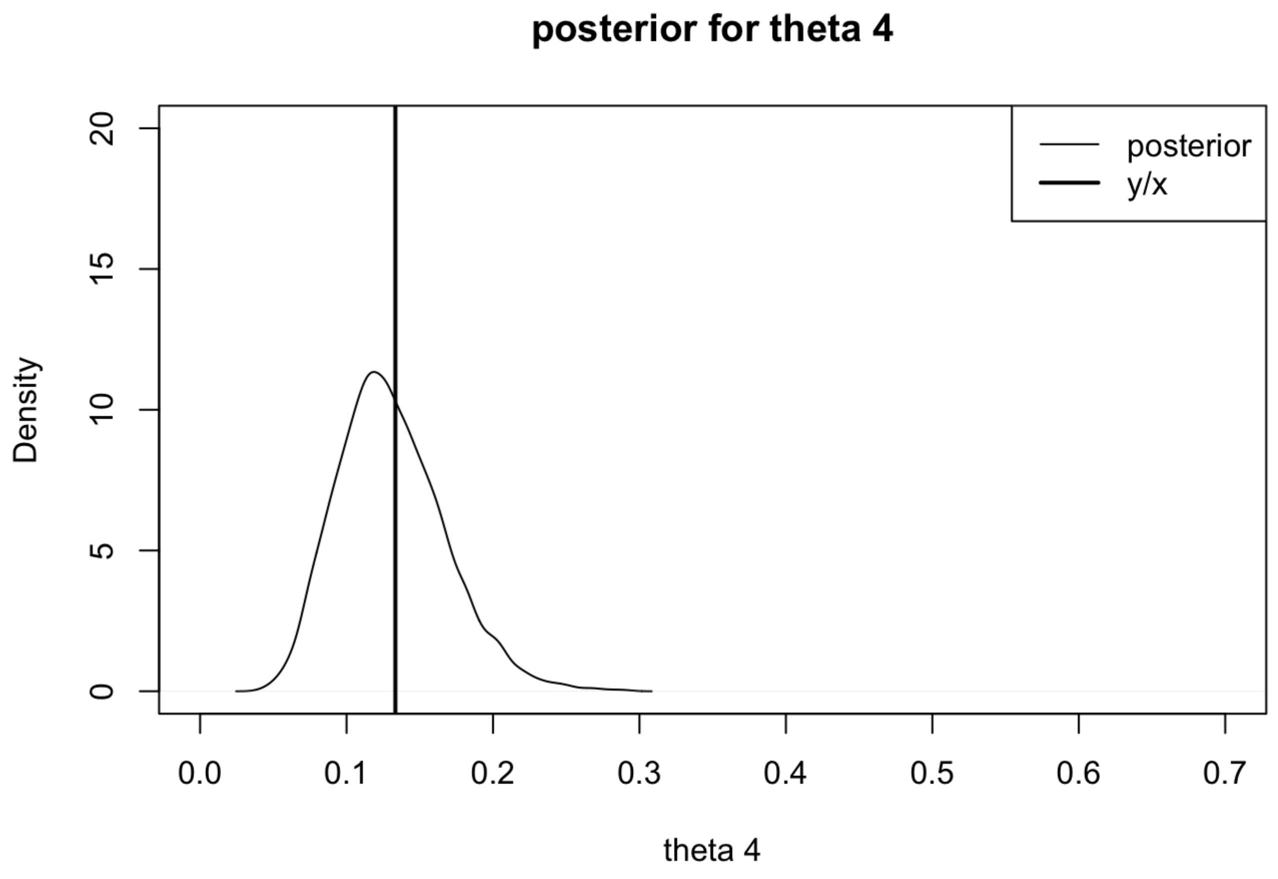
Problem 3 (c)

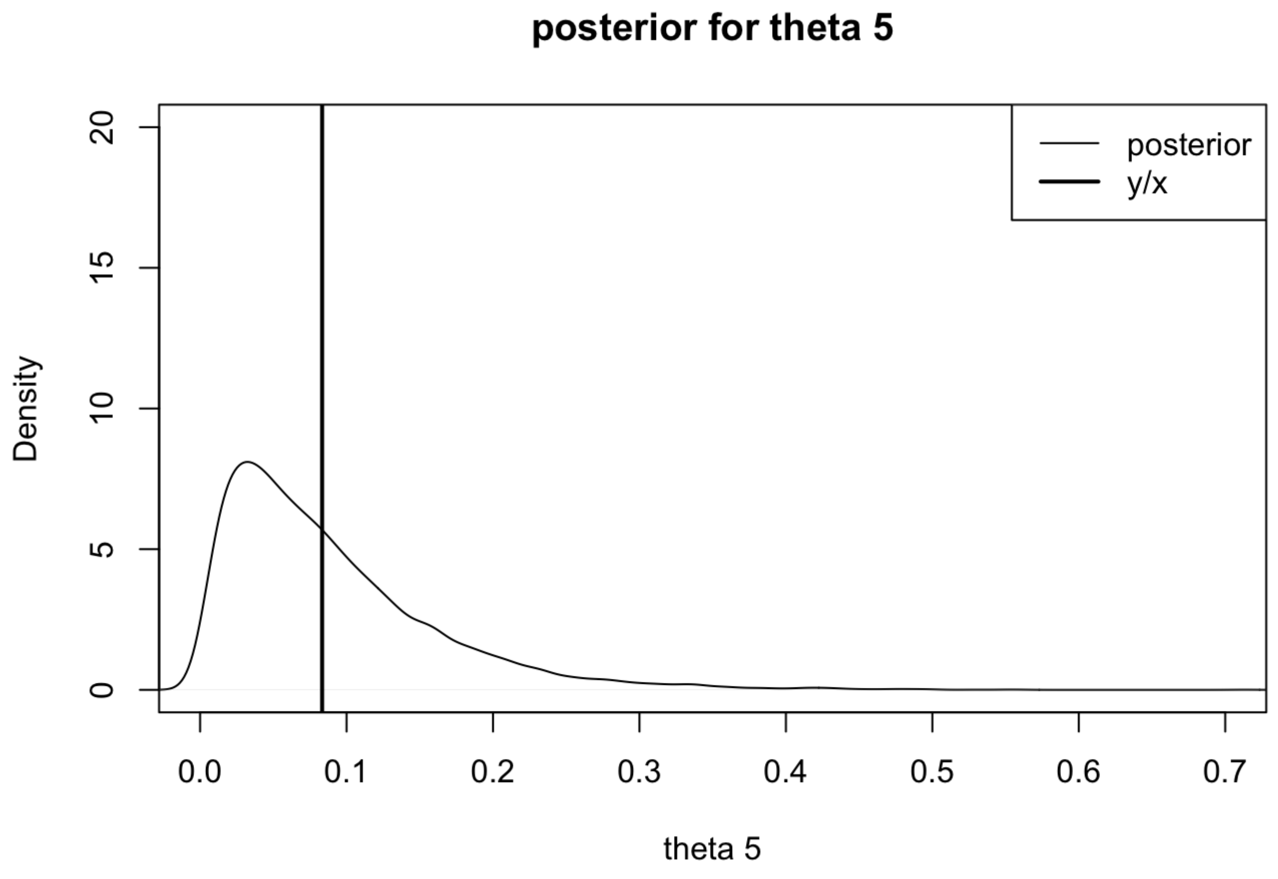
```
# (i)
for(i in 1:6) {
    plot(density(THETA[,i]), xlab=paste("theta",i), main=paste("posterior for theta",i),
xlim=c(0,0.7), ylim=c(0,20))
    abline(v=y[i]/x[i],lwd=2)
    legend("topright", c("posterior", "y/x"), lwd = c(1, 2))
}
```

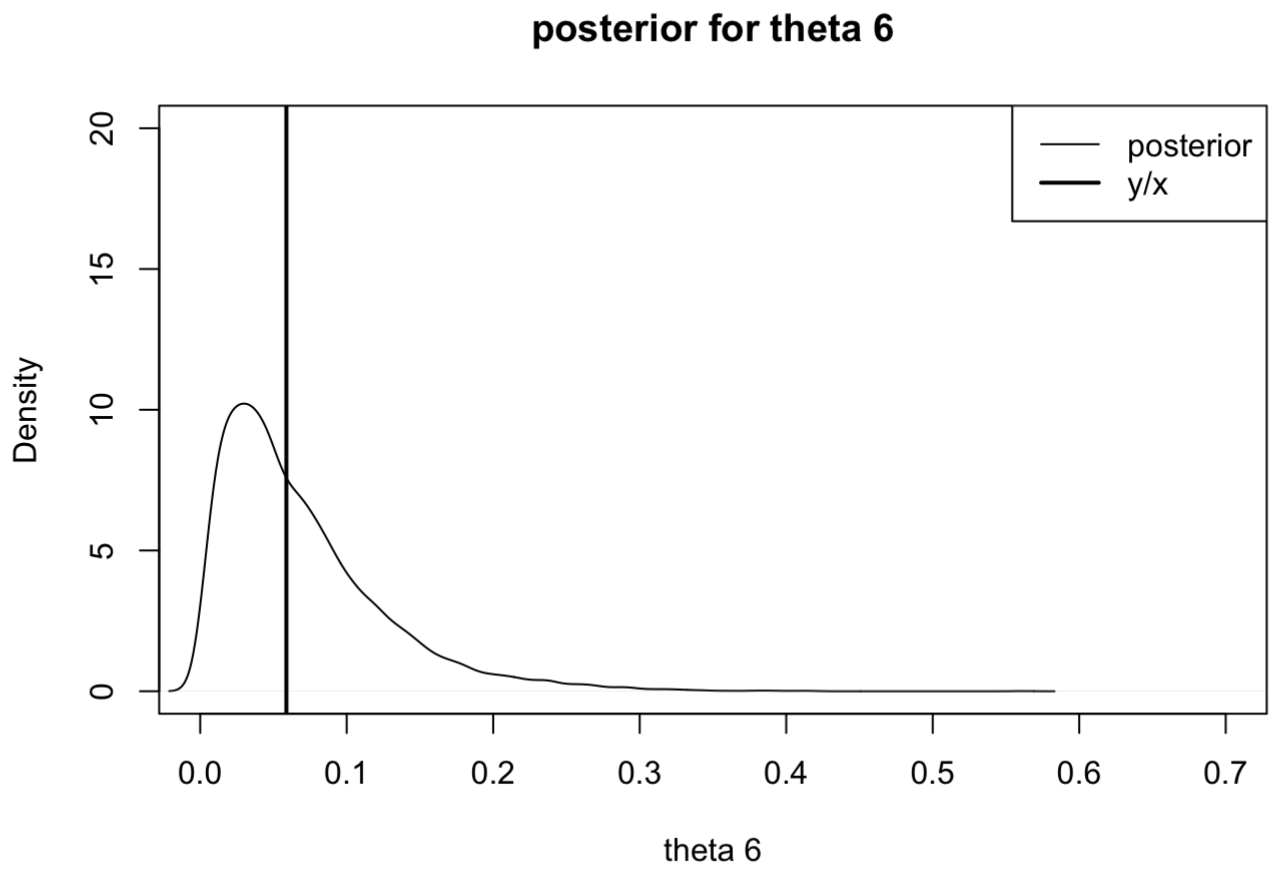

posterior for theta 1
Density
theta 1
posterior
y/x
0
5
10
15
20
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7


posterior for theta 2
Density
theta 2
posterior
y/x
0
5
10
15
20
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7

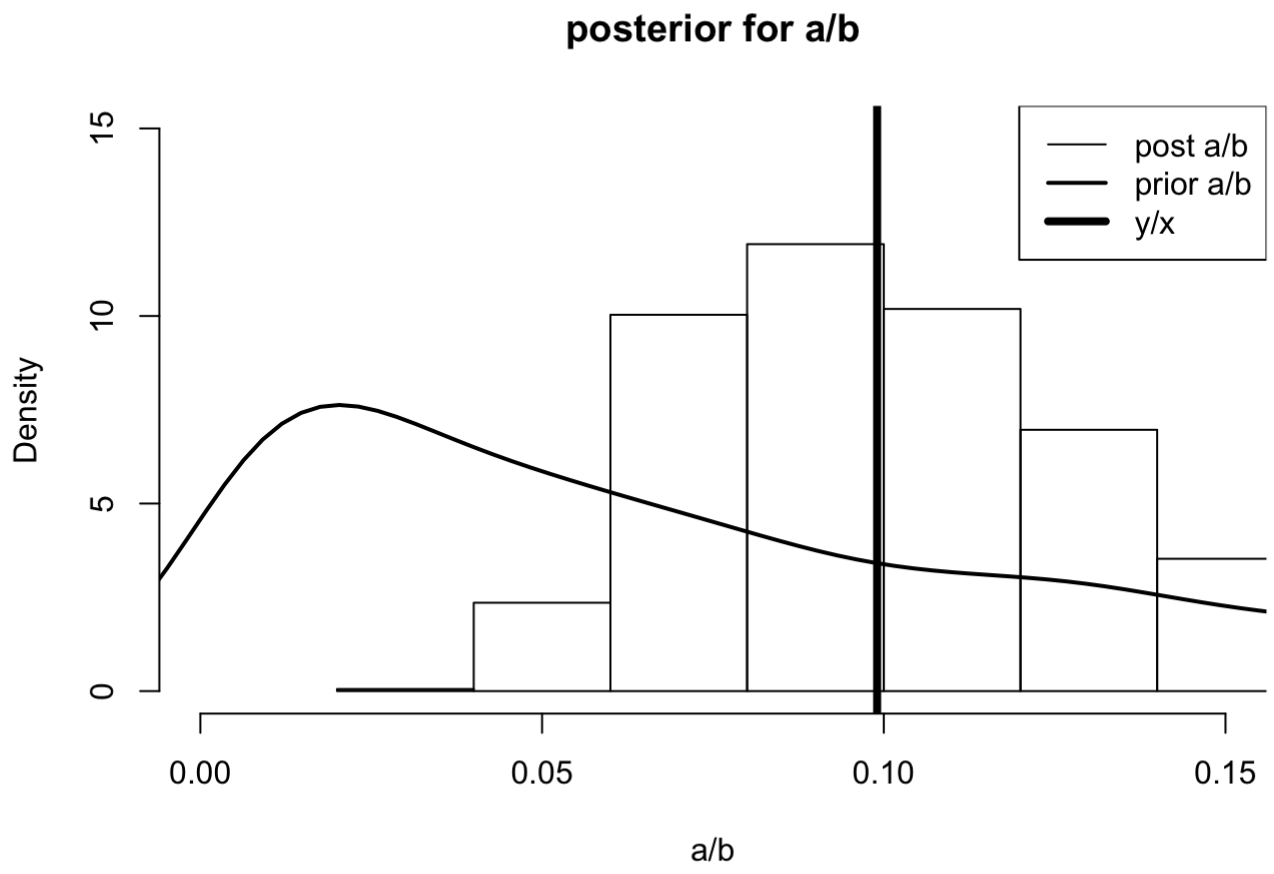

posterior for theta 3
Density
theta 3
posterior
y/x
0
5
10
15
20
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7


posterior for theta 4
Density
theta 4
posterior
y/x
0
5
10
15
20
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7

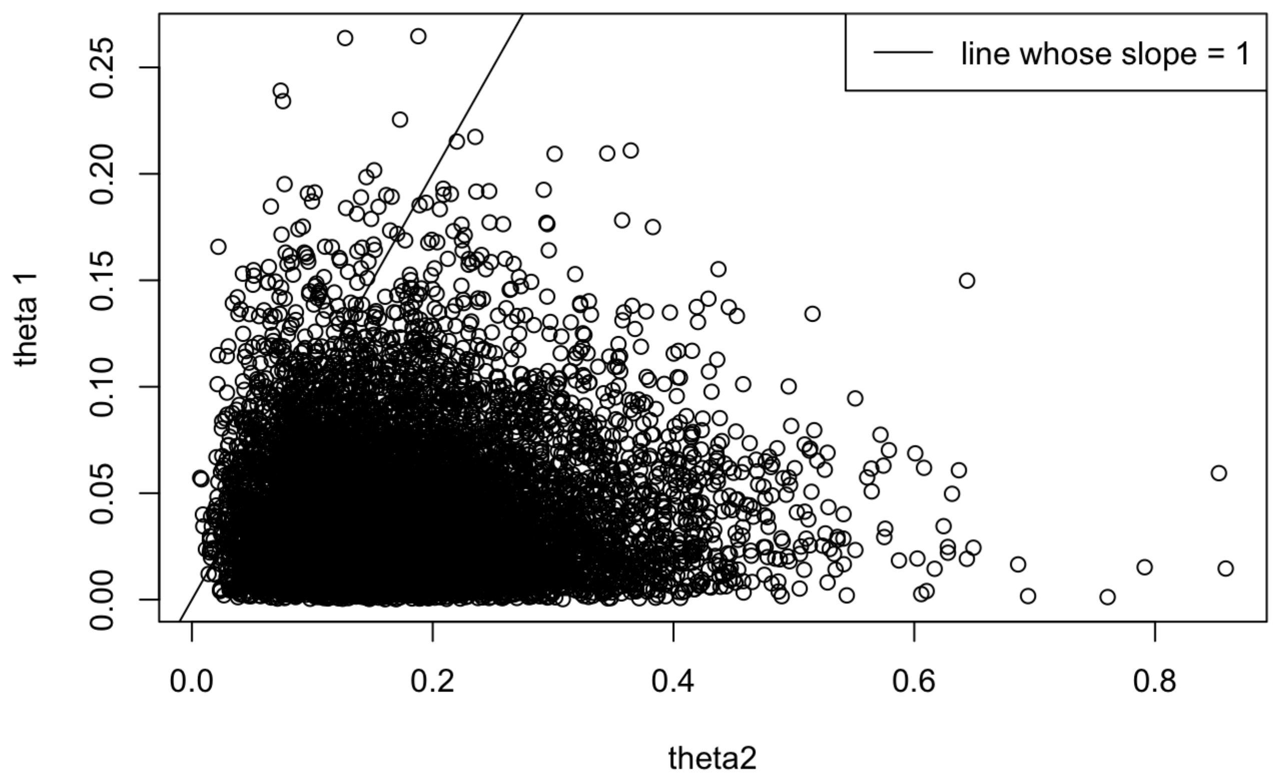

posterior for theta 5
Density
theta 5
posterior
y/x
0
5
10
15
20
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7

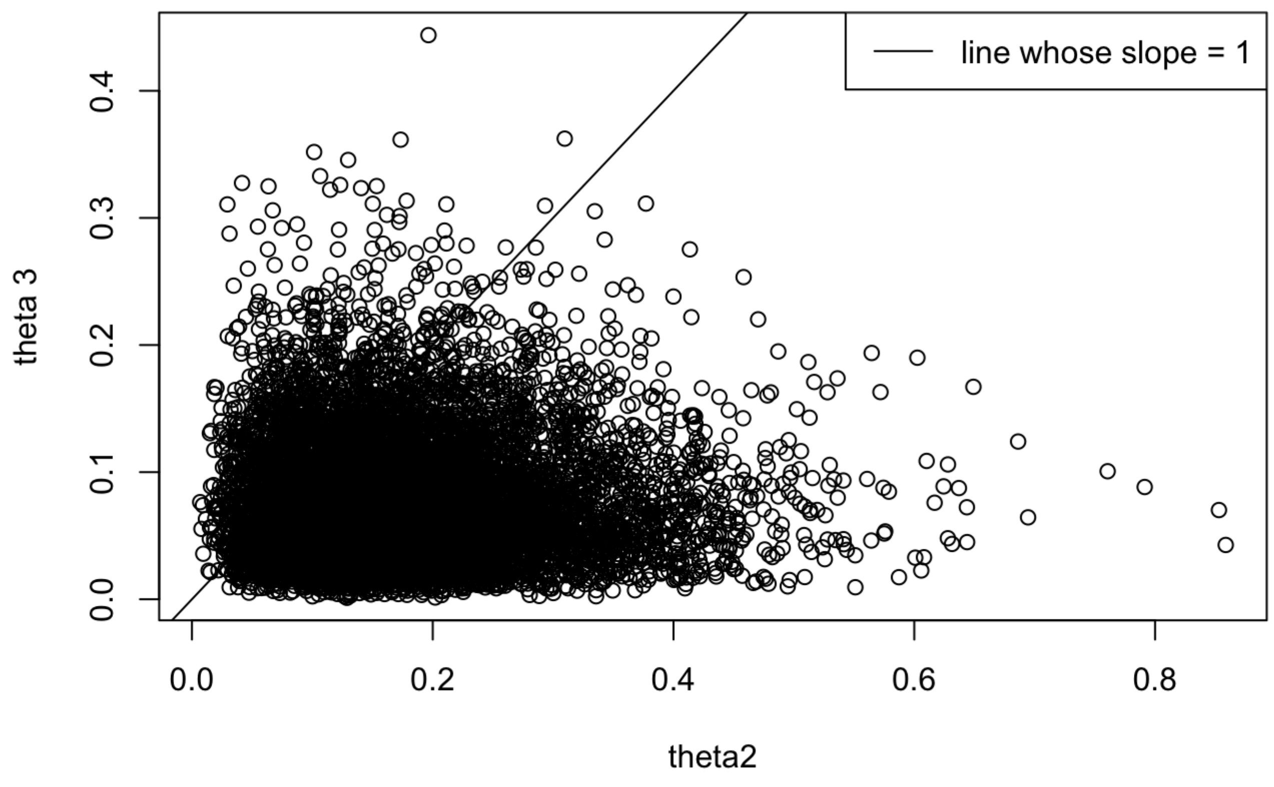


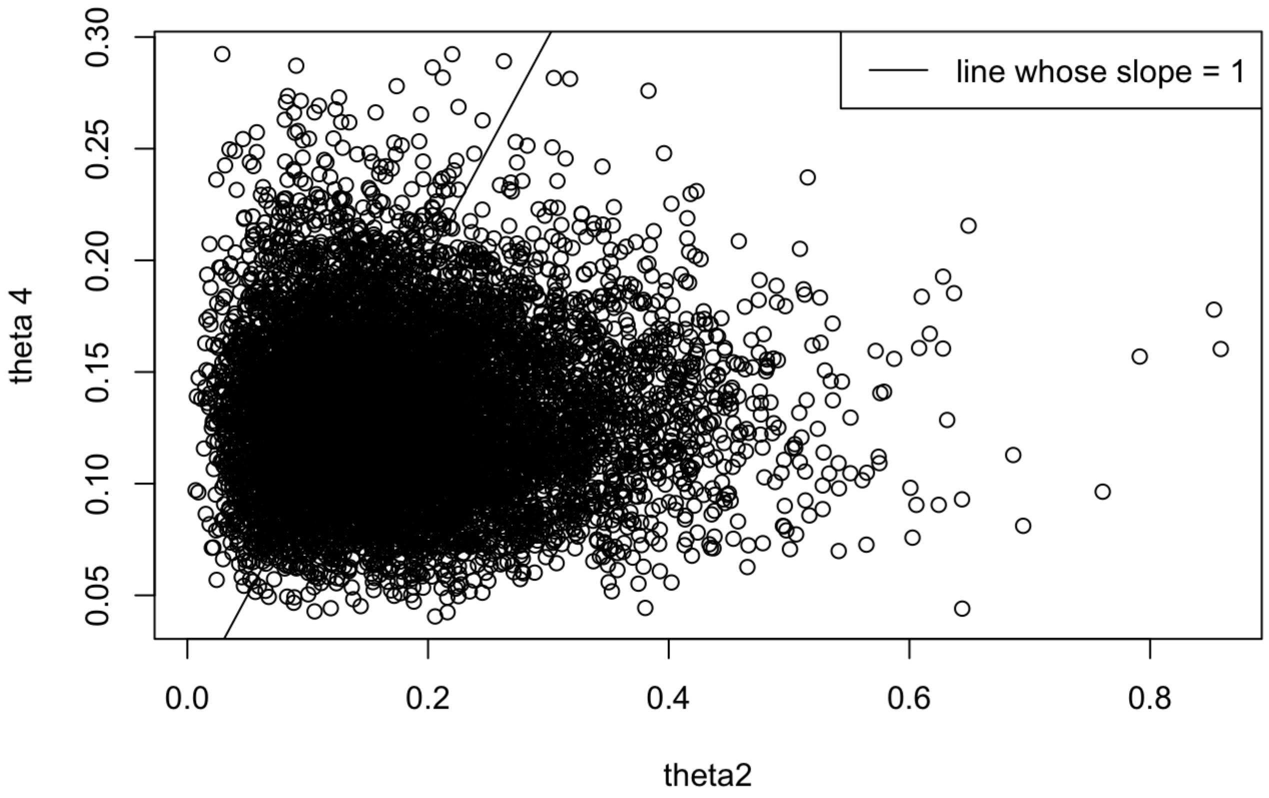
```
# (ii)
prior_a <- rgamma(9000, 1, 1)
prior_b <- rgamma(9000, 10, 1)
hist(ALPHA/BETA,xlab="a/b", freq=FALSE, main = "posterior for a/b", ylim = c(0,15), xlim
= c(0, 0.15))
lines( density(prior_a/prior_b), lwd=2)
abline(v=mean(y/x),lwd=4)
legend("topright", c("post a/b", "prior a/b", "y/x"), lwd = c(1, 2, 4))
```

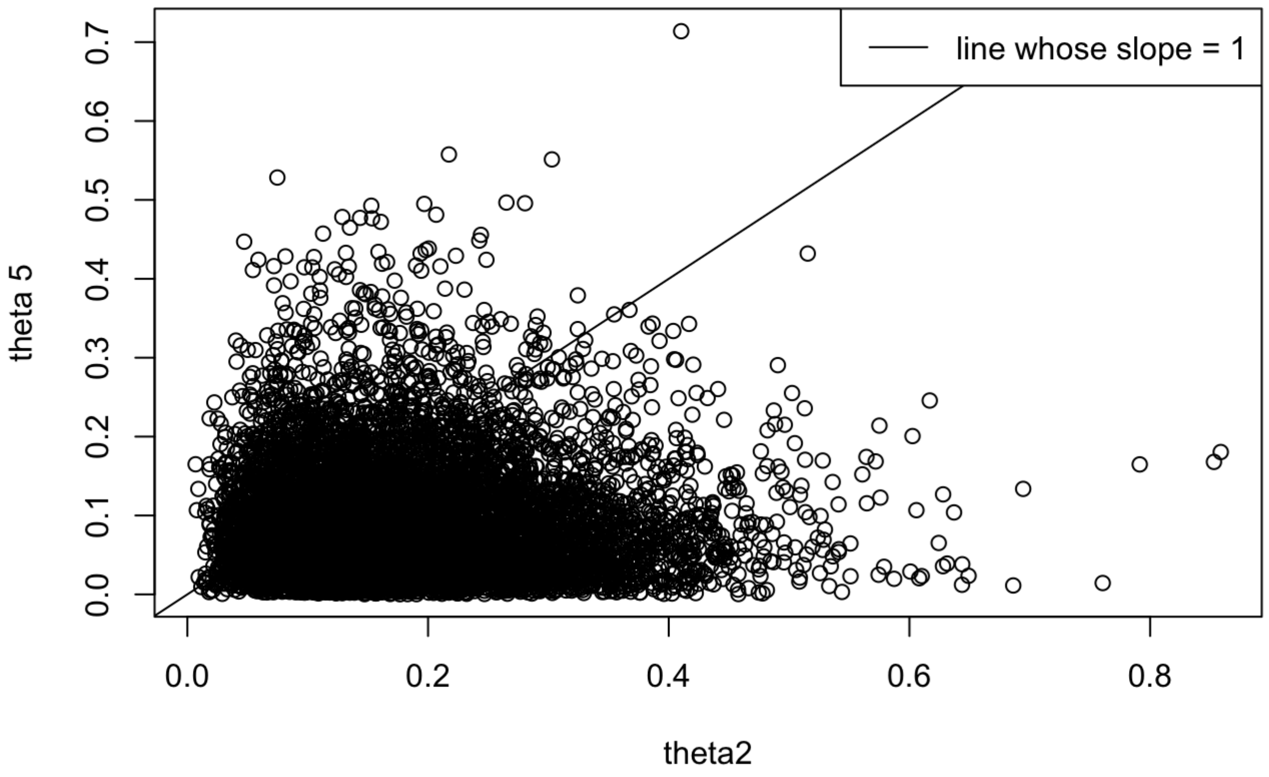


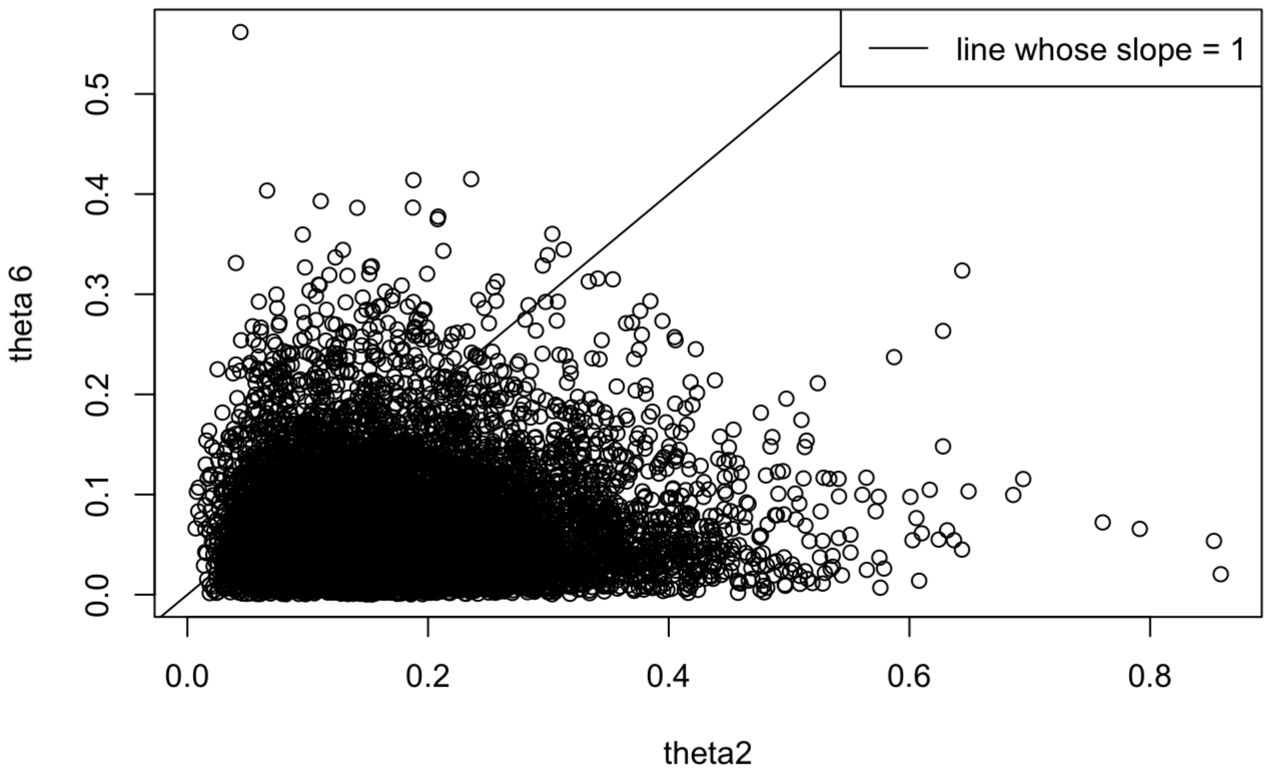
```
# (iii)
for(i in c(1,3,4,5,6))
  {
    plot(THETA[,2], THETA[,i], type="p", xlab="theta2", ylab=paste("theta", i))
    abline(0,1,lwd=1)
    legend("topright", c("line whose slope = 1"), lwd = 1)
}
```


line whose slope = 1
theta 1
theta2
0.00
0.05
0.10
0.15
0.20
0.25
0.0
0.2
0.4
0.6
0.8


line whose slope = 1
theta 3
theta2
0.0
0.1
0.2
0.3
0.4
0.0
0.2
0.4
0.6
0.8


line whose slope = 1
theta 4
theta2
0.05
0.10
0.15
0.20
0.25
0.30
0.0
0.2
0.4
0.6
0.8


line whose slope = 1
theta 5
theta2
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.0
0.2
0.4
0.6
0.8


line whose slope = 1
theta 6
theta2
0.0
0.1
0.2
0.3
0.4
0.5
0.0
0.2
0.4
0.6
0.8

```
p_21 <- mean( THETA[,2] > THETA[,1] )
p_23 <- mean( THETA[,2] > THETA[,3] )
p_24 <- mean( THETA[,2] > THETA[,4] )
p_25 <- mean( THETA[,2] > THETA[,5] )
p_26 <- mean( THETA[,2] > THETA[,6] )
cat("probability that theta2 > theta 1 is:", p_21, "\n")
```

```
## probability that theta2 > theta 1 is: 0.9506721
```

```
cat("probability that theta2 > theta 3 is:", p_23, "\n")
```

```
## probability that theta2 > theta 3 is: 0.844795
```

```
cat("probability that theta2 > theta 4 is:", p_24, "\n")
```

```
## probability that theta2 > theta 4 is: 0.6449283
```

```
cat("probability that theta2 > theta 5 is:", p_25, "\n")
```

```
## probability that theta2 > theta 5 is: 0.8016887
```

```
cat("probability that theta2 > theta 6 is:", p_26, "\n")
```

```
## probability that theta2 > theta 6 is: 0.8652372
```

```
max <- apply(THETA[,],1,max)
p_2largest <- mean(THETA[,2] == max)
cat("probability that theta2 is the largest is:", p_2largest, "\n")
```

```
## probability that theta2 is the largest is: 0.5360515
```

```
cat("yi/xi is:", y/x)
```

```
## yi/xi is: 0.03030303 0.2142857 0.07407407 0.1333333 0.08333333 0.05882353
```

The probability that theta2 is larger than each of the other counties tells us that ingeneral, county2 has a possibility of 83% or more to have higher nongenetic birth defect rate than another single county (except that county2 has a possibility of 55% or more to have more birth defects than county 5). The probability that theta2 is the largest among all theta's tells us that county2 has around 50% chance to have the highest nongenetic birth defect rate among all counties. If we just examine yi/xi for each county i, since yi/xi can't involve posterior distribution, then we would easily think that county2 definitely has the highest defect birth rate, but this possibility is actually just around 50%.