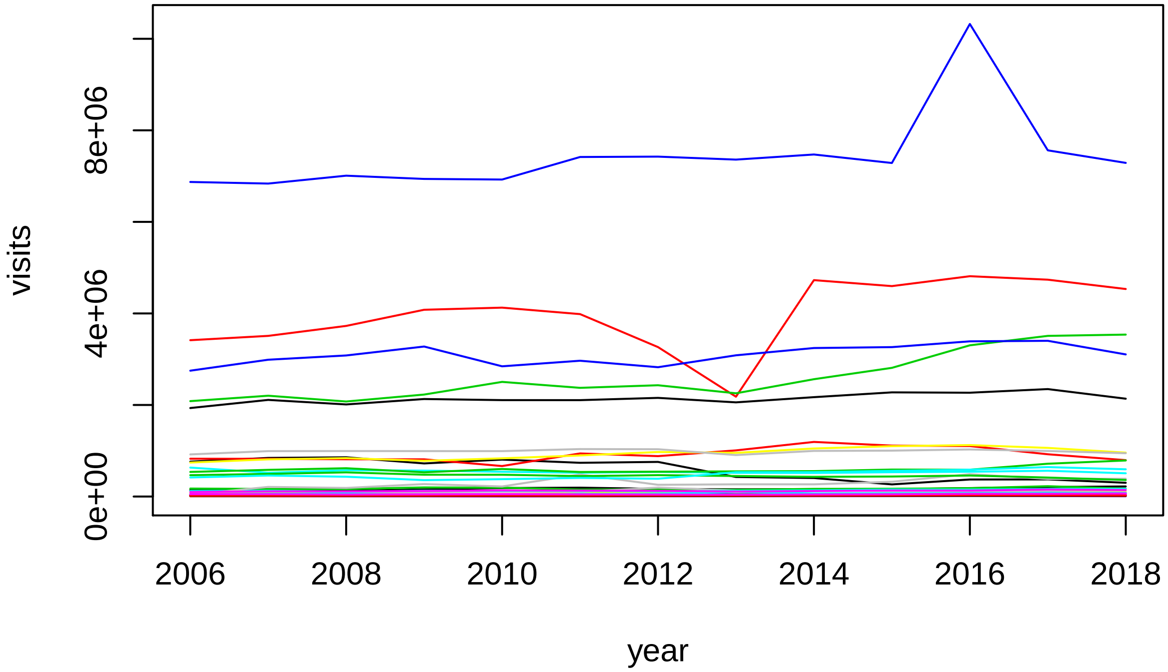
STAT 578 (Spring 2020) HW5 Solution

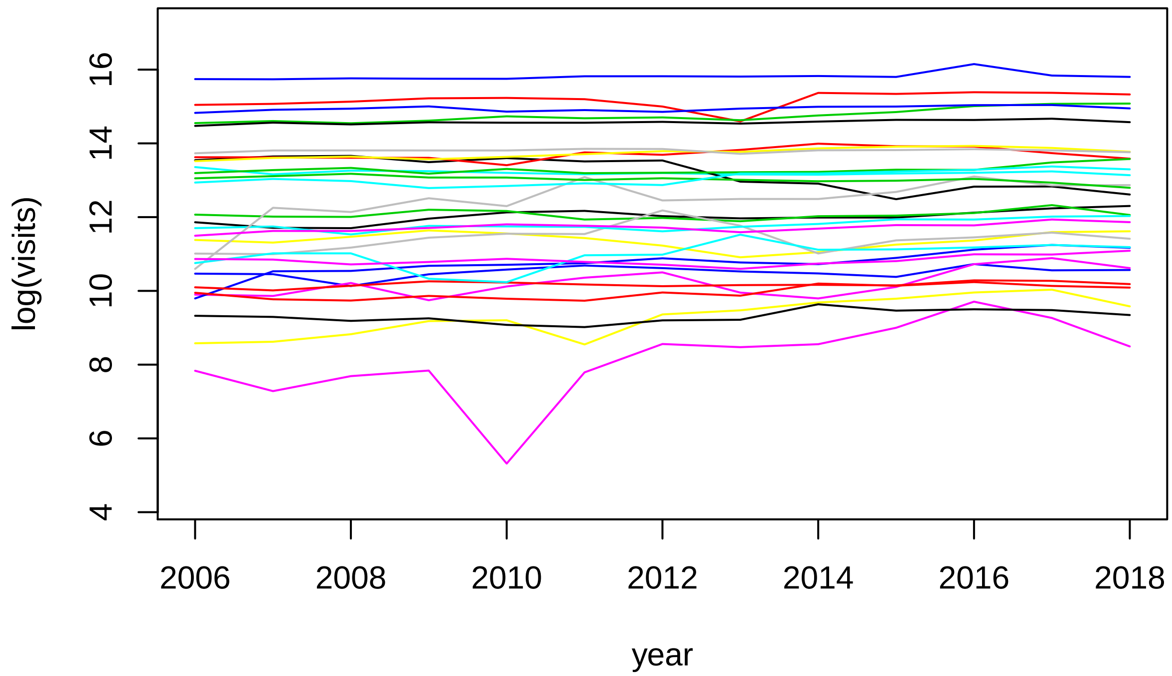
(a) (i)

```
visit <- read.csv("~/UIUC/STAT578_20Spring/HW5/usparkvisits.csv",
                  header = T, row.names=1)
plot(2006:2018, visit[1, ], type='l', col=1,
     xlab='year', ylab='visits', ylim=c(0, max(visit)))
for(i in 2:nrow(visit)){
  lines(2006:2018, visit[i, ], col=i)
}
```



(ii)

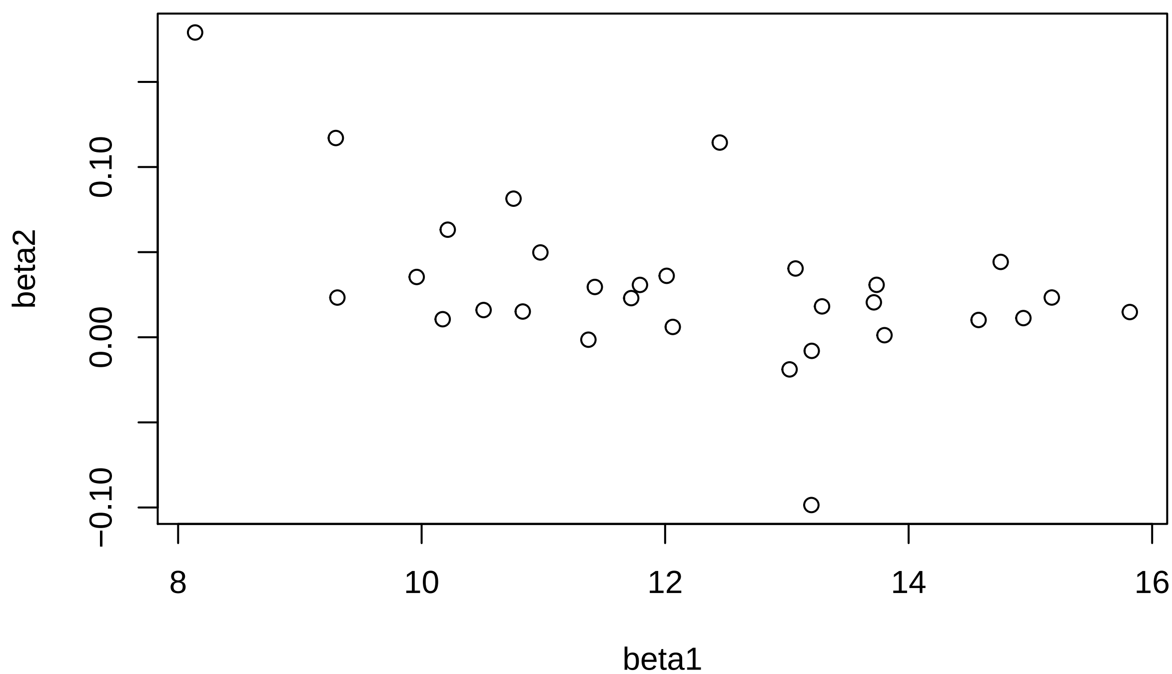
```
logvisit <- log(visit)
plot(2006:2018, logvisit[1, ], type='l', col=1,
     xlab='year', ylab='log(visits)', ylim=c(min(logvisit)-1, max(logvisit)+1))
for(i in 2:nrow(logvisit)){
  lines(2006:2018, logvisit[i, ], col=i)
}
```



(b) (i)

```r
year <- 2006:2018 - mean(2006:2018)

beta <- matrix(0, nrow(logvisit), 2)
colnames(beta) <- c('beta1', 'beta2')
for (j in 1:nrow(logvisit)) {
  beta[j, ] <- lm(as.numeric(logvisit[j, ]) ~ year)$coef
}
plot(beta)
```



(ii) Average of $\hat{\beta}_1^{(j)} = 12.18$ and average of $\hat{\beta}_2^{(j)} = 0.03$

```r
colMeans(beta)
```

```
##       beta1       beta2
## 12.17658206  0.03062706
```

(iii) Sample variance of $\hat{\beta}_1^{(j)} = 3.79$ and sample variance of $\hat{\beta}_2^{(j)} = 0.002$

```r
apply(beta, 2, var)
```

```
##       beta1       beta2
## 3.791482749 0.002286344
```

(iv) Sample correlation between $\hat{\beta}_1^{(j)}$ and $\hat{\beta}_2^{(j)}$ is -0.48.

```r
cor(beta[, 1], beta[, 2])
```

```
## [1] -0.4762208
```

(c) (i) JAGS model:

```
data {
  dimY <- dim(logvisit)
}
model {
  for (j in 1:dimY[1]) {
    for (i in 1:dimY[2]) {
      logvisit[j,i] ~ dnorm(beta[1,j] + beta[2,j]*year[i], sigmasq_y_inv)
    }
    beta[1:2,j] ~ dmnorm(mu_beta, Sigma_beta_inv)
  }

  mu_beta ~ dmnorm(mu_beta0, Sigma_mu_beta0_inv)

  Sigma_beta_inv ~ dwish(2*Sigma0, 2)
  Sigma_beta <- inverse(Sigma_beta_inv)

  sigmasq_y_inv ~ dgamma(0.0001, 0.0001)
  sigmasq_y <- 1/sigmasq_y_inv

  rho <- Sigma_beta[1,2] / sqrt(Sigma_beta[1,1] * Sigma_beta[2,2])
}
```

R code:

```r
library(rjags)
```

```
## Loading required package: coda

## Linked to JAGS 4.3.0

## Loaded modules: basemod,bugs
```

```r
d1 <- list(logvisit = logvisit,
           year = year,
           mu_beta0 = c(0, 0),
           Sigma_mu_beta0_inv = diag(1e-6, 2),
           Sigma0 = diag(c(10, 0.01), 2))

inits1 <- list(list(sigmasq_y_inv = 10,
                    mu_beta = c(100, 100),
```

```r
                    Sigma_beta_inv = diag(100, 2),
                    .RNG.name="base::Wichmann-Hill", .RNG.seed=12),
               list(sigmasq_y_inv = 0.01,
                    mu_beta = c(-100, 100),
                    Sigma_beta_inv = diag(100, 2),
                    .RNG.name="base::Wichmann-Hill", .RNG.seed=34),
               list(sigmasq_y_inv = 10,
                    mu_beta = c(100, -100),
                    Sigma_beta_inv = diag(0.01, 2),
                    .RNG.name="base::Wichmann-Hill", .RNG.seed=56),
               list(sigmasq_y_inv = 0.01,
                    mu_beta = c(-100, -100),
                    Sigma_beta_inv = diag(0.01, 2),
                    .RNG.name="base::Wichmann-Hill", .RNG.seed=78))

m1 <- jags.model("~/UIUC/STAT578_20Spring/HW5/m1.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: 1303
##
## Initializing model
```

```r
update(m1, 10000)  # burn-in
x1 <- coda.samples(m1, c("mu_beta", "Sigma_beta", "sigmasq_y", "rho"), n.iter=2000)
gelman.diag(x1, autoburnin=FALSE, multivariate=FALSE)
```

```
## Potential scale reduction factors:
##
##                 Point est. Upper C.I.
## Sigma_beta[1,1]          1          1
## Sigma_beta[2,1]          1          1
## Sigma_beta[1,2]          1          1
## Sigma_beta[2,2]          1          1
## mu_beta[1]               1          1
## mu_beta[2]               1          1
## rho                      1          1
## sigmasq_y                1          1
```

```r
effectiveSize(x1[,c("mu_beta[1]", "mu_beta[2]",
                    "Sigma_beta[1,1]", "Sigma_beta[1,2]", "Sigma_beta[2,2]", "sigmasq_y", "rho")])
```

```
##      mu_beta[1]      mu_beta[2] Sigma_beta[1,1] Sigma_beta[1,2] Sigma_beta[2,2]
##        8502.942        7097.947        7925.375        6626.459        6080.931
```

```
##       sigmasq_y             rho
##        5858.075        6209.695
```

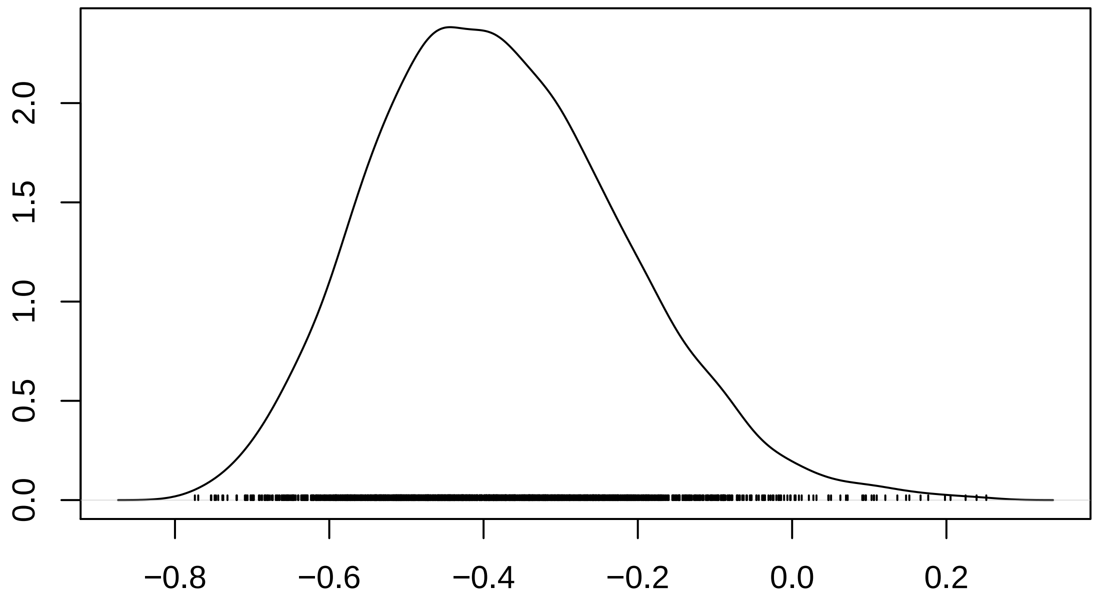
(ii)

```r
summary(x1[,c("mu_beta[1]", "mu_beta[2]",
              "Sigma_beta[1,1]", "Sigma_beta[1,2]", "Sigma_beta[2,2]",
              "sigmasq_y", "rho")])
```

```
##
## Iterations = 10001:12000
## Thinning interval = 1
## Number of chains = 4
## Sample size per chain = 2000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                      Mean        SD  Naive SE Time-series SE
## mu_beta[1]      12.182311 0.3936209 4.401e-03      4.287e-03
## mu_beta[2]       0.030550 0.0104170 1.165e-04      1.238e-04
## Sigma_beta[1,1]  4.648315 1.2952601 1.448e-02      1.456e-02
## Sigma_beta[1,2] -0.044795 0.0253099 2.830e-04      3.111e-04
## Sigma_beta[2,2]  0.002922 0.0008711 9.739e-06      1.124e-05
## sigmasq_y        0.057864 0.0045169 5.050e-05      5.909e-05
## rho             -0.378768 0.1627172 1.819e-03      2.065e-03
##
## 2. Quantiles for each variable:
##
##                      2.5%       25%       50%       75%     97.5%
## mu_beta[1]      11.425473 11.928898 12.178999 12.434598 12.978387
## mu_beta[2]       0.009986  0.023726  0.030712  0.037339  0.051256
## Sigma_beta[1,1]  2.763043  3.732521  4.431315  5.333303  7.786636
## Sigma_beta[1,2] -0.102486 -0.058409 -0.042206 -0.027843 -0.003380
## Sigma_beta[2,2]  0.001646  0.002308  0.002777  0.003386  0.005055
## sigmasq_y        0.049708  0.054649  0.057610  0.060879  0.067157
## rho             -0.661301 -0.496416 -0.390725 -0.273933 -0.033924
```

(iii) Approximate 95% central posterior credible interval for ρ: $(-0.66, -0.03)$

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



N = 2000 Bandwidth = 0.02858

(iv) $P(\rho < 0 \mid Data) = 0.98$. Bayes factor favoring $\rho < 0$ versus $\rho > 0$: $\frac{P(\rho<0|Data)}{P(\rho>0|Data)} = 56.55$.

There is strong data evidence that $\rho < 0$.

```
mean(unlist(x1[,"rho"]) < 0)
```

```
## [1] 0.982625
```

```
mean(unlist(x1[,"rho"]) < 0) / (1 - mean(unlist(x1[,"rho"]) < 0))
```

```
## [1] 56.55396
```

(v) Approximate 95% central posterior credible interval for $e^{12\mu_{\beta_2}}$: $(1.13, 1.85)$

```
exp(summary(x1[,"mu_beta[2]"])$quantiles[c(1,5)] * 12)
```

```
##     2.5%    97.5%
## 1.127306 1.849797
```

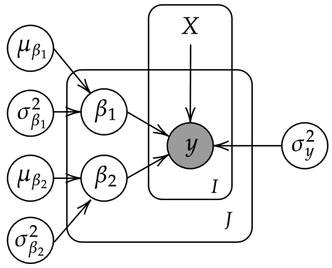
(vi) Effective number of parameters: 57.94

Plummer's DIC: 52.2

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

```
## Mean deviance:  -5.736
## penalty 57.94
## Penalized deviance: 52.2
```

(d) (i)



(ii) JAGS model:

```
data {
  dimY <- dim(logvisit)
}
model {
  for (j in 1:dimY[1]) {
    for (i in 1:dimY[2]) {
      logvisit[j,i] ~ dnorm(beta[1,j] + beta[2,j]*year[i], sigmasq_y_inv)
    }
    beta[1,j] ~ dnorm(mu_beta1, 1/sigma_beta1^2)
    beta[2,j] ~ dnorm(mu_beta2, 1/sigma_beta2^2)
  }
  mu_beta1 ~ dnorm(0, 0.000001)
  mu_beta2 ~ dnorm(0, 0.000001)

  sigma_beta1 ~ dunif(0, 1000)
  sigmasq_beta1 <- sigma_beta1^2
  sigma_beta2 ~ dunif(0, 1000)
  sigmasq_beta2 <- sigma_beta2^2

  sigmasq_y_inv ~ dgamma(0.0001, 0.0001)
  sigmasq_y <- 1/sigmasq_y_inv
}
```

R code:

```
d2 <- list(logvisit = logvisit,
           year = year)

inits2 <- list(list(sigmasq_y_inv = 10, mu_beta1 = 100, mu_beta2 = 100,
                    sigma_beta1 = 100, sigma_beta2 = 100,
                    .RNG.name="base::Wichmann-Hill", .RNG.seed=12),
               list(sigmasq_y_inv = 0.001, mu_beta1 = -100, mu_beta2 = 100,
                    sigma_beta1 = 100, sigma_beta2 = 100,
                    .RNG.name="base::Wichmann-Hill", .RNG.seed=34),
               list(sigmasq_y_inv = 10, mu_beta1 = 100, mu_beta2 = -100,
                    sigma_beta1 = 0.01, sigma_beta2 = 0.01,
                    .RNG.name="base::Wichmann-Hill", .RNG.seed=56),
               list(sigmasq_y_inv = 0.001, mu_beta1 = -100, mu_beta2 = -100,
                    sigma_beta1 = 0.01, sigma_beta2 = 0.01,
                    .RNG.name="base::Wichmann-Hill", .RNG.seed=78))

m2 <- jags.model("~/UIUC/STAT578_20Spring/HW5/m2.bug",
                 d2, inits2,
```

```r
                  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: 65
##    Total graph size: 1261
##
## Initializing model
```

```r
update(m2, 10000)  # burn-in
x2 <- coda.samples(m2, c("mu_beta1", "mu_beta2",
                         "sigmasq_beta1", "sigmasq_beta2",
                         "sigmasq_y"), n.iter=20000)
gelman.diag(x2, autoburnin=FALSE)
```

```
## Potential scale reduction factors:
##
##               Point est. Upper C.I.
## mu_beta1               1          1
## mu_beta2               1          1
## sigmasq_beta1          1          1
## sigmasq_beta2          1          1
## sigmasq_y              1          1
##
## Multivariate psrf
##
## 1
```

```r
effectiveSize(x2[,c("mu_beta1", "mu_beta2",
                    "sigmasq_beta1", "sigmasq_beta2",
                    "sigmasq_y")])
```

```
##      mu_beta1      mu_beta2 sigmasq_beta1 sigmasq_beta2     sigmasq_y
##      81306.73      62330.62      19767.46      35623.61      59039.02
```

(iii)

```r
summary(x2[,c("mu_beta1", "mu_beta2",
              "sigmasq_beta1", "sigmasq_beta2",
              "sigmasq_y")])
```

```
##
## Iterations = 11001:31000
## Thinning interval = 1
## Number of chains = 4
## Sample size per chain = 20000
##
## 1. Empirical mean and standard deviation for each variable,
```

```
##    plus standard error of the mean:
##
##                    Mean        SD  Naive SE Time-series SE
## mu_beta1      12.179559 0.3748949 1.325e-03      1.316e-03
## mu_beta2       0.030623 0.0092136 3.258e-05      3.690e-05
## sigmasq_beta1  4.225142 1.2267873 4.337e-03      1.951e-02
## sigmasq_beta2  0.002221 0.0007391 2.613e-06      3.949e-06
## sigmasq_y      0.058051 0.0045456 1.607e-05      1.872e-05
##
## 2. Quantiles for each variable:
##
##                    2.5%       25%       50%       75%     97.5%
## mu_beta1      11.434823 11.932305 12.181364 12.426445 12.913582
## mu_beta2       0.012419  0.024544  0.030624  0.036715  0.048784
## sigmasq_beta1  2.458429  3.368483  4.019133  4.834569  7.195842
## sigmasq_beta2  0.001163  0.001702  0.002097  0.002593  0.004014
## sigmasq_y      0.049817  0.054875  0.057813  0.060961  0.067640
```

(iv) Approximate 95% central posterior credible interval for $e^{12\mu_{\beta_2}}$: $(1.16, 1.80)$, which is narrower than the previous one.

```
exp(summary(x2[,"mu_beta2"])$quantiles[c(1,5)] * 12)
```

```
##     2.5%    97.5%
## 1.160705 1.795713
```

(v) Effective number of parameters: 57.67

Plummer's DIC: 53.17

```
dic.samples(m2, 100000)
```

```
## Mean deviance:  -4.5
## penalty 57.67
## Penalized deviance: 53.17
```

(vi) Plummer's DIC for this model is larger than the one for the first model, indicating that we prefer the more complex model.