The state dataset of the USA in the 1970s

Hao Zhang

2024-08-27

1. Abstract

This dataset encompasses state-level information from the early 1970s in the USA. This analysis investigates life expectancy across various regions, utilizing a linear model to explore its correlation with other variables.

The North Central region has the highest life expectancy, and the analysis reveals that murder and illiteracy rates have the strongest correlation with it. Income, high school graduation rates, and the mean number of days with a minimum temperature below freezing are also significant factors.

2. Data

2.1 Reading in and manipulating data

The dataset includes state information from the early 1970s in the USA, focusing on various factors such as population, income, illiteracy, life expectancy, murder rates, high school graduation rates, frost days, and area.

state.abb: character vector of 2-letter abbreviations for the state names.

state.region: factor giving the region (Northeast, South, North Central, West) that each state belongs to.

state.x77: matrix with 50 rows and 8 columns giving the following statistics in the respective columns.

- Population: population estimate as of July 1, 1975
- Income: per capita income (1974)
- Illiteracy: illiteracy (1970, percent of population)
- Life Exp: life expectancy in years (1969–71)
- Murder: murder and non-negligent manslaughter rate per 100,000 population (1976)
- HS Grad: percent high-school graduates (1970)
- Frost: mean number of days with minimum temperature below freezing (1931–1960) in capital or large city
- Area: land area in square miles

To better understand the relationship between life expectancy and other variables, we also calculate population density by dividing the population by the area. This enriched dataset allows for a comprehensive analysis of how these factors influence life expectancy across different regions.

```
data("state") # Load data set "state"
dat <- data.frame(state.x77) # Transform matrix into data frame
dat$PopDensity <- dat$Population/dat$Area # Add new variable PopDensity
dat <- cbind(state.abb, dat, state.region) # Combine the three data sets
## Rename columns
colnames(dat)[1] <- "State"
colnames(dat)[11] <- "Region"
colnames(dat)[5] <- "LifeExp"
colnames(dat)[7] <- "HSGrad"
## We will remove some rows from the data set which contain missing values.
dat <- na.omit(dat)

head(dat)
```

```
##            State Population Income Illiteracy LifeExp Murder HSGrad Frost
## Alabama       AL       3615   3624        2.1   69.05   15.1   41.3    20
## Alaska        AK        365   6315        1.5   69.31   11.3   66.7   152
## Arizona       AZ       2212   4530        1.8   70.55    7.8   58.1    15
## Arkansas      AR       2110   3378        1.9   70.66   10.1   39.9    65
## California    CA      21198   5114        1.1   71.71   10.3   62.6    20
## Colorado      CO       2541   4884        0.7   72.06    6.8   63.9   166
##              Area   PopDensity Region
## Alabama     50708 0.0712905261  South
## Alaska     566432 0.0006443845   West
## Arizona    113417 0.0195032491   West
## Arkansas    51945 0.0406198864  South
## California 156361 0.1355708904   West
## Colorado   103766 0.0244877898   West
```

```
str(dat)
```

```
## 'data.frame':    50 obs. of  11 variables:
##  $ State     : chr  "AL" "AK" "AZ" "AR" ...
##  $ Population: num  3615 365 2212 2110 21198 ...
##  $ Income    : num  3624 6315 4530 3378 5114 ...
##  $ Illiteracy: num  2.1 1.5 1.8 1.9 1.1 0.7 1.1 0.9 1.3 2 ...
##  $ LifeExp   : num  69 69.3 70.5 70.7 71.7 ...
##  $ Murder    : num  15.1 11.3 7.8 10.1 10.3 6.8 3.1 6.2 10.7 13.9 ...
##  $ HSGrad    : num  41.3 66.7 58.1 39.9 62.6 63.9 56 54.6 52.6 40.6 ...
##  $ Frost     : num  20 152 15 65 20 166 139 103 11 60 ...
##  $ Area      : num  50708 566432 113417 51945 156361 ...
##  $ PopDensity: num  0.071291 0.000644 0.019503 0.04062 0.135571 ...
##  $ Region    : Factor w/ 4 levels "Northeast","South",..: 2 4 4 2 4 4 1 2 2 2 ...
```

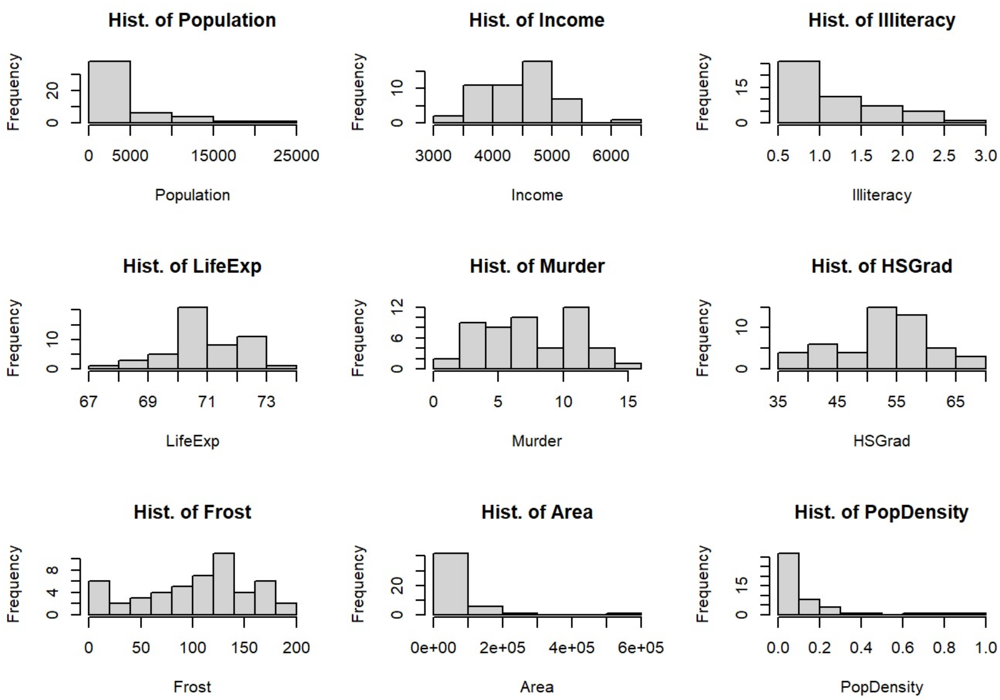
```
summary(dat)
```

```
##     State             Population        Income       Illiteracy   
##  Length:50          Min.   :  365   Min.   :3098   Min.   :0.500  
##  Class :character   1st Qu.: 1080   1st Qu.:3993   1st Qu.:0.625  
##  Mode  :character   Median : 2838   Median :4519   Median :0.950  
##                     Mean   : 4246   Mean   :4436   Mean   :1.170  
##                     3rd Qu.: 4968   3rd Qu.:4814   3rd Qu.:1.575  
##                     Max.   :21198   Max.   :6315   Max.   :2.800  
##     LifeExp          Murder           HSGrad          Frost       
##  Min.   :67.96   Min.   : 1.400   Min.   :37.80   Min.   :  0.00  
##  1st Qu.:70.12   1st Qu.: 4.350   1st Qu.:48.05   1st Qu.: 66.25  
##  Median :70.67   Median : 6.850   Median :53.25   Median :114.50  
##  Mean   :70.88   Mean   : 7.378   Mean   :53.11   Mean   :104.46  
##  3rd Qu.:71.89   3rd Qu.:10.675   3rd Qu.:59.15   3rd Qu.:139.75  
##  Max.   :73.60   Max.   :15.100   Max.   :67.30   Max.   :188.00  
##       Area          PopDensity                 Region  
##  Min.   :  1049   Min.   :0.0006444   Northeast    : 9  
##  1st Qu.: 36985   1st Qu.:0.0253352   South        :16  
##  Median : 54277   Median :0.0730154   North Central:12  
##  Mean   : 70736   Mean   :0.1492245   West         :13  
##  3rd Qu.: 81163   3rd Qu.:0.1442828                     
##  Max.   :566432   Max.   :0.9750033                     
```

2.2 Visualizing data

To begin the analysis, the distributions of the variables are examined through histograms.

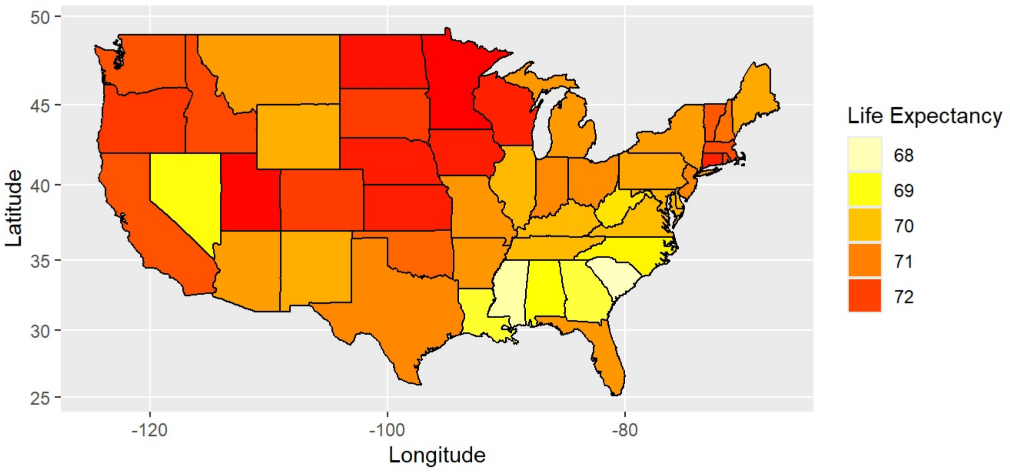
```r
par(mfrow = c(3, 3))
a <- colnames(dat)[2:10]
for (i in 1:length(a)){
 hist(dat[a[i]][,1], main = paste("Hist. of", a[i], sep = " "), xlab = a[i])
}
```



These visualizations reveal that Population, Illiteracy, Area, and Population Density are left-skewed, while other variables are approximately normally distributed.

A choropleth map is also employed to provide a geographical overview of life expectancy across the states, highlighting regions with higher and lower life expectancy and identifying outliers such as Nevada and areas around Massachusetts.

```
library("maps")
library(ggplot2)
dat$region <- tolower(state.name)  # Lowercase states' names
states <- map_data("state")        # Extract state data
map <- merge(states, dat, by = "region", all.x = T)  # Merge states and state.x77 data
map <- map[order(map$order), ]     # Must order first
ggplot(map, aes(x = long, y = lat, group = group)) +
  geom_polygon(aes(fill = LifeExp)) +
  geom_path() +
  scale_fill_gradientn(colours = rev(heat.colors(10))) +
  coord_map() +
  labs(x = "Longitude", y = "Latitude") +
  guides(fill = guide_legend(title = "Life Expectancy"))
```



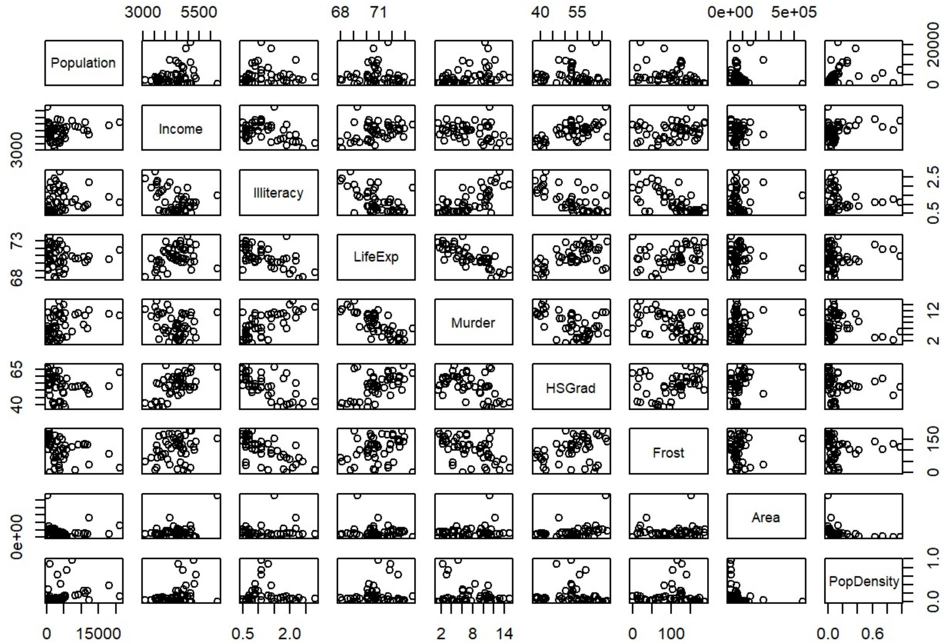
The map shows that the top and left are close to red while the bottom right and left are close to yellow, which means that life expectancies are higher in northcentral and northwest states but less in south and east states.

There are also two exceptions: Nevada in the west, which has a low life expectancy, and areas around Massachusetts in the east, which have a high life expectancy which the other variables, such as income, illiteracy, and murder rate, should explain.

2.3 Analyzing the relationship among variables

Let's look at pairwise scatter plots of the nine variables.

```
pairs(dat[,2:10])
```



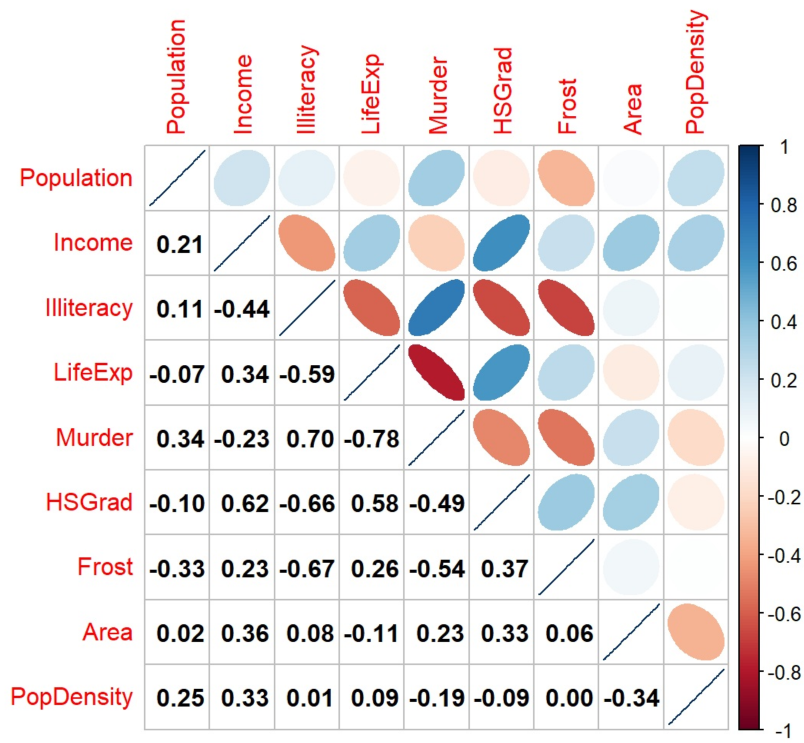
The analysis of pairwise scatter plots indicates strong correlations among several variables. For instance, Illiteracy and high school graduation rates display a close linear relationship.

Such collinearity can pose challenges in linear regression models, leading to unstable estimates. To address this, we formally estimate the correlations using the corrplot package, which quantifies the degree of correlation between variables.

```
library("corrplot")
```

```
## corrplot 0.94 loaded
```

```
Cor = cor(as.matrix(dat[, 2:10])) # Calculate correlation matrix
corrplot(Cor, type="upper", method="ellipse", tl.pos="lt")
corrplot(Cor, type="lower", method="number", col="black",
         add=TRUE, diag=FALSE, tl.pos="n", cl.pos="n") # Mix plots of "number" and "ellipse"
```



We can see that LifeExp has a high negative correlation with Murder, a mild negative correlation with Illiteracy, a mild positive correlation with HSGrad, and a slight positive correlation with Income and Frost. In addition, Illiteracy has a high negative correlation with HSGrad and Frost and a high positive correlation with Murder.

So, in our cases, if we want to use the linear model to discover how the other variables correlate to LifeExp, one option is to choose Income and Illiteracy as explanatory variables. In sections 3.1 and 3.2, we choose a more rigorous method for selecting explanatory variables.

3. Linear model analysis

In this section, we aim to identify the variables most closely associated with life expectancy. The analysis begins with variable selection, using a linear model where the coefficients are standardized. This approach helps determine the significance of each variable in predicting life expectancy. Two models are constructed: the first includes a broader set of variables, while the second refines the selection based on the results from the initial model. Model comparison using the Deviance Information Criterion (DIC) suggests that the second model, which focuses on murder rates, high school graduation rates, and frost days, provides a better fit for the data.

3.1 Variable selection

One primary goal of this analysis is to find out which variables are related to the presence of life expectancy. This objective is often called “variable selection.” One way to do this is to use a linear model where the priors for the coefficients in the linear equation favour values near 0 (indicating a weak relationship). This way, the burden of establishing an association lies with the data. We assume it doesn’t exist if there is no strong signal.

Rather than tailoring a prior for each coefficient based on the scale its covariate takes values on, it is customary to subtract the mean and divide by the standard deviation for each variable.

```
X = scale(dat[,c(3,4,6,7,8,10)], center=TRUE, scale=TRUE) # Normalize the explanatory variables
head(X)
```

```
##                Income Illiteracy     Murder     HSGrad      Frost  PopDensity
## Alabama    -1.3211387   1.525758  2.0918101 -1.4619293 -1.6248292 -0.35263218
## Alaska      3.0582456   0.541398  1.0624293  1.6828035  0.9145676 -0.67228881
## Arizona     0.1533029   1.033578  0.1143154  0.6180514 -1.7210185 -0.58695703
## Arkansas   -1.7214837   1.197638  0.7373617 -1.6352611 -0.7591257 -0.49140937
## California  1.1037155  -0.114842  0.7915396  1.1751891 -1.6248292 -0.06177915
## Colorado    0.7294092  -0.771082 -0.1565742  1.3361400  1.1838976 -0.56440319
```

```
colMeans(X) # Mean of the normalized explanatory variables
```

```
##        Income    Illiteracy        Murder        HSGrad         Frost
## -3.005235e-16  1.165734e-16 -3.186687e-17  3.807718e-16  1.099121e-16
##    PopDensity
##  6.383782e-18
```

```
apply(X, 2, sd) # Standard deviation of the normalized explanatory variables
```

```
##     Income Illiteracy     Murder     HSGrad      Frost PopDensity
##          1          1          1          1          1          1
```

3.2 The first model

We’ll apply the linear hierarchical model to this data set on life expectancy, incorporating the region variable to select explanatory variables.

```
library("coda")  # Loading required package: coda
library("rjags") # Linked to JAGS
```

```
## Linked to JAGS 4.3.1
```

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

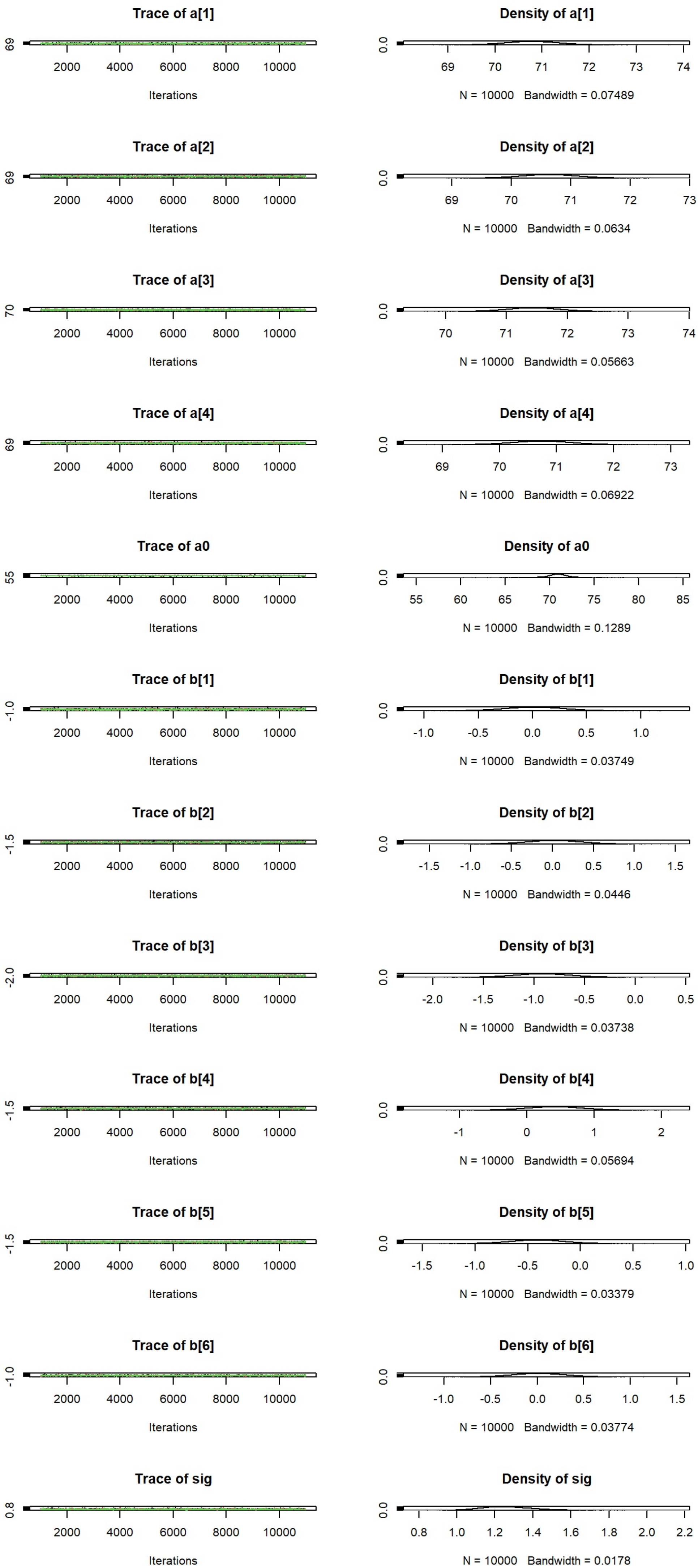
```
mod1_string = " model {
   for (i in 1:length(y)) {
    y[i] ~ dnorm(mu[i], prec)
    mu[i] =  a[Region[i]] + b[1]*Income[i] + b[2]*Illiteracy[i] + b[3]*Murder[i]
        + b[4]*HSGrad[i] + b[5]*Frost[i] + b[6]*PopDensity[i]
   }

   for (j in 1:max(Region)) {
    a[j] ~ dnorm(a0, prec_a)
   }

   a0 ~ dnorm(70, 1.0/1.0e6)
   prec_a ~ dgamma(1/2.0, 1*10.0/2.0)
   tau = sqrt( 1.0 / prec_a )

   for (j in 1:6) {
        b[j] ~ dnorm(0.0, 1.0)
   }

   prec ~ dgamma(5/2.0, 5*10.0/2.0)
   sig = sqrt( 1.0 / prec )
} "

set.seed(32)
data1_jags = list(y=dat$LifeExp, Income=X[,"Income"],Illiteracy=X[,"Illiteracy"],
                 Murder=X[,"Murder"],HSGrad=X[,"HSGrad"],Frost=X[,"Frost"],
                 PopDensity=X[,"PopDensity"], Region=as.numeric(dat$Region))

mod1 = jags.model(textConnection(mod1_string), data=data1_jags, n.chains=3)
```

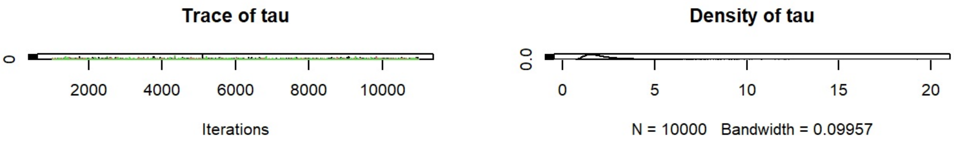
```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 50
##    Unobserved stochastic nodes: 13
##    Total graph size: 735
##
## Initializing model
```

```
update(mod1, 1000) # Burn-in

params1 = c("a0", "a", "b", "sig", "tau")
mod1_sim = coda.samples(model=mod1, variable.names=params1, n.iter=1e4)

mod1_csim = as.mcmc(do.call(rbind, mod1_sim)) # Combine multiple chains
```

Before we check the inferences from the model, we perform convergence diagnostics for our Markov chains.

```
plot(mod1_sim)
```


Trace of a[1]
Iterations
Density of a[1]
N = 10000 Bandwidth = 0.07489
Trace of a[2]
Iterations
Density of a[2]
N = 10000 Bandwidth = 0.0634
Trace of a[3]
Iterations
Density of a[3]
N = 10000 Bandwidth = 0.05663
Trace of a[4]
Iterations
Density of a[4]
N = 10000 Bandwidth = 0.06922
Trace of a0
Iterations
Density of a0
N = 10000 Bandwidth = 0.1289
Trace of b[1]
Iterations
Density of b[1]
N = 10000 Bandwidth = 0.03749
Trace of b[2]
Iterations
Density of b[2]
N = 10000 Bandwidth = 0.0446
Trace of b[3]
Iterations
Density of b[3]
N = 10000 Bandwidth = 0.03738
Trace of b[4]
Iterations
Density of b[4]
N = 10000 Bandwidth = 0.05694
Trace of b[5]
Iterations
Density of b[5]
N = 10000 Bandwidth = 0.03379
Trace of b[6]
Iterations
Density of b[6]
N = 10000 Bandwidth = 0.03774
Trace of sig
Iterations
Density of sig
N = 10000 Bandwidth = 0.0178



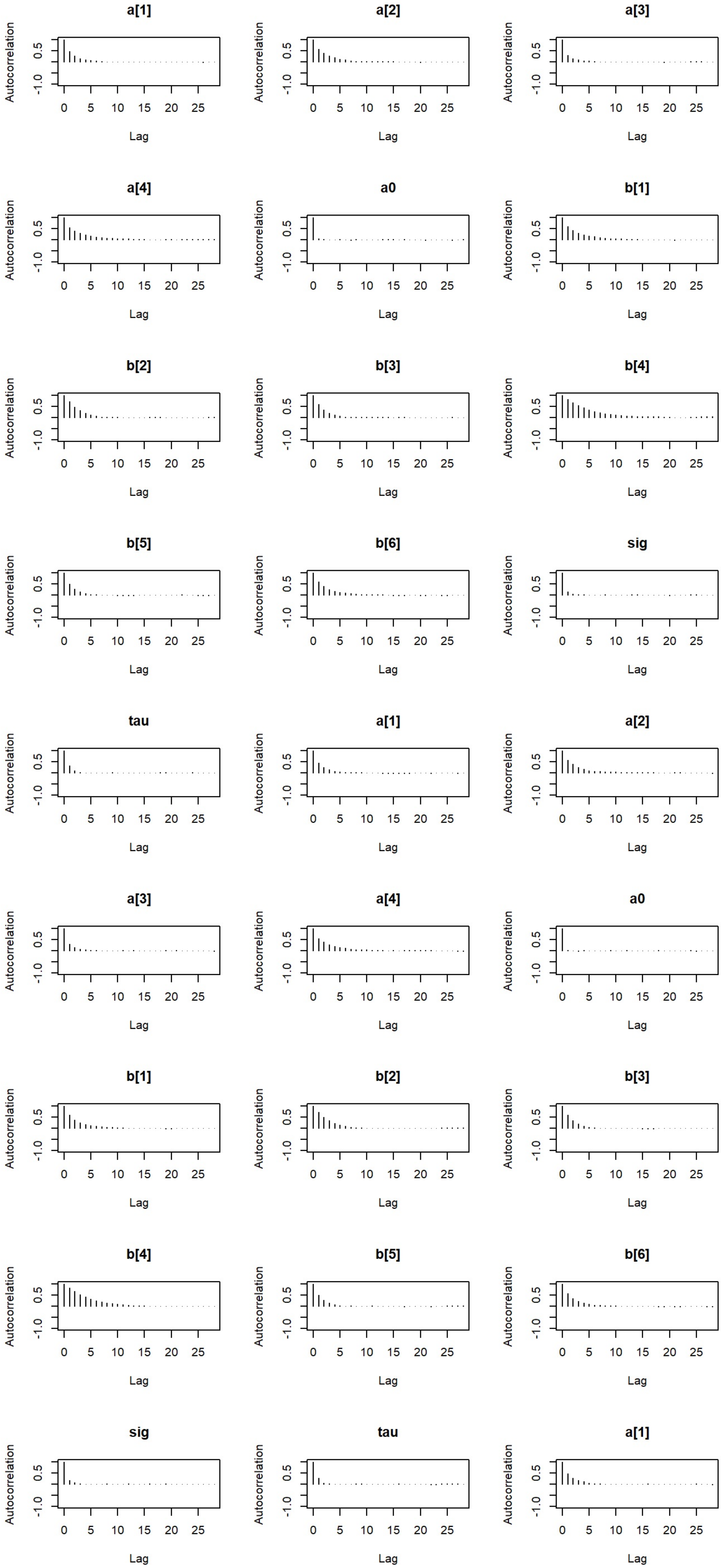

```
gelman.diag(mod1_sim)
```

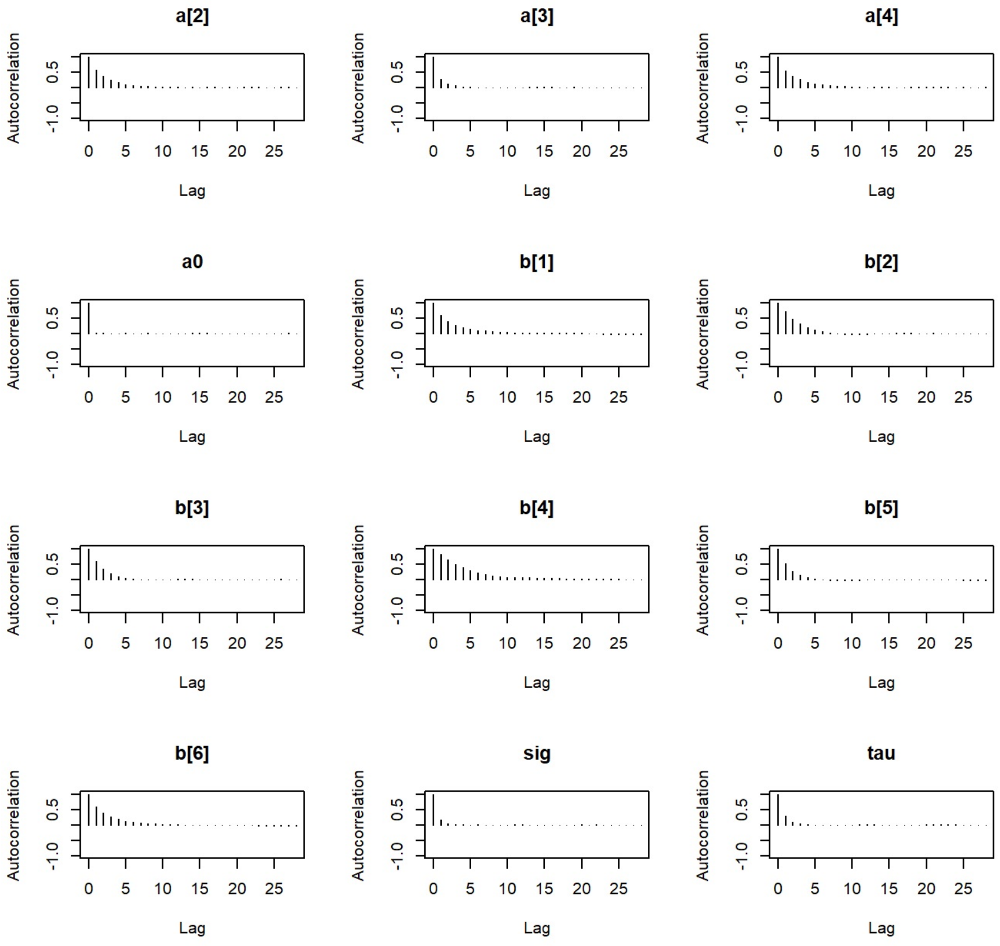
```
## Potential scale reduction factors:
##
##      Point est. Upper C.I.
## a[1]          1       1.00
## a[2]          1       1.00
## a[3]          1       1.00
## a[4]          1       1.00
## a0            1       1.00
## b[1]          1       1.00
## b[2]          1       1.00
## b[3]          1       1.00
## b[4]          1       1.00
## b[5]          1       1.00
## b[6]          1       1.01
## sig           1       1.00
## tau           1       1.00
##
## Multivariate psrf
##
## 1
```

```
autocorr.diag(mod1_sim)
```

```
##                a[1]         a[2]         a[3]         a[4]            a0
## Lag 0   1.000000000  1.000000000  1.000000000  1.000000000  1.0000000000
## Lag 1   0.464206740  0.568609177  0.284015222  0.547862236  0.0244351636
## Lag 5   0.050355843  0.114002493  0.031945826  0.147138690  0.0172870930
## Lag 10 -0.009492406  0.021987268 -0.008389121  0.040497948 -0.0009016396
## Lag 50  0.002918229 -0.002158828 -0.002889233 -0.004761976 -0.0019262910
##                b[1]         b[2]        b[3]       b[4]         b[5]
## Lag 0  1.0000000000  1.000000000 1.000000000 1.00000000  1.000000000
## Lag 1  0.5942637385  0.718867436 0.589881786 0.81823118  0.498096418
## Lag 5  0.1446550784  0.125696218 0.051579059 0.32205911  0.024125644
## Lag 10 0.0346812868 -0.006730182 0.003598046 0.09634618 -0.010661853
## Lag 50 0.0009906403  0.005963785 0.002427121 0.00389652 -0.006166992
##                b[6]           sig           tau
## Lag 0   1.000000000  1.0000000000  1.0000000000
## Lag 1   0.586032232  0.1583619876  0.2949548952
## Lag 5   0.114713425 -0.0018944180 -0.0009553241
## Lag 10  0.017717686 -0.0028250914 -0.0018056583
## Lag 50 -0.003908673  0.0001367124 -0.0006883706
```

```
autocorr.plot(mod1_sim)
```


a[1]
a[2]
a[3]
a[4]
a0
b[1]
b[2]
b[3]
b[4]
b[5]
b[6]
sig
tau
a[1]
a[2]
a[3]
a[4]
a0
b[1]
b[2]
b[3]
b[4]
b[5]
b[6]
sig
tau
a[1]
Autocorrelation
Lag




```
effectiveSize(mod1_sim)
```

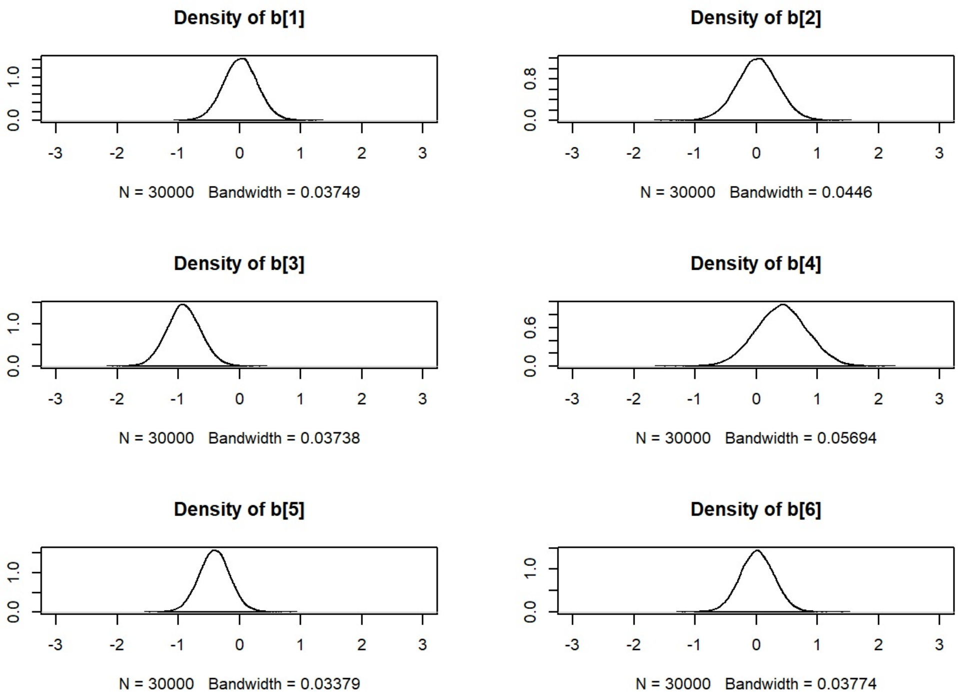
```
##      a[1]      a[2]      a[3]      a[4]        a0      b[1]      b[2]      b[3]
##  9417.125  6711.861 13748.716  6145.775 28958.910  6237.915  5775.663  8164.950
##      b[4]      b[5]      b[6]       sig       tau
##  3373.810 10131.464  7002.110 20311.309 16567.927
```

It looks like our model converges well. So, we can use the following posterior summary of the parameters in our model to get an inference.

```
summary(mod1_sim)
```

```
##
## Iterations = 1001:11000
## Thinning interval = 1
## Number of chains = 3
## Sample size per chain = 10000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##          Mean     SD Naive SE Time-series SE
## a[1] 70.77597 0.5622 0.003246      0.0057990
## a[2] 70.63683 0.4754 0.002745      0.0058068
## a[3] 71.43519 0.4245 0.002451      0.0036247
## a[4] 70.73665 0.5132 0.002963      0.0065637
## a0   70.88787 1.1955 0.006902      0.0070492
## b[1]  0.01588 0.2780 0.001605      0.0035382
## b[2]  0.01611 0.3351 0.001935      0.0044117
## b[3] -0.91445 0.2797 0.001615      0.0030961
## b[4]  0.41019 0.4230 0.002442      0.0073110
## b[5] -0.41214 0.2535 0.001463      0.0025203
## b[6]  0.01352 0.2798 0.001616      0.0033484
## sig   1.26795 0.1335 0.000771      0.0009374
## tau   2.09658 1.0535 0.006082      0.0082164
##
## 2. Quantiles for each variable:
##
##         2.5%     25%      50%     75%    97.5%
## a[1] 69.6673 70.4060 70.77680 71.1501 71.88275
## a[2] 69.7077 70.3206 70.63525 70.9506 71.57527
## a[3] 70.6060 71.1532 71.43406 71.7159 72.27065
## a[4] 69.7323 70.3920 70.73652 71.0797 71.74920
## a0   68.5169 70.2501 70.88540 71.5309 73.23582
## b[1] -0.5269 -0.1712  0.01566  0.2013  0.56652
## b[2] -0.6539 -0.2040  0.01702  0.2391  0.67232
## b[3] -1.4593 -1.1001 -0.91683 -0.7287 -0.35898
## b[4] -0.4178  0.1256  0.41242  0.6914  1.23932
## b[5] -0.9097 -0.5798 -0.41224 -0.2440  0.08638
## b[6] -0.5358 -0.1748  0.01314  0.2012  0.56101
## sig   1.0406  1.1738  1.25675  1.3506  1.56066
## tau   0.9940  1.4404  1.82556  2.4297  4.84841
```

```
par(mfrow=c(3,2))
densplot(mod1_csim[,6:11], xlim=c(-3.0, 3.0))
```




```
colnames(X)  # Variable names
```

```
## [1] "Income"     "Illiteracy" "Murder"     "HSGrad"     "Frost"
## [6] "PopDensity"
```

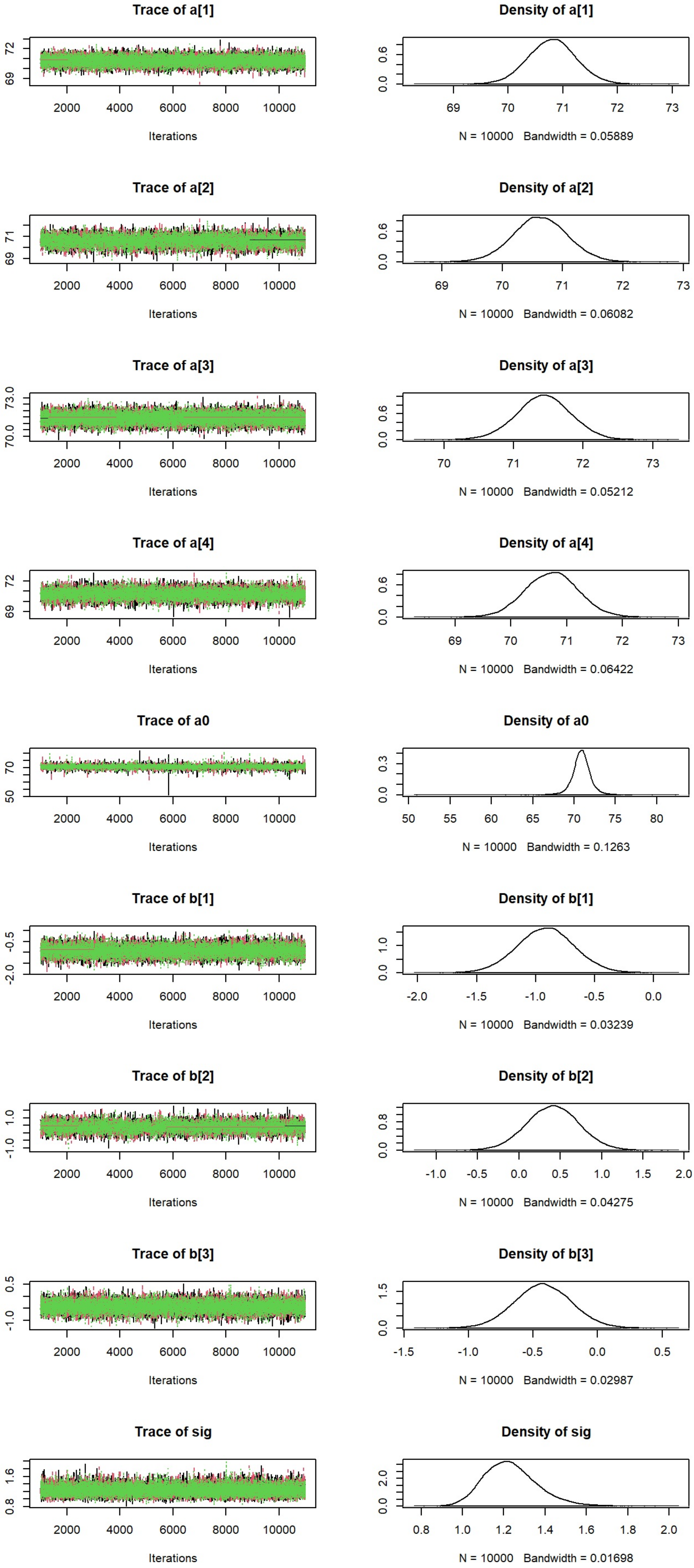
It is clear that the coefficients for variables Murder, HSGrad, and Frost are not 0. The posterior distribution for the coefficient of Income, Illiteracy and PopDensity is almost centred on 0, which we choose to drop out. So we choose Murder, HSGrad and Frost as the explanatory variables for our second model.

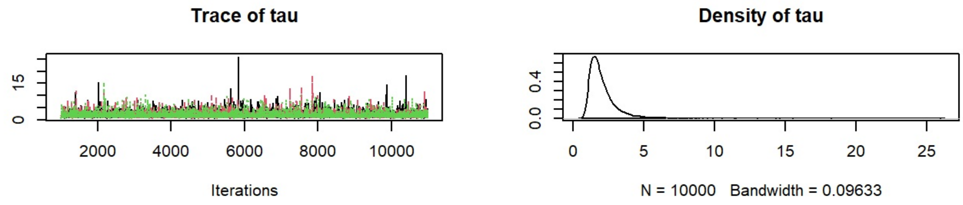
3.3 The second model

```
mod2_string = " model {
   for (i in 1:length(y)) {
    y[i] ~ dnorm(mu[i], prec)
    mu[i] =  a[Region[i]] + b[1]*Murder[i] + b[2]*HSGrad[i] + b[3]*Frost[i]
   }

   for (j in 1:max(Region)) {
    a[j] ~ dnorm(a0, prec_a)
   }

   a0 ~ dnorm(70, 1.0/1.0e6)
   prec_a ~ dgamma(1/2.0, 1*10.0/2.0)
   tau = sqrt( 1.0 / prec_a )

   for (j in 1:3) {
        b[j] ~ dnorm(0.0, 1.0)
   }

   prec ~ dgamma(5/2.0, 5*10.0/2.0)
   sig = sqrt( 1.0 / prec )
} "

set.seed(32)
data2_jags = list(y=dat$LifeExp, Murder=X[,"Murder"],HSGrad=X[,"HSGrad"],
                 Frost=X[,"Frost"], Region=as.numeric(dat$Region))

mod2 = jags.model(textConnection(mod2_string), data=data2_jags, n.chains=3)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 50
##    Unobserved stochastic nodes: 10
##    Total graph size: 462
##
## Initializing model
```

```
update(mod2, 1000) # Burn-in

params2 = c("a0", "a", "b", "sig", "tau")
mod2_sim = coda.samples(model=mod2, variable.names=params2, n.iter=1e4)

mod2_csim = as.mcmc(do.call(rbind, mod2_sim)) # Combine multiple chains

plot(mod2_sim)
```


Trace of a[1]
Iterations
Density of a[1]
N = 10000 Bandwidth = 0.05889
Trace of a[2]
Iterations
Density of a[2]
N = 10000 Bandwidth = 0.06082
Trace of a[3]
Iterations
Density of a[3]
N = 10000 Bandwidth = 0.05212
Trace of a[4]
Iterations
Density of a[4]
N = 10000 Bandwidth = 0.06422
Trace of a0
Iterations
Density of a0
N = 10000 Bandwidth = 0.1263
Trace of b[1]
Iterations
Density of b[1]
N = 10000 Bandwidth = 0.03239
Trace of b[2]
Iterations
Density of b[2]
N = 10000 Bandwidth = 0.04275
Trace of b[3]
Iterations
Density of b[3]
N = 10000 Bandwidth = 0.02987
Trace of sig
Iterations
Density of sig
N = 10000 Bandwidth = 0.01698



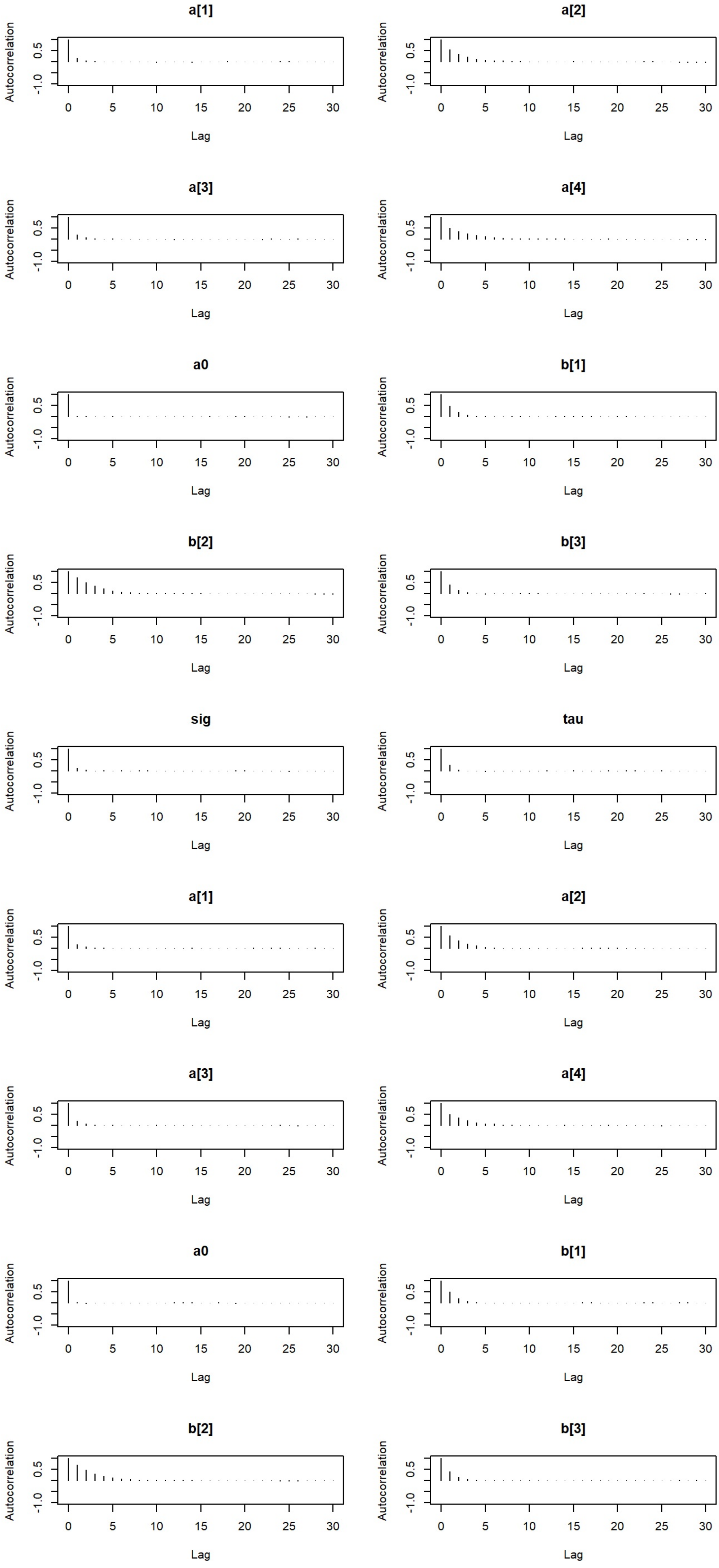

```
gelman.diag(mod2_sim)
```

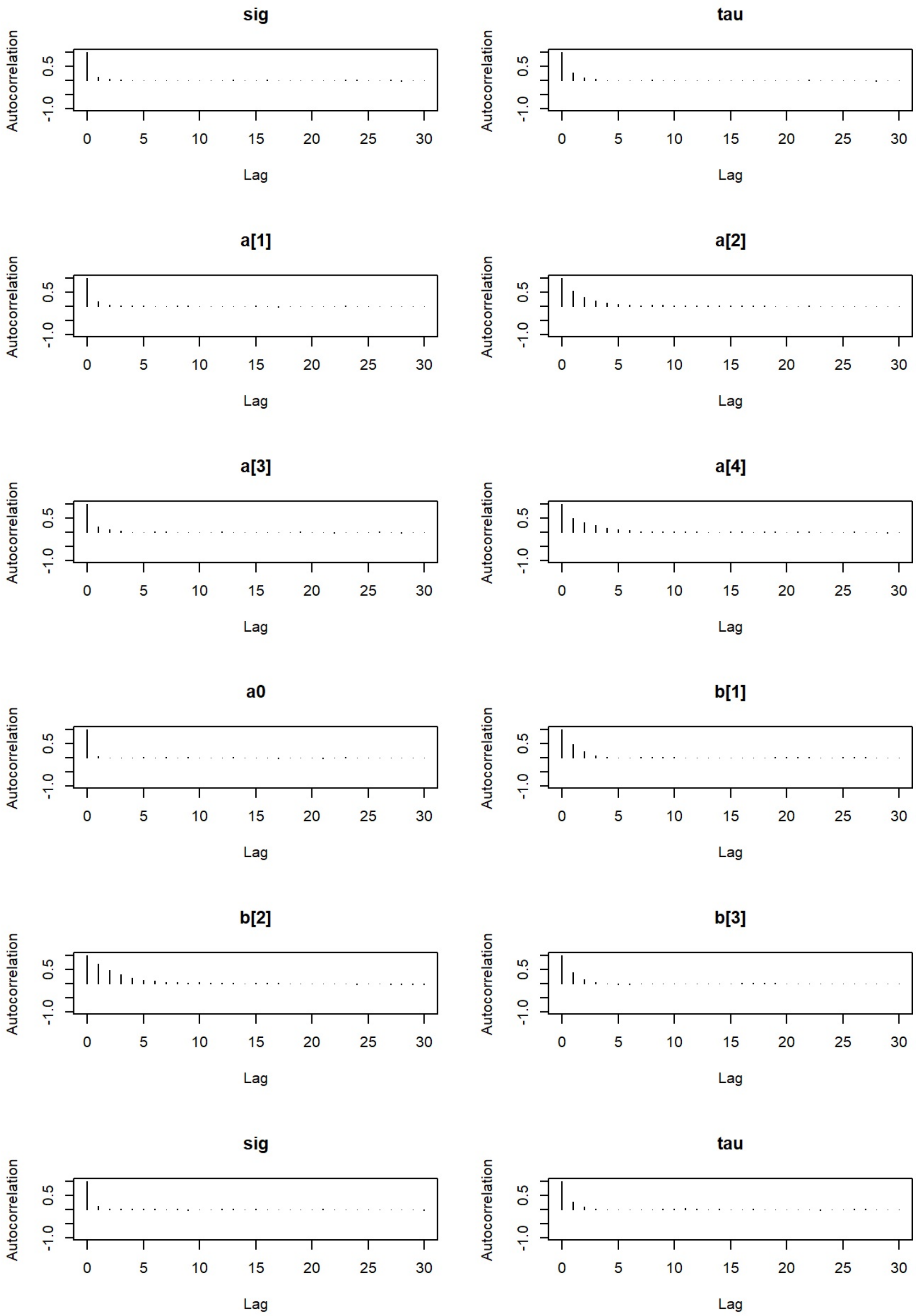
```
## Potential scale reduction factors:
##
##       Point est. Upper C.I.
## a[1]          1          1
## a[2]          1          1
## a[3]          1          1
## a[4]          1          1
## a0            1          1
## b[1]          1          1
## b[2]          1          1
## b[3]          1          1
## sig           1          1
## tau           1          1
##
## Multivariate psrf
##
## 1
```

```
autocorr.diag(mod2_sim)
```

```
##                 a[1]         a[2]         a[3]         a[4]           a0
## Lag 0   1.000000000  1.000000000  1.000000000  1.000000000  1.000000000
## Lag 1   0.173093927  0.551983493  0.191467440  0.500458939  0.024593794
## Lag 5   0.007001683  0.063844896  0.007925891  0.095316398  0.010309581
## Lag 10 -0.009702144  0.010276168  0.001928454  0.012226606 -0.005373661
## Lag 50  0.014069538 -0.001613744 -0.003950655 -0.006101309  0.004062713
##                 b[1]         b[2]         b[3]          sig         tau
## Lag 0   1.000000000  1.000000000  1.000000000  1.0000000000  1.00000000
## Lag 1   0.477214438  0.698297979  0.384128911  0.1135794656  0.27333756
## Lag 5   0.010281396  0.127243126 -0.013292707 -0.0031584097 -0.01037119
## Lag 10  0.007745621  0.021397219  0.006097074  0.0013806923  0.00917921
## Lag 50 -0.006063727 -0.005921175 -0.006718882  0.0006074689 -0.01243647
```

```
autocorr.plot(mod2_sim)
```


a[1]
a[2]
a[3]
a[4]
a0
b[1]
b[2]
b[3]
sig
tau
a[1]
a[2]
a[3]
a[4]
a0
b[1]
b[2]
b[3]
Autocorrelation
-1.0
0.5
Lag
0
5
10
15
20
25
30




```
effectiveSize(mod2_sim)
```

```
##      a[1]      a[2]      a[3]      a[4]        a0      b[1]      b[2]      b[3] 
## 19781.980  8129.579 18624.047  7544.095 29739.607 11387.485  5849.688 14309.304 
##       sig       tau 
## 22145.146 17097.114
```

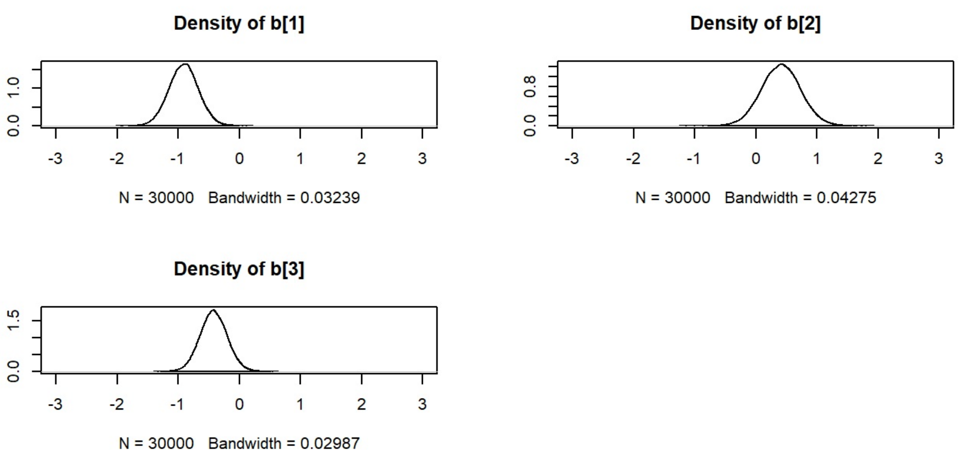
Our second model also converges well.

```
summary(mod2_sim)
```

```
##
## Iterations = 1001:11000
## Thinning interval = 1
## Number of chains = 3
## Sample size per chain = 10000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##          Mean     SD  Naive SE Time-series SE
## a[1] 70.8014 0.4386 0.0025323      0.0031192
## a[2] 70.6276 0.4538 0.0026199      0.0050345
## a[3] 71.4304 0.3911 0.0022582      0.0028692
## a[4] 70.7327 0.4762 0.0027494      0.0054893
## a0   70.8932 1.1682 0.0067446      0.0067868
## b[1] -0.9134 0.2409 0.0013907      0.0022570
## b[2]  0.4092 0.3170 0.0018302      0.0041462
## b[3] -0.4229 0.2226 0.0012853      0.0018627
## sig   1.2338 0.1287 0.0007432      0.0008651
## tau   2.0765 1.0416 0.0060137      0.0079674
##
## 2. Quantiles for each variable:
##
##          2.5%     25%     50%     75%    97.5%
## a[1] 69.9444 70.5074 70.8040 71.0925 71.66813
## a[2] 69.7374 70.3260 70.6257 70.9303 71.52274
## a[3] 70.6530 71.1721 71.4306 71.6900 72.19714
## a[4] 69.8092 70.4104 70.7355 71.0505 71.67061
## a0   68.5883 70.2644 70.8911 71.5197 73.21183
## b[1] -1.3844 -1.0751 -0.9132 -0.7533 -0.43921
## b[2] -0.2109  0.1964  0.4096  0.6218  1.03048
## b[3] -0.8577 -0.5715 -0.4245 -0.2747  0.01562
## sig   1.0115  1.1431  1.2232  1.3118  1.51783
## tau   0.9956  1.4325  1.8119  2.3897  4.78472
```

```
par(mfrow=c(3,2))
densplot(mod2_csim[,6:8], xlim=c(-3.0, 3.0))
colnames(X)[c(3,4,5)]  # Variable names
```

```
## [1] "Murder" "HSGrad" "Frost"
```



Then, we use a quantity known as the deviance information criterion (DIC), which essentially calculates the posterior mean of the log-likelihood and adds a penalty for model complexity. Let’s compare these two models using DIC.

```
dic.samples(mod1, n.iter=1e3)
```

```
## Mean deviance:  137.7
## penalty 10.45
## Penalized deviance: 148.1
```

```
dic.samples(mod2, n.iter=1e3)
```

```
## Mean deviance:  133.7
## penalty 7.894
## Penalized deviance: 141.6
```

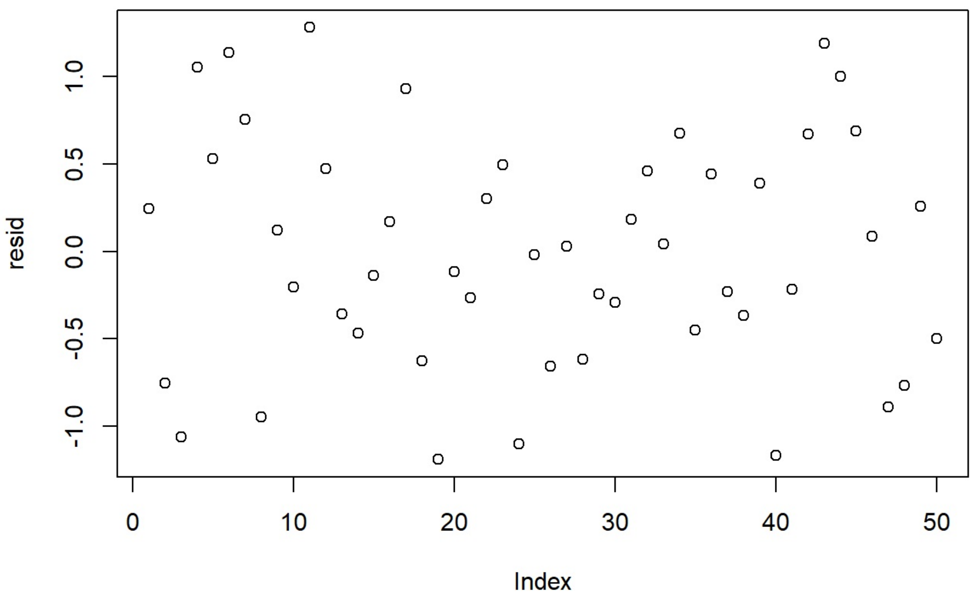
The second model has a better DIC, so we chose it to infer the life expectancy.

Then, we check our second model's residuals (the difference between LifeExp and the model's prediction for that value). This is important with linear models since residuals can reveal violations of the assumptions we made to specify the model. In particular, we are looking for any sign that the model is not linear or normally distributed or that the observations are not independent (conditional on covariates).

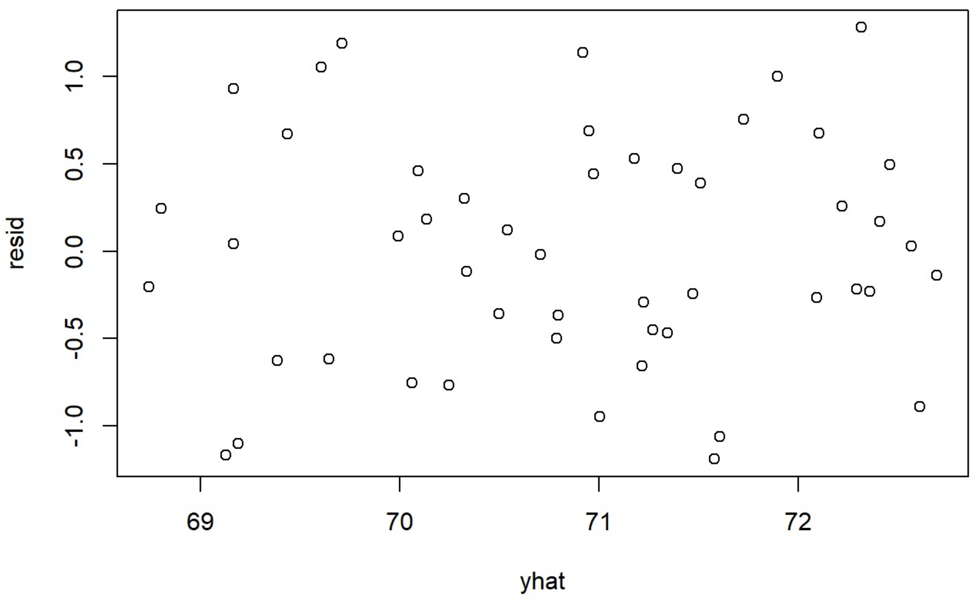
```
pm2_params = colMeans(mod2_csim) # Posterior mean

yhat=X[,c(3,4,5)] %*%  pm2_params[6:8]
for (i in 1: nrow(X)){
  yhat[i] = yhat[i] + pm2_params[as.numeric(dat[i,"Region"])]
}

resid = data2_jags$y - yhat

mean(resid) # Mean of residuals
```

```
## [1] -0.0002714809
```

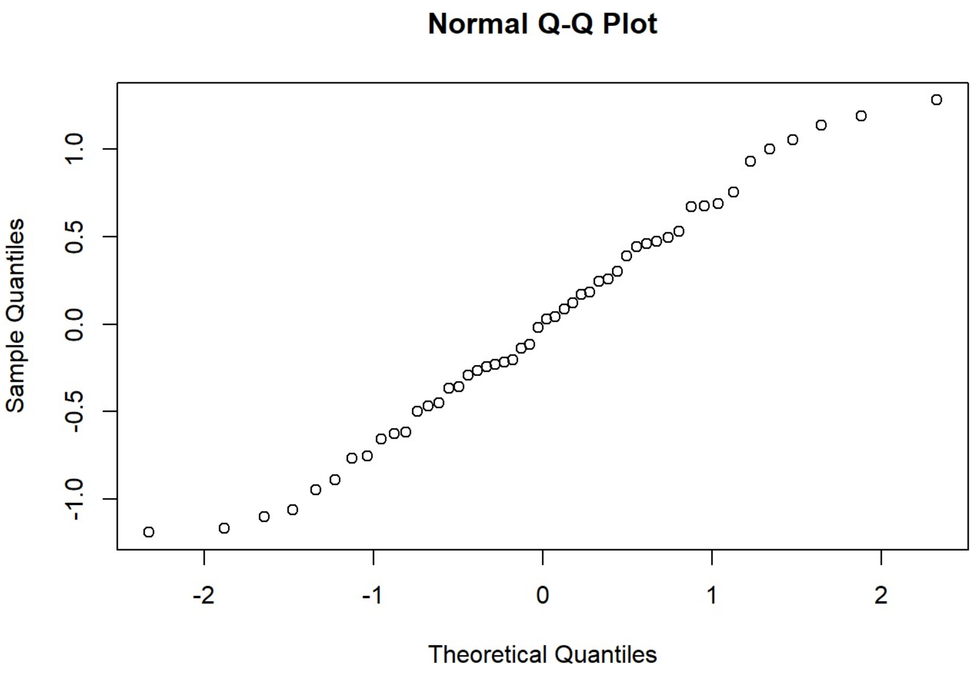
```
plot(resid) # Against data index
```




```
plot(yhat, resid) # Against predicted values
```




```r
qqnorm(resid) # Checking normality of residuals
```



The residuals look pretty good here (no patterns or shapes). Together with the correlations in section 2.3, we have the following conclusion.

4. Conclusion

The analysis concludes that murder rates and illiteracy are the most strongly correlated with life expectancy, followed by income, high school graduation rates, and frost days. Regional differences also play a significant role, with the North Central region exhibiting the highest average life expectancy.