STAT 8700 Homework 10

Brian Detweiler

Tuesday, November 22th, 2016

```
##
## *********************************************************************
## Kruschke, J. K. (2015). Doing Bayesian Data Analysis, Second Edition:
## A Tutorial with R, JAGS, and Stan. Academic Press / Elsevier.
## *********************************************************************
```

1. Consider the data presented in Table 7.3

```
blue.earth <- c(5.0, 13.0, 7.2, 6.8, 12.8,  5.8,  9.5, 6.0, 3.8,  14.3, 1.8,  6.9, 4.7, 9.5)
blue.earth.level <- c(0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0)
clay <- c(0.9, 12.9, 2.6, 3.5, 26.6,  1.5, 13.0, 8.8, 19.5,  2.5, 9.0, 13.1, 3.6, 6.9)
clay.level <- c(1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1)
goodhue <-    c(14.3, 6.9, 7.6, 9.8,  2.6, 43.5,  4.9, 3.5,  4.8,  5.6, 3.5,  3.9, 6.7)
goodhue.level <- c(0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
radon.df <- cbind(blue.earth, blue.earth.level, clay, clay.level, goodhue, goodhue.level)
radon.df <- as.data.frame(radon.df)
radon.df
```

```
##    blue.earth blue.earth.level clay clay.level goodhue goodhue.level
## 1         5.0                0  0.9          1    14.3             0
## 2        13.0                0 12.9          0     6.9             1
## 3         7.2                0  2.6          0     7.6             0
## 4         6.8                0  3.5          1     9.8             1
## 5        12.8                0 26.6          0     2.6             0
## 6         5.8                1  1.5          0    43.5             0
## 7         9.5                0 13.0          0     4.9             0
## 8         6.0                0  8.8          0     3.5             0
## 9         3.8                0 19.5          0     4.8             0
## 10       14.3                1  2.5          1     5.6             0
## 11        1.8                0  9.0          0     3.5             0
## 12        6.9                0 13.1          0     3.9             0
## 13        4.7                0  3.6          0     6.7             0
## 14        9.5                0  6.9          1    14.3             0
```

```
basements.blue.earth <- ((1 + blue.earth.level) %% 2)
basements.clay <- ((1 + clay.level) %% 2)
basements.goodhue <- ((1 + goodhue.level) %% 2)

basement.data <- data.frame(y=sum(basements.blue.earth), N=length(basements.blue.earth))
basement.data <- rbind(basement.data, data.frame(y=sum(basements.clay), N=length(basements.clay)))
basement.data <- rbind(basement.data, data.frame(y=sum(basements.goodhue), N=length(basements.goodhue)))
basement.data
```

```
##    y  N
```

```
## 1 12 14
## 2 10 14
## 3 11 13
```

(a) Let θ_1, θ_2, and θ_3 represent the proportion of houses that have basements in Blue Earth, Clay, and Goodhue counties respectively. Fit a hierarchical model to the data, and use it to obtain posterior summaries for θ_1, θ_2, and θ_3.

```
# Log Posterior (u, v space)
log.post2 <- function(u, v) {

  basements <- basement.data$y
  houses <- basement.data$N
  alpha <- exp(u + v) / (1 + exp(u))
  beta <- exp(v) / (1 + exp(u))
  ldens <- 0

  # Loop over each of the 71 experiments
  for(i in 1:length(houses)) {

    # There is no gamma function in R - have to use Log Gamma, so the density is logged
    # This is why it's addative rather than multiplicative
    # basements[i] is the same as y_i
    # houses[i] is the same as n_i
    ldens <- (ldens
              + (lgamma(alpha + beta) + lgamma(alpha + basements[i]) + lgamma(beta + houses[i] - basemen
              - (lgamma(alpha) + lgamma(beta) + lgamma(alpha + beta + houses[i])))
  }

  # Return the final posterior density, which is still in logged form
  ldens - 5 / 2 * log(alpha + beta) + log(alpha) + log(beta)
}

# Just defines the size of each contour: 0.05, 0.15, 0.25, ..., 0.95
contours <- seq(0.05, 0.95, 0.1)

# Do the same steps as above, but refine the grid space
# Also, I changed the length to 200, because 2001 was slowing my computer to a crawl
u2 <- seq(-4, 4, length = 200)
v2 <- seq(-5, 13, length = 200)
logdens2 <- outer(u2, v2, log.post2)
# dens2 is a 200x200 matrix of probabilities for the u2, v2 values of alpha and beta
# For instance, u2[1] = -2.3, and v2[1] = 1. dens2[1, 1] = 1.978023e-14
# This is equivalent to saying p(alpha = -2.3, beta = 1) = 1.978023e-14
dens2 <- exp(logdens2 - max(logdens2))

fileName <- "Assignment_10_1_a"

modelString ="
model {

  for (j in 1:count) {
```

```
    y[j] ~ dbin(theta[j], N[j])
    theta[j] ~ dbeta(alpha, beta)
  }

  lnx <- log(alpha / beta)
  lny <- log(alpha + beta)

  alpha <- u / pow(v, 2)
  beta <- (1 - u) / pow(v, 2)

  u ~ dunif(0, 1)
  v ~ dunif(0, 1)
}
"

writeLines(modelString, con=fileName)

basementsModel = jags.model(file=fileName,
                            data=list(y=basement.data$y,
                                      N=basement.data$N,
                                      count=length(basement.data$N)),
                            n.chains=4)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 3
##    Unobserved stochastic nodes: 5
##    Total graph size: 27
##
## Initializing model
```

```
update(basementsModel, n.iter=10000)


basementsSamples <- coda.samples(basementsModel,
                                 n.iter=200000,
                                 variable.names=c("alpha", "beta", "theta", "y", "lnx", "lny"),
                                 thin=20)

basementsSamples.M <- as.matrix(basementsSamples)
summary(basementsSamples.M[,"theta[1]"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.3839  0.7609  0.8162  0.8104  0.8673  0.9956
```

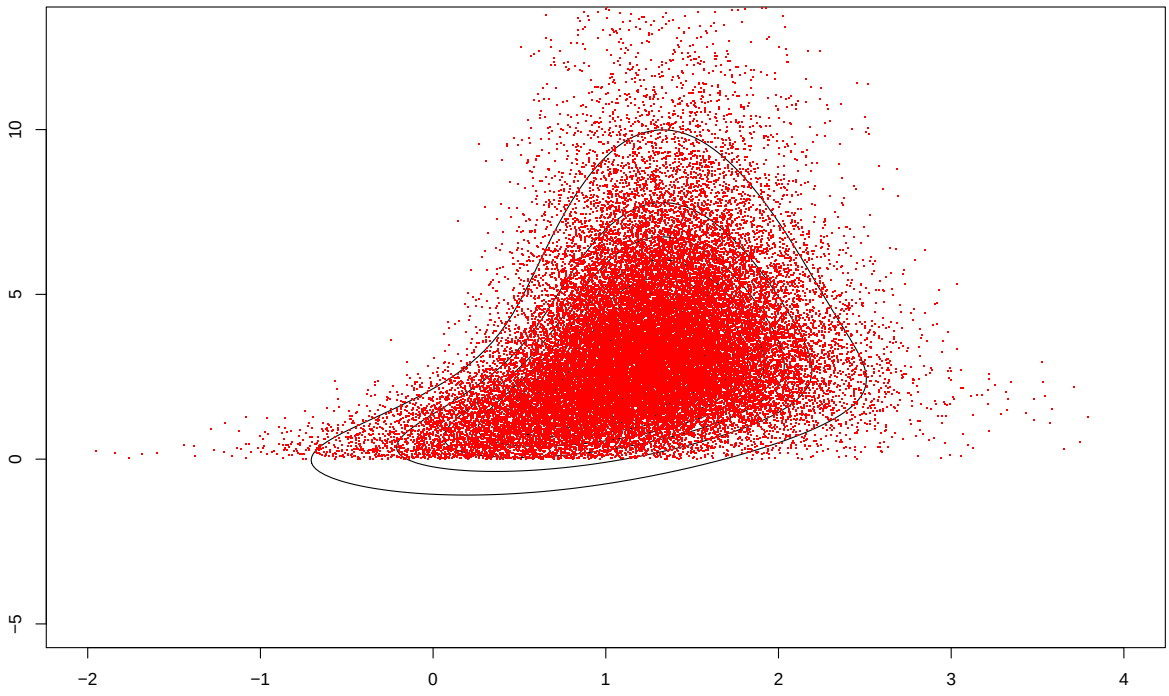
```
summary(basementsSamples.M[,"theta[2]"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.2269  0.6960  0.7618  0.7501  0.8166  0.9776
```

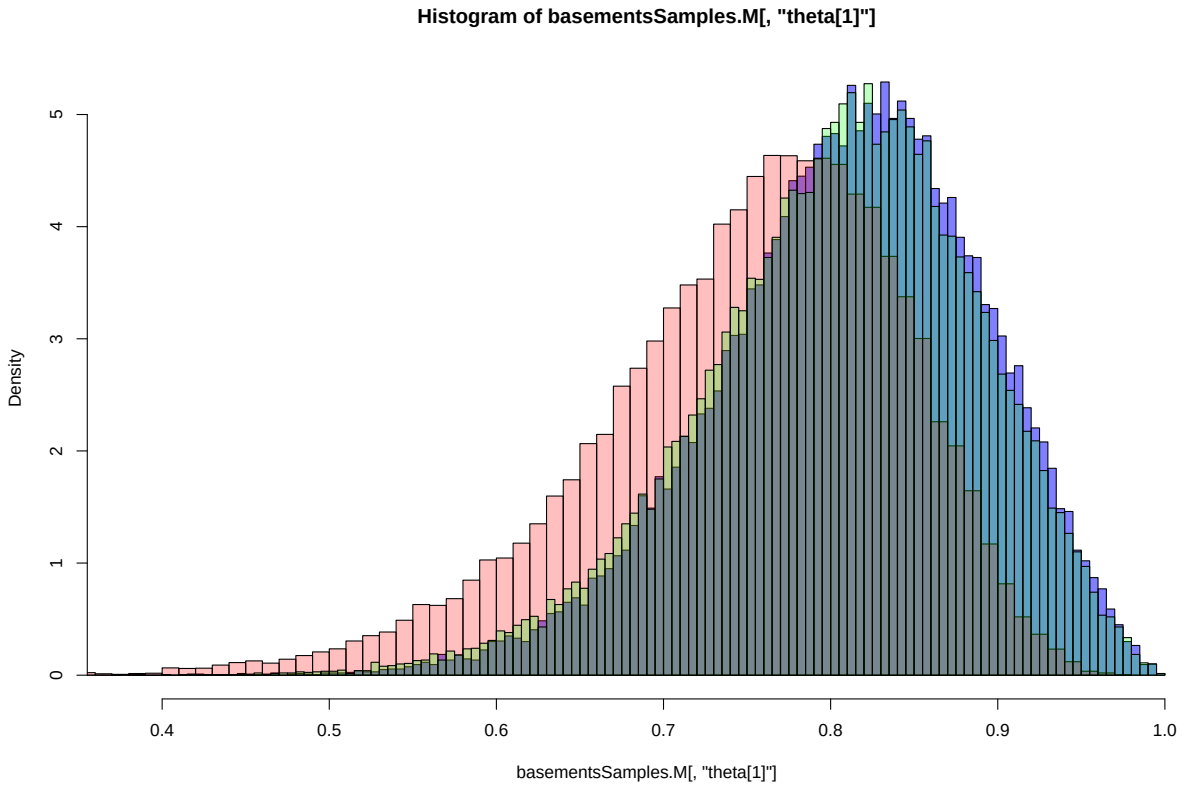
```r
summary(basementsSamples.M[,"theta[3]"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.3205  0.7537  0.8112  0.8045  0.8623  0.9990
```

```r
contour(u2, v2, dens2, levels = contours, drawlabels = FALSE, xlim=c(-2, 4), ylim=c(-5, 13))
points(basementsSamples.M[,"lnx"], basementsSamples.M[,"lny"], col="red", pch=".", xlim=c(-2, 4), ylim=
```




```r
hist(basementsSamples.M[,"theta[1]"], breaks=100, freq=F, col=rgb(0, 0, 1, .5))
hist(basementsSamples.M[,"theta[2]"], breaks=100, freq=F, col=rgb(1, 0, 0, .25), add=T)
hist(basementsSamples.M[,"theta[3]"], breaks=100, freq=F, col=rgb(0, 1, 0, .25), add=T)
```



(b) Fit a linear regression to the natural log of the radon measurements, with indicator variables for the three counties and for weather a measurement was recorded on the first floor or basement, do not include an intercept term. Present posterior summaries for parameters and summarize your posterior inferences in non-technical terms (int, for each parameter β, what does e^{β} represent?)

```
radon.linreg <- data.frame(radon=log(radon.df$blue.earth),
                           basement=as.numeric(!as.logical(radon.df$blue.earth.level)),
                           blue.earth=rep(1, length(radon.df$blue.earth)),
                           clay=rep(0, length(radon.df$blue.earth)),
                           goodhue=rep(0, length(radon.df$blue.earth)))

radon.linreg <- rbind(radon.linreg,
                      data.frame(radon=log(radon.df$clay),
                           basement=as.numeric(!as.logical(radon.df$clay.level)),
                           blue.earth=rep(0, length(radon.df$clay)),
                           clay=rep(1, length(radon.df$clay)),
                           goodhue=rep(0, length(radon.df$clay))))

radon.linreg <- rbind(radon.linreg,
                      data.frame(radon=log(radon.df$goodhue),
                           basement=as.numeric(!as.logical(radon.df$goodhue.level)),
                           blue.earth=rep(0, length(radon.df$goodhue)),
                           clay=rep(0, length(radon.df$goodhue)),
```

```
                          goodhue=rep(1, length(radon.df$goodhue))))

radon.linreg
```

```
##         radon basement blue.earth clay goodhue
## 1   1.6094379        1          1    0       0
## 2   2.5649494        1          1    0       0
## 3   1.9740810        1          1    0       0
## 4   1.9169226        1          1    0       0
## 5   2.5494452        1          1    0       0
## 6   1.7578579        0          1    0       0
## 7   2.2512918        1          1    0       0
## 8   1.7917595        1          1    0       0
## 9   1.3350011        1          1    0       0
## 10  2.6602595        0          1    0       0
## 11  0.5877867        1          1    0       0
## 12  1.9315214        1          1    0       0
## 13  1.5475625        1          1    0       0
## 14  2.2512918        1          1    0       0
## 15 -0.1053605        0          0    1       0
## 16  2.5572273        1          0    1       0
## 17  0.9555114        1          0    1       0
## 18  1.2527630        0          0    1       0
## 19  3.2809112        1          0    1       0
## 20  0.4054651        1          0    1       0
## 21  2.5649494        1          0    1       0
## 22  2.1747517        1          0    1       0
## 23  2.9704145        1          0    1       0
## 24  0.9162907        0          0    1       0
## 25  2.1972246        1          0    1       0
## 26  2.5726122        1          0    1       0
## 27  1.2809338        1          0    1       0
## 28  1.9315214        0          0    1       0
## 29  2.6602595        1          0    0       1
## 30  1.9315214        0          0    0       1
## 31  2.0281482        1          0    0       1
## 32  2.2823824        0          0    0       1
## 33  0.9555114        1          0    0       1
## 34  3.7727609        1          0    0       1
## 35  1.5892352        1          0    0       1
## 36  1.2527630        1          0    0       1
## 37  1.5686159        1          0    0       1
## 38  1.7227666        1          0    0       1
## 39  1.2527630        1          0    0       1
## 40  1.3609766        1          0    0       1
## 41  1.9021075        1          0    0       1
## 42  2.6602595        1          0    0       1
```

```
fileName <- "Assignment_10_1_b"

modelString ="
model {
```

```
  for (j in 1:count) {
    y[j] ~ dnorm(mu[j], tau)
    mu[j] <- beta[1] * basement[j] + beta[2] * blueearth[j] + beta[3] * clay[j] + beta[4] * goodhue[j]
  }

  yclay ~ dnorm(beta[1], tau)

  # Prior for beta
  for(j in 1:4){
    beta[j] ~ dnorm(0,0.0001)
  }

  # Prior for the inverse variance
  tau   ~ dgamma(0.01, 0.01)

}
"

writeLines(modelString, con=fileName)

radonModel = jags.model(file=fileName,
                            data=list(y=radon.linreg$radon,
                                      basement=radon.linreg$basement,
                                      blueearth=radon.linreg$blue.earth,
                                      clay=radon.linreg$clay,
                                      goodhue=radon.linreg$goodhue,
                                      count=length(radon.linreg$radon)),
                            n.chains=4)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 42
##    Unobserved stochastic nodes: 6
##    Total graph size: 241
##
## Initializing model
```

```
update(radonModel, n.iter=10000)


radonSamples <- coda.samples(radonModel,
                                n.iter=1000000,
                                variable.names=c("beta", "yclay"),
                                thin=50)
summary(radonSamples)
```

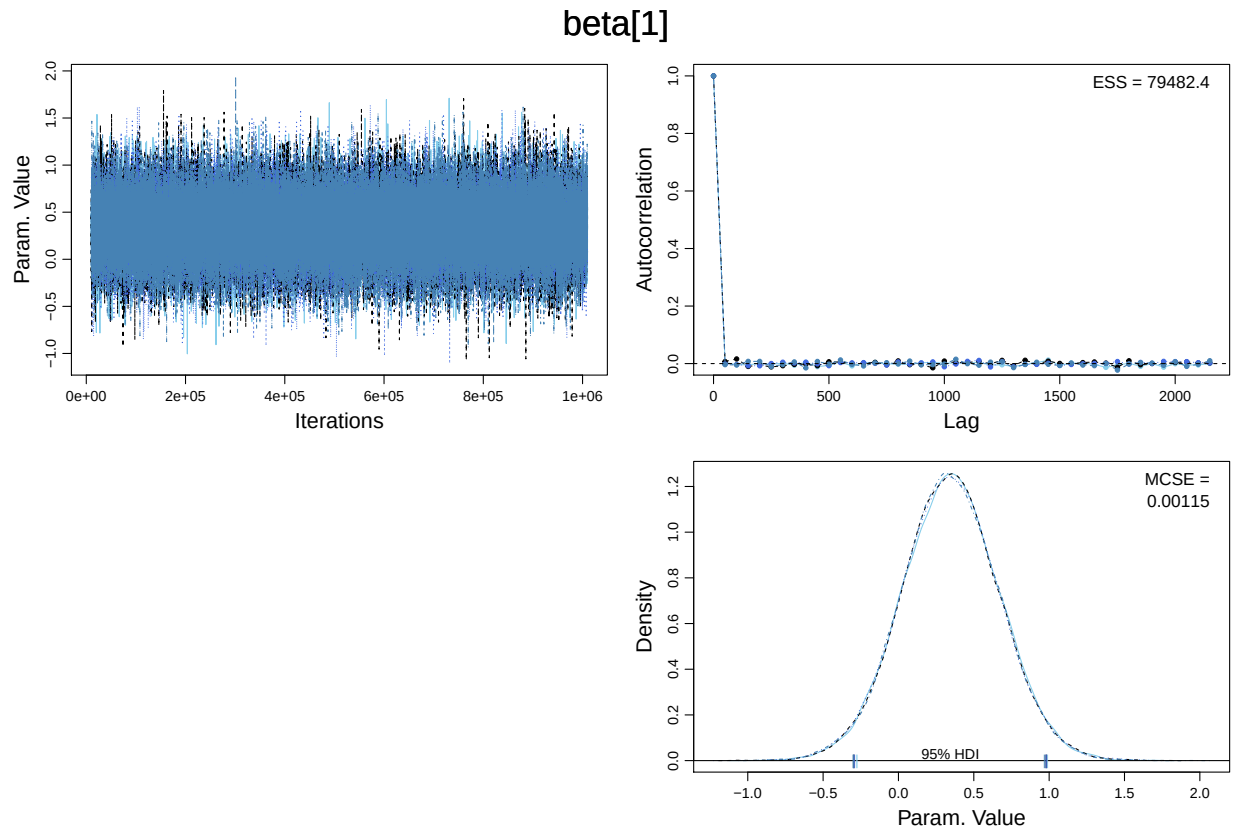
```
##
## Iterations = 10050:1010000
## Thinning interval = 50
## Number of chains = 4
## Sample size per chain = 20000
```

```
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##           Mean     SD Naive SE Time-series SE
## beta[1] 0.3444 0.3233 0.001143       0.001147
## beta[2] 1.6136 0.3527 0.001247       0.001255
## beta[3] 1.5367 0.3168 0.001120       0.001120
## beta[4] 1.6288 0.3525 0.001246       0.001245
## yclay   0.3487 0.8696 0.003074       0.003069
##
## 2. Quantiles for each variable:
##
##            2.5%     25%    50%    75% 97.5%
## beta[1] -0.2934  0.1304 0.3433 0.5591 0.980
## beta[2]  0.9167  1.3805 1.6123 1.8466 2.307
## beta[3]  0.9170  1.3256 1.5367 1.7469 2.161
## beta[4]  0.9292  1.3956 1.6291 1.8622 2.325
## yclay   -1.3630 -0.2261 0.3512 0.9235 2.066
```

```
diagMCMC(codaObject = radonSamples)
```

```
## [1] "Warning: coda::gelman.plot fails for beta[1]"
```

```
radonSamples.M <- as.matrix(radonSamples)
```



```r
summary(radonSamples.M)
```

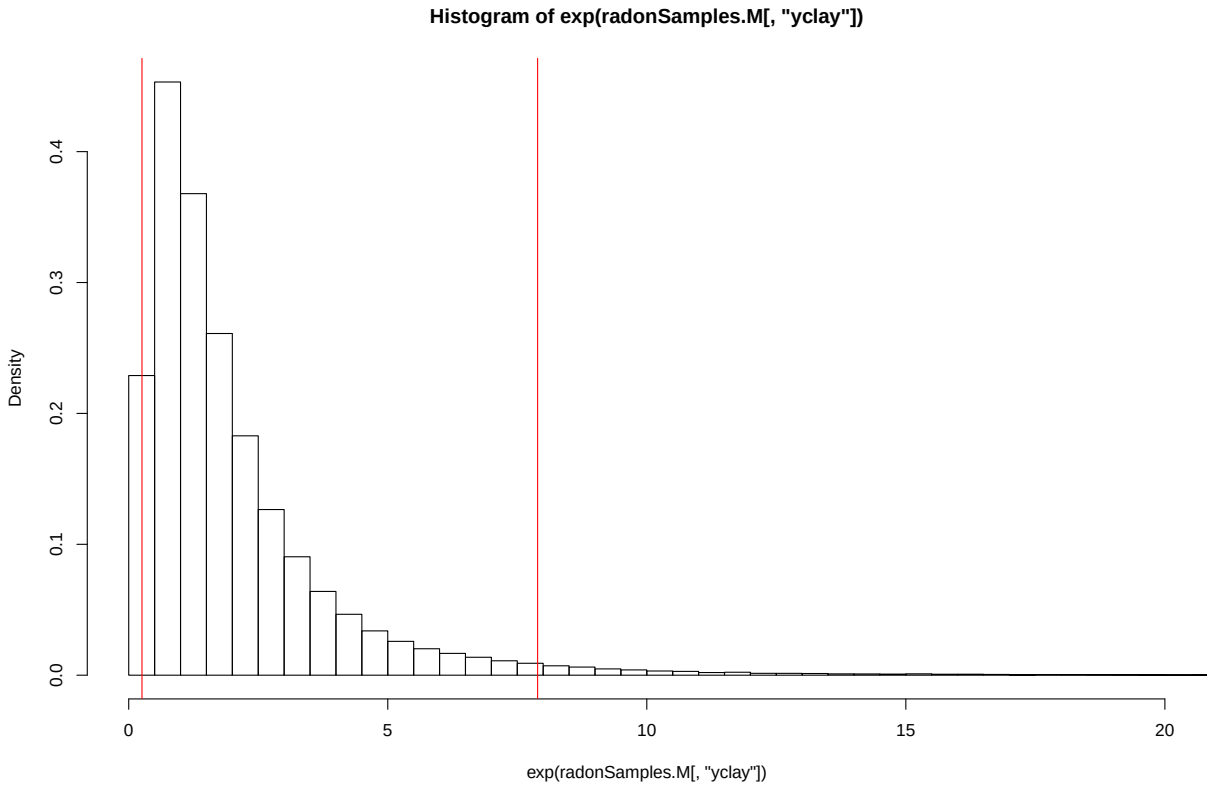
```
##     beta[1]           beta[2]           beta[3]           beta[4]
##  Min.   :-1.1075   Min.   :0.007564   Min.   :0.04772   Min.   :-0.07992
##  1st Qu.: 0.1304   1st Qu.:1.380549   1st Qu.:1.32556   1st Qu.: 1.39560
##  Median : 0.3433   Median :1.612310   Median :1.53672   Median : 1.62906
##  Mean   : 0.3444   Mean   :1.613600   Mean   :1.53666   Mean   : 1.62879
##  3rd Qu.: 0.5591   3rd Qu.:1.846609   3rd Qu.:1.74689   3rd Qu.: 1.86218
##  Max.   : 1.9471   Max.   :3.430915   Max.   :2.98612   Max.   : 3.48379
##      yclay
##  Min.   :-3.3283
##  1st Qu.:-0.2261
##  Median : 0.3512
##  Mean   : 0.3487
##  3rd Qu.: 0.9235
##  Max.   : 4.1062
```

β_1 represents whether the measurement was taken in a basement or not. It is clear that having a basement has a positive effect on y.

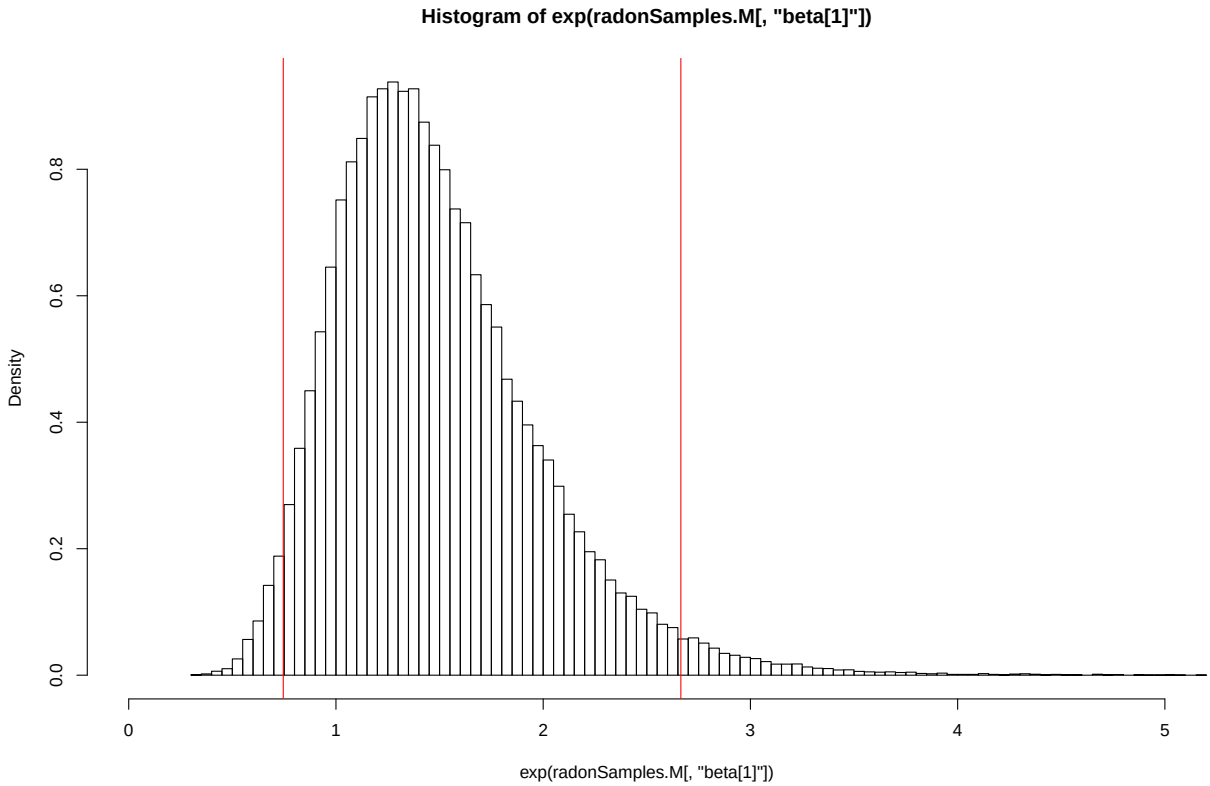
$\beta_2, \beta_3, \beta_4$ represents which county the measurement was taken in.

(c) Suppose another house is sampled at random from Clay County, simulate values from the posterior predictive distribution for its radon measurements anf give an 95% predictive interval. Express the interval of the original unlogged scale. (Hint: You must consider whether or not the randomly chosen house has a basement)

```r
yclay.quant <- quantile(exp(radonSamples.M[, "yclay"]), probs = c(0.025, 0.975))
hist(exp(radonSamples.M[,"yclay"]), breaks=100, freq=F, xlim=c(0, 20))
abline(v=c(yclay.quant[[1]], yclay.quant[[2]]), col="red")
```




```
beta1.quant <- quantile(exp(radonSamples.M[, "beta[1]"]), probs = c(0.025, 0.975))
hist(exp(radonSamples.M[,"beta[1]"]), breaks=100, freq=F, xlim=c(0, 5))
abline(v=c(beta1.quant[[1]], beta1.quant[[2]]), col="red")
```


Histogram of exp(radonSamples.M[, "beta[1]"])
Density
0.0
0.2
0.4
0.6
0.8
0
1
2
3
4
5
exp(radonSamples.M[, "beta[1]"])

■

2. The file `drinks.txt` contains the amount of time needed by a company employee to refill an automatic vending machine. For each refill, the number of cases of product and the distance walked (in feet) is also recorded.

```
drinks <- read.table('drinks.txt', header=TRUE)
drinks$Intercept <- rep(1, length(drinks$Time))
drinks
```

```
##     Time Cases Distance Intercept
## 1  16.68     7      560         1
## 2  11.50     3      220         1
## 3  12.03     3      340         1
## 4  14.88     4       80         1
## 5  13.75     6      150         1
## 6  18.11     7      330         1
## 7   8.00     2      110         1
## 8  17.83     7      210         1
## 9  79.24    30     1460         1
## 10 21.50     5      605         1
## 11 40.33    16      688         1
## 12 21.00    10      215         1
## 13 13.50     4      255         1
## 14 19.75     6      462         1
## 15 24.00     9      448         1
## 16 29.00    10      776         1
## 17 15.35     6      200         1
## 18 19.00     7      132         1
## 19  9.50     3       36         1
## 20 35.10    17      770         1
## 21 17.90    10      140         1
## 22 52.32    26      810         1
## 23 18.75     9      450         1
## 24 19.83     8      635         1
## 25 10.75     4      150         1
```

(a) Fit a linear regression model for the time taken, with number of cases and distance walked as explanatory variables (include an intercept term).

```
fileName <- "Assignment_10_2_a"

modelString ="
model {

  for (j in 1:count) {
    y[j] ~ dnorm(mu[j], tau)

    mu[j] <- beta[1] * x1[j] + beta[2] * x2[j] + beta[3] * x3[j]
```

```
  }

  # Prior for beta
  for(j in 1:3){
    beta[j] ~ dnorm(0,0.0001)
  }

  # Prior for the inverse variance
  tau ~ dgamma(0.01, 0.01)

  # Predictive values
  ypred ~ dnorm(muavg, tau)
  muavg <- beta[1] * x1avg * beta[2] * x2avg + beta[3] * x3avg

  # For RB2
  Svar <- pow(sd(y), 2)
  sigma2 <- 1 / tau

  RB2 <- 1 - (sigma2 / Svar)
}
"

writeLines(modelString, con=fileName)

drinksModel = jags.model(file=fileName,
                         data=list(y = drinks$Time,
                                   x1 = drinks$Intercept,
                                   x2 = drinks$Distance,
                                   x3 = drinks$Cases,
                                   x1avg = mean(drinks$Intercept),
                                   x2avg = mean(drinks$Distance),
                                   x3avg = mean(drinks$Cases),
                                   count = length(drinks$Time)),
                         n.chains=4)
```

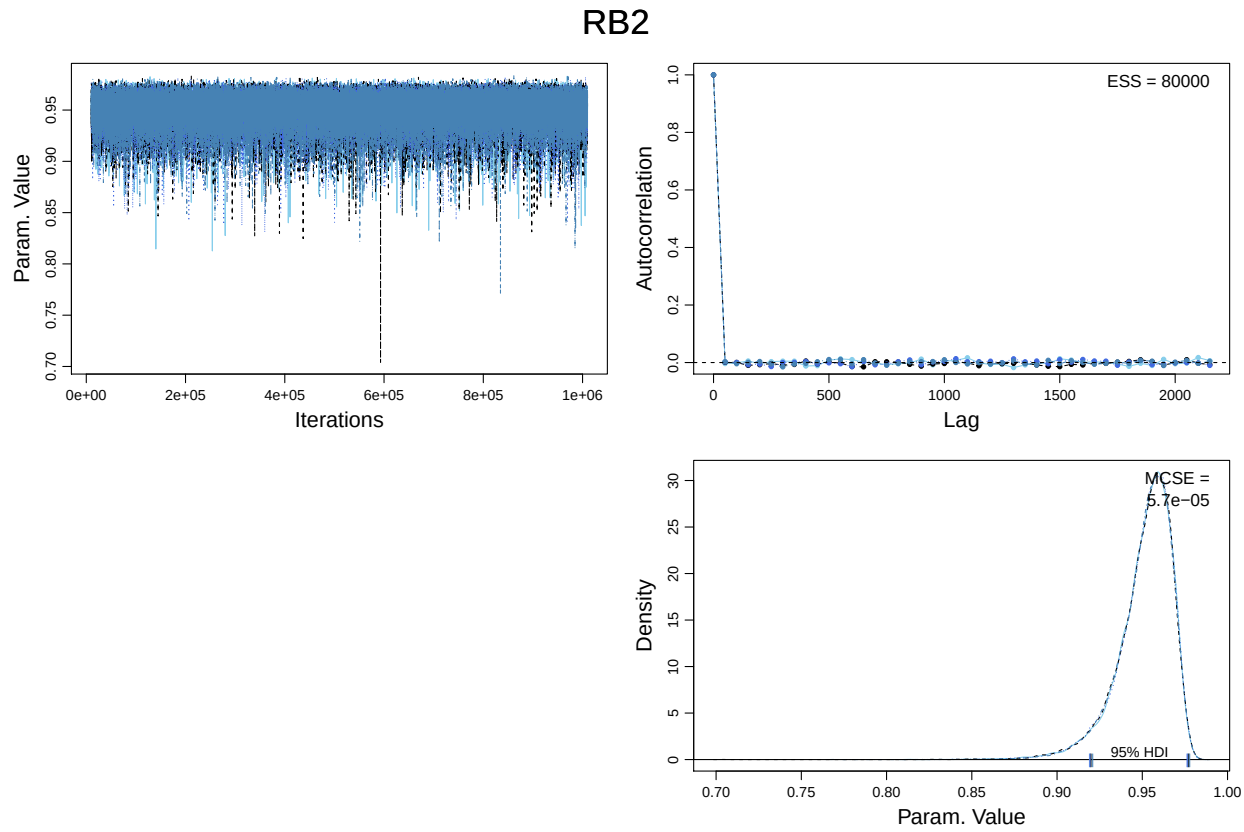
```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 25
##    Unobserved stochastic nodes: 5
##    Total graph size: 196
##
## Initializing model
```

```
update(drinksModel, n.iter=10000)


drinksSamples <- coda.samples(drinksModel,
                              n.iter=1000000,
                              variable.names=c("beta", "RB2", "Svar", "sigma2", "ypred"),
                              thin=50)
diagMCMC(codaObject = drinksSamples)
```

```
## [1] "Warning: coda::gelman.plot fails for RB2"
```

```
drinksSamples.M <- as.matrix(drinksSamples)
```




```
rb2 <- quantile(drinksSamples.M[,"RB2"], probs = c(0.025, 0.975))
summary(drinksSamples.M)
```

```
##       RB2              Svar        beta[1]          beta[2]
##  Min.   :0.7038   Min.   :241   Min.   :-2.966   Min.   :-0.007718
##  1st Qu.:0.9438   1st Qu.:241   1st Qu.: 1.584   1st Qu.: 0.011922
##  Median :0.9546   Median :241   Median : 2.339   Median : 0.014403
##  Mean   :0.9516   Mean   :241   Mean   : 2.337   Mean   : 0.014407
##  3rd Qu.:0.9628   3rd Qu.:241   3rd Qu.: 3.088   3rd Qu.: 0.016906
##  Max.   :0.9843   Max.   :241   Max.   : 8.280   Max.   : 0.039061
##     beta[3]           sigma2           ypred
##  Min.   :0.6703   Min.   : 3.784   Min.   :-13.96
##  1st Qu.:1.4983   1st Qu.: 8.964   1st Qu.: 23.00
##  Median :1.6156   Median :10.949   Median : 27.37
##  Mean   :1.6150   Mean   :11.677   Mean   : 27.53
##  3rd Qu.:1.7320   3rd Qu.:13.539   3rd Qu.: 31.92
##  Max.   :2.4724   Max.   :71.386   Max.   : 70.55
```

```
ypredCI <- quantile(drinksSamples.M[, "ypred"], probs = c(0.025, 0.975))
```

(b) In Classical Statistics, one way the quality of a regression model can be analyzed is by calculating something called the $AdjustedR^2$ value, it is basically the pro-portion of variation in the response variable that is explained by the explanatory variables, adjusted for the number of variables. Obviously, the closer this number is to 1, the better. The Bayesian equivalent R2B is defined as

$$R_B^2 = 1 - \frac{\sigma^2}{S_Y^2}$$

where σ^2 is the variance of the regression model and s_Y^2 is the sample variance of the response variable data. Note that since in the Bayesian framework σ^2 is a random variable, so is R_B^2. Obtain a 95% credible interval for R_B^2. Does it seem like the model is a good for the data?

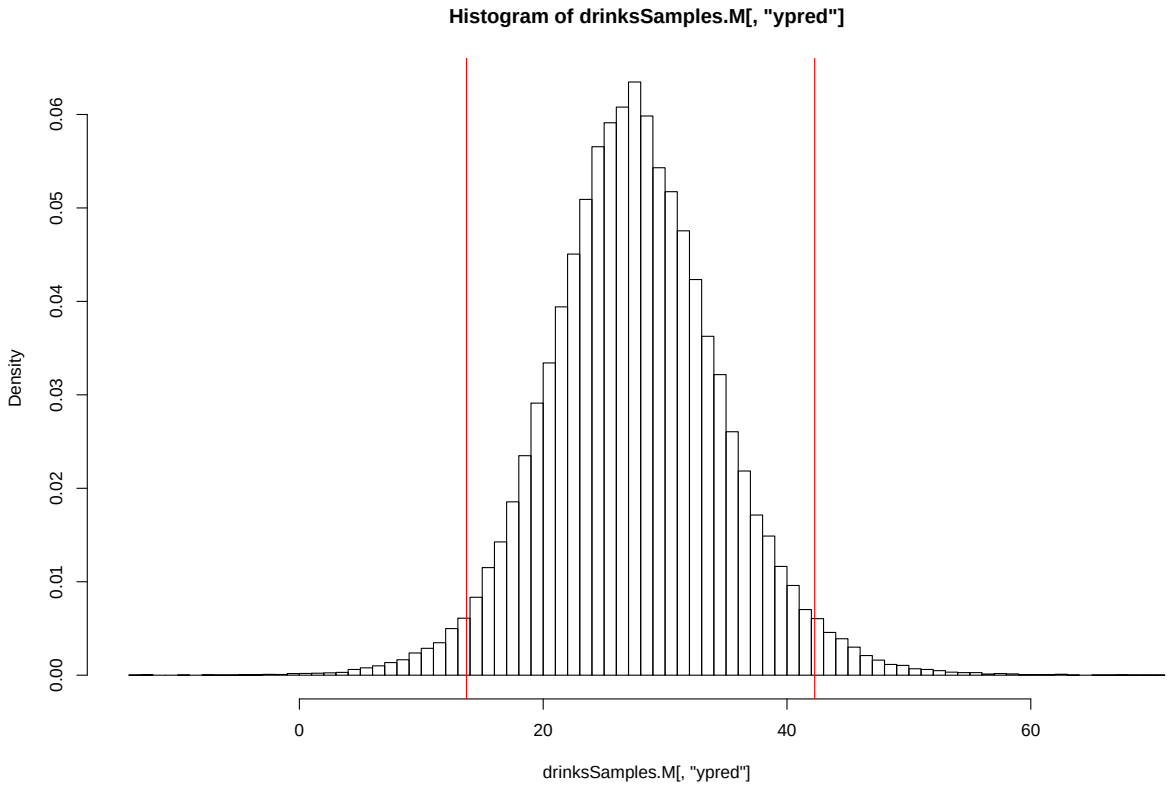
The CI for the R_B^2 for this model is (0.9117538, 0.9735834), which indicates that the model is a good fit.

(c) Obtain a 95% predictive interval for how long it would take to restock the vending machine if the number of cases and distance were at their average (mean) values.

```
hist(drinksSamples.M[, "ypred"], breaks=100, freq=F)
ypredCI <- quantile(drinksSamples.M[, "ypred"], probs = c(0.025, 0.975))
ypredCI
```

```
##     2.5%    97.5%
## 13.70967 42.26849
```

```
abline(v=c(ypredCI[[1]], ypredCI[[2]]), col="red")
```


Histogram of drinksSamples.M[, "ypred"]
Density
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0
20
40
60
drinksSamples.M[, "ypred"]

■

3. Revisit the data in Question 1. Fit a Two-Way ANOVA to the natural log of the radon measurements. For each county in separately, construct a 95% credible interval for the difference between the average (unlogged) radon measurement in houses with basements and in houses without basements. Test the hypothesis that the average radon measurement in houses with basements is greater than in houses without basements.

```
blue.earth.basements <- radon.df %>% filter(blue.earth.level == 0) %>% select(blue.earth)
blue.earth.no.basements <- radon.df %>% filter(blue.earth.level == 1) %>% select(blue.earth)
clay.basements <- radon.df %>% filter(clay.level == 0) %>% select(clay)
clay.no.basements <- radon.df %>% filter(clay.level == 1) %>% select(clay)
goodhue.basements <- radon.df %>% filter(goodhue.level == 0) %>% select(goodhue)
goodhue.no.basements <- radon.df %>% filter(goodhue.level == 1) %>% select(goodhue)


fileName <- "Assignment_10_3_a"

modelString ="
model {

  for (i1 in 1:length(blue.earth.basements)) {
    blue.earth.basements[i1] ~ dnorm(mu.blue.earth.basements, tau)
  }
  mu.blue.earth.basements <- m + alpha.blue.earth + beta.basements + gamma.blue.earth.basements

  for (i2 in 1:length(blue.earth.no.basements)) {
    blue.earth.no.basements[i2] ~ dnorm(mu.blue.earth.no.basements, tau)
  }
  mu.blue.earth.no.basements <- m + alpha.blue.earth + beta.no.basements + gamma.blue.earth.no.basement

  for (i3 in 1:length(clay.basements)) {
    clay.basements[i3] ~ dnorm(mu.clay.basements, tau)
  }
  mu.clay.basements <- m + alpha.clay + beta.basements + gamma.clay.basements

  for (i4 in 1:length(clay.no.basements)) {
    clay.no.basements[i4] ~ dnorm(mu.clay.no.basements, tau)
  }
  mu.clay.no.basements <- m + alpha.clay + beta.no.basements + gamma.clay.no.basements

  for (i5 in 1:length(goodhue.basements)) {
    goodhue.basements[i5] ~ dnorm(mu.goodhue.basements, tau)
  }
  mu.goodhue.basements <- m + alpha.goodhue + beta.basements + gamma.goodhue.basements

  for (i6 in 1:length(goodhue.no.basements)) {
    goodhue.no.basements[i6] ~ dnorm(mu.goodhue.no.basements, tau)
  }
  mu.goodhue.no.basements <- m + alpha.goodhue + beta.no.basements + gamma.goodhue.no.basements
```

```
  m ~ dnorm(0, 1.0001)
  alpha.blue.earth ~ dnorm(0, 1.0001)
  alpha.clay ~ dnorm(0, 1.0001)
  alpha.goodhue ~ dnorm(0, 1.0001)
  beta.basements ~ dnorm(0, 1.0001)
  beta.no.basements ~ dnorm(0, 1.0001)
  gamma.blue.earth.basements ~ dnorm(0, 1.0001)
  gamma.blue.earth.no.basements <- -gamma.blue.earth.basements
  gamma.clay.basements ~ dnorm(0, 1.0001)
  gamma.clay.no.basements <- -gamma.clay.basements
  gamma.goodhue.basements <- -gamma.blue.earth.basements - gamma.clay.basements
  gamma.goodhue.no.basements <- -gamma.goodhue.basements

  tau ~ dgamma(0.01, 0.01)

  delta.blue.earth <- exp(mu.blue.earth.basements) - exp(mu.blue.earth.no.basements)
  delta.clay <- exp(mu.clay.basements) - exp(mu.clay.no.basements)
  delta.goodhue <- exp(mu.goodhue.basements) - exp(mu.goodhue.no.basements)
  delta.all <- delta.blue.earth + delta.clay + delta.goodhue

}
"

writeLines(modelString, con=fileName)

anovaModel = jags.model(file=fileName,
                            data=list(blue.earth.basements = log(blue.earth.basements[,1]),
                                      blue.earth.no.basements = log(blue.earth.no.basements[,1]),
                                      clay.basements = log(clay.basements[,1]),
                                      clay.no.basements = log(clay.no.basements[,1]),
                                      goodhue.basements = log(goodhue.basements[,1]),
                                      goodhue.no.basements = log(goodhue.no.basements[,1])),
                            n.chains=4)
```

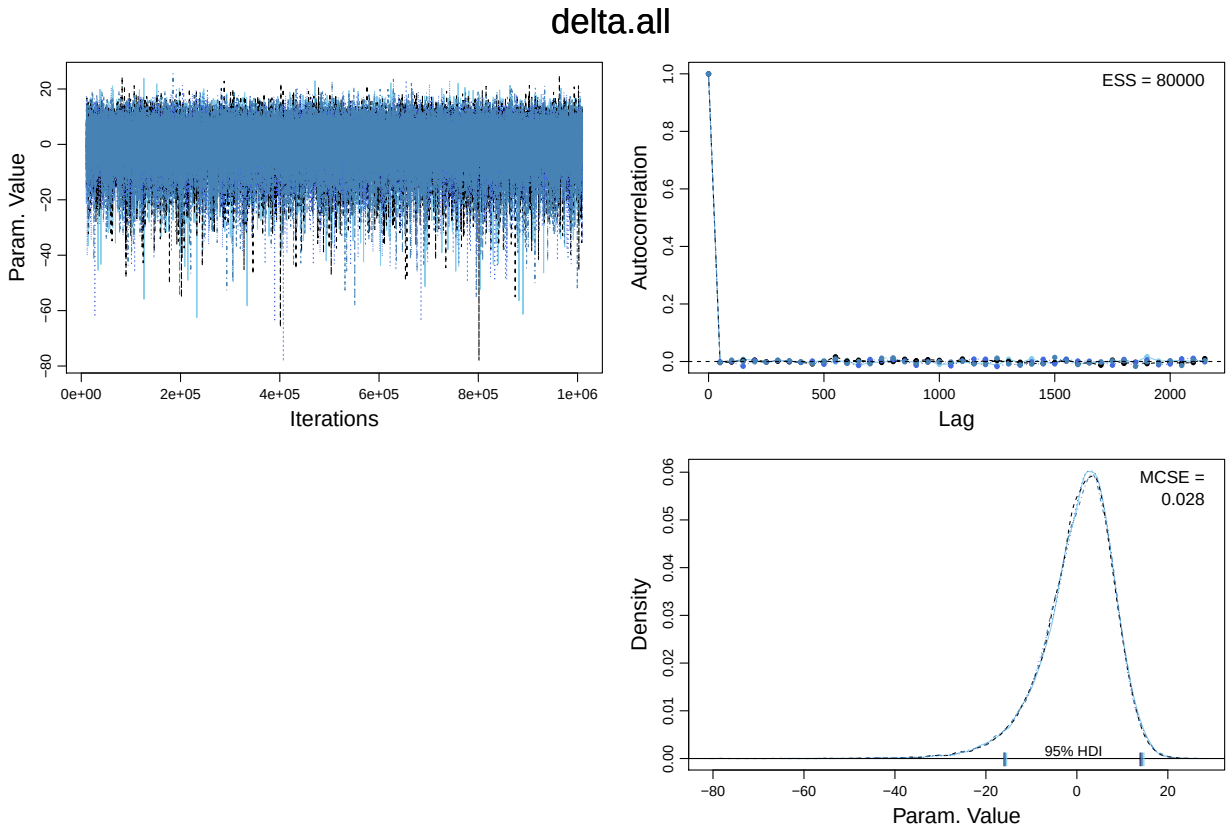
```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 42
##    Unobserved stochastic nodes: 9
##    Total graph size: 89
##
## Initializing model
```

```
update(anovaModel, n.iter=10000)


anovaSamples <- coda.samples(anovaModel,
                            n.iter=1000000,
                            variable.names=c("delta.blue.earth", "delta.clay", "delta.goodhue", "delta
                            thin=50)
diagMCMC(codaObject = anovaSamples)
```

```
## [1] "Warning: coda::gelman.plot fails for delta.all"
```

```
anovaSamples.M <- as.matrix(anovaSamples)
```




```
delta.blue.earth <- quantile(anovaSamples.M[,"delta.blue.earth"], probs = c(0.025, 0.975))
delta.blue.earth
```

```
##       2.5%      97.5%
## -16.312842   4.590973
```

```
hist(anovaSamples.M[,"delta.blue.earth"], breaks = 100)
abline(v=c(delta.blue.earth[[1]], delta.blue.earth[[2]]), col="red")
blue.earth.p.val <-
  length(anovaSamples.M[,"delta.blue.earth"][which(anovaSamples.M[,"delta.blue.earth"] > 0)]) /
  length(anovaSamples.M[,"delta.blue.earth"])

delta.clay <- quantile(anovaSamples.M[,"delta.clay"], probs = c(0.025, 0.975))
delta.clay
```

```
##       2.5%      97.5%
##  0.9654498 10.3190840
```

```
hist(anovaSamples.M[,"delta.clay"], breaks = 100)
abline(v=c(delta.clay[[1]], delta.clay[[2]]), col="red")
clay.p.val <-
```

```
  length(anovaSamples.M[,"delta.clay"][which(anovaSamples.M[,"delta.clay"] > 0)]) /
  length(anovaSamples.M[,"delta.clay"])

delta.goodhue <- quantile(anovaSamples.M[,"delta.goodhue"], probs = c(0.025, 0.975))
delta.goodhue
```

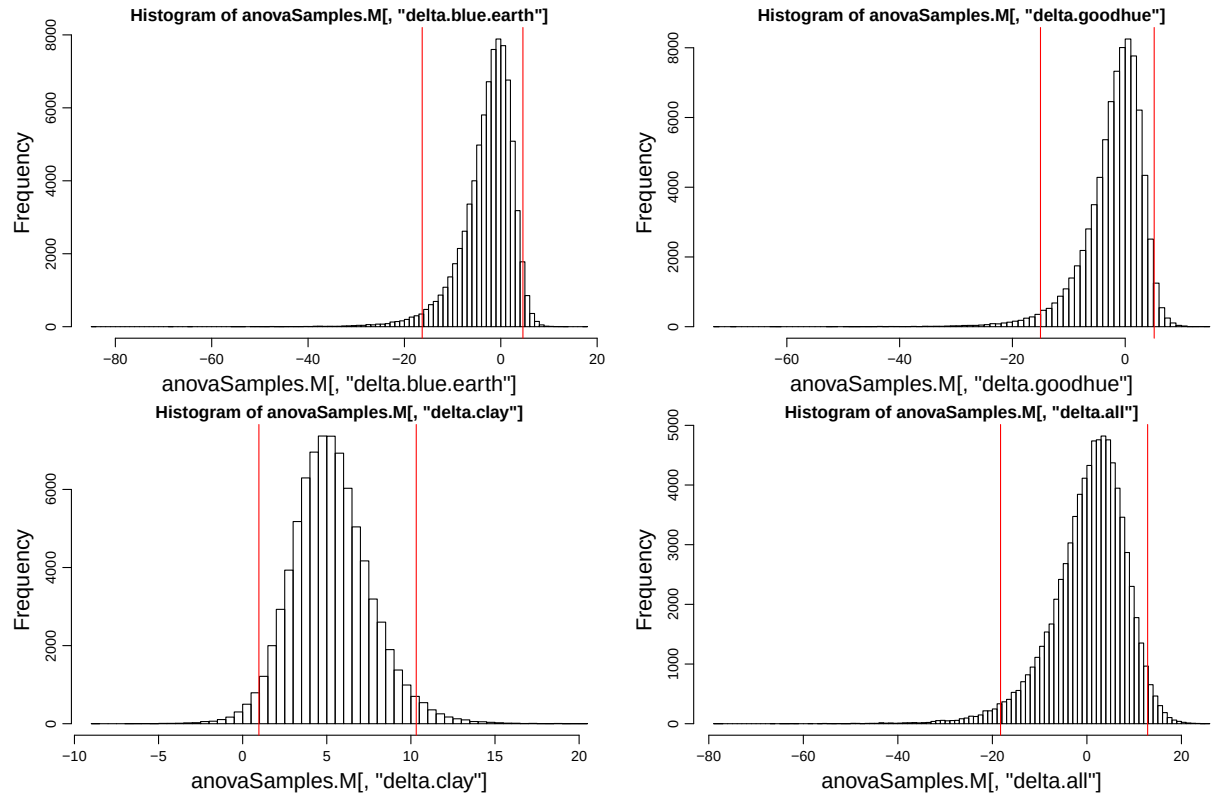
```
##       2.5%      97.5%
## -15.023975   5.139381
```

```
hist(anovaSamples.M[,"delta.goodhue"], breaks = 100)
abline(v=c(delta.goodhue[[1]], delta.goodhue[[2]]), col="red")
goodhue.p.val <-
  length(anovaSamples.M[,"delta.goodhue"][which(anovaSamples.M[,"delta.goodhue"] > 0)]) /
  length(anovaSamples.M[,"delta.goodhue"])

delta.all <- quantile(anovaSamples.M[,"delta.all"], probs = c(0.025, 0.975))
delta.all
```

```
##      2.5%     97.5%
## -18.27455  12.84836
```

```
hist(anovaSamples.M[,"delta.all"], breaks = 100)
abline(v=c(delta.all[[1]], delta.all[[2]]), col="red")
```



```
all.p.val <-
  length(anovaSamples.M[,"delta.all"][which(anovaSamples.M[,"delta.all"] > 0)]) /
  length(anovaSamples.M[,"delta.all"])
```

H_0 : Houses with basements do not have greater radon measurements than houses with basements.

H_1 : Houses with basements have greater radon measurements than houses without.

$p(radon_{basements} - radon_{nobasement} \leq 0) = 0.421175$

The p-value is not in the critical region, and thus we cannot reject the null hypothesis. There appears to be no difference, on average, in radon measurements for houses with basements vs. houses without (However, in Clay county, this is clearly not the case, with a p-value of 0.0097875).

■