Project Report - MAS8405

Lloyd Bates

25/02/2022

Data Description

Original Data

```
# First few entries of data
head(as.data.frame(Reisby))
```

```
##    id hd week   lnimi   lndmi female reactive_depression
## 1 101 23    0 4.04305 4.20469      0                    1
## 2 101 12    1 3.93183 4.81218      0                    1
## 3 101  9    2 4.33073 4.96284      0                    1
## 4 101  8    3 4.36945 4.96284      0                    1
## 5 103 13    0 2.77259 5.23644      1                    1
## 6 103 22    1 3.46574 5.20949      1                    1
```

The data set contains 250 observations of 7 variables: “id”,“hd”,“week”,“lnimi” “lndmi”,“female”,“reactive_depression”. Here one of the main variables that we will be observing is “lnimi” which represents the log concentration the antidepressant drug Imipramine (IMI) in a patients blood. The question of this report is observe the effectiveness of this drug on a patients depression.

Data Preprocessing

01-A File

Before analysis can be completed it is important to format the data correctly, one detail about the data set is that each row is not a unique person in total there are:

```
# Number of different patients
groups = unique(reisby$id)
length(groups)
```

```
## [1] 66
```

```
# Re-encode id vector
reisby$id = match((reisby$id), groups)
```

This number means that each person was not measured every week. As well the id column has been modified to go from 1:66. Further data transformations have taken place including: converting the week column to a 0/1 representing a placebo week or a week where the drug was administered, also, the Hamilton depression index has been encoded to show the 4 possible levels of depression. Finally, for both a normalized and raw format, the data as been split into test and train data. Note groups are not split as train[200] is 53 and test[201] = 54

```
head(reisby)
```

```
##   id hd week   lnimi   lndmi female reactive_depression
## 1  1  2    0 4.04305 4.20469      0                    1
## 2  1  1    0 3.93183 4.81218      0                    1
## 3  1  1    0 4.33073 4.96284      0                    1
## 4  1  1    0 4.36945 4.96284      0                    1
## 5  2  1    0 2.77259 5.23644      1                    1
## 6  2  2    0 3.46574 5.20949      1                    1
```

Next the structure of the data frame can be viewed to see if how the variables are stored:

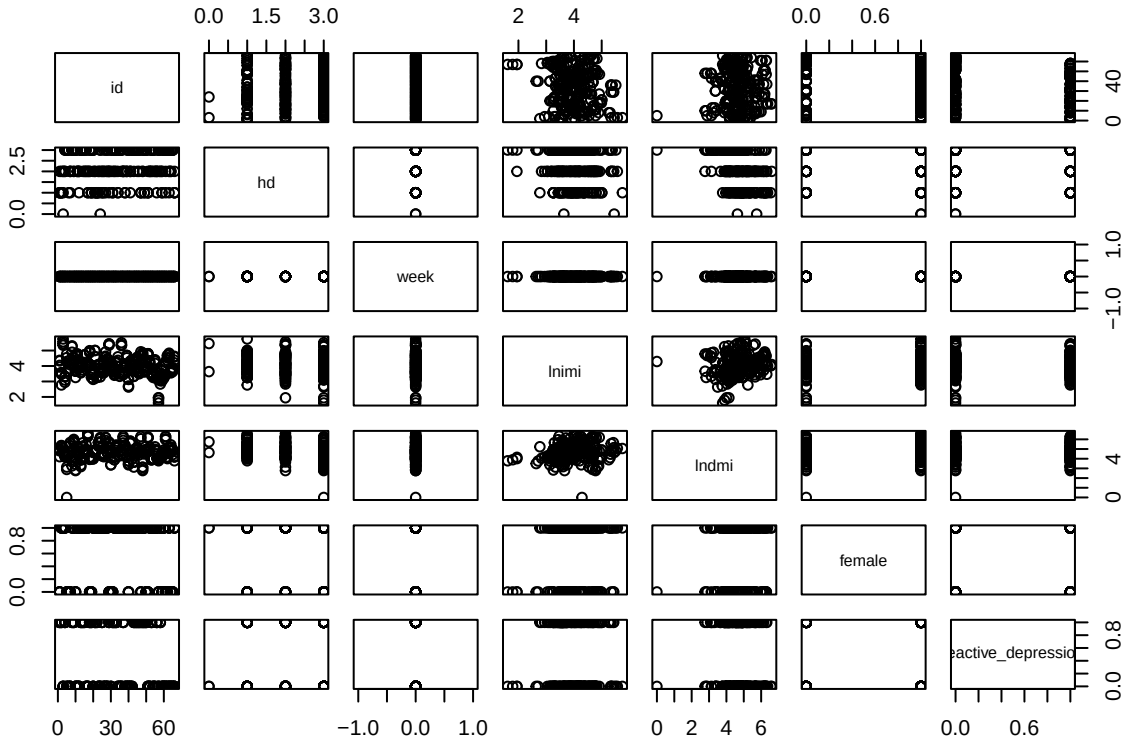
```
# View how data is stored
str(reisby)
```

```
## 'data.frame':    250 obs. of  7 variables:
##  $ id                 : int  1 1 1 1 2 2 2 2 3 3 ...
##  $ hd                 : num  2 1 1 1 1 2 1 1 2 1 ...
##  $ week               : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ lnimi              : num  4.04 3.93 4.33 4.37 2.77 ...
##  $ lndmi              : num  4.2 4.81 4.96 4.96 5.24 ...
##  $ female             : num  0 0 0 0 1 1 1 1 1 1 ...
##  $ reactive_depression: num  1 1 1 1 1 1 1 1 0 0 ...
```

Exploritory Data Analysis

Firstly to view any relationships between the variables the sactterplotmattrix can be viewed:

```
# View scatterplot matrix
pairs(reisby)
```



Regression to Predict Efficacy of Imipramine

Fit a multiple linear regression to the reisby data set and report the posterior mean and a 95% HPD interval for each parameter. To test the model different priors will be used with both a standardized and raw version of the data. We will use the lnimi as the response.

Jags Code

For the first multiple linear regression the JAGS code is:

```
modelstring="
  model {
    k = 10^3
    b0 ~ dnorm(0, k)

    for (j in 1:p) {
      b[j] ~ dnorm(0, k)
    }

    tau ~ dgamma(0.001, 0.001)
    sd = pow(tau, -0.5)

    for (i in 1:N) {
```

```
      y[i] ~ dnorm(mu[i], tau)
      mu[i] = b0 + inprod(b, x[i,])
    }
}"
```

Prior Selection

For the first model the prior used will be a uninformed, normal distribution as to make b's to be almost flat.

R Code

```
y = reisby$lndmi
x = reisby[,-4]
x = x[,-1]

jags_data = list(y=y, x=x, N=nrow(x), p=ncol(x))
model = jags.model(textConnection(modelstring), data=jags_data, n.chains=4)
```

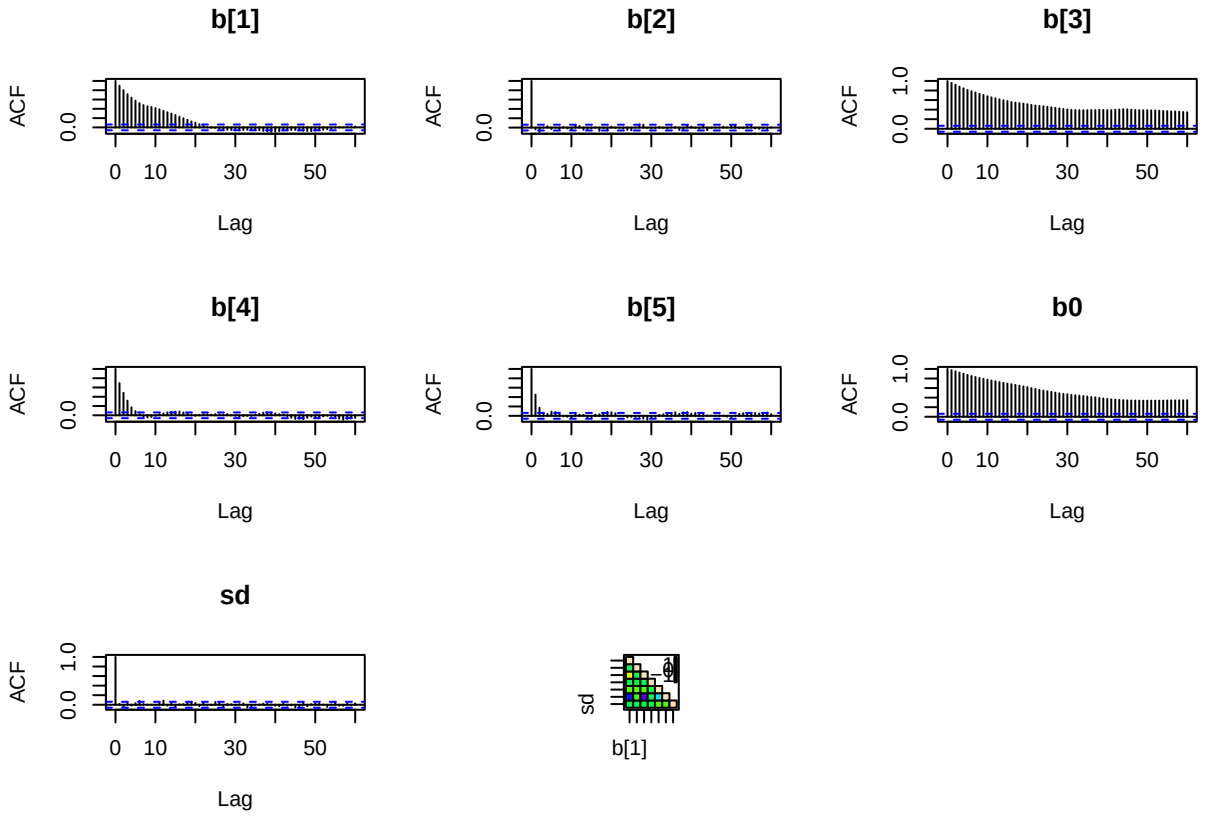
```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 250
##    Unobserved stochastic nodes: 7
##    Total graph size: 2228
##
## Initializing model
```

```
update(model, n.iter=1000)
samples = coda.samples(model, variable.names=c("b0", "sd", "b"),n.iter=1000)
```

MCMC Diagnostics

Note for this first model only will the diagnostics be shown, it can be assumed that in all following models the same diagnostics will be used.

```
# ACF
mcmc_mat = as.matrix(samples[[1]])
par(mfrow=c(3,3))
for (i in 1:7) {
  acf(mcmc_mat[,i], lag.max=60, main=colnames(mcmc_mat)[i])
}
crosscorr.plot(samples)
```



From the acf plots it is clear that lag 30 is still significant, so a thinning interval of around 32 should be sufficient, now the new JAGS code will look like:

```
jags_data = list(y=y, x=x, N=nrow(x), p=ncol(x))
model = jags.model(textConnection(modelstring), data=jags_data, n.chains=4)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 250
##    Unobserved stochastic nodes: 7
##    Total graph size: 2228
##
## Initializing model
```

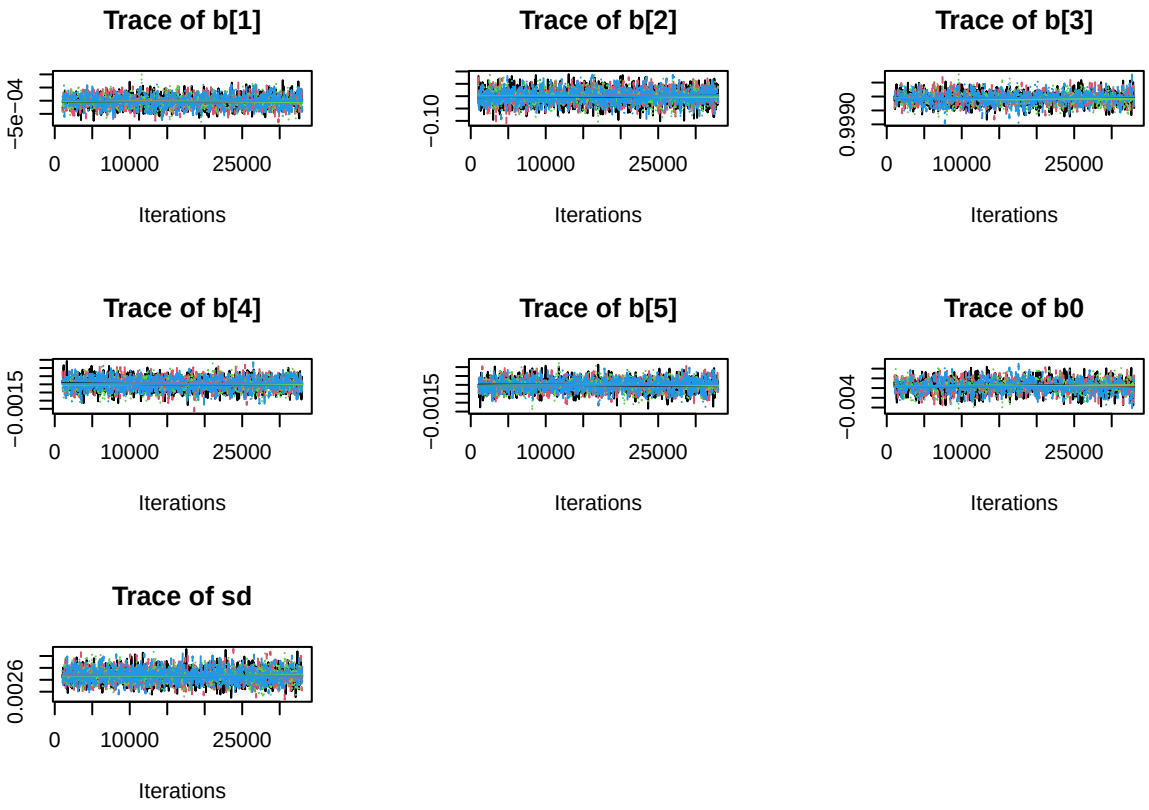
```
update(model, n.iter=1000)
th = 32
samples = coda.samples(model,
                       variable.names=c("b0", "sd", "b"), thin=th,
                       n.iter=th*1000)

# View how well the chains mixed
gelman.diag(samples, multivariate = FALSE)
```

```
## Potential scale reduction factors:
```

```
##
##      Point est. Upper C.I.
## b[1]       1.00       1.00
## b[2]       1.00       1.00
## b[3]       1.01       1.02
## b[4]       1.00       1.00
## b[5]       1.00       1.01
## b0         1.01       1.02
## sd         1.00       1.01
```

```
# Trace
par(mfrow=c(3,3))
plot(samples, auto.layout=FALSE, density=FALSE)
```



Summary

From the mcmc output the following summaries can be found:

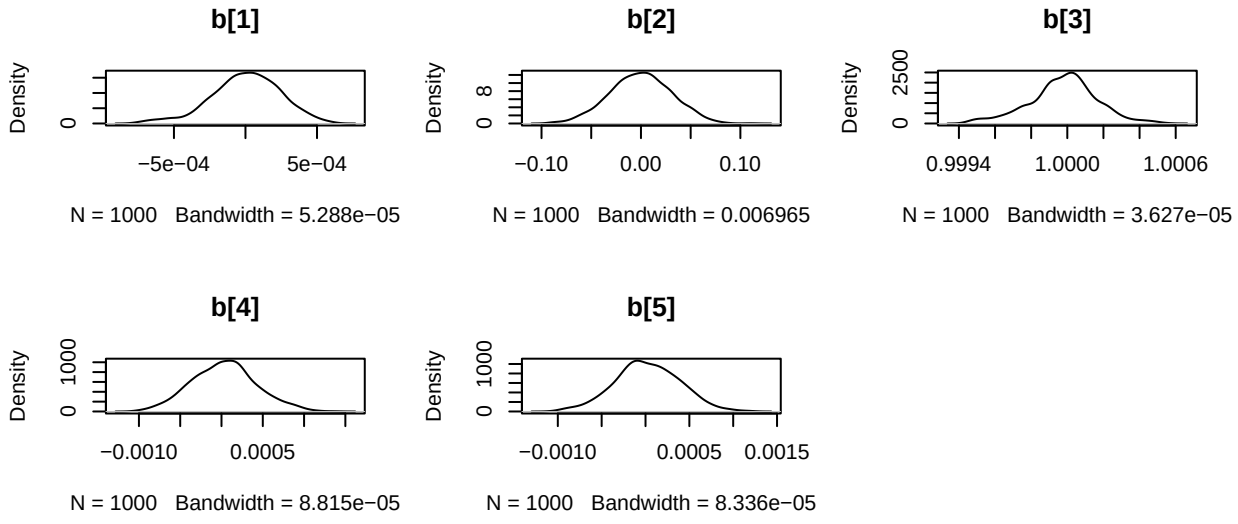
```
## Mean deviance:  -2463
## penalty 6.167
## Penalized deviance: -2457
```

```
##              lower        upper          mean
## b[1] -0.0004988776 0.0004440358 -3.108843e-05
```

```
## b[2] -0.0601772392 0.0646975886 -5.083504e-04
## b[3]  0.9994549328 1.0003763739  9.999327e-01
## b[4] -0.0007178183 0.0007992537  2.065917e-05
## b[5] -0.0007122569 0.0007350541 -1.668417e-05
```

Posterior density plots based on MCMC output:

```
par(mfrow=c(3,3))
for (i in 1:5) {
    d = density(mcmc_mat[,i])
    plot(d, main=colnames(mcmc_mat)[i])
}
```



These show that the posterior distributions are approximately normal.