Question 1: Exercise 4.1, p.217, Carlin & Louis.

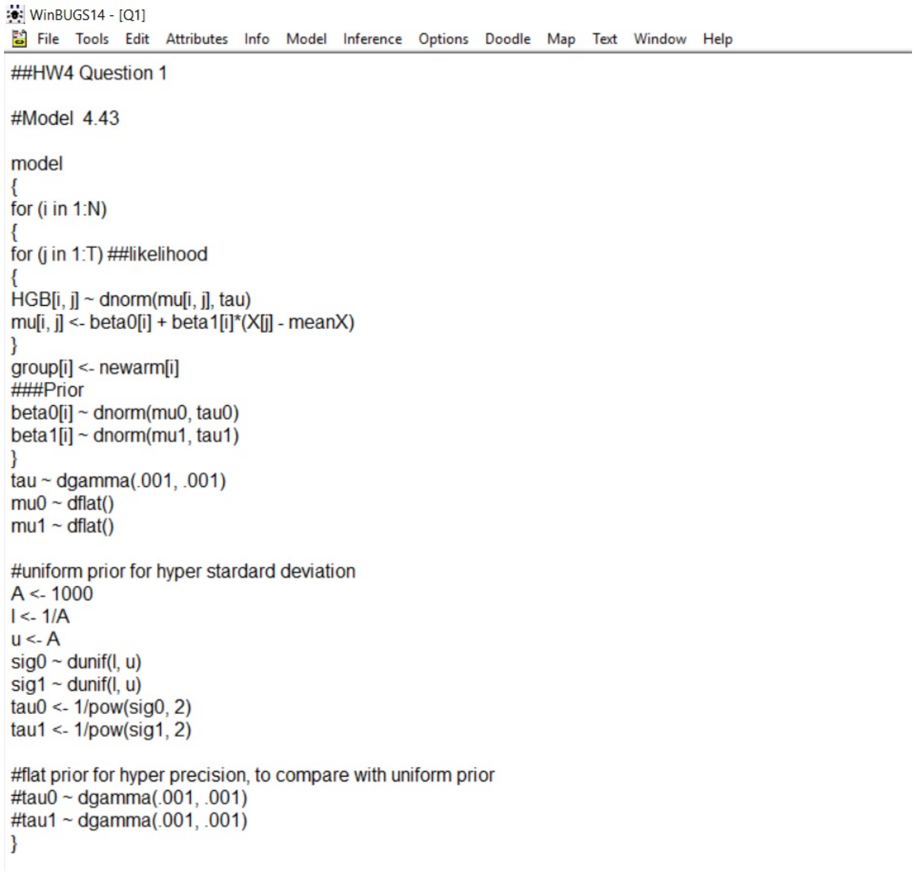
Solution 1. *For part (a):*

Given the hierarchical model:

$$Y_{ij} = \beta_{0i} + \beta_{1i}(X_j - \mu_X) + \epsilon_{ij}, i = 1, \ldots, 365, j = 1, \ldots, 22,$$

where $X_j = j, \mu_X = \frac{1}{22}\sum_{j=1}^{22} X_j = 11.5$, $\epsilon_{ij} \overset{iid}{\sim} N(0, \tau)$, $\beta_{0i} \overset{iid}{\sim} N(\mu_0, \tau_0)$, *and* $\beta_{1i} \overset{iid}{\sim} N(\mu_1, \tau_1)$. *Here* τ, τ_0 *and* τ_1 *represent precisions.*

We fit the model in WinBUGS:


WinBUGS14 - [Q1]
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Help

```
##HW4 Question 1

#Model 4.43

model
{
for (i in 1:N)
{
for (j in 1:T) ##likelihood
{
HGB[i, j] ~ dnorm(mu[i, j], tau)
mu[i, j] <- beta0[i] + beta1[i]*(X[j] - meanX)
}
group[i] <- newarm[i]
###Prior
beta0[i] ~ dnorm(mu0, tau0)
beta1[i] ~ dnorm(mu1, tau1)
}
tau ~ dgamma(.001, .001)
mu0 ~ dflat()
mu1 ~ dflat()

#uniform prior for hyper stardard deviation
A <- 1000
l <- 1/A
u <- A
sig0 ~ dunif(l, u)
sig1 ~ dunif(l, u)
tau0 <- 1/pow(sig0, 2)
tau1 <- 1/pow(sig1, 2)

#flat prior for hyper precision, to compare with uniform prior
#tau0 ~ dgamma(.001, .001)
#tau1 ~ dgamma(.001, .001)
}
```

For the choice of priors, we have:

$$\mu_0 \sim dflat(), \mu_1 \sim dflat(), \tau \sim dgamma(0.001, 0.001)$$

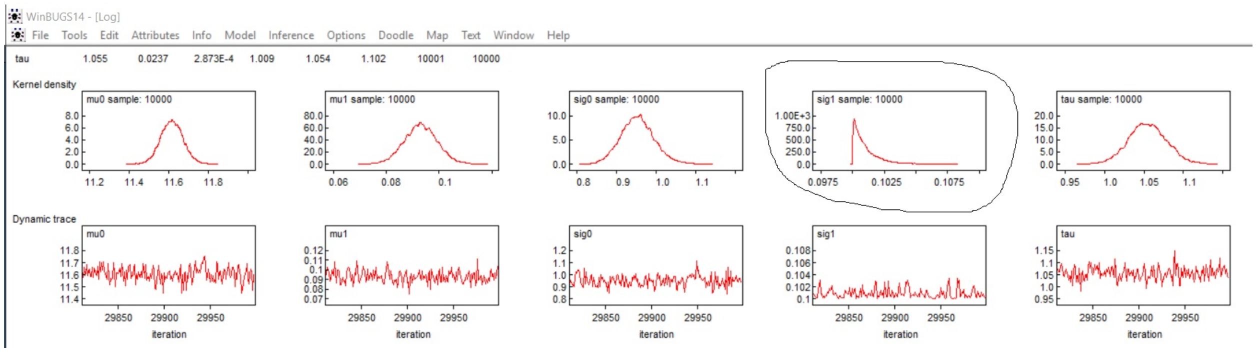
As for τ_0 *and* τ_1, *we first consider the uniform prior for standard deviations:*

$$sig0 \sim dunif(\frac{1}{A}, A), sig1 \sim dunif(\frac{1}{A}, A)$$

and then let

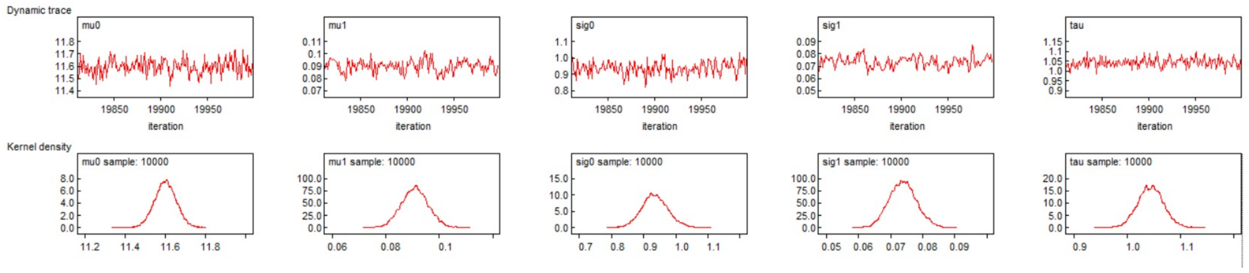
$$\tau_0 = \frac{1}{pow(sig0, 2)} \text{ and } \tau_1 = \frac{1}{pow(sig1, 2)}$$

I tried different values of A. For example, when $A = 10$, the uniform prior is $U(0.1, 10)$ and we find that the density of sig1 exhibits a "truncated" shape:



```
tau     1.055   0.0237  2.873E-4  1.009   1.054   1.102   10001   10000
```

This indicates that A is not large enough. Eventually we chose $A = 1000$, and it looks much better on the density plot:

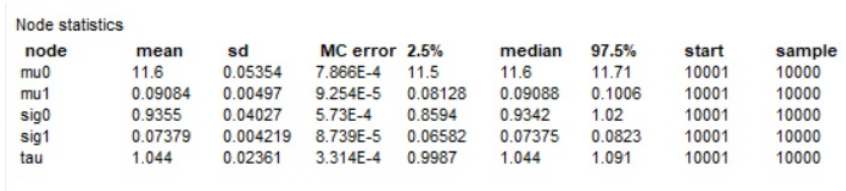


On the other hand, we also considered the choice of flat prior directly on τ_0 and τ_1:

$$\tau_0 \sim dgamma(0.001, 0.001) \text{ and } \tau_1 \sim dgamma(0.001, 0.001)$$

It turns out, the point estimate, as well as the 95% credible intervals are quite close to the results we got from using the uniform prior on τ_0 and τ_1.

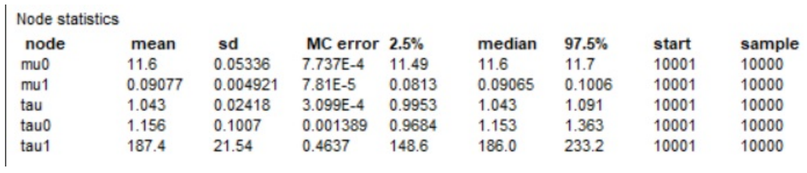
From uniform prior on sig0 and sig1:



Node statistics

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
mu0	11.6	0.05354	7.866E-4	11.5	11.6	11.71	10001	10000
mu1	0.09084	0.00497	9.254E-5	0.08128	0.09088	0.1006	10001	10000
sig0	0.9355	0.04027	5.73E-4	0.8594	0.9342	1.02	10001	10000
sig1	0.07379	0.004219	8.739E-5	0.06582	0.07375	0.0823	10001	10000
tau	1.044	0.02361	3.314E-4	0.9987	1.044	1.091	10001	10000

From flat prior on τ_0 and τ_1:



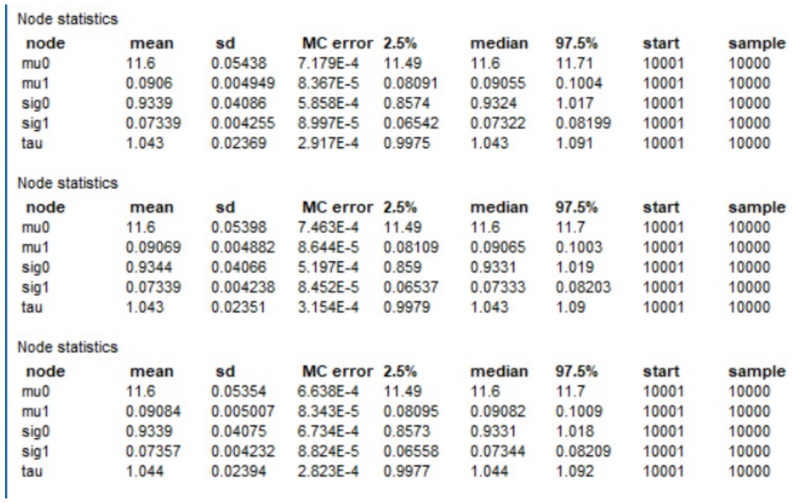
Node statistics

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
mu0	11.6	0.05336	7.737E-4	11.49	11.6	11.7	10001	10000
mu1	0.09077	0.004921	7.81E-5	0.0813	0.09065	0.1006	10001	10000
tau	1.043	0.02418	3.099E-4	0.9953	1.043	1.091	10001	10000
tau0	1.156	0.1007	0.001389	0.9684	1.153	1.363	10001	10000
tau1	187.4	21.54	0.4637	148.6	186.0	233.2	10001	10000

Throughout the rest of solution for Question 1*, we generate results using the uniform prior for sig0 and sig1, per suggestion from class, as well as by Gelman.*

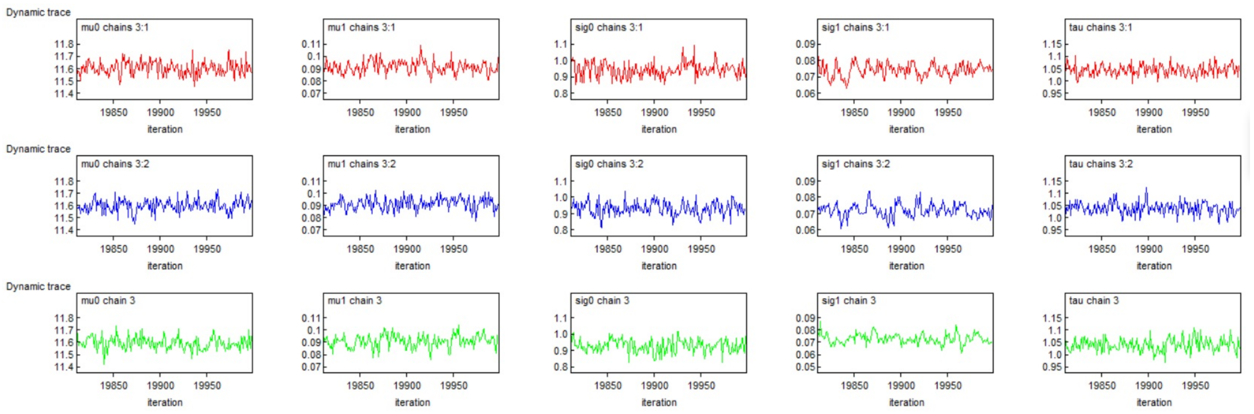
We also consider three chains to help better assess convergence. We simulate 10000 *times for each chain, after burning the first* 10000 *iterations.*

For point estimates and credible intervals, all 3 *chains give similar results:*

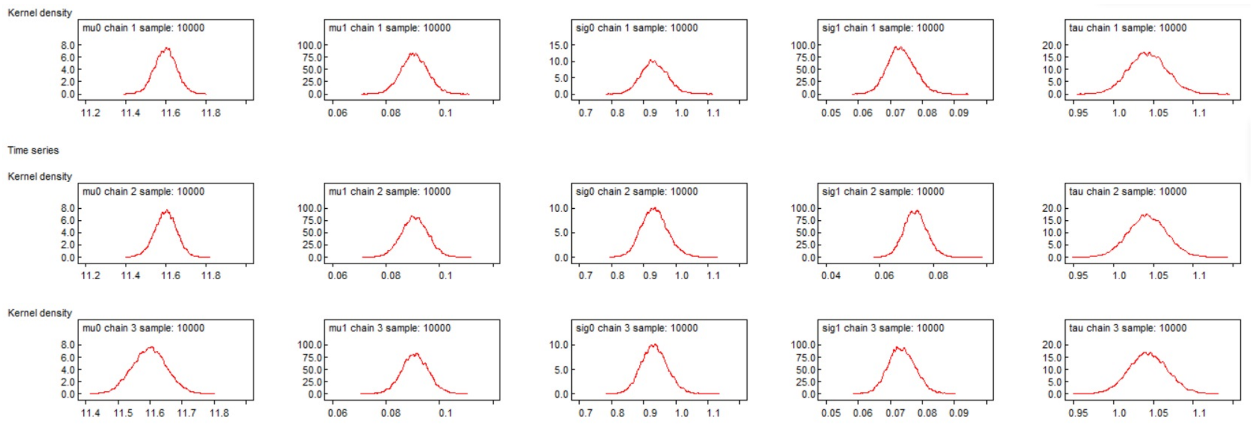


```
Node statistics
 node    mean      sd        MC error  2.5%     median   97.5%    start   sample
 mu0     11.6      0.05438   7.179E-4  11.49    11.6     11.71    10001   10000
 mu1     0.0906    0.004949  8.367E-5  0.08091  0.09055  0.1004   10001   10000
 sig0    0.9339    0.04086   5.858E-4  0.8574   0.9324   1.017    10001   10000
 sig1    0.07339   0.004255  8.997E-5  0.06542  0.07322  0.08199  10001   10000
 tau     1.043     0.02369   2.917E-4  0.9975   1.043    1.091    10001   10000

Node statistics
 node    mean      sd        MC error  2.5%     median   97.5%    start   sample
 mu0     11.6      0.05398   7.463E-4  11.49    11.6     11.7     10001   10000
 mu1     0.09069   0.004882  8.644E-5  0.08109  0.09065  0.1003   10001   10000
 sig0    0.9344    0.04066   5.197E-4  0.859    0.9331   1.019    10001   10000
 sig1    0.07339   0.004238  8.452E-5  0.06537  0.07333  0.08203  10001   10000
 tau     1.043     0.02351   3.154E-4  0.9979   1.043    1.09     10001   10000

Node statistics
 node    mean      sd        MC error  2.5%     median   97.5%    start   sample
 mu0     11.6      0.05354   6.638E-4  11.49    11.6     11.7     10001   10000
 mu1     0.09084   0.005007  8.343E-5  0.08095  0.09082  0.1009   10001   10000
 sig0    0.9339    0.04075   6.734E-4  0.8573   0.9331   1.018    10001   10000
 sig1    0.07357   0.004232  8.824E-5  0.06558  0.07344  0.08209  10001   10000
 tau     1.044     0.02394   2.823E-4  0.9977   1.044    1.092    10001   10000
```

For trace plot to check convergence, all three chains indicate a good sign of convergence.

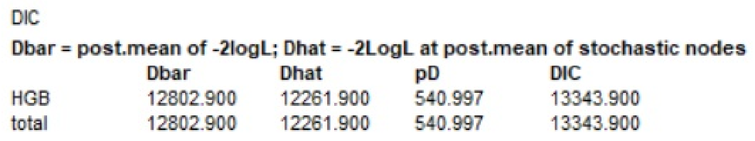


and all three chains have similar density plots for the parameters:

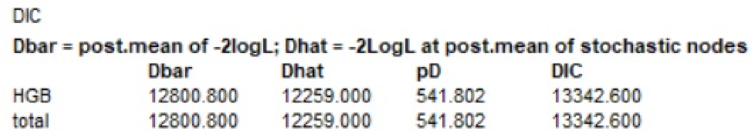


The above plots indicates that the convergence is good, and our results from different chains are consistent.

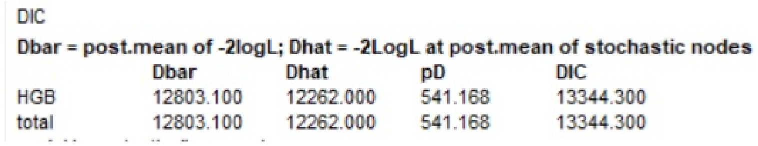
We can also look at DICs (for separate chains): they are also pretty close as well.



```
DIC
Dbar = post.mean of -2logL; Dhat = -2LogL at post.mean of stochastic nodes
            Dbar        Dhat        pD        DIC
HGB         12802.900   12261.900   540.997   13343.900
total       12802.900   12261.900   540.997   13343.900
```



```
DIC
Dbar = post.mean of -2logL; Dhat = -2LogL at post.mean of stochastic nodes
            Dbar        Dhat        pD        DIC
HGB         12800.800   12259.000   541.802   13342.600
total       12800.800   12259.000   541.802   13342.600
```

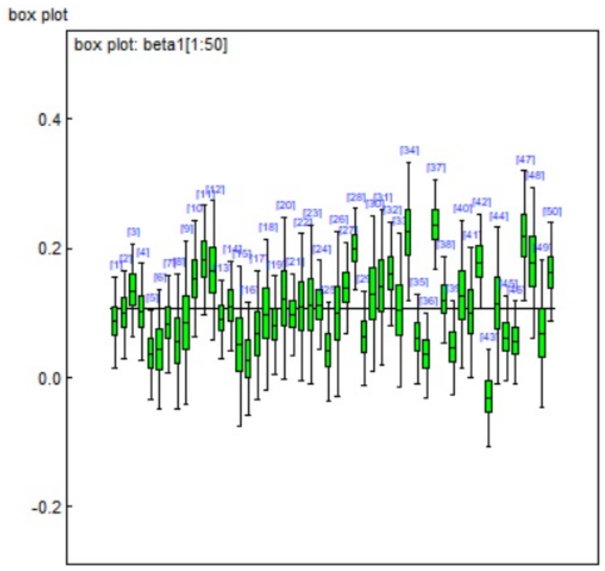


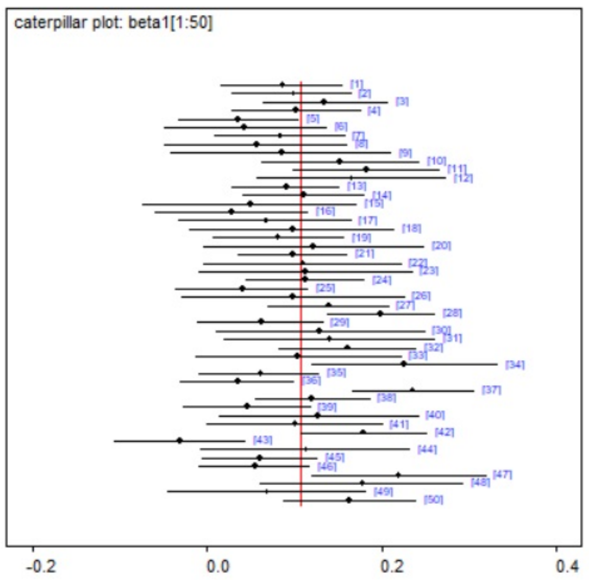
```
DIC
Dbar = post.mean of -2logL; Dhat = -2LogL at post.mean of stochastic nodes
            Dbar        Dhat        pD        DIC
HGB         12803.100   12262.000   541.168   13344.300
total       12803.100   12262.000   541.168   13344.300
```

Since there are 365 estimates for each β_0 and β_1, we did not show them here. To interpret though, the posterior distribution of μ_1 means the mean (or average) rate of change for HGB among all patients posteriori, and the posterior distribution of β_{1i} means the individual rate of change of HGB of a specific patient posteriori.

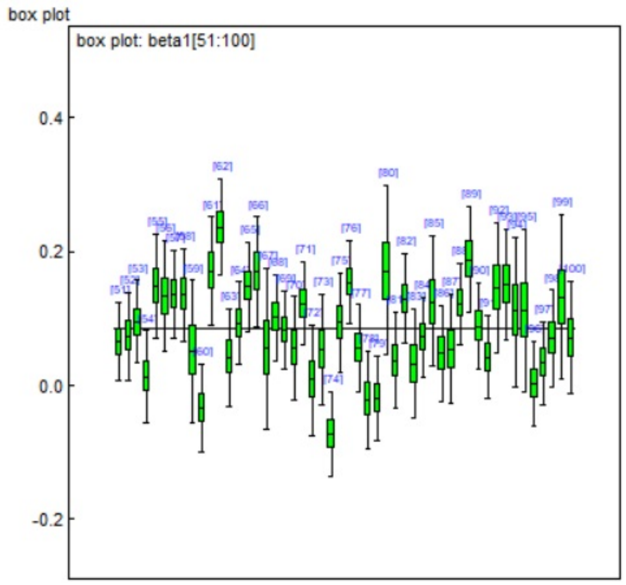
We now use compare function in winBUGS to generate boxplot and caterpillar plot for β_1. Since there are 365 of them, it would be hard to observe if we put all patients information in one plot. Instead we group them by every 50 patients:

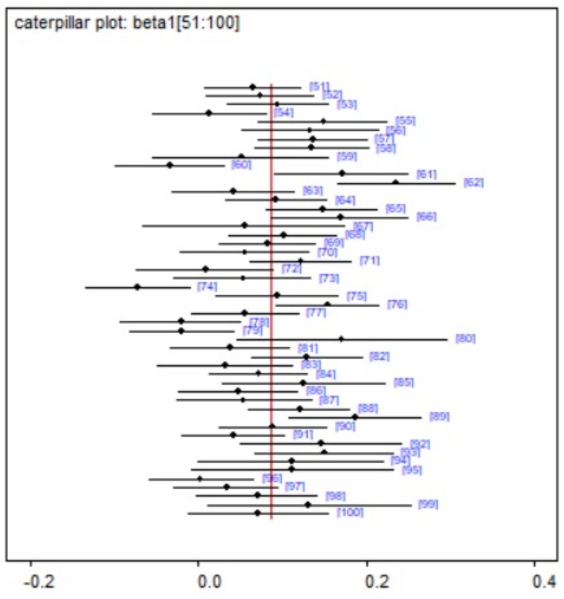
For $\beta_1[1]$ to $\beta_1[50]$:



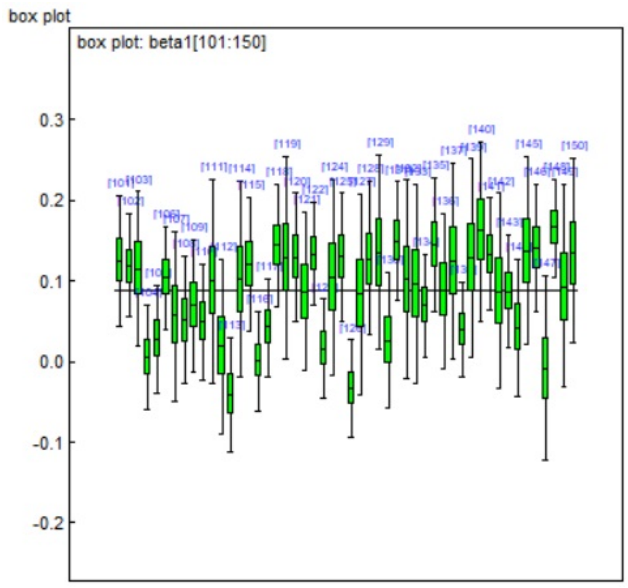


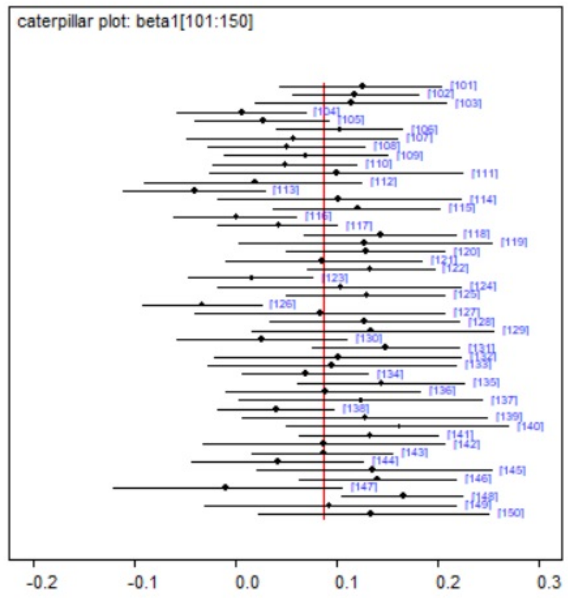
For $\beta_1[51]$ *to* $\beta_1[100]$



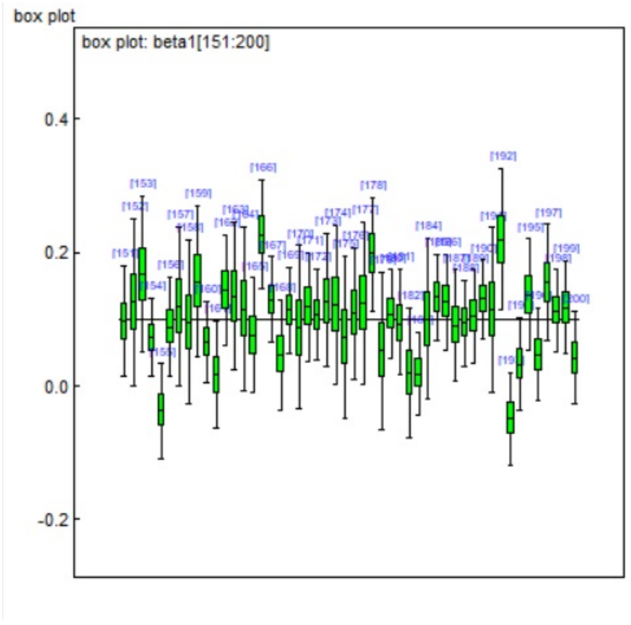


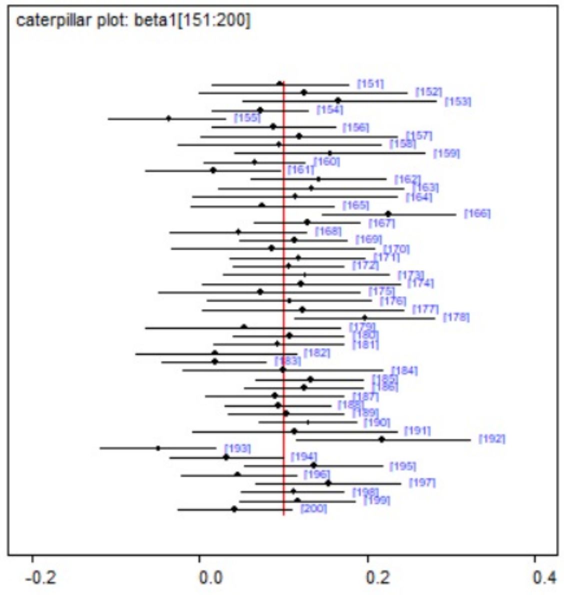
For $\beta_1[101]$ *to* $\beta_1[150]$



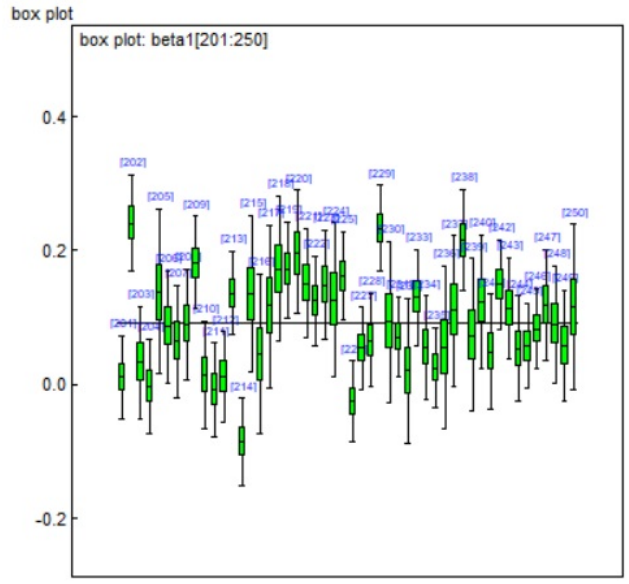


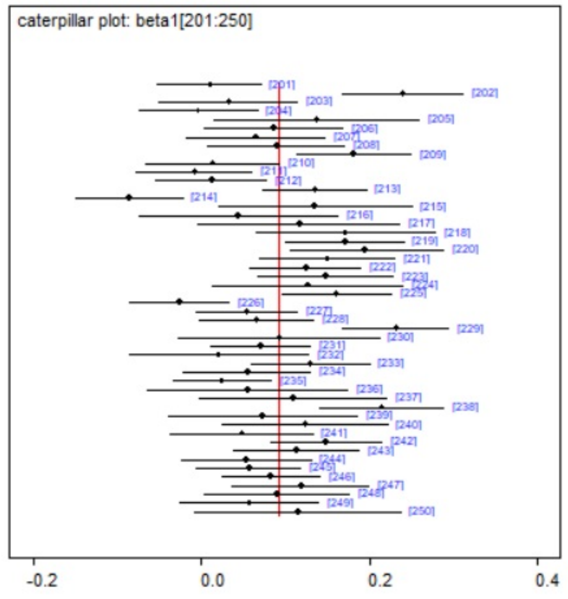
For $\beta_1[151]$ *to* $\beta_1[200]$



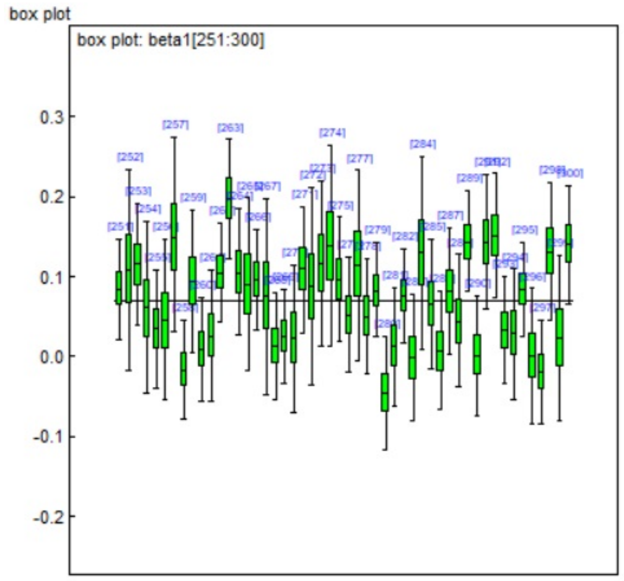


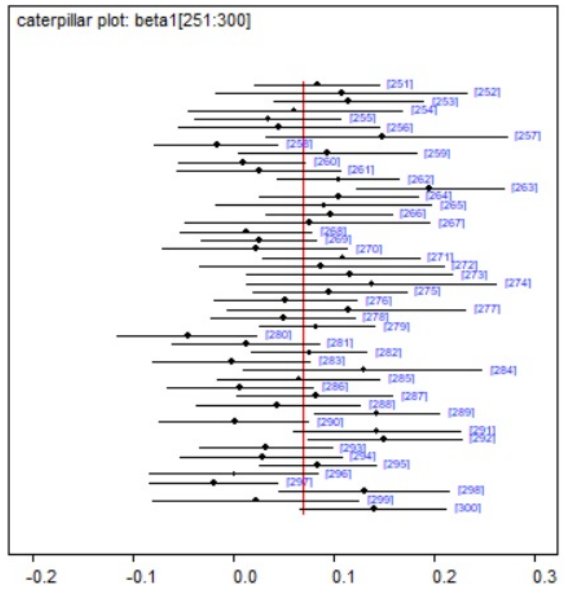
For $\beta_1[201]$ *to* $\beta_1[250]$



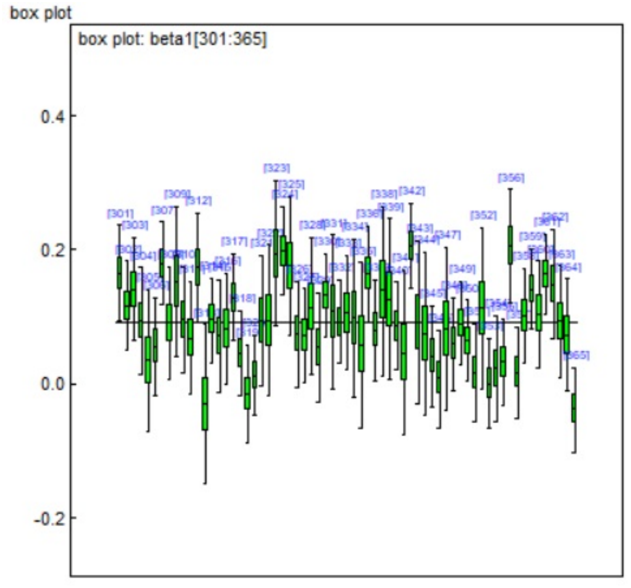


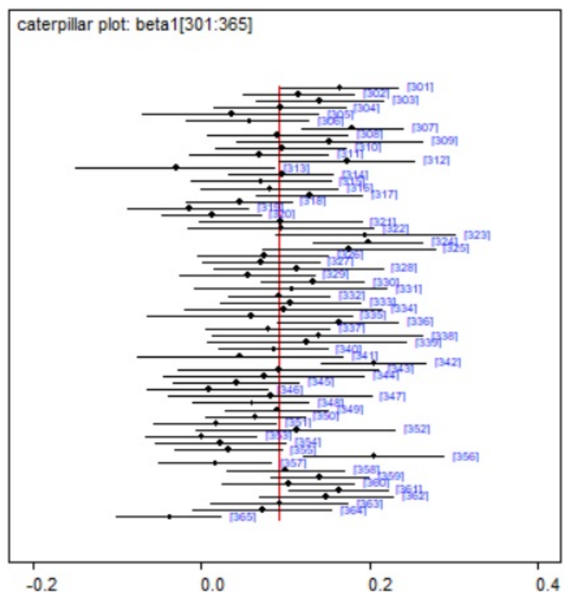
For $\beta_1[251]$ *to* $\beta_1[300]$





For $\beta_1[301]$ *to* $\beta_1[365]$

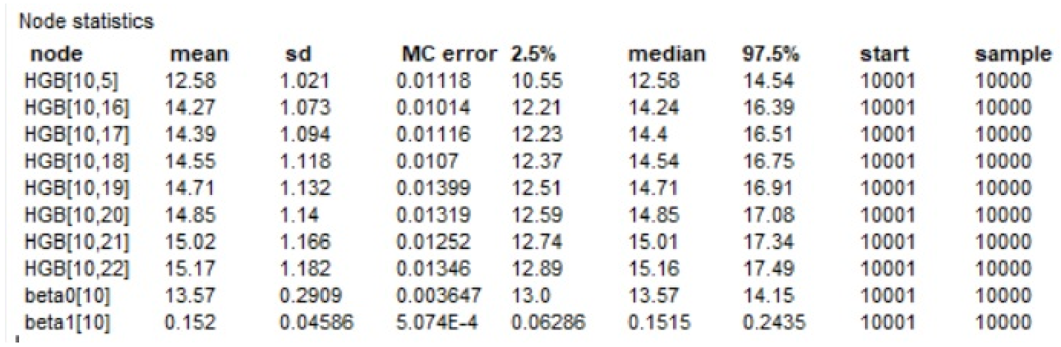




As we can easily observe that, for some patients i, the rate of change $\beta_{1i} > 0$ *and for some others* $\beta_{1i} < 0$. *Some have their whole* 95% *credible interval positive, for example, patients* 202, 209, *and some have their whole* 95% *credible interval negative, for example, patients* 201, 212, 214 *etc. So not all participants HGB measurements are improving over time.*

For part (b):

We get the following estimates for patient number 10 (same code as part (a), but add $HGB[10,]$ *as the stochastic node that we monitor):*



Node statistics

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
HGB[10,5]	12.58	1.021	0.01118	10.55	12.58	14.54	10001	10000
HGB[10,16]	14.27	1.073	0.01014	12.21	14.24	16.39	10001	10000
HGB[10,17]	14.39	1.094	0.01116	12.23	14.4	16.51	10001	10000
HGB[10,18]	14.55	1.118	0.0107	12.37	14.54	16.75	10001	10000
HGB[10,19]	14.71	1.132	0.01399	12.51	14.71	16.91	10001	10000
HGB[10,20]	14.85	1.14	0.01319	12.59	14.85	17.08	10001	10000
HGB[10,21]	15.02	1.166	0.01252	12.74	15.01	17.34	10001	10000
HGB[10,22]	15.17	1.182	0.01346	12.89	15.16	17.49	10001	10000
beta0[10]	13.57	0.2909	0.003647	13.0	13.57	14.15	10001	10000
beta1[10]	0.152	0.04586	5.074E-4	0.06286	0.1515	0.2435	10001	10000

Notice that HGB increase slowly in time, which is consistent with the estimate for $\beta_{1,10}$ *that is positive.*

The estimated standard deviations of the imputed valuesincrease for the later weeks (goes from around 1.00 *to around* 1.20*), after the participant was lost to follow-up; uncertainty increases as we move further from the bulk of the data.*

For part (c):

We improve the model as following:

$$Y_{ij} = \beta_{0i} + \beta_{1i}(X_j - \mu_X) + \epsilon_{ij}, i = 1, \ldots, 365, j = 1, \ldots, 22$$

where $X_j = j$, $\epsilon_{ij} \overset{iid}{\sim} N(0, \tau)$ *and*

$$\beta_{0i} \overset{iid}{\sim} N(\mu_0 + \gamma_0 \cdot (newarm[i] - 1), \tau_0)$$
$$\beta_{1i} \overset{iid}{\sim} N(\mu_1 + \gamma_1 \cdot (newarm[i] - 1), \tau_1)$$

That is to say, for those participants in $newarm[i] = 1$, *we have:*

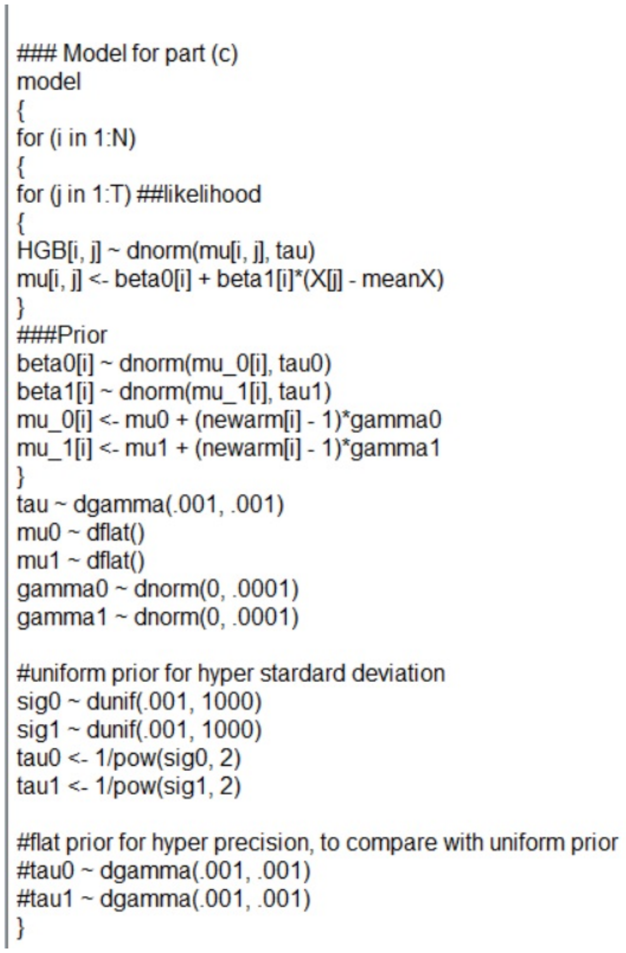
$$\beta_{0i} \overset{iid}{\sim} N(\mu_0, \tau_0)$$
$$\beta_{1i} \overset{iid}{\sim} N(\mu_1, \tau_1)$$

for those participants in $newarm[i] = 2$, *we have:*

$$\beta_{0i} \overset{iid}{\sim} N(\mu_0 + \gamma_0, \tau_0)$$
$$\beta_{1i} \overset{iid}{\sim} N(\mu_1 + \gamma_1, \tau_1)$$

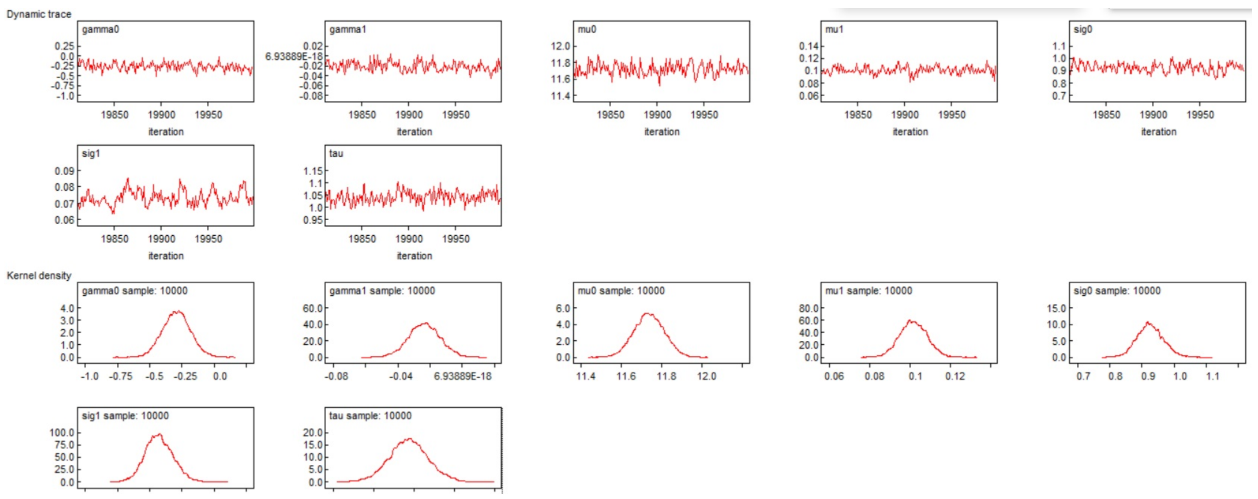
For priors, we still take $\mu_0 \sim dflat(), \mu_1 \sim dflat()$, $\tau \sim dgamma(.001, .001)$, $\sigma_0 \sim dunif(.001, 1000)$, $\sigma_1 \sim dunif(.001, 1000)$, *and we take* $\gamma_0 \sim dnorm(0, .0001)$ *and* $\gamma_1 \sim dnorm(0, .0001)$.

The code for the improved model is:



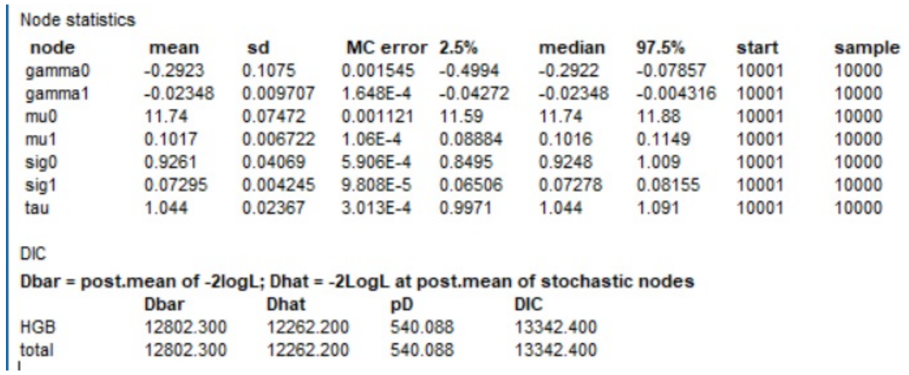
```
### Model for part (c)
model
{
for (i in 1:N)
{
for (j in 1:T) ##likelihood
{
HGB[i, j] ~ dnorm(mu[i, j], tau)
mu[i, j] <- beta0[i] + beta1[i]*(X[j] - meanX)
}
###Prior
beta0[i] ~ dnorm(mu_0[i], tau0)
beta1[i] ~ dnorm(mu_1[i], tau1)
mu_0[i] <- mu0 + (newarm[i] - 1)*gamma0
mu_1[i] <- mu1 + (newarm[i] - 1)*gamma1
}
tau ~ dgamma(.001, .001)
mu0 ~ dflat()
mu1 ~ dflat()
gamma0 ~ dnorm(0, .0001)
gamma1 ~ dnorm(0, .0001)

#uniform prior for hyper stardard deviation
sig0 ~ dunif(.001, 1000)
sig1 ~ dunif(.001, 1000)
tau0 <- 1/pow(sig0, 2)
tau1 <- 1/pow(sig1, 2)

#flat prior for hyper precision, to compare with uniform prior
#tau0 ~ dgamma(.001, .001)
#tau1 ~ dgamma(.001, .001)
}
```

The trace plot and density plot is:



The convergence seems to be doing pretty well.(we burned first 10000 *iterations.)*

The estimate and DIC is:



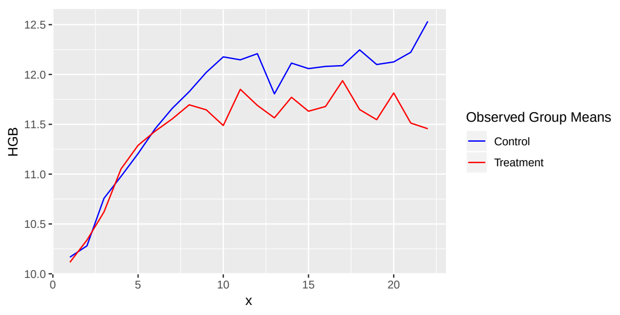
```
Node statistics
 node      mean      sd        MC error  2.5%      median    97.5%      start   sample
 gamma0    -0.2923   0.1075    0.001545  -0.4994   -0.2922   -0.07857   10001   10000
 gamma1    -0.02348  0.009707  1.648E-4  -0.04272  -0.02348  -0.004316  10001   10000
 mu0       11.74     0.07472   0.001121  11.59     11.74     11.88      10001   10000
 mu1       0.1017    0.006722  1.06E-4   0.08884   0.1016    0.1149     10001   10000
 sig0      0.9261    0.04069   5.906E-4  0.8495    0.9248    1.009      10001   10000
 sig1      0.07295   0.004245  9.808E-5  0.06506   0.07278   0.08155    10001   10000
 tau       1.044     0.02367   3.013E-4  0.9971    1.044     1.091      10001   10000

DIC
Dbar = post.mean of -2logL; Dhat = -2LogL at post.mean of stochastic nodes
          Dbar        Dhat        pD       DIC
HGB       12802.300   12262.200   540.088  13342.400
total     12802.300   12262.200   540.088  13342.400
```

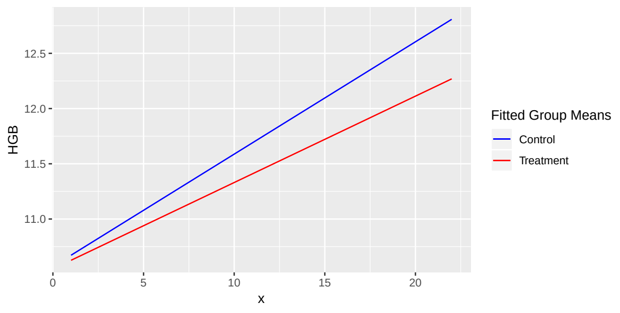
The DIC is very close to the reduced model (slightly smaller than all three DICs from different chains), this suggests that we may not really need the "complete" model here.

From the estimates we see that $\gamma_0(-0.29)$ and $\gamma_1(-0.02)$ are both significantly negative, but compared to the grand intercept ($\mu_0 = 11.74$) and slope ($\mu_1 = 0.10$) without treatment effect, the difference is not very significant.

The following R code plot graphs of fitted grand means and observed grand means between treatment groups. (I call those with newarm$[i] = 1$ as control and those with newarm$[i] = 2$ as treatment).

```r
#HW4

#Question 1

library(ggplot2)

X <- c(1:22); meanX <- mean(X); N <- 365; Time <- 22
mu0 <- 11.74; mu1 <-0.1017; gamma0 <- -0.2923; gamma1 <- -0.02348

Control <- function(x){
  d <-mu0 +mu1*(x - meanX)
}
Treatment <- function(x){
  d <- mu0 + gamma0 + (mu1 + gamma1)*(x - meanX)
}

p <- ggplot(data = data.frame(x = 0), mapping = aes(x = x)) +
  stat_function(fun=Control, geom="line", aes(colour="Control")) +
  stat_function(fun=Treatment, geom="line", aes(colour="Treatment")) +
  labs(y = "HGB") +
  #theme_grey() +
  scale_x_continuous(limits = c(1, 22)) +
  #scale_y_continuous(limits = c(10.5, 13.5))
  scale_color_manual(name = "Fitted Group Means",
                     values = c("blue", "red"), # Color specification
                     labels = c("Control", "Treatment"))

#read in the raw data of Question 1
Q1data <- read.table("C:\\akira\\data\\Q1_rawdata.txt", header = FALSE)

#create the data frame to be used in ggplot
Q1data_control <- subset(Q1data, Q1data$V1 == 1)
Q1data_treatment <- subset(Q1data, Q1data$V1 == 2)
Observed_control <- colMeans(Q1data_control, na.rm = TRUE)
Observed_treatment <- colMeans(Q1data_treatment, na.rm = TRUE)
Observed_control <- Observed_control[2:23]
Observed_treatment <- Observed_treatment[2:23]
Q1observed <- data.frame("Control" = Observed_control,
                         "Treatment" = Observed_treatment,
                         x = 1:22)
library(tidyr)
Q1observed_long <- gather(Q1observed, Treatment, HGB,
                          Control:Treatment, factor_key = TRUE)
p2 <- ggplot(Q1observed_long, mapping = aes(x = x, y = HGB, colour = Treatment)) +
      geom_line() + scale_x_continuous(limits = c(1, 22)) +
      scale_color_manual(name = "Observed Group Means",
                         values = c("blue", "red"), # Color specification
                         labels = c("Control", "Treatment"))
library(gridExtra)

grid.arrange(p2, p, nrow = 2)
```





As we can see from the plot that the two groups are pretty close at baseline, but due to the different slopes the treatment effects are different over the time.

Question 2. Parts (a) and (c) of Exercise 4.5, p.220, Carlin & Louis.

Solution 2. *For part* (a)*:*

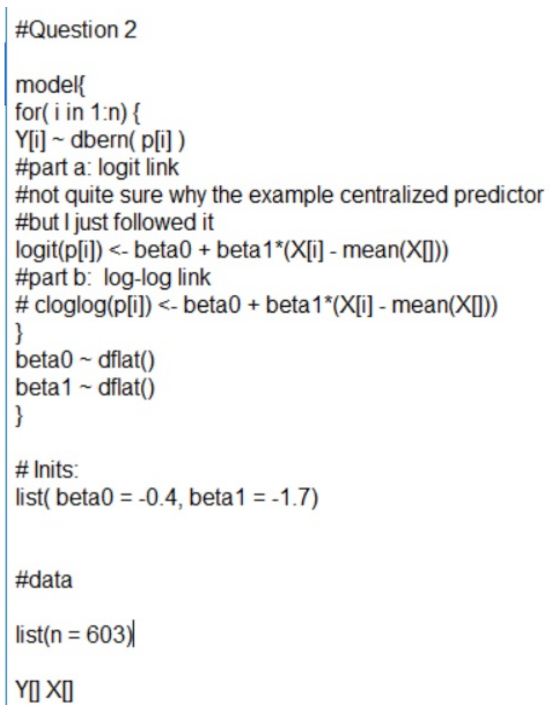
Our model is:

$$logit(p_i) = \beta_0 + \beta_1 X_i, i = 1, \dots, n$$

with $n = 602$*. Follow the code from example* 4.4*, we also centralize* X *so the model becomes*

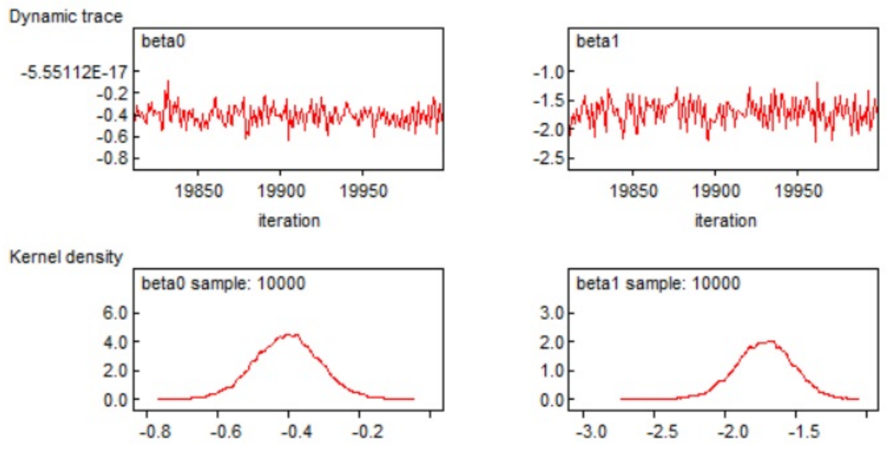
$$logit(p_i) = \beta_0 + \beta_1(X_i - mean(X))$$

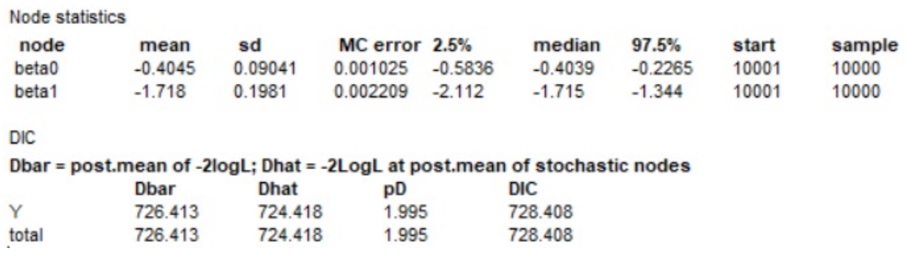
The winBUGs code is:



```
#Question 2

model{
for( i in 1:n) {
Y[i] ~ dbern( p[i] )
#part a: logit link
#not quite sure why the example centralized predictor
#but I just followed it
logit(p[i]) <- beta0 + beta1*(X[i] - mean(X[]))
#part b:  log-log link
# cloglog(p[i]) <- beta0 + beta1*(X[i] - mean(X[]))
}
beta0 ~ dflat()
beta1 ~ dflat()
}

# Inits:
list( beta0 = -0.4, beta1 = -1.7)


#data

list(n = 603)

Y[] X[]
```

We burn the first 10000 *iterations, got the following output:*





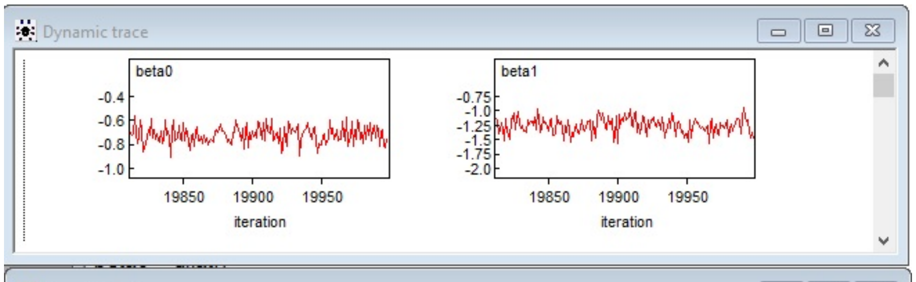
```
Node statistics
 node       mean      sd        MC error  2.5%      median    97.5%     start    sample
 beta0      -0.4045   0.09041   0.001025  -0.5836   -0.4039   -0.2265   10001    10000
 beta1      -1.718    0.1981    0.002209  -2.112    -1.715    -1.344    10001    10000

DIC
Dbar = post.mean of -2logL; Dhat = -2LogL at post.mean of stochastic nodes
           Dbar      Dhat      pD        DIC
Y          726.413   724.418   1.995     728.408
total      726.413   724.418   1.995     728.408
```

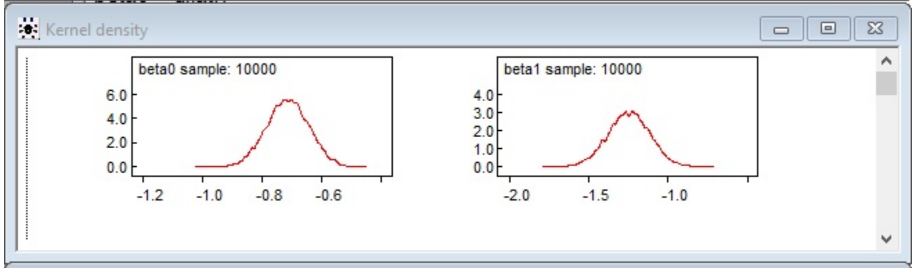
From the trace and density plot we think the convergence is done very well. The estimate of β_1 *is* -1.718 *with* 95% *credible interval as* $(-2.112, -1.344)$. *This tells that* β_1 *is significantly negative.*

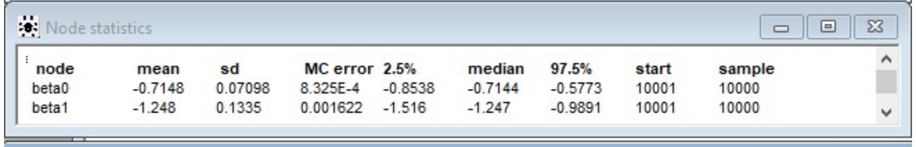
(If the log odds has negative rate of change, so is the odds. This is to say that as the distance to the forest edge increase, the odds of two species co-presence decrease.)

Now for part (c), replace logit link with the complementary log-log link $log[-log(1-p_i)]$. *The code is the same as above except we comment out the line for logit link and use the line for complementary log-log link.*

The output is as following (burn first 10000 *iterations, then sample* 10000 *more times):*

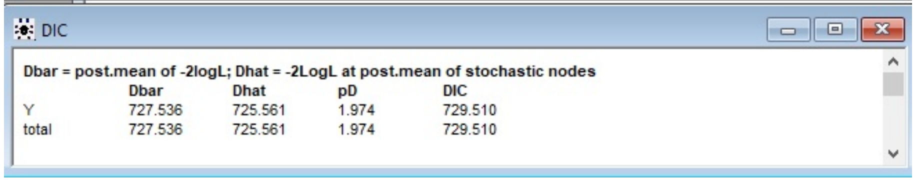






Node statistics

```
 node     mean     sd       MC error  2.5%     median   97.5%    start   sample
 beta0    -0.7148  0.07098  8.325E-4  -0.8538  -0.7144  -0.5773  10001   10000
 beta1    -1.248   0.1335   0.001622  -1.516   -1.247   -0.9891  10001   10000
```



DIC

```
Dbar = post.mean of -2logL; Dhat = -2LogL at post.mean of stochastic nodes
        Dbar      Dhat      pD      DIC
Y       727.536   725.561   1.974   729.510
total   727.536   725.561   1.974   729.510
```

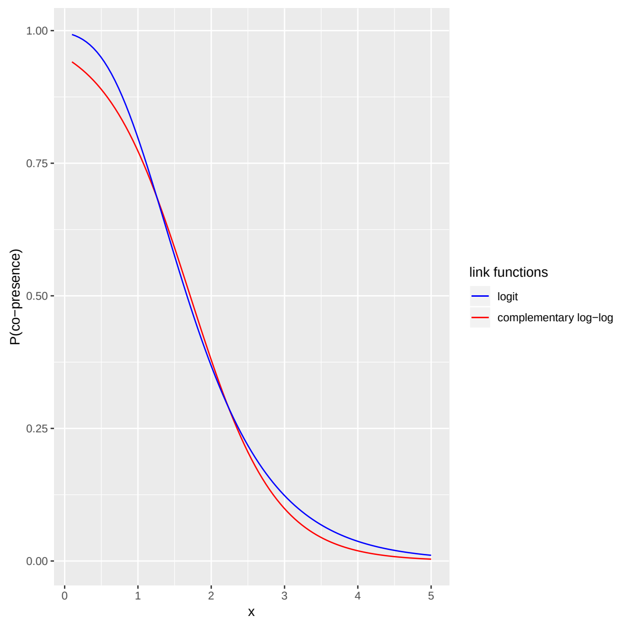
So the estimate for β_1 *is* -1.248, *with* 95% *credible interval as* $(-1.516, -0.989)$. *Thus under the complementary log-log link,* β_1 *is also significantly negative.*

The DIC of these two models are quite close (728.408 *and* 729.510)*, hence the DIC is not particularly in favor of either one.*

The following R code plot the two fitted regression lines and compare:

```r
#Question 2.
Q2data <- read.table("C:\\akira\\data\\copresence.txt", header = TRUE)

Xmean <- mean(Q2data$X)

beta01 <- -0.4045; beta11 <- -1.718 #logit link
beta02 <- -0.7148; beta12 <- -1.248 #log-log link
```

```r
library(gtools)#for logit and inv.logit
library(LaplacesDemon) #for cloglog and invcloglog

logit <- function(x){
  d <- inv.logit(beta01 + beta11*(x - Xmean))
}
loglog <- function(x){
  d <- invcloglog(beta02 + beta12*(x - Xmean))
}

p2 <- ggplot(data = data.frame(x = 0), mapping = aes(x = x)) + #create dummy dataset
  stat_function(fun=logit, geom="line", aes(colour="logit")) +
  stat_function(fun=loglog, geom="line", aes(colour="complementary log-log")) +
  labs(y = "P(co-presence)") +
  #theme_grey() +
  scale_x_continuous(limits = c(0.1, 5)) +
  #scale_y_continuous(limits = c(10.5, 13.5))
  scale_color_manual(name = "link functions",
                     values = c("blue", "red"), # Color specification
                     labels = c("logit", "complementary log-log"))
p2
```



As we can see that the two lines are pretty close to each other, which is consistent with our information from DIC that not any one of these two models are favored more than the other.