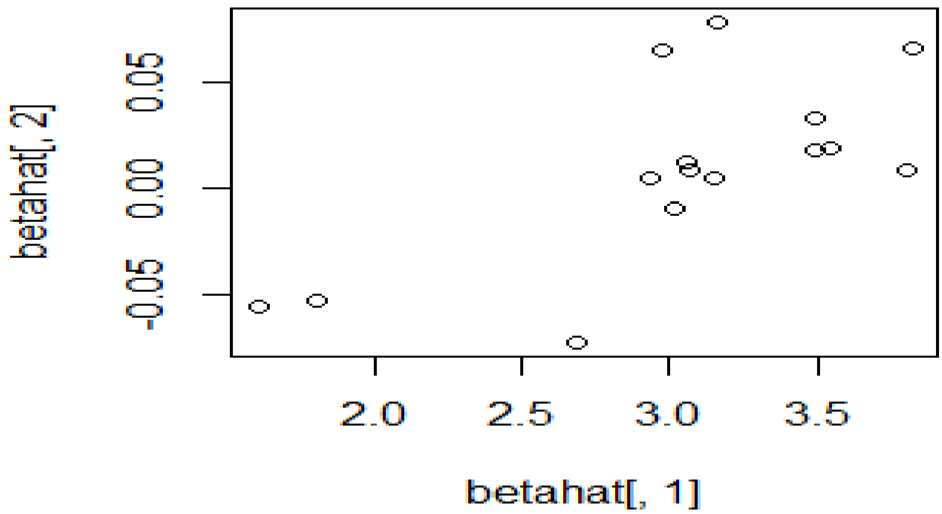
Assignment 5

```
> stock = read.table("stockprices.txt",row.names = 1, head
er = TRUE)
> logstock = log(stock)
> betahat <- matrix(NA, nrow(logstock), 2)
> for(j in 1:nrow(logstock))
+ betahat[j,] <- lsfit(rbind(1,2,3,4,5,6,7,8,9,10,11,12)-
6.5, t(logstock[j,]))$coef
```

(a)

```
> plot(betahat[,1], betahat[,2])
> apply(betahat, 2, mean)
[1] 3.040272605 0.008148221
```



(b)(i)

data {

    dimY <- dim(price)

    monthcent <- month - mean(month)

}

model {

    for (j in 1:dimY[1]) {

```
        for (i in 1:dimY[2]) {
            price[j,i] ~ dnorm(beta[1,j] + beta[2,j]*monthcent[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(price = logstock, month = c(1,2,3,4,5,6,7,8,9,
10,11,12), mubeta0 = c(0,0),Sigmamubetainv = rbind(c(0.000
001, 0),c(0, 0.000001)), Sigma0 = rbind(c(1,0),c(0,0.01)))
> inits1 <- list(list(sigmasqyinv = 10, mubeta = c(1000, 1
000), Sigmabetainv = rbind(c(100,0),c(0,100))),list(sigma
sqyinv = 0.001, mubeta = c(-1000, 1000), Sigmabetainv = rb
ind(c(100,0),c(0,100))),list(sigmasqyinv = 10, mubeta = c
(1000,-1000),Sigmabetainv = rbind(c(0.001,0),c(0,0.001))),
list(sigmasqyinv = 0.001, mubeta = c(-1000,-1000), Sigmabe
tainv = rbind(c(0.001,0),c(0,0.001))))
```

```
> library(rjags)
Loading required package: coda
Linked to JAGS 4.3.0
Loaded modules: basemod,bugs
> m1 <- jags.model("stockprices.bug", d1, inits1, n.chains
=4, n.adapt=10000)
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: 180
   Unobserved stochastic nodes: 18
   Total graph size: 639

Initializing model

> update(m1, 10000)
  |**************************************************| 10
0%
> x1 <- coda.samples(m1, c("mubeta","Sigmabeta", "sigmasqy
"), n.iter=20000)
  |**************************************************| 10
0%
```

(ii)

```
> summary(x1)

Iterations = 10001:30000
Thinning interval = 1
Number of chains = 4
Sample size per chain = 20000

1. Empirical mean and standard deviation for each variabl
e,
   plus standard error of the mean:

                 Mean       SD  Naive SE Time-series SE
```

```
Sigmabeta[1,1]     5.5998 9.410e+01 3.327e-01      1.894e+0
0
Sigmabeta[2,1]    -0.3315 1.029e+01 3.638e-02      1.924e-0
1
Sigmabeta[1,2]    -0.3315 1.029e+01 3.638e-02      1.924e-0
1
Sigmabeta[2,2]     0.1205 2.481e+00 8.771e-03      4.436e-0
2
mubeta[1]         8.8757 4.358e+01 1.541e-01      6.077e+00
mubeta[2]        17.2642 6.287e+01 2.223e-01      1.854e+01
sigmasqy      53191.7590 2.160e+05 7.638e+02      3.932e+0
4

2. Quantiles for each variable:

                      2.5%       25%      50%      75%
Sigmabeta[1,1]   0.276812  0.424244 0.549905 0.753270
Sigmabeta[2,1]  -0.311486  0.009560 0.017302 0.027351
Sigmabeta[1,2]  -0.311486  0.009560 0.017302 0.027351
Sigmabeta[2,2]   0.001694  0.002614 0.003418 0.004719
mubeta[1]      -40.880776  2.906871 3.046556 3.191920
mubeta[2]       -0.023981 -0.001243 0.009675 0.022029
sigmasqy         0.016613  0.019194 0.020846 0.022918
                    97.5%
Sigmabeta[1,1] 1.079e+01
Sigmabeta[2,1] 1.042e-01
Sigmabeta[1,2] 1.042e-01
Sigmabeta[2,2] 1.803e-01
mubeta[1]      1.504e+02
mubeta[2]      2.597e+02
sigmasqy       8.901e+05
```

(iii)

```
> x1 <- coda.samples(m1, c("beta","mubeta","Sigmabeta","si
gmasqy","rho"), n.iter=20000)
  |**************************************************| 10
0%
> effectiveSize(x1[,c("mubeta[1]","mubeta[2]","Sigmabeta
[1,1]","Sigmabeta[1,2]","Sigmabeta[2,2]","sigmasqy","rho
")])
    mubeta[1]     mubeta[2]
     80889.25      73640.25
```

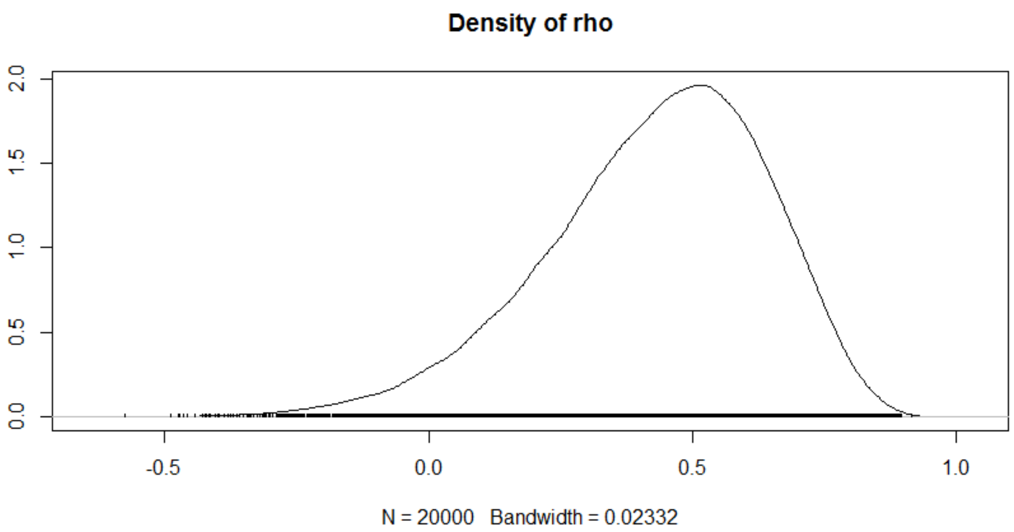
```
Sigmabeta[1,1] Sigmabeta[1,2]
      68006.88       67248.43
Sigmabeta[2,2]       sigmasqy
      64618.21       58132.24
           rho
      67640.73
```

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

Iterations = 30001:50000
Thinning interval = 1
Number of chains = 4
Sample size per chain = 20000

1. Empirical mean and standard deviation for each variabl
e,
   plus standard error of the mean:

                   Mean       SD  Naive SE
mubeta[1]      3.039538 0.197732 6.991e-04
mubeta[2]      0.008038 0.015843 5.601e-05
Sigmabeta[1,1] 0.584143 0.251770 8.901e-04
Sigmabeta[1,2] 0.020206 0.015019 5.310e-05
Sigmabeta[2,2] 0.003608 0.001576 5.571e-06
sigmasqy      0.020736 0.002422 8.563e-06
rho          0.428763 0.210433 7.440e-04
               Time-series SE
mubeta[1]           6.953e-04
mubeta[2]           5.843e-05
Sigmabeta[1,1]       9.656e-04
Sigmabeta[1,2]       5.797e-05
Sigmabeta[2,2]       6.199e-06
sigmasqy            1.005e-05
rho                8.099e-04

2. Quantiles for each variable:

                    2.5%       25%
mubeta[1]       2.648495  2.911936
mubeta[2]      -0.023346 -0.002180
Sigmabeta[1,1]  0.275564  0.415618
Sigmabeta[1,2] -0.001698  0.010679
```

```
Sigmabeta[2,2]  0.001665  0.002552
sigmasqy        0.016519  0.019029
rho            -0.042702  0.299432
                   50%      75%   97.5%
mubeta[1]      3.039608 3.166192 3.43113
mubeta[2]      0.008049 0.018312 0.03921
Sigmabeta[1,1] 0.529173 0.687886 1.21414
Sigmabeta[1,2] 0.017752 0.026873 0.05606
Sigmabeta[2,2] 0.003266 0.004262 0.00757
sigmasqy       0.020558 0.022238 0.02600
rho            0.453263 0.582603 0.76997
```

95% central posterior credible interval for rho is (-0.042702, 0.76997)

```
> par(mfrow=c(2,2))
> densplot(x1[,c("mubeta[1]","mubeta[2]","Sigmabeta[1,1]",
"Sigmabeta[1,2]","Sigmabeta[2,2]","sigmasqy","rho")])
```



(iv)

The posterior probability that $\rho > 0$ is:

```
> mean(as.numeric(unlist(x1[,c("rho")]))>0)
[1] 0.964425
```

Since prior odds = 1, BF(ρ>0;ρ<0)=posterior odds which is:

```
> plz = mean(as.numeric(unlist(x1[,c("rho")]))>0)
> plz / (1-plz)
[1] 27.10963
```

It is strong data evidence for ρ>0 versus ρ<0.

(v)

```
> quantile(unlist(x1[,c("mubeta[2]")]),c(0.025,0.975))
       2.5%       97.5%
-0.02334579  0.03921436
> quandata = quantile(unlist(x1[,c("mubeta[2]")]),c(0.025,
0.975))
> exp(11*quandata)
     2.5%     97.5%
0.7735201 1.5393465
```

126.1% is within the 95% central posterior credible interval. It demonstrates that the change of NASDAQ composite index fits in the model.

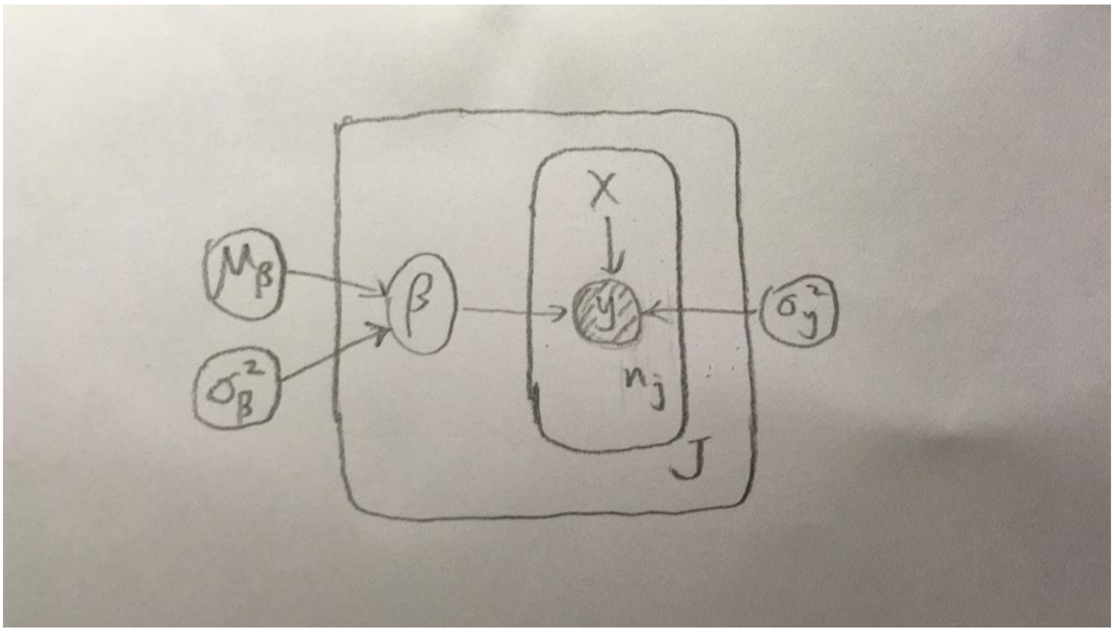
(vi)

```
> dic.samples(m1,200000)
  |**************************************************| 10
0%
Mean deviance:  -188.1
penalty 30.61
Penalized deviance: -157.5
```

Effective number of parameters: about 31

Plummer's DIC: -157.5

(c)(i)



(ii)

```
data {
    dimY <- dim(price)
    monthcent <- month - mean(month)
}
model {
    for (j in 1:dimY[1]) {
        for (i in 1:dimY[2]) {
            price[j,i] ~ dnorm(beta[1,j] + beta[2,j]*monthcent[i], sigmasqyinv)
        }
        beta[1,j] ~ dnorm(mubeta1, Sigmasqbeta1inv)
        beta[2,j] ~ dnorm(mubeta2, Sigmasqbeta2inv)
    }
```

```
    mubeta1 ~ dnorm(0, 0.000001)

    mubeta2 ~ dnorm(0, 0.000001)

    Sigmasqbeta1inv <- 1/Sigmabeta1^2

    Sigmabeta1 ~ dunif(0, 1000)

    Sigmasqbeta2inv <- 1/Sigmabeta2^2

    Sigmabeta2 ~ dunif(0, 1000)

    Sigmasqbeta1 <- Sigmabeta1^2

    Sigmasqbeta2 <- Sigmabeta2^2

    sigmasqyinv ~ dgamma(0.0001, 0.0001)

    sigmasqy <- 1/sigmasqyinv

}
```

```
> d1 <- list(price = logstok, month = c(1,2,3,4,5,6,7,8,9,
10,11,12))
> inits1 <- list(list(sigmasqyinv = 10, mubeta1 = 1000, mu
beta2 = 1000, Sigmabeta1inv=100, sigmabeta2inv=100),list(s
igmasqyinv=0.001, mubeta1=-1000,mubeta2=1000,Sigmabeta1in
v=100,Sigmabeta2inv=100),list(sigmasqyinv=10,mubeta1=1000,
mubeta2=-1000,Sigmabeta1inv=0.001,Sigmabeta2inv=0.001),li
st(sigmasqyinv=0.001,mubeta1=-1000,mubeta2=-1000,Sigmabet
a1inv=0.001,Sigmabeta2inv=0.001))
> library(rjags)
> m1 <- jags.model("stockprices.bug",d1,inits1,n.chains=4,
n.adapt=10000)
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: 180
   Unobserved stochastic nodes: 35
   Total graph size: 612

Initializing model

  |++++++++++++++++++++++++++++++++++++++++++++++++++| 10
0%
> update(m1,10000)
  |**************************************************| 10
0%
> x1 <- coda.samples(m1, c("mubeta1","mubeta2","Sigmasqbet
a1","Sigmasqbeta2","sigmasqy"), n.iter=20000)
  |**************************************************| 10
0%
> effectiveSize(x1[,c("mubeta1","mubeta2","Sigmasqbeta1",
"Sigmasqbeta2","sigmasqy")])
     mubeta1      mubeta2
    79403.07     70378.12
Sigmasqbeta1 Sigmasqbeta2
    26840.38     23626.16
    sigmasqy
    56074.36
```

(iii)

```
> summary(x1[,c("mubeta1","mubeta2","Sigmasqbeta1","Sigma
sqbeta2","sigmasqy")])

Iterations = 20001:40000
Thinning interval = 1
Number of chains = 4
Sample size per chain = 20000

1. Empirical mean and standard deviation for each variabl
e,
   plus standard error of the mean:

                 Mean       SD  Naive SE Time-series SE
mubeta1      3.040157 0.183873 6.501e-04      6.527e-04
mubeta2      0.008204 0.012782 4.519e-05      4.820e-05
Sigmasqbeta1 0.505978 0.239444 8.466e-04      1.464e-03
```

```
Sigmasqbeta2 0.002308 0.001172 4.144e-06     7.657e-06
sigmasqy     0.020806 0.002434 8.606e-06     1.028e-05

2. Quantiles for each variable:

                2.5%       25%      50%      75%    97.5%
mubeta1      2.6756645 2.922e+00 3.040903 3.158332 3.40328
8
mubeta2     -0.0172419 5.732e-05 0.008225 0.016402 0.03346
8
Sigmasqbeta1  0.2251471 3.481e-01 0.451054 0.597221 1.1149
05
Sigmasqbeta2  0.0009499 1.543e-03 0.002034 0.002751 0.0052
92
sigmasqy     0.0165834 1.909e-02 0.020630 0.022311 0.02605
4
```

(iv)

```
> quantile(unlist(x1[,c("mubeta2")]),c(0.025,0.975))
      2.5%      97.5%
-0.01724190  0.03346795
> quandata <- quantile(unlist(x1[,c("mubeta2")]),c(0.025,
0.975))
> exp(11*quandata)
     2.5%     97.5%
0.8272396 1.4450551
```

Previous results:

```
     2.5%     97.5%
0.7735201 1.5393465
```

Compared with previous results, current results are narrower interval which means the current model is more restricted than previous model.

(v)

```
> dic.samples(m1,200000)
  |**************************************************| 10
0%
Mean deviance:  -187.5
penalty 30.54
Penalized deviance: -157
```

Effective number of parameters: about 31

Plummer's DIC: -157

(vi)

Previous result: Plummer's DIC: -157.5

They are almost the same, no preference, either one is OK.

(d)(i)

Two ways, first, make it a multivariable regression, add β_3 multiplied by stock index into the model. Totally different construction, more parameters. Second, modify the generation of β_1 and β_2, add stock index and corresponding coefficients into both, stock index can be centralized, and corresponding coefficients are generated based on some reliable distribution. In this way, more hyperparameters are needed.

(ii)

Homoscedasticity, as time increases, if there is correlation between time value and outcome variable, for each value of explanatory variable, the distribution of residuals does not have the same variance anymore. The assumption of simple linear regression model is violated.