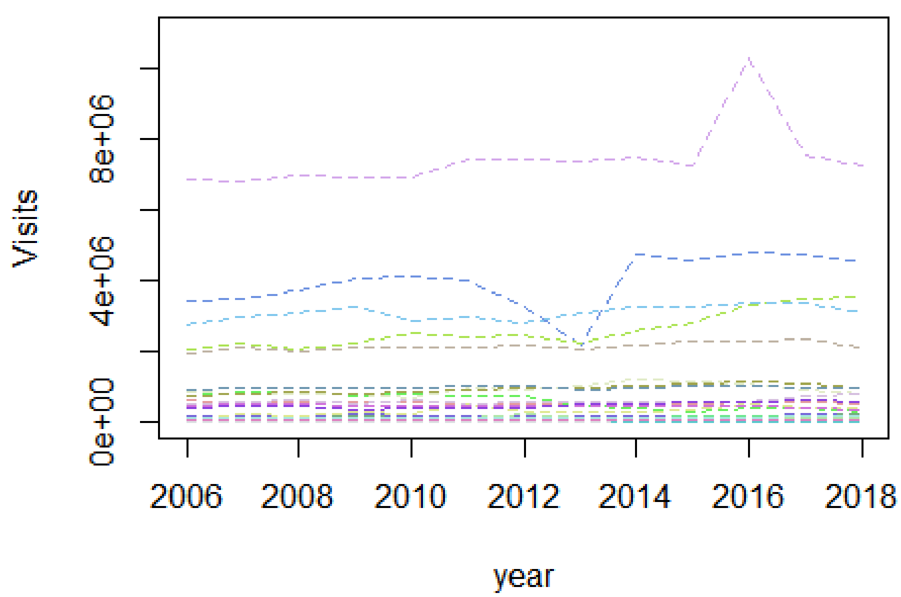
# STAT578-HW5

## a)(i)

```
library(randomcoloR)

df <- read.csv("C:\\temp\\usparkvisits.csv",header = TRUE)
num_parks <- nrow(df)
num_years <- ncol(df)

palette <- distinctColorPalette(num_parks)
year <- 2006:2018
plot(year,(df[1,2:num_years]),ylim = c(0, 11000000),col = palette[0],ylab =
"Visits",lty =2)
for (i in 1:num_parks)
    lines(year,(df[i,2:num_years]),ylim = c(0, 11000000),col = palette[i],lty
=2)
```

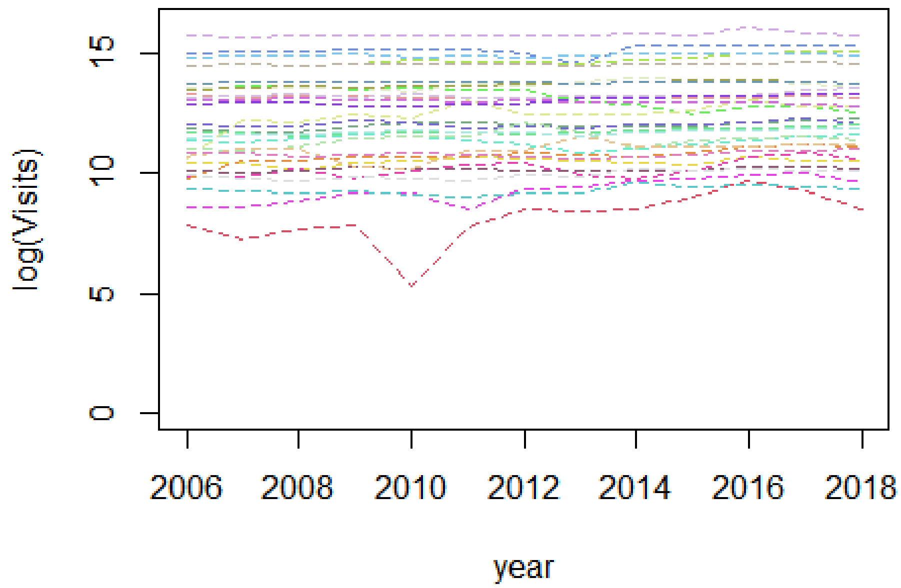



## a)(ii)

```
plot(year,log(df[1,2:num_years]),ylim = c(0, log(11000000)),col = palette[0],
ylab = "log(Visits)",lty=2)

for (i in 1:num_parks)
```

```
    lines(year,log(df[i,2:num_years]),ylim = c(0, log(11000000)),col =
palette[i],lty =2)
```



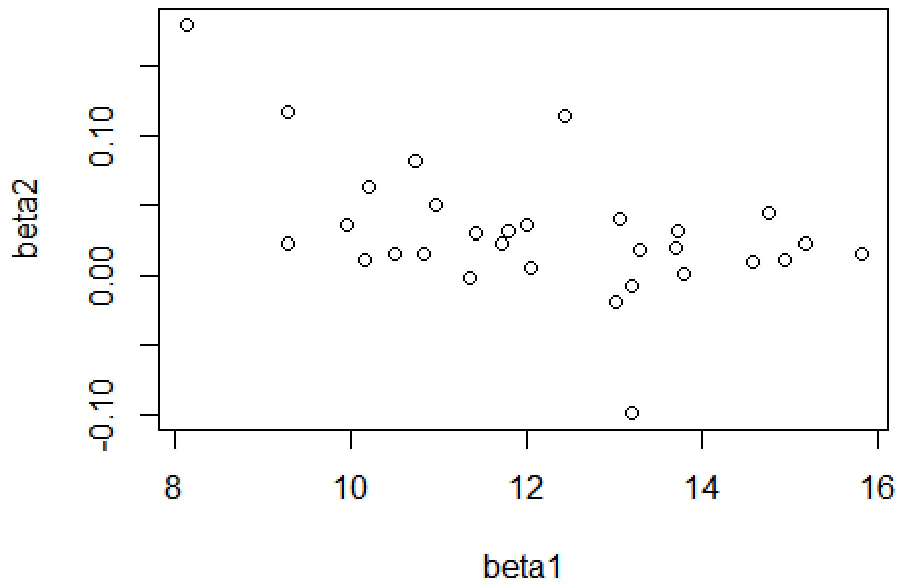

## b)(i)

```
df1 <- read.csv("C:\\temp\\usparks.csv",header = TRUE)
park_names = colnames(df1)[3:ncol(df1)]

# Center the year.
df1$Year <- df1$Year - mean(df1$Year)

beta1 <- c()
beta2 <- c()
beta_all <-matrix(NA, ncol(df1), 2)
for (i in 1:num_parks)
{
  mod = lm(
    as.formula(paste("log(", park_names[i],")", "~","Year",sep = "" )),
    data=df1)
  beta <- mod$coefficients
  beta_all[i,] <- beta
  beta1 <-  c(beta1,beta[1])
  beta2 <-  c(beta2,beta[2])
}
```

```r
plot(beta1,beta2)
```




## b)(ii)

```r
mean(beta1)

## [1] 12.17658

mean(beta2)

## [1] 0.03062706
```

## b)(iii)

```r
var(beta1)

## [1] 3.791483

var(beta2)

## [1] 0.002286344
```

## b)(iv)

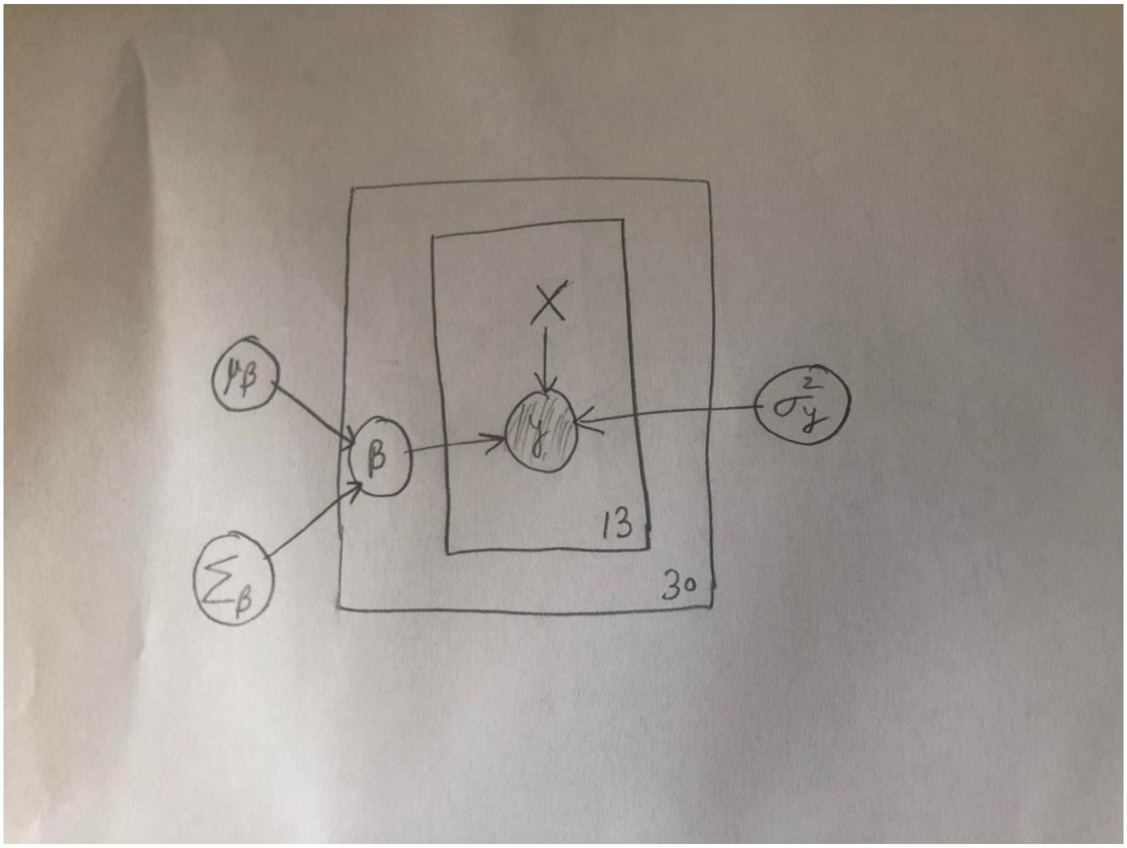
```r
cor(beta1,beta2)

## [1] -0.4762208
```

## ##c(i)

*JAGS MODEL*

```
data {
dimY <- dim(visits)
yearcent <- year - mean(year)
}
model {
for (j in 1:dimY[1]) {
for (i in 1:dimY[2]) {
visits[j,i] ~ dnorm(beta[1,j] + beta[2,j]*yearcent[i], sigmasqyinv)
}
beta[1:2,j] ~ dmnorm(mubeta, Sigmabetainv)
}
mubeta ~ dmnorm(mubeta0, Sigmamubetainv)
Sigmabetainv ~ dwish(2*Sigma0, 2)
sigmasqyinv ~ dgamma(0.0001, 0.0001)
Sigmabeta <- inverse(Sigmabetainv)
rho <- Sigmabeta[1,2] / sqrt(Sigmabeta[1,1] * Sigmabeta[2,2])
sigmasqy <- 1/sigmasqyinv
}
```




```
d1 <- list(visits = log(df[,-1]),year = c(2006:2018),mubeta0 = c(0, 0),
Sigmamubetainv = rbind(c(0.000001, 0),c(0, 0.000001)),
Sigma0 = rbind(c(10, 0),c(0, 0.01)))

inits1 <- list(
list(sigmasqyinv = 10, mubeta = c(50, 50),Sigmabetainv = rbind(c(10000, 0),
c(0, 10000)),.RNG.name="base::Wichmann-Hill",.RNG.seed=77),
list(sigmasqyinv = 0.001, mubeta = c(-50, 50), Sigmabetainv = rbind(c(10000,
0), c(0, 10000)),.RNG.name="base::Wichmann-Hill",.RNG.seed=78),
list(sigmasqyinv = 10, mubeta = c(50, -50), Sigmabetainv = rbind(c(0.001,
0),c(0, 0.001)),.RNG.name="base::Wichmann-Hill",.RNG.seed=79),
list(sigmasqyinv = 0.001, mubeta = c(-50, -50),Sigmabetainv = rbind(c(0.001,
0), c(0, 0.001)),.RNG.name="base::Wichmann-Hill",.RNG.seed=80))

library(rjags)

m1 <- jags.model("C:\\temp\\parkvisit1.bug", d1, inits1, n.chains=4,
n.adapt=1000)

## Compiling data graph
##    Resolving undeclared variables
##    Allocating nodes
##    Initializing
##    Reading data back into data table
## Compiling model graph
```

```
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 390
##    Unobserved stochastic nodes: 33
##    Total graph size: 1316
##
## Initializing model
```

```
update(m1, 2000) # burn-in
x1 <- coda.samples(m1, c("mubeta","Sigmabeta","sigmasqy","rho"),
n.iter=50000)
gelman.diag(x1, autoburnin=FALSE, multivariate=FALSE)
```

```
## Potential scale reduction factors:
##
##                 Point est. Upper C.I.
## Sigmabeta[1,1]          1          1
## Sigmabeta[2,1]          1          1
## Sigmabeta[1,2]          1          1
## Sigmabeta[2,2]          1          1
## mubeta[1]               1          1
## mubeta[2]               1          1
## rho                     1          1
## sigmasqy                1          1
```

```
effectiveSize(x1[,c("mubeta[1]","mubeta[2]","Sigmabeta[1,1]","Sigmabeta[1,2]"
,"Sigmabeta[2,2]","sigmasqy","rho")])
```

```
##      mubeta[1]      mubeta[2] Sigmabeta[1,1] Sigmabeta[1,2] Sigmabeta[2,2]
##       198080.8       163483.5       188370.4       162508.4       144852.7
##       sigmasqy            rho
##       147948.2       157563.2
```

## c(ii)

```
summary(x1[,c("mubeta[1]","mubeta[2]","Sigmabeta[1,1]","Sigmabeta[1,2]","Sigm
abeta[2,2]","sigmasqy","rho")])
```

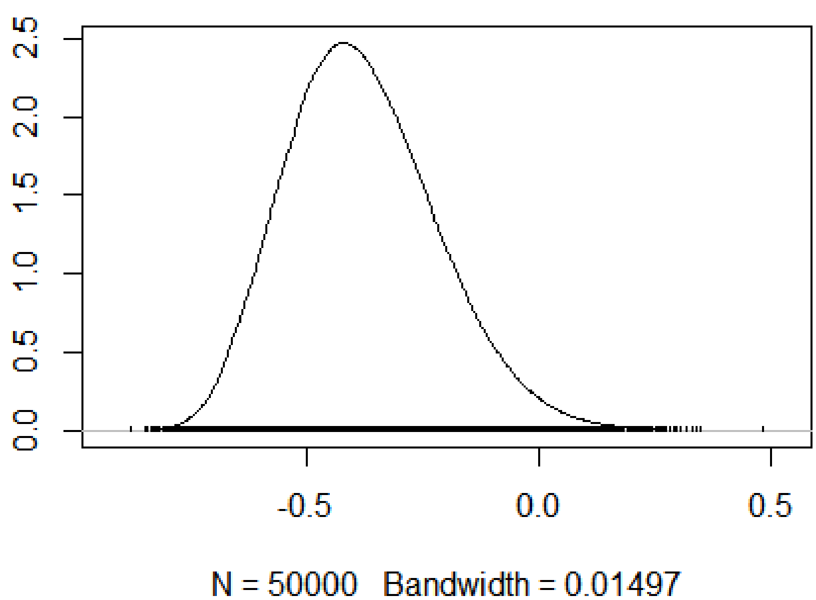
```
##
## Iterations = 2001:52000
## Thinning interval = 1
## Number of chains = 4
## Sample size per chain = 50000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                     Mean        SD  Naive SE Time-series SE
## mubeta[1]      12.176639 0.3932035 8.792e-04      8.835e-04
## mubeta[2]       0.030601 0.0103980 2.325e-05      2.572e-05
## Sigmabeta[1,1]  4.641285 1.2896068 2.884e-03      2.972e-03
```

```
## Sigmabeta[1,2] -0.044942 0.0250845 5.609e-05      6.224e-05
## Sigmabeta[2,2]  0.002922 0.0008701 1.946e-06      2.286e-06
## sigmasqy        0.057892 0.0045237 1.012e-05      1.176e-05
## rho            -0.380779 0.1622743 3.629e-04      4.089e-04
##
## 2. Quantiles for each variable:
##
##                     2.5%       25%       50%       75%     97.5%
## mubeta[1]       11.40007 11.916344 12.177173 12.436363 12.950936
## mubeta[2]        0.01020  0.023707  0.030587  0.037451  0.051081
## Sigmabeta[1,1]   2.76477  3.730703  4.429051  5.314420  7.738573
## Sigmabeta[1,2]  -0.10239 -0.058684 -0.042180 -0.028146 -0.003358
## Sigmabeta[2,2]   0.00165  0.002308  0.002782  0.003375  0.005015
## sigmasqy         0.04968  0.054742  0.057653  0.060804  0.067413
## rho             -0.66250 -0.496964 -0.392766 -0.276852 -0.032544
```

## c(iii)

95% posterior for $\rho$ is (-0.66250, -0.032544)

```
densplot(x1[,c("rho")])
```




## c(iv)

Posterior probability of $\rho$<0

```
post.samp <- as.matrix(x1)
post_rho_neg<- mean(post.samp[,"rho"] < 0)
post_rho_pos<- mean(post.samp[,"rho"] > 0)
bayes_factor <-post_rho_neg/post_rho_pos
post_rho_neg

## [1] 0.982945

bayes_factor

## [1] 57.63383
```

Since bayes factor is ~57 this indicates strong evidence in favor of *rho* <0

## c(v)

```
post.samp.change = exp(12*post.samp[,"mubeta[2]"])
quantile(post.samp.change,c(0.025,0.975))

##     2.5%    97.5%
## 1.130208 1.845914
```

95% central interval for $e^{12\mu_{\beta_2}}$ is (1.130208 1.845914)

## c(vi)

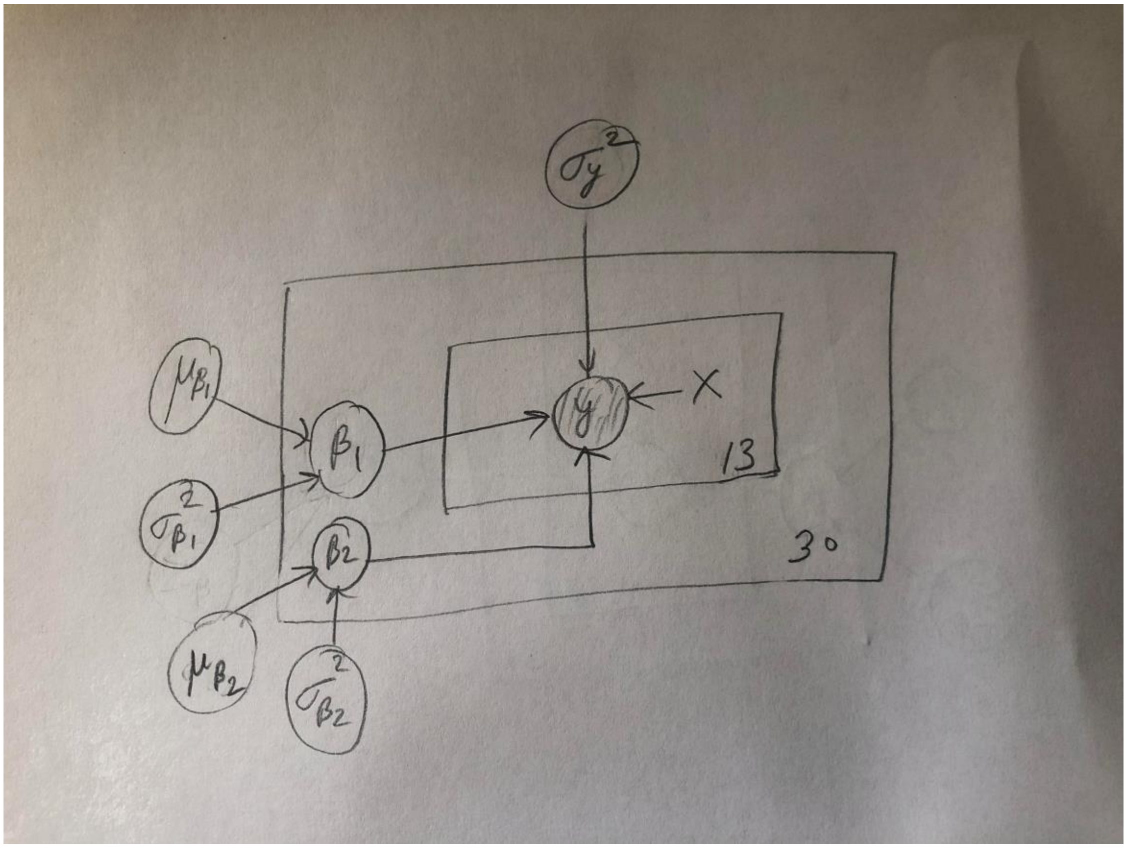
The effective no of parameters is 57.96 and DIC is 52.26

```
dic.samples(m1,100000)

## Mean deviance:  -5.696
## penalty 57.96
## Penalized deviance: 52.26
```

## d(i)

DAG




## d(ii)

```
JAGS MODEL
data {
dimY <- dim(visits)
yearcent <- year - mean(year)
}
model {
for (j in 1:dimY[1]) {
for (i in 1:dimY[2]) {
visits[j,i] ~ dnorm(beta[1,j] + beta[2,j]*yearcent[i], sigmasqyinv)
}
beta[1,j] ~ dnorm(mubeta[1], beta1_precision)
```

*beta[2,j] ~ dnorm(mubeta[2], beta2_precision)*

*}*

*mubeta ~ dmnorm(mubeta0, Sigmamubetainv)*

*Sigmabeta1~ dunif(0, 1000)*

*Sigmabeta2~ dunif(0, 1000)*

*Sigmabeta1_sq <- Sigmabeta1^2*

*Sigmabeta2_sq <- Sigmabeta2^2*

*beta1_precision <- 1/(Sigmabeta1_sq)*

*beta2_precision <- 1/(Sigmabeta2_sq)*

*sigmasqyinv ~ dgamma(0.0001, 0.0001)*

*sigmasqy <- 1/sigmasqyinv*

*}*

```
d1 <- list(visits = log(df[,-1]),year = c(2006:2018),mubeta0 = c(0, 0),
Sigmamubetainv = rbind(c(0.000001, 0),c(0, 0.000001)))
library(rjags)
m1 <- jags.model("C:\\temp\\parkvisit2.bug", d1, inits1, n.chains=4,
n.adapt=1000)

## Compiling data graph
##    Resolving undeclared variables
##    Allocating nodes
##    Initializing
##    Reading data back into data table
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 390
##    Unobserved stochastic nodes: 64
##    Total graph size: 1282

## Initializing model

update(m1, 2000) # burn-in
x1 <- coda.samples(m1,
c("mubeta","sigmasqy","Sigmabeta1_sq","Sigmabeta2_sq"), n.iter=50000)
gelman.diag(x1, autoburnin=FALSE, multivariate=FALSE)

## Potential scale reduction factors:
##
##               Point est. Upper C.I.
## Sigmabeta1_sq          1          1
```

```
## Sigmabeta2_sq          1          1
## mubeta[1]              1          1
## mubeta[2]              1          1
## sigmasqy               1          1
```

```
effectiveSize(x1[,c("mubeta[1]","mubeta[2]","Sigmabeta1_sq","Sigmabeta2_sq","sigmasqy")])
```

```
##     mubeta[1]     mubeta[2] Sigmabeta1_sq Sigmabeta2_sq      sigmasqy
##     201296.57     153481.68     104942.48      99129.66     148261.50
```

## d(iii)

```
summary(x1[,c("mubeta[1]","mubeta[2]","Sigmabeta1_sq","Sigmabeta2_sq","sigmasqy")])
```

```
## 
## Iterations = 3001:53000
## Thinning interval = 1 
## Number of chains = 4 
## Sample size per chain = 50000 
## 
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
## 
##                   Mean        SD  Naive SE Time-series SE
## mubeta[1]     12.176393 0.3751660 8.389e-04      8.362e-04
## mubeta[2]      0.030609 0.0091755 2.052e-05      2.342e-05
## Sigmabeta1_sq  4.226757 1.2151295 2.717e-03      3.755e-03
## Sigmabeta2_sq  0.002214 0.0007339 1.641e-06      2.331e-06
## sigmasqy       0.058048 0.0045460 1.017e-05      1.181e-05
## 
## 2. Quantiles for each variable:
## 
##                    2.5%       25%       50%       75%     97.5%
## mubeta[1]     11.434059 11.929191 12.175927 12.424544 12.913098
## mubeta[2]      0.012520  0.024555  0.030627  0.036643  0.048701
## Sigmabeta1_sq  2.465673  3.372466  4.020480  4.856217  7.172742
## Sigmabeta2_sq  0.001159  0.001697  0.002089  0.002587  0.003998
## sigmasqy       0.049818  0.054874  0.057815  0.060961  0.067623
```

## d(iv)

```
post.samp <- as.matrix(x1)
post.samp.change = exp(12*post.samp[,"mubeta[2]"])
quantile(post.samp.change,c(0.025,0.975))
```

```
##     2.5%    97.5% 
## 1.162118 1.793928
```

95% central posterior $e^{12\mu_{\beta_2}}$ in univariate model (1.16,1.79) is comparable (albeit narrower) than bivariate model (1.13 1.84)

## d(v)

```
dic.samples(m1,100000)
```

```
## Mean deviance:  -4.517 
## penalty 57.69 
## Penalized deviance: 53.17
```

## d (vi)

The DIC value of the bivariate prior model(52.26 ) is lower than univariate model (53.17), the bivariate model is a prefered model based on DIC.