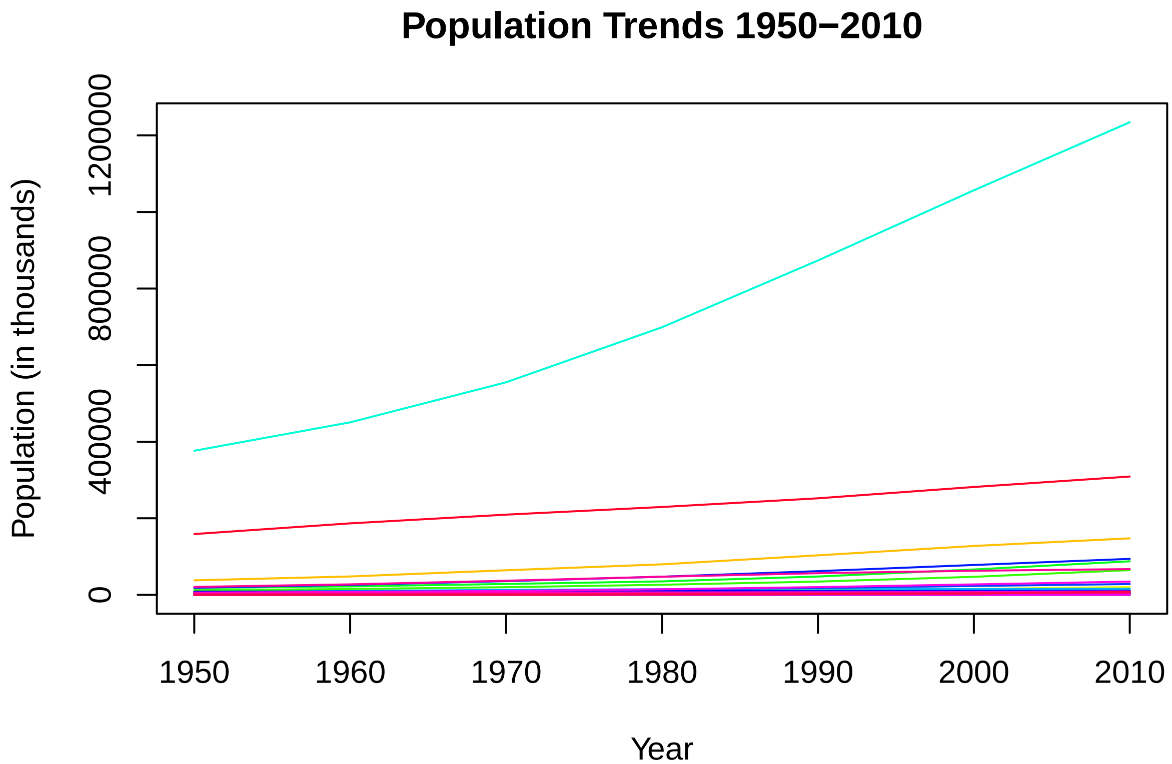
File countrypop.csv contains total population numbers (in thousands) for 40 different countries for the years 1950, 1960, 1970, 1980, 1990, 2000, and 2010. Build and compare two different varying-coefficient hierarchical normal regression models for the log-scale numbers, using JAGS and rjags.

```
data<- read.csv("./countrypop.csv", header=TRUE)
head(data,10)
```

```
##       Country  Year1950  Year1960  Year1970  Year1980   Year1990   Year2000
## 1     Andorra     6.198    13.410    24.275    36.063     54.508     65.390
## 2       Aruba    38.052    54.208    59.070    60.097     62.152     90.866
## 3     Austria  6936.442  7070.773  7516.238  7609.750   7723.954   8069.276
## 4  Azerbaijan  2927.926  3895.398  5180.032  6150.735   7242.758   8122.743
## 5     Bahrain   115.612   162.429   212.607   359.897    495.927    664.610
## 6  Bangladesh 37894.671 48013.505 64232.486 79639.498 103171.957 127657.862
## 7     Belarus  7745.004  8124.881  8913.549  9569.847  10151.135   9871.635
## 8    Cameroon  4307.021  5176.920  6519.754  8621.409  11780.086  15513.944
## 9     Comoros   159.459   191.122   230.055   307.831    411.598    542.358
## 10    Croatia  3850.294  4192.641  4423.069  4598.125   4776.370   4428.075
##      Year2010
## 1      84.454
## 2     101.665
## 3    8409.945
## 4    9032.465
## 5    1240.864
## 6  147575.433
## 7    9420.576
## 8   20341.236
## 9     689.696
## 10   4328.163
```

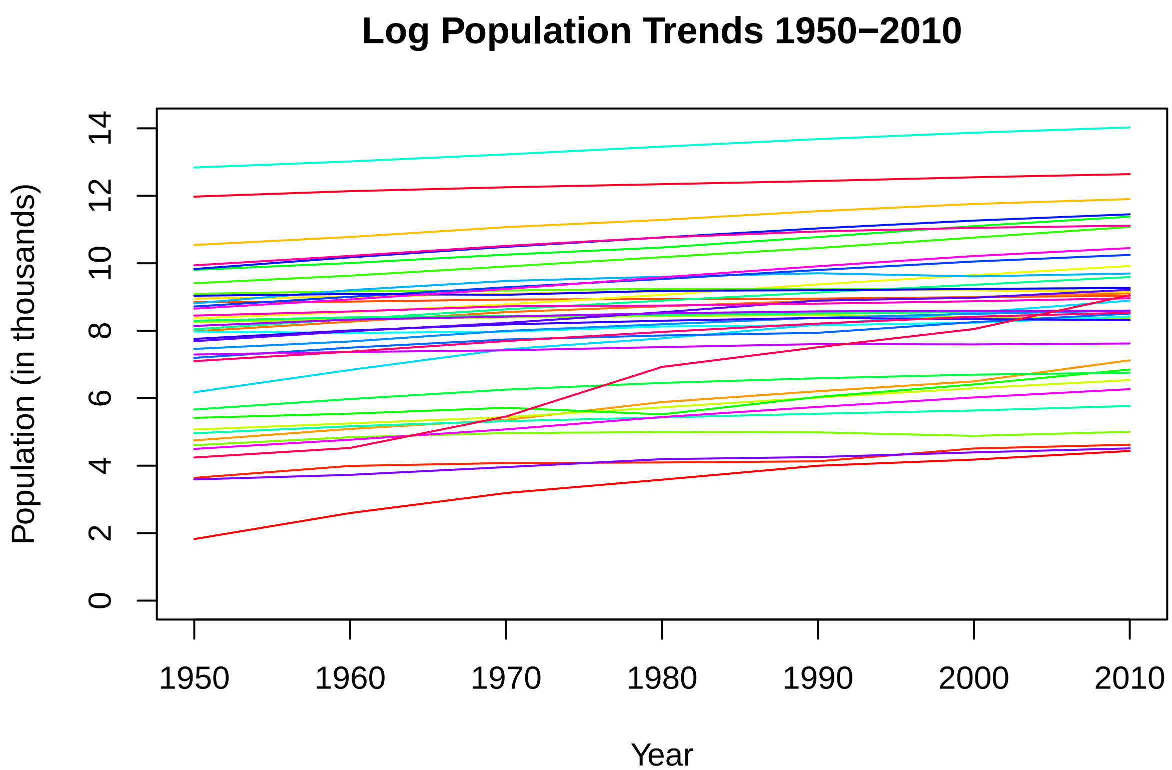
(a)
(i) On the same set of axes, plot segmented lines, one for each country, representing the population (in thousands) versus the year (1950, 1960, …). Distinguish the lines for different countries by using different colors or line types.

```
xs <- seq(1950, 2010, by=10)
plot(xs, data[1, -1], type="l", col=rainbow(nrow(data))[1], ylim=c(0, max(data[-1])), xlab="Year", ylab
for (i in 2:nrow(data)) {
  lines(xs, data[i, -1], col=rainbow(nrow(data))[i])
}
```



(ii) Repeat the previous part using the natural logarithm of the population numbers.

```
data[, -1] <- log(data[, -1])
plot(xs, data[1, -1], type="l", col=rainbow(nrow(data))[1], ylim=c(0, max(data[-1])), xlab="Year", ylab=
for (i in 2:nrow(data)) {
  lines(xs, data[i, -1], col=rainbow(nrow(data))[i])
}
```



Let y_{ij} be the natural logarithm of the population (in thousands) of country j in the year indexed by i ($i = 1, \ldots, 7$ corresponding to 1950, ..., 2010), and $j = 1, \ldots, 40$. For each country, let the log-population be modeled as a simple linear regression on the centered year index:

$$y_{ij} | \beta^{(j)}, \sigma_y^2 \sim \text{indep. } N\left(\beta_1^{(j)} + \beta_2^{(j)}(x_i - \bar{x}), \sigma_y^2\right)$$

where

$$\beta^{(j)} = \begin{pmatrix} \beta_1^{(j)} \\ \beta_2^{(j)} \end{pmatrix}, \quad j = 1, \ldots, 40$$

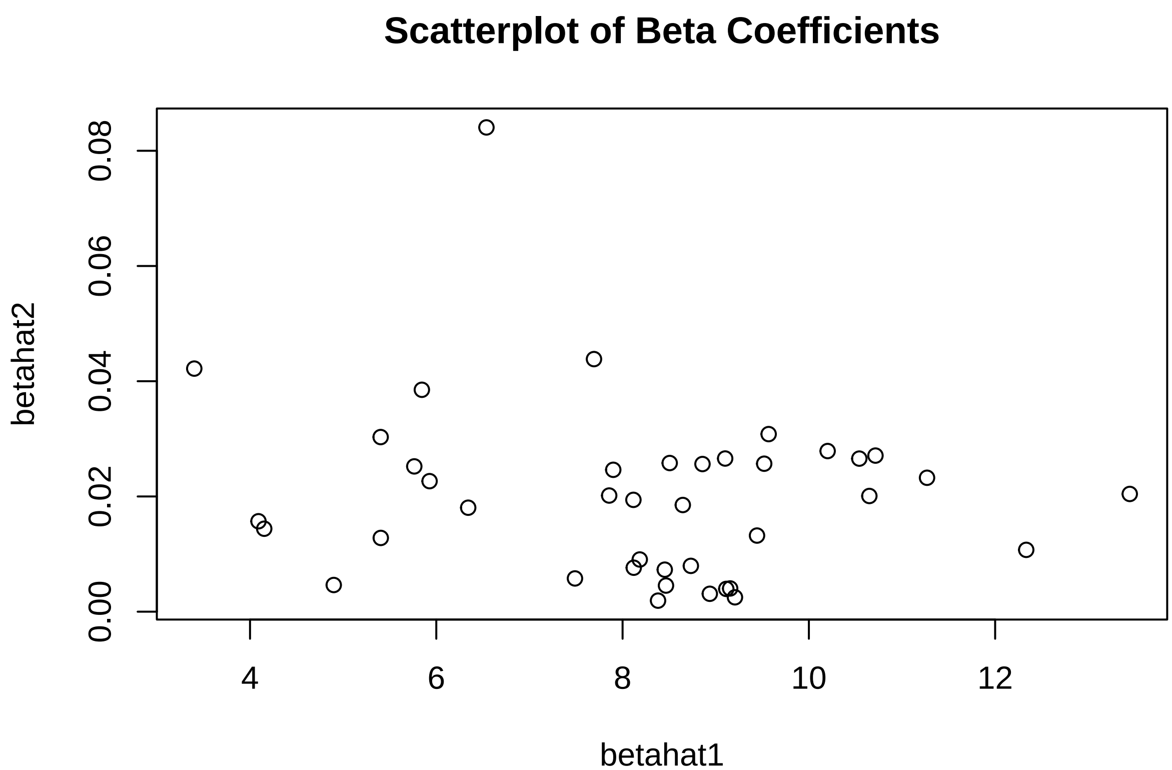
and

$$x_i = i, \quad i = 1, \ldots, 7$$

Note that the coefficients are allowed to depend on the country, but the variance is not.
(b) Let $\hat{\beta}_1^{(j)}$ and $\hat{\beta}_2^{(j)}$ be the ordinary least squares estimates of $\beta_1^{(j)}$ and $\beta_2^{(j)}$, estimated for country j.
(i) Produce a scatterplot of the pairs $(\hat{\beta}_1^{(j)}, \hat{\beta}_2^{(j)})$, $j = 1, \ldots, 40$.

```
years <- c(1950, 1960, 1970, 1980, 1990, 2000, 2010)
centered_years <- years - 1980
betahat <- matrix(NA, nrow = nrow(data), ncol = 2)
for (j in 1:nrow(data)) {
  population_data <- t(data[j, -1])
  fit <- lsfit(x = centered_years, y = population_data)
  betahat[j, ] <- fit$coef
}

plot(betahat[, 1], betahat[, 2], xlab = "betahat1", ylab = "betahat2", main = "Scatterplot of Beta Coef
```



(ii) Give the average (sample mean) of $\hat{\beta}_1^{(j)}$ and also of $\hat{\beta}_2^{(j)}$.

```
means = apply(betahat, 2, mean)
cat("mean_betahat1: ", means[1], "\n", "mean_betahat2: ", means[2])
```

```
## mean_betahat1:  8.159197
##  mean_betahat2:  0.01991429
```

(iii) Give the sample variance of $\hat{\beta}_1^{(j)}$ and also of $\hat{\beta}_2^{(j)}$.

```
var_betahat1 = var(betahat)[1,1]
var_betahat2 = var(betahat)[2,2]
cat("var_betahat1: ", var_betahat1, "\n", "var_betahat2: ", var_betahat2, "\n")
```

```
## var_betahat1:  4.946005
##  var_betahat2:  0.0002313094
```

(iv) Give the sample correlation between $\hat{\beta}_1^{(j)}$ and $\hat{\beta}_2^{(j)}$.

```
betahat_cov <- cov(betahat[, 1], betahat[, 2])
cat("betahat_cov: ", betahat_cov, "\n")
```

```
## betahat_cov:  -0.00566316
```

(c) Consider the bivariate prior

$$\beta^{(j)} | \mu_\beta, \Sigma_\beta \sim \text{iid } N(\mu_\beta, \Sigma_\beta)$$

with

$$\mu_\beta = \begin{pmatrix} \mu_{\beta 1} \\ \mu_{\beta 2} \end{pmatrix}$$

and covariance matrix

$$\Sigma_\beta = \begin{pmatrix} \sigma^2_{\beta 1} & \rho\sigma_{\beta 1}\sigma_{\beta 2} \\ \rho\sigma_{\beta 1}\sigma_{\beta 2} & \sigma^2_{\beta 2} \end{pmatrix}$$

with hyperpriors

$$\mu_\beta \sim N\left(0, 1000^2 I\right)$$

and

$$\Sigma_\beta^{-1} \sim \text{Wishart}_2\left(\left(\Sigma_0^{-1}\right)/2\right)$$

in the notation used in the lecture videos. For your analysis, use

$$\Sigma_0 = \begin{pmatrix} 5 & 0 \\ 0 & 0.02 \end{pmatrix}$$

based on preliminary analyses. Let the prior on σ_y^2 be

$$\sigma_y^2 \sim \text{Inv-gamma}(0.0001, 0.0001)$$

(i) List an appropriate JAGS model. Make sure to create nodes for Σ_β^{-1}, ρ, and σ_y^2.

```
d1 <- list(population = data[,-1],
           year = xs,
           mubeta0 = c(0, 0),
           Sigmamubetainv = rbind(c(0.000001, 0),
                                  c(0, 0.000001)),
           Sigma0 = rbind(c(5, 0),
                          c(0, 0.02)))

inits1 <- list(list(sigmasqyinv = 10, mubeta = c(1000, 1000),
                    Sigmabetainv = rbind(c(100, 0),
                                         c(0, 100))),
               list(sigmasqyinv = 0.001, mubeta = c(-1000, 1000),
                    Sigmabetainv = rbind(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),
                    Sigmabetainv = rbind(c(0.001, 0),
                                         c(0, 0.001))))
library(rjags)
```

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

```
## Linked to JAGS 4.3.2
```

```
## Loaded modules: basemod,bugs
```

```
#print the model
cat (readLines('countrypop_1.bug'), sep= '\n')
```

```
## data {
##   dimY <- dim(population)
##   yearcent <- year - mean(year)
## }
##
## model {
##   for (j in 1:dimY[1]) {
##     for (i in 1:dimY[2]) {
##       population[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
## }
```

```r
m1 <- jags.model("countrypop_1.bug", d1, inits1, n.chains=4, n.adapt=5000)
```

```
## 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: 280
##    Unobserved stochastic nodes: 43
##    Total graph size: 1004
##
## Initializing model
```

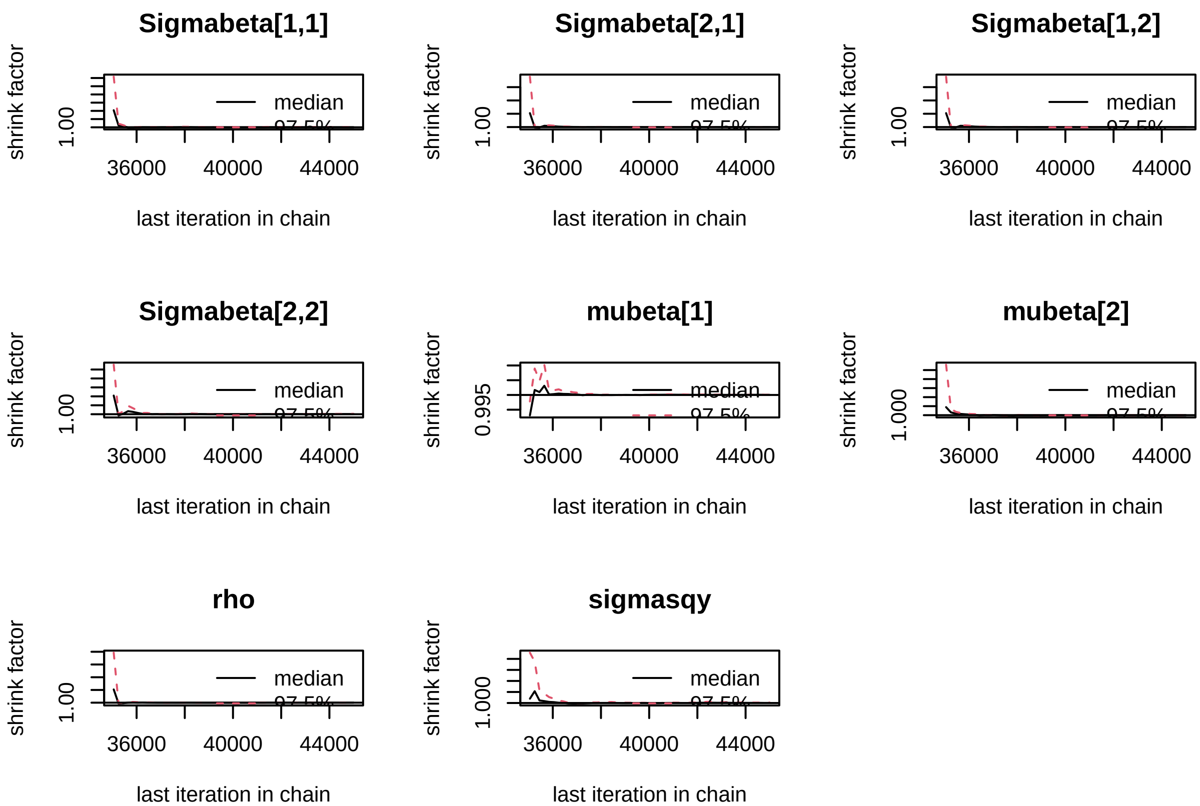
(ii) Display the coda summary of the results for the monitored parameters.

```r
update(m1, 35000) # burn-in
x1 <- coda.samples(m1, c("mubeta","Sigmabeta","sigmasqy", "rho"), n.iter=10000)

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
```

```r
gelman.plot(x1, c("mubeta","Sigmabeta","sigmasqy", "rho"), autoburnin=FALSE)
```




```r
effectiveSize(x1)
```

```
## Sigmabeta[1,1] Sigmabeta[2,1] Sigmabeta[1,2] Sigmabeta[2,2]      mubeta[1]
##       38198.64       38389.14       38389.14       37984.53       40726.68
##      mubeta[2]            rho       sigmasqy
##       40000.00       38371.68       22209.47
```

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

```
##                        Mean           SD     Naive SE Time-series SE
## mubeta[1]       8.160194389 0.3655607441 1.827804e-03   1.811932e-03
## mubeta[2]       0.019886477 0.0056703000 2.835150e-05   2.835235e-05
## Sigmabeta[1,1]  5.337944385 1.2484454503 6.242227e-03   6.388376e-03
## Sigmabeta[1,2] -0.005834996 0.0136789745 6.839487e-05   6.982092e-05
## Sigmabeta[2,2]  0.001291679 0.0003046300 1.523150e-06   1.564426e-06
## sigmasqy        0.009867360 0.0009941911 4.970955e-06   6.674398e-06
## rho            -0.069189490 0.1551427443 7.757137e-04   7.920312e-04
```

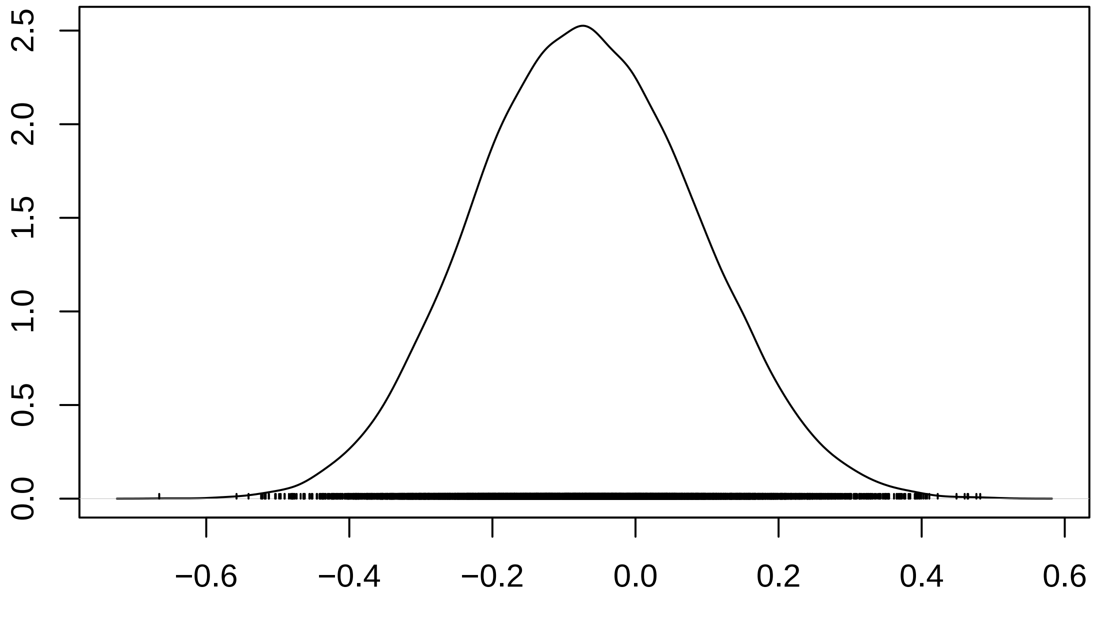
```r
summary(x1[,c("mubeta[1]","mubeta[2]","Sigmabeta[1,1]","Sigmabeta[1,2]",
              "Sigmabeta[2,2]","sigmasqy","rho")])$quantiles
```

```
##                        2.5%          25%          50%         75%       97.5%
## mubeta[1]       7.436360860  7.917841372  8.159036179 8.405882025 8.876009106
## mubeta[2]       0.008637544  0.016158242  0.019879969 0.023624875 0.031082424
## Sigmabeta[1,1]  3.418676524  4.452191897  5.165893634 6.025735764 8.269149271
## Sigmabeta[1,2] -0.034176613 -0.014162094 -0.005559062 0.002829731 0.020678222
## Sigmabeta[2,2]  0.000825890  0.001077522  0.001248754 0.001456135 0.002011983
```

```
## sigmasqy        0.008116119  0.009168586  0.009798569 0.010498616 0.011986597
## rho            -0.366604770 -0.176387227 -0.071420504 0.036303220 0.238140476
```

(iii) Give an approximate 95% central posterior interval for the correlation parameter ρ, and also produce a graph of its (estimated) posterior density.
An approximate 95% central posterior interval for the correlation parameter ρ is (-0.3666, 0.2381).

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



(iv) Approximate the posterior probability that $\rho < 0$. Also, approximate the Bayes factor favoring $\rho < 0$ versus $\rho >= 0$. Describe the level of data evidence for $\rho < 0$.

```r
# posterior probability for rho < 0
post.pro = mean(as.matrix(x1[,c("rho")]) < 0)
# Bayes factor
factor <- mean(as.matrix(x1[,c("rho")]) < 0) / (mean(as.matrix(x1[,c("rho")]) >= 0))

cat("The posterior probability for rho < 0 is:", post.pro, ",
    \n and Bayes factor favoring rho < 0 over rho >= 0 is:", factor,
    ", \n which means the data evidence is barely mentionable. \n")
```

```
## The posterior probability for rho < 0 is: 0.673 ,
##
##  and Bayes factor favoring rho < 0 over rho >= 0 is: 2.058104 ,
##  which means the data evidence is barely mentionable.
```

(v) The model implies that, over the 6 decades from 1950 to 2010, the median population change factor should be $e^{6\mu_{\beta_2}}$. Form an approximate 95% central posterior interval for this factor.

```r
cat("An approximate 95% central posterior interval for the median population change factor is:
    \n (", exp(6 * 0.0087),",", exp(6 * 0.0311), "). \n")
```

```
## An approximate 95% central posterior interval for the median population change factor is:
##
```

```
##  ( 1.053586 , 1.205145 ).
```

(vi) Use the rjags function dic.samples to compute the effective number of parameters ("penalty") and Plummer's DIC ("Penalized deviance"). Use at least 100,000 iterations.

```
dic.samples(m1, 200000)
```

```
## Mean deviance:  -499.9
## penalty 81.72
## Penalized deviance: -418.2
```

(d) Now consider a different model with "univariate" hyperpriors for the model coefficients, which do not allow for a coefficient correlation parameter:

$$\beta_1^{(j)} | \mu_{\beta_1}, \sigma_{\beta_1} \sim \text{iid } N(\mu_{\beta_1}, \sigma^2_{\beta_1})$$

$$\beta_2^{(j)} | \mu_{\beta_2}, \sigma_{\beta_2} \sim \text{iid } N(\mu_{\beta_2}, \sigma^2_{\beta_2})$$

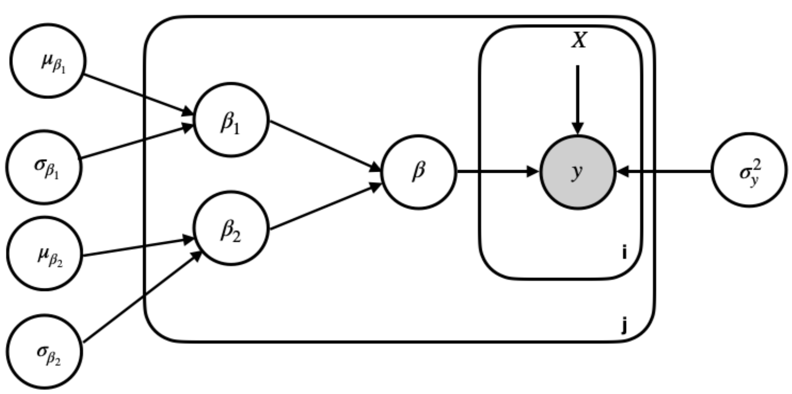
with hyperpriors

$$\mu_{\beta_1}, \mu_{\beta_2} \sim \text{iid } N(0, 1000^2)$$

$$\sigma_{\beta_1}, \sigma_{\beta_2} \sim \text{iid } U(0, 1000)$$

(i) Draw a complete DAG for this new model.

```
library(png)
img <- readPNG("dag.png")
plot(NA, xlim = c(0, 1.5), ylim = c(0,1.1), type = "n", xlab = "", ylab = "", xaxt = "n",
     yaxt = "n", bty = "n", asp = 1)
rasterImage(img, 0, 0, 1.9, 1.1)
```



(ii) List an appropriate JAGS model. Make sure that there are nodes for σ_{β_1}, σ_{β_2}, and σ^2_y.

```
d2 <- list(population = data[,-1],
           year = xs)
```

```r
inits2 <- list(list(sigmasqyinv = 10, mubeta1 = 1000, mubeta2 = 1000,
                    sigmabeta1 = 1000, sigmabeta2 = 1000),
               list(sigmasqyinv = 0.001, mubeta1 = -1000, mubeta2 = 1000,
                    sigmabeta1 = 1000, sigmabeta2 = 1000),
               list(sigmasqyinv = 10, mubeta1 = 1000, mubeta2 = -1000,
                    sigmabeta1 = 0.001, sigmabeta2 = 0.001),
               list(sigmasqyinv = 0.001, mubeta1 = -1000, mubeta2 = -1000,
                    sigmabeta1 = 0.001, sigmabeta2 = 0.001))
library(rjags)
```

```r
#print the model
cat (readLines('countrypop_2.bug'), sep= '\n')
```

```
## data {
##   dimY <- dim(population)
##   yearcent <- year - mean(year)
## }
##
## model {
##   for (j in 1:dimY[1]) {
##     for (i in 1:dimY[2]) {
##       population[j,i] ~ dnorm(beta[1,j] + beta[2,j]*yearcent[i], sigmasqyinv)
##     }
##     beta[1,j] ~ dnorm(mubeta1, sigmabeta1sqinv)
##     beta[2,j] ~ dnorm(mubeta2, sigmabeta2sqinv)
##   }
##   mubeta1 ~ dnorm(0, 0.000001)
##   mubeta2 ~ dnorm(0, 0.000001)
##   sigmabeta1 ~ dunif(0, 1000)
##   sigmabeta2 ~ dunif(0, 1000)
##   sigmasqyinv ~ dgamma(0.0001, 0.0001)
##
##   sigmabeta1sqinv <- 1/sigmabeta1^2
##   sigmabeta2sqinv <- 1/sigmabeta2^2
##   sigmasqy <- 1/sigmasqyinv
## }
```

```r
m2 <- jags.model("countrypop_2.bug", d2, inits2, n.chains=4, n.adapt=3000)
```

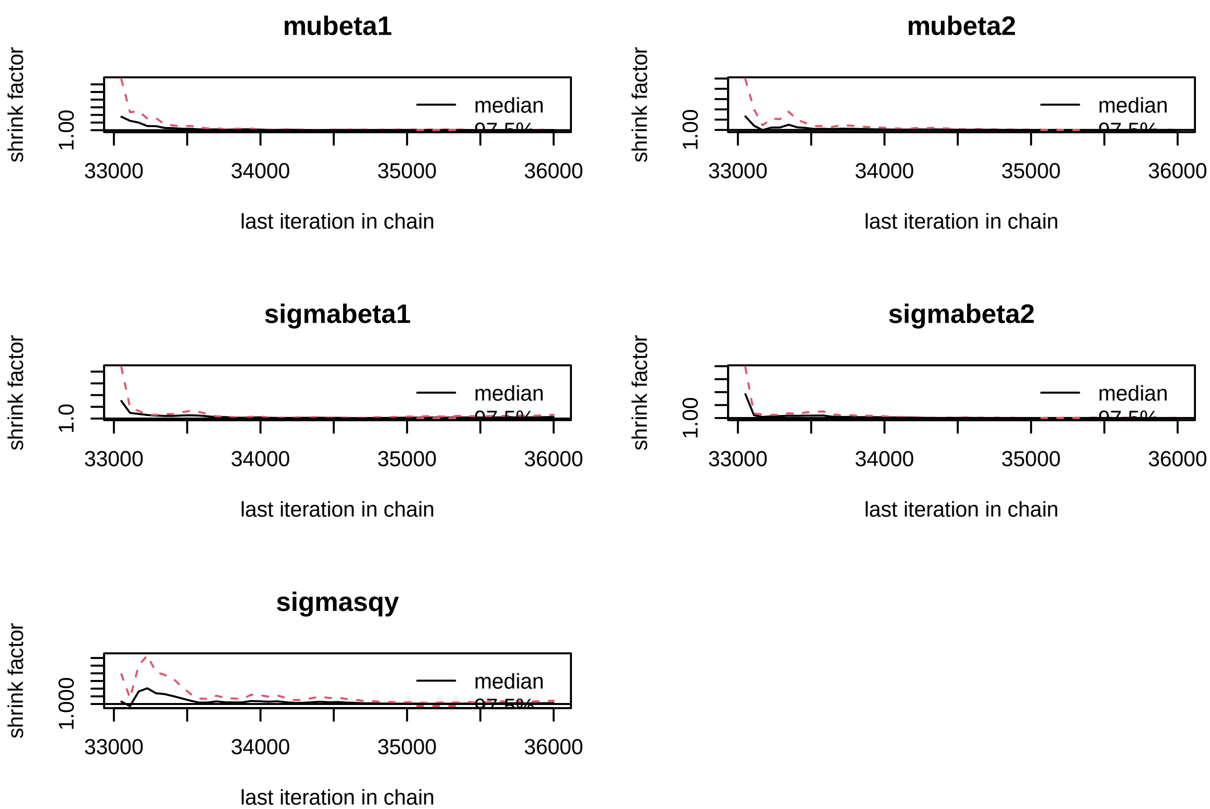
```
## Compiling data graph
##    Resolving undeclared variables
##    Allocating nodes
##    Initializing
##    Reading data back into data table
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 280
##    Unobserved stochastic nodes: 85
##    Total graph size: 952
##
## Initializing model
```

(iii) Display the coda summary of the results for the monitored parameters.

```r
update(m2, 30000) # burn-in
x2 <- coda.samples(m2, c("mubeta1","mubeta2","sigmabeta1","sigmabeta2", "sigmasqy"), n.iter=3000)
gelman.diag(x2, autoburnin=FALSE, multivariate=FALSE)
```

```
## Potential scale reduction factors:
##
##            Point est. Upper C.I.
## mubeta1          1.00       1.00
## mubeta2          1.00       1.00
## sigmabeta1       1.01       1.03
## sigmabeta2       1.00       1.00
## sigmasqy         1.00       1.00
```

```r
gelman.plot(x2, autoburnin=FALSE)
```




```r
effectiveSize(x2)
```

```
##    mubeta1    mubeta2 sigmabeta1 sigmabeta2   sigmasqy
##  13993.993  11554.684   4769.414   9494.022   6709.319
```

```r
summary(x2)$statistics
```

```
##                   Mean           SD     Naive SE Time-series SE
## mubeta1    8.155365306 0.3603122344 3.289186e-03   3.063226e-03
## mubeta2    0.019886437 0.0024989401 2.281210e-05   2.326968e-05
## sigmabeta1 2.281863860 0.2555192557 2.332561e-03   9.107543e-03
## sigmabeta2 0.015619229 0.0018442580 1.683570e-05   1.896227e-05
## sigmasqy   0.009898141 0.0009978783 9.109341e-06   1.223398e-05
```

```r
summary(x2)$quantiles
```

```
##                   2.5%         25%         50%        75%      97.5%
## mubeta1    7.450848172 7.916140833 8.155707468 8.39519273 8.86460378
## mubeta2    0.014993240 0.018235940 0.019866919 0.02153941 0.02482687
## sigmabeta1 1.851591931 2.100474872 2.257439450 2.43847124 2.84876203
## sigmabeta2 0.012493525 0.014316361 0.015454024 0.01670218 0.01974435
## sigmasqy   0.008131373 0.009193654 0.009835128 0.01051720 0.01201931
```

(iv) Form an approximate 95% central posterior interval for median change factor, and compare it with the previous results.

```r
cat("An approximate 95% central posterior interval for the median change factor is:
    \n (", exp(6 * 0.0150), ",", exp(6 * 0.0248), "). \n")
```

```
## An approximate 95% central posterior interval for the median change factor is:
##
##  ( 1.094174 , 1.160441 ).
```

Comparing with the previous result (1.053586 , 1.205145), the new one has similar but relatively more restrictive interval.

(v) Use the rjags function dic.samples to compute the effective number of parameters ("penalty") and Plummer's DIC ("Penalized deviance"). Use at least 100,000 iterations.

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

```
## Mean deviance:  -499.8
## penalty 81.26
## Penalized deviance: -418.5
```

(vi) Compare the (Plummer's) DIC values for this model and the previous one (-418.3), the conclusion is: These two models have very similar DICs, so it is hard to tell which model would be prefered by only using DICs.