Homework 3 Suglia

Elena Suglia

11/17/2019

Question 1

Using the "litterbags.csv" data set, create and run a model in JAGS that corresponds to this one:

```
lmer(N_min_rate~Celastrus + (1|Plot), data=litterbags)
```

This data set contains nitrogen mineralization rates at 7 different plots where the treatment was whether or not the invasive liana Celastrus orbiculatus was present or not.

In your answer, please include the model code and the means and standard deviations of the intercept, slope, group-level (i.e. among-plot) variance and individual-level (within-plot) variance. Briefly report what you did to check model convergence.

Model code in text file

```
 model {

   # Likelihood
  for (i in 1:n) {
    y[i] ~ dnorm(y.hat[i], tau.y)
    y.hat[i] <- a[Plot[i]] + b*x[i] # regression with intercept for each plot
  }

  # group-level model
  for (j in 1:7) { # 7 plots
    a[j] ~ dnorm(mu.a, tau.a) # separate intercepts for every plot, no pooling of information
  }

  #priors
  mu.a ~ dnorm(0, 0.1) # overall mean
  sigma.a ~ dnorm(0, 0.5)T(0,) # weakly informative prior on the group-level standard deviation
  tau.a <- pow(sigma.a, -2) # group-level precision

  # individual-level priors
  b ~ dnorm(0, 0.1) #  prior for slope
  sigma.y ~ dnorm(0, 0.1)T(0,) # weakly informative prior on the individual-level sd
  tau.y <- pow(sigma.y, -2) # individual-level precision
}
```

Read in data

```
d = read_csv("litterbags.csv")
```

Define data needed for model

```r
length(unique(d$Plot)) # 7 plots
d2.data <- list(n=nrow(d), y=d$N_min_rate, x=d$Celastrus,
plot.index=d$Plot, n.plots=length(unique(d$Plot)))
```

Have jags generate starting values

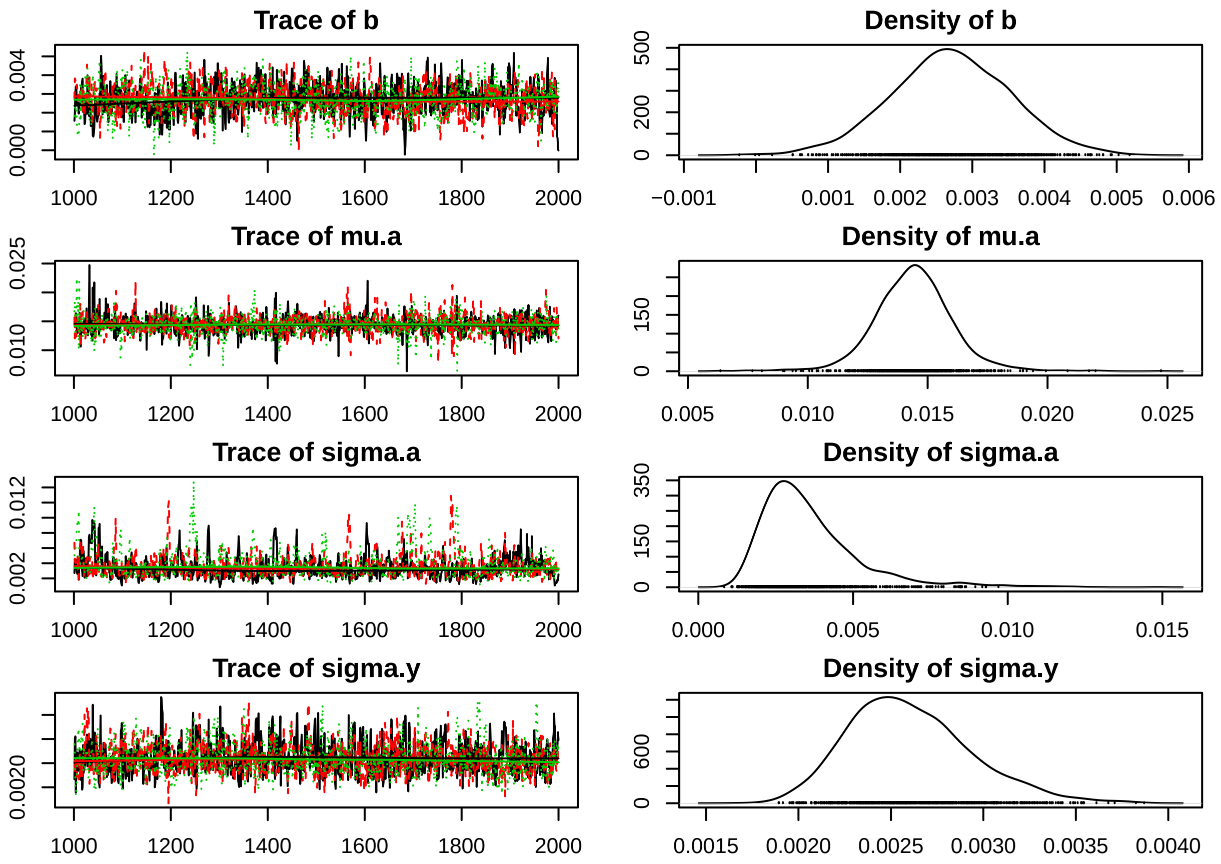
```r
d2.inits <- list(list(sigma.a = 1, sigma.y=2), list(sigma.a = 2,
sigma.y=1), list(sigma.a = 5, sigma.y=0.4))
d2 <- jags.model("litterbags_jags_hw3.txt", data=d2.data, inits = d2.inits,
n.chains=3, n.adapt=1000)
```

Do a trial run with some parameters, and check convergence

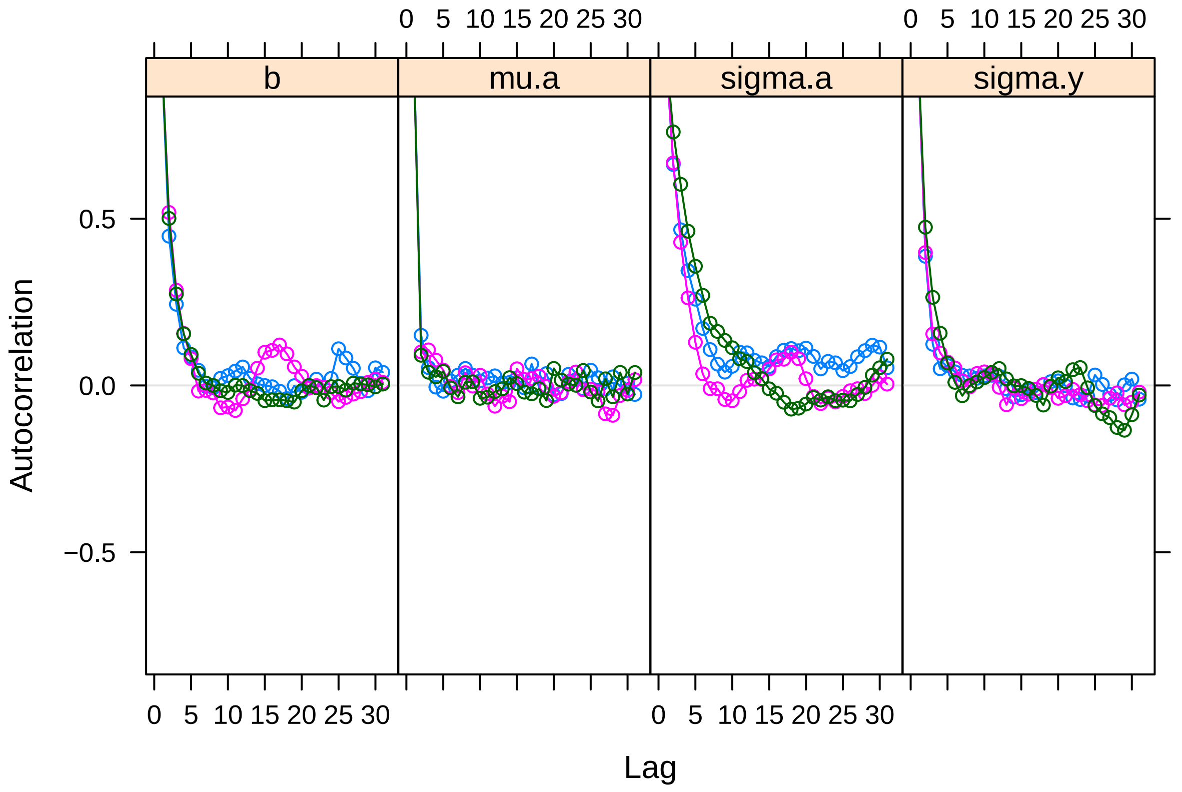
```r
d2.samp <- coda.samples(d2, c("b", "sigma.y", "mu.a", "sigma.a"), n.iter=1000)
summary(d2.samp)
```

```
##
## Iterations = 1001:2000
## Thinning interval = 1
## Number of chains = 3
## Sample size per chain = 1000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##              Mean        SD  Naive SE Time-series SE
## b       0.002702 0.0008266 1.509e-05      2.628e-05
## mu.a    0.014418 0.0016406 2.995e-05      3.550e-05
## sigma.a 0.003610 0.0016476 3.008e-05      7.688e-05
## sigma.y 0.002603 0.0003317 6.055e-06      9.707e-06
##
## 2. Quantiles for each variable:
##
##             2.5%      25%      50%      75%    97.5%
## b       0.001023 0.002165 0.002697 0.003268 0.004338
## mu.a    0.011286 0.013450 0.014412 0.015348 0.017777
## sigma.a 0.001607 0.002520 0.003213 0.004243 0.008381
## sigma.y 0.002041 0.002364 0.002569 0.002801 0.003336
```

```r
par(mar=rep(2, 4))
plot(d2.samp)
```



```r
acfplot(d2.samp)
```



```
gelman.diag(d2.samp)
```

```
## Potential scale reduction factors:
##
##         Point est. Upper C.I.
## b             1.00       1.00
## mu.a          1.00       1.00
## sigma.a       1.03       1.07
## sigma.y       1.00       1.01
##
## Multivariate psrf
##
## 1.01
```

Add more iterations and include all parameters of interest

```
d2.samp <- coda.samples(d2, c("a", "b", "sigma.y", "mu.a", "sigma.a", "y.hat"), n.iter=5000, thin=5)
```

Look at the results

```
d2.summ <- summary(d2.samp)
d2.stats <- as.data.frame(d2.summ$statistics)
head(d2.stats, 100)
```

```
##                    Mean           SD     Naive SE Time-series SE
## a[1]        0.018094452 0.0011581855 2.114548e-05   2.114688e-05
## a[2]        0.012893579 0.0010864376 1.983555e-05   1.981748e-05
## a[3]        0.017677102 0.0012783733 2.333980e-05   2.273521e-05
## a[4]        0.012115220 0.0011581635 2.114508e-05   2.081226e-05
## a[5]        0.014956385 0.0011040788 2.015763e-05   2.144425e-05
## a[6]        0.012622608 0.0010816347 1.974786e-05   2.019552e-05
## a[7]        0.012773676 0.0011091395 2.025002e-05   1.948926e-05
## b           0.002665172 0.0008477275 1.547732e-05   1.615134e-05
## mu.a        0.014433530 0.0015623524 2.852452e-05   2.852923e-05
## sigma.a     0.003491457 0.0015403687 2.812316e-05   3.243161e-05
## sigma.y     0.002616677 0.0003394741 6.197921e-06   6.332734e-06
## y.hat[1]    0.020759624 0.0011665434 2.129807e-05   2.063974e-05
## y.hat[2]    0.020759624 0.0011665434 2.129807e-05   2.063974e-05
## y.hat[3]    0.020759624 0.0011665434 2.129807e-05   2.063974e-05
## y.hat[4]    0.018094452 0.0011581855 2.114548e-05   2.114688e-05
## y.hat[5]    0.018094452 0.0011581855 2.114548e-05   2.114688e-05
## y.hat[6]    0.018094452 0.0011581855 2.114548e-05   2.114688e-05
## y.hat[7]    0.015558751 0.0010988376 2.006194e-05   1.976811e-05
## y.hat[8]    0.015558751 0.0010988376 2.006194e-05   1.976811e-05
## y.hat[9]    0.015558751 0.0010988376 2.006194e-05   1.976811e-05
## y.hat[10] 0.012893579 0.0010864376 1.983555e-05   1.981748e-05
## y.hat[11] 0.012893579 0.0010864376 1.983555e-05   1.981748e-05
## y.hat[12] 0.012893579 0.0010864376 1.983555e-05   1.981748e-05
## y.hat[13] 0.020342275 0.0012156266 2.219420e-05   2.218613e-05
## y.hat[14] 0.020342275 0.0012156266 2.219420e-05   2.218613e-05
## y.hat[15] 0.020342275 0.0012156266 2.219420e-05   2.218613e-05
## y.hat[16] 0.017677102 0.0012783733 2.333980e-05   2.273521e-05
## y.hat[17] 0.017677102 0.0012783733 2.333980e-05   2.273521e-05
## y.hat[18] 0.014780392 0.0011507925 2.101050e-05   2.164535e-05
## y.hat[19] 0.014780392 0.0011507925 2.101050e-05   2.164535e-05
## y.hat[20] 0.014780392 0.0011507925 2.101050e-05   2.164535e-05
## y.hat[21] 0.012115220 0.0011581635 2.114508e-05   2.081226e-05
## y.hat[22] 0.012115220 0.0011581635 2.114508e-05   2.081226e-05
## y.hat[23] 0.012115220 0.0011581635 2.114508e-05   2.081226e-05
## y.hat[24] 0.017621557 0.0010966608 2.002220e-05   2.002001e-05
## y.hat[25] 0.017621557 0.0010966608 2.002220e-05   2.002001e-05
## y.hat[26] 0.017621557 0.0010966608 2.002220e-05   2.002001e-05
## y.hat[27] 0.014956385 0.0011040788 2.015763e-05   2.144425e-05
## y.hat[28] 0.014956385 0.0011040788 2.015763e-05   2.144425e-05
## y.hat[29] 0.014956385 0.0011040788 2.015763e-05   2.144425e-05
## y.hat[30] 0.015287781 0.0011196661 2.044221e-05   1.997699e-05
## y.hat[31] 0.015287781 0.0011196661 2.044221e-05   1.997699e-05
## y.hat[32] 0.015287781 0.0011196661 2.044221e-05   1.997699e-05
## y.hat[33] 0.012622608 0.0010816347 1.974786e-05   2.019552e-05
## y.hat[34] 0.012622608 0.0010816347 1.974786e-05   2.019552e-05
## y.hat[35] 0.012622608 0.0010816347 1.974786e-05   2.019552e-05
## y.hat[36] 0.015438848 0.0011179322 2.041056e-05   2.039300e-05
## y.hat[37] 0.015438848 0.0011179322 2.041056e-05   2.039300e-05
## y.hat[38] 0.015438848 0.0011179322 2.041056e-05   2.039300e-05
## y.hat[39] 0.012773676 0.0011091395 2.025002e-05   1.948926e-05
## y.hat[40] 0.012773676 0.0011091395 2.025002e-05   1.948926e-05
## y.hat[41] 0.012773676 0.0011091395 2.025002e-05   1.948926e-05
```

```r
y.hat.rows <- grep("y.hat", rownames(d2.stats))
y.hat <- d2.stats$Mean[y.hat.rows] # To get fitted values of the model, look at the stats and pull out
resids <- d$N_min_rate - y.hat
# for convenience, we can then put those values into our data frame and examine them
d <- cbind(d, y.hat = y.hat, resid = resids)
```

Some summary statistics:

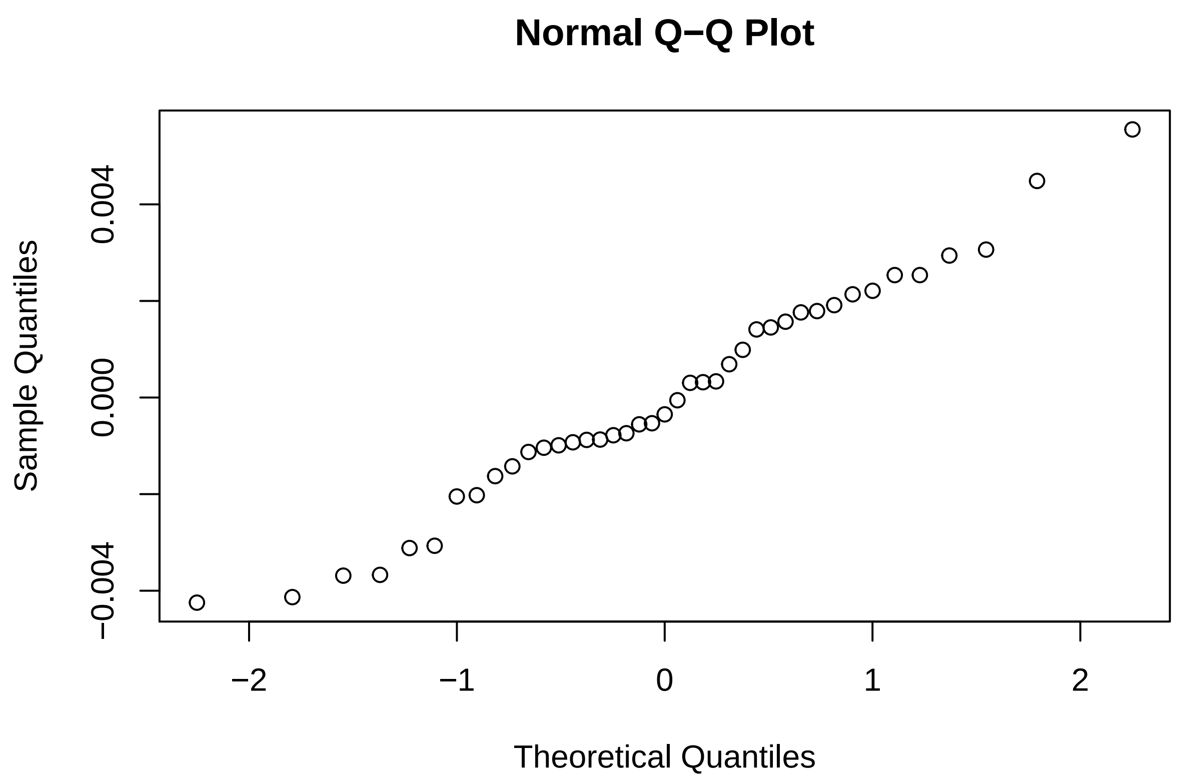
The means and standard deviations of:

- slope = b
- intercept = mu.a
- group-level (i.e. among-plot) variance = sigma.a
- individual-level (within-plot) variance = sigma.y

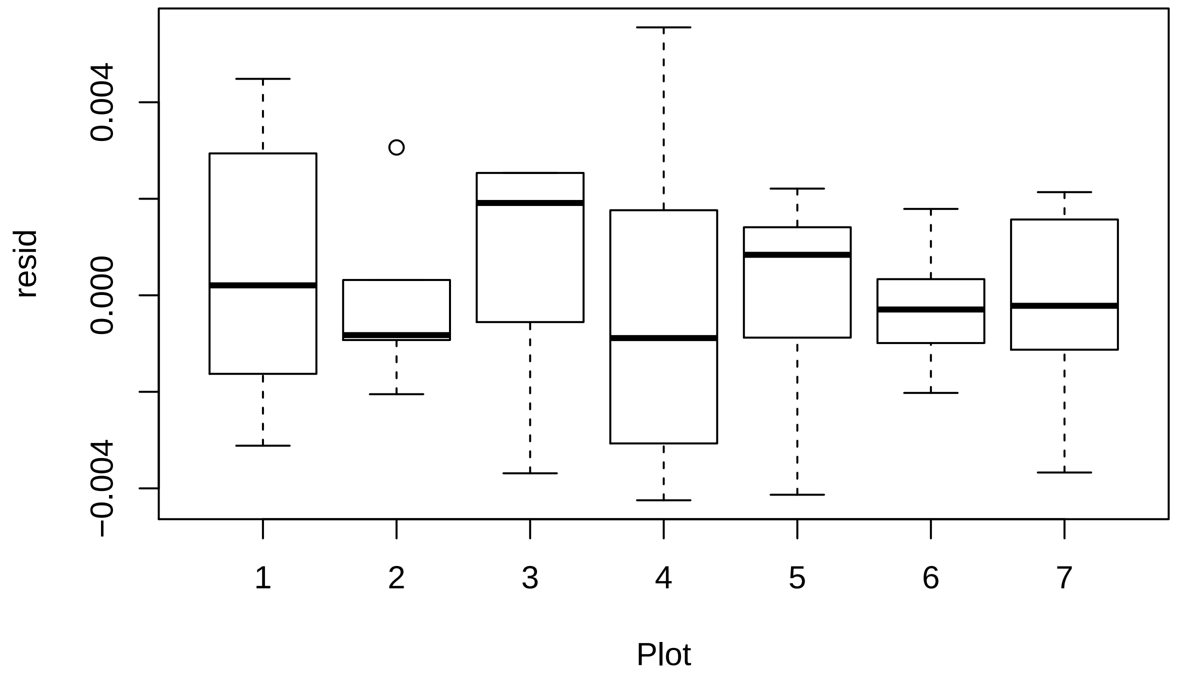
```
##                    Mean           SD
## a[1]       0.018094452 0.0011581855
## a[2]       0.012893579 0.0010864376
## a[3]       0.017677102 0.0012783733
## a[4]       0.012115220 0.0011581635
## a[5]       0.014956385 0.0011040788
## a[6]       0.012622608 0.0010816347
## a[7]       0.012773676 0.0011091395
## b          0.002665172 0.0008477275
## mu.a       0.014433530 0.0015623524
## sigma.a    0.003491457 0.0015403687
## sigma.y    0.002616677 0.0003394741
## y.hat[1]  0.020759624 0.0011665434
## y.hat[2]  0.020759624 0.0011665434
## y.hat[3]  0.020759624 0.0011665434
## y.hat[4]  0.018094452 0.0011581855
## y.hat[5]  0.018094452 0.0011581855
## y.hat[6]  0.018094452 0.0011581855
## y.hat[7]  0.015558751 0.0010988376
## y.hat[8]  0.015558751 0.0010988376
## y.hat[9]  0.015558751 0.0010988376
## y.hat[10] 0.012893579 0.0010864376
## y.hat[11] 0.012893579 0.0010864376
## y.hat[12] 0.012893579 0.0010864376
## y.hat[13] 0.020342275 0.0012156266
## y.hat[14] 0.020342275 0.0012156266
## y.hat[15] 0.020342275 0.0012156266
## y.hat[16] 0.017677102 0.0012783733
## y.hat[17] 0.017677102 0.0012783733
## y.hat[18] 0.014780392 0.0011507925
## y.hat[19] 0.014780392 0.0011507925
## y.hat[20] 0.014780392 0.0011507925
## y.hat[21] 0.012115220 0.0011581635
## y.hat[22] 0.012115220 0.0011581635
## y.hat[23] 0.012115220 0.0011581635
## y.hat[24] 0.017621557 0.0010966608
## y.hat[25] 0.017621557 0.0010966608
## y.hat[26] 0.017621557 0.0010966608
## y.hat[27] 0.014956385 0.0011040788
## y.hat[28] 0.014956385 0.0011040788
## y.hat[29] 0.014956385 0.0011040788
## y.hat[30] 0.015287781 0.0011196661
## y.hat[31] 0.015287781 0.0011196661
## y.hat[32] 0.015287781 0.0011196661
## y.hat[33] 0.012622608 0.0010816347
## y.hat[34] 0.012622608 0.0010816347
## y.hat[35] 0.012622608 0.0010816347
## y.hat[36] 0.015438848 0.0011179322
## y.hat[37] 0.015438848 0.0011179322
## y.hat[38] 0.015438848 0.0011179322
## y.hat[39] 0.012773676 0.0011091395
## y.hat[40] 0.012773676 0.0011091395
## y.hat[41] 0.012773676 0.0011091395
```

See how well the model ran

```
qqnorm(d$resid)
```



```
boxplot(resid~Plot, data=d)
```



```
# Now that random intercepts for county are included in the model, does it look like the residuals diff
```

Look at DIC

```
d2.DIC <- dic.samples(d2, n.iter=5000, thin=5)
d2.DIC
```

```
## Mean deviance:  -373.2
## penalty 8.931
## Penalized deviance: -364.2
```

For comparison, let's quickly fit the complete-pooling model that has no county effects.

#d1.data <- list(n=nrow(d), y=d$N_m in_r ate, x = d$Celastrus) # for this simple model we can let jags generate starting values #d1 <- jags.model("d_completepool_jags.txt", data=d1.data, n.chains=3, n.adapt=1000) #d1.samp <- coda.samples(d1, c("a", "b", "sigma.y"), n.iter=1000) #d1.DIC <- dic.samples(d1, n.iter=1000)

Question 2

Fit a hierarchical (multilevel) model in JAGS using the data sets "immunity.csv" and "patient_age.csv". The model should include "immune.level" as the response variable, "time" as an individual-level predictor, and "age" as a group-level predictor.

These data are repeated measures on individual patients, so each patient is a "group" in the data set. Each row in the data set "immunity.csv" is one observation on one patient. Each row in the data set "patient_age.csv" is the age of each patient at time the study began – so this is a group level predictor, with one row of data per patient. In other words, here "age" is analogous to county-level bedrock uranium content in the Gelman & Hill radon example.

Hints: You can use the column "patient" in "immunity.csv" to index the random intercept for patient. You can then use the column "age" in "patient_age.csv" in the group-level regression that explains some of the variation in the random intercepts. There is a model like this on page 361 of Gelman & Hill.

In your answer, include the model you created. Also report the means and standard deviations of: - the slopes of the individual-data-point-level and "group"-level (i.e. patient-level) regressions - the individual-data-point-level and "group"-level variance parameters