Statistical Computing HW5

Wei-Chen Chang r12227118

Due: 2024-05-22

Setup

The Model

$$E[Y_i] = \frac{1}{e_i} = \beta_0 + \frac{\beta_1}{(N_i + \alpha_1)} + \frac{\beta_2}{(P_i + \alpha_2)} + \frac{\beta_3}{(K_i + \alpha_3)}$$

The nutrient values of different levels in N, P, K are N=0, 100, 200, 400, P=0, 22, 44, 88, and K=0, 42, 84, 168. respectively. 168.

And the list of priors:

$$\begin{cases} Y_i \sim gamma(\nu, \nu e_i) \\ \nu \sim gamma(0.01, 0.01) \\ \alpha_1 \sim N(\mu = 40, \sigma^2 = 100) \\ \alpha_2 \sim N(\mu = 22, \sigma^2 = 100) \\ \alpha_3 \sim N(\mu = 32, \sigma^2 = 100) \\ \beta_0 \sim N(\mu = 0, \sigma^2 = 10000) \\ \beta_j \sim N(\mu = 0, \sigma^2 = 10000) I[\beta_j > 0],\ j = 1, 2, 3 \end{cases}$$

Specifying Model in JAGS

```
# Load Dataset
data(bermuda.grass, package = "LearnBayes")
b.grass.list <- c(as.list(bermuda.grass),
                  list("N" = nrow(bermuda.grass))
)
# JAGS model
modelstring ="
model{
  for(i in 1:N){
    y[i] ~ dgamma(nu, mu[i]); mu[i] <- nu*eta[i]
    eta[i] <- 1/yhat[i]
    yhat[i] <- beta0 + (beta1/(100*Nit[i]+alpha[1]))+(beta2/(22*Phos[i]+alpha[2]))+(beta3/(42*Pot[i]+al
  }
## priors ##
  nu~dgamma(0.01, 0.01)
  alpha[1]~dnorm(40, 1/100)
  alpha[2]~dnorm(22, 1/100)
  alpha[3]~dnorm(32, 1/100)
  beta0~dnorm(0, 1/10000)
  beta1~dnorm(0, 1/10000) T(0,) # truncated below 0
  beta2~dnorm(0, 1/10000) T(0,) # truncated below 0
  beta3~dnorm(0, 1/10000) T(0,) # truncated below 0

}
"
writeLines(modelstring, con="bermuda.bug")
```

Analysis

Here `R2jags` package is used to done the MCMC. The Posterior Mean, S.D., and other statistics can be seen in the output. One can see that every parameters have `Rhat` ($\hat{R}$, Gelman-Rubin Statistic) values lower than 1.01, the convergence of the MCMC seems to be OK.

```
library(coda)
library(rjags)
library(R2jags)
param.names <- c("nu", "alpha", paste0("beta",0:3))
# set initial values
bayes.mod.inits <- function(){
 list("nu" = rgamma(1, 0.1, 0.1),
      "alpha" = c(rnorm(1, 40, sd=5),
                  rnorm(1, 22, sd=5),
                  rnorm(1, 32, sd=5)),
      "beta0" = rnorm(1, sd=5), # such initial for beta0 seems to generate error
      "beta1" = abs(rnorm(1,0, sd=5)),
      "beta2" = abs(rnorm(1,0, sd=5)),
      "beta3" = abs(rnorm(1,0, sd=5))
      )
}
inits <- list(bayes.mod.inits(), bayes.mod.inits(), bayes.mod.inits())
set.seed(907)
ber.jags<- jags(data = b.grass.list,
     # inits = inits, # Note: errors happends a lot when using self-specified inits
     parameters.to.save = param.names,
     n.chains=3, n.iter=10000, n.burnin=1000,
     model.file = "bermuda.bug")
```

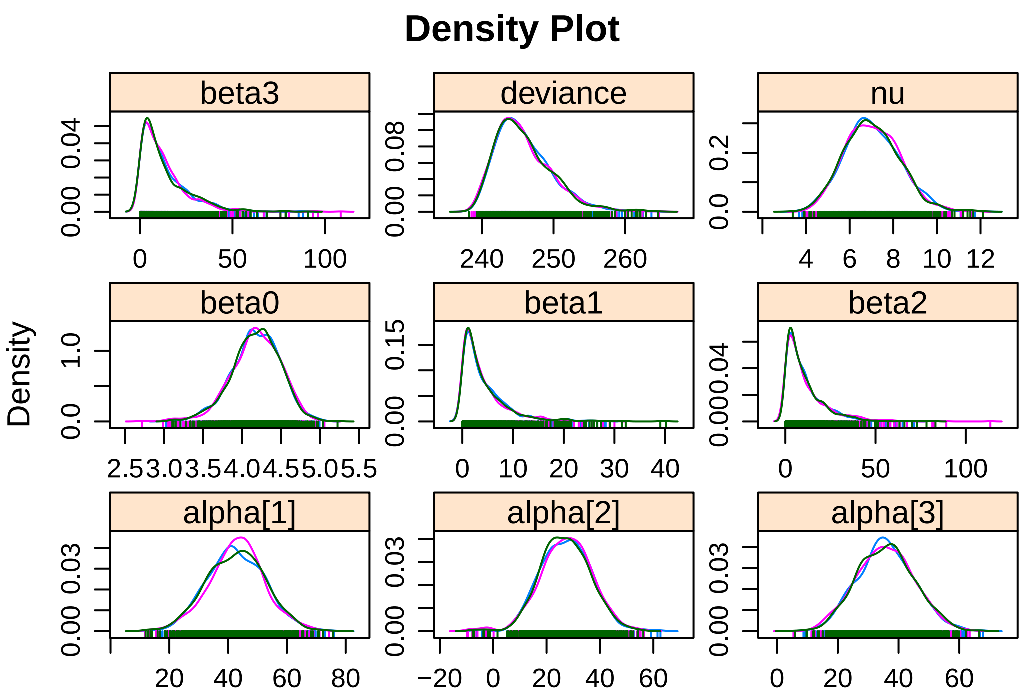
```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 64
##    Unobserved stochastic nodes: 8
##    Total graph size: 504
##
## Initializing model
```

```
ber.jags |> print()
```

```
## Inference for Bugs model at "bermuda.bug", fit using jags,
##  3 chains, each with 10000 iterations (first 1000 discarded), n.thin = 9
##  n.sims = 3000 iterations saved
##          mu.vect sd.vect    2.5%     25%     50%     75%   97.5%  Rhat n.eff
## alpha[1]  42.834   9.747  23.932  36.294  42.877  49.406  62.121 1.001  3000
## alpha[2]  27.238   9.532   9.546  20.795  27.290  33.634  45.306 1.001  3000
## alpha[3]  35.771   9.362  18.232  29.281  35.694  42.000  54.093 1.001  3000
## beta0      4.180   0.312   3.511   3.986   4.191   4.393   4.735 1.001  3000
## beta1      4.192   4.582   0.145   1.102   2.686   5.723  17.109 1.001  2200
## beta2     10.867  11.388   0.318   2.975   7.232  14.696  40.603 1.001  3000
## beta3     12.819  12.444   0.391   3.788   9.103  18.099  43.084 1.001  3000
## nu         7.039   1.260   4.718   6.151   6.974   7.862   9.648 1.001  3000
## deviance 245.880   3.915 240.128 243.033 245.170 248.163 255.161 1.001  3000
##
## For each parameter, n.eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor (at convergence, Rhat=1).
##
## DIC info (using the rule, pD = var(deviance)/2)
## pD = 7.7 and DIC = 253.5
## DIC is an estimate of expected predictive error (lower deviance is better).
```

For visulaizing purpose, the posterior density plot were shown below.

```
library(lattice)
ber.mcmc <- as.mcmc(ber.jags)
densityplot(ber.mcmc, layout=c(3,3), aspect="fill",
       main = "Density Plot")
```



One can see the density plot of 3 chains seems aligned.

Model Diagnosis

Convergence

For diagnosis of the Markov Chains, here Geweke statistic were computed and convergence plot were drawn to check for convergence.

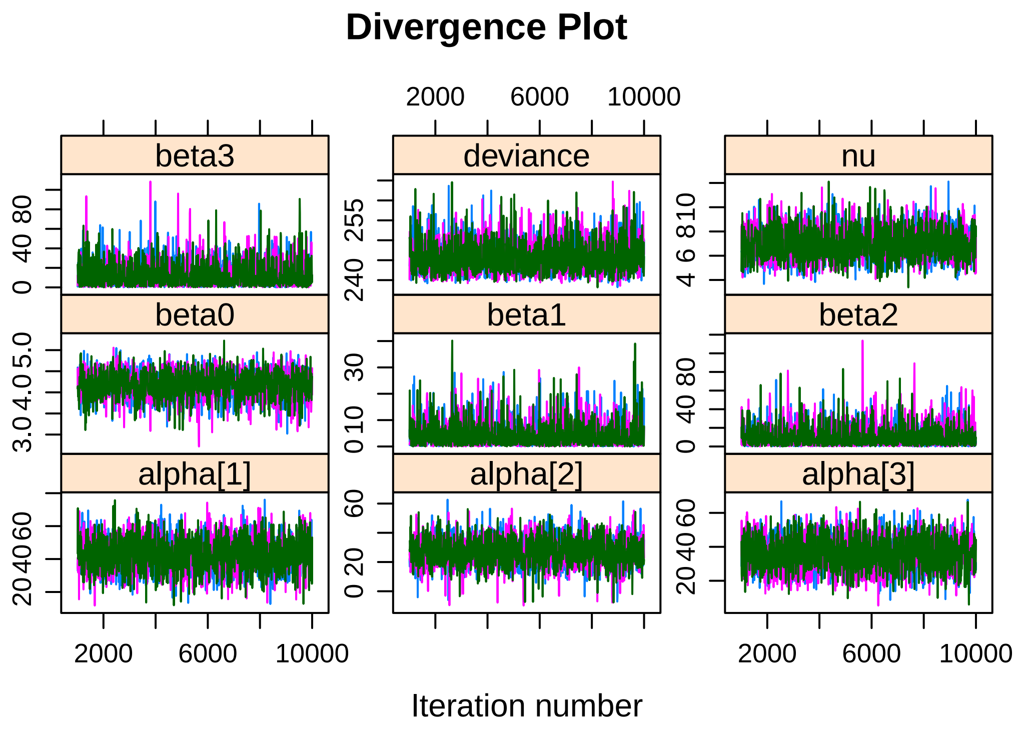
```
geweke.diag(ber.mcmc)
```

```
## [[1]]
##
## Fraction in 1st window = 0.1
## Fraction in 2nd window = 0.5
##
## alpha[1] alpha[2] alpha[3]    beta0    beta1    beta2    beta3 deviance
##  1.47266 -0.70218  0.48073  1.76330  0.53679 -1.20705  0.13131  0.05902
##       nu
## -0.79822
##
##
## [[2]]
##
## Fraction in 1st window = 0.1
## Fraction in 2nd window = 0.5
##
```

```
## alpha[1] alpha[2] alpha[3]    beta0    beta1    beta2    beta3 deviance
##  -0.5824  -1.2659   0.4523   2.0434   0.2369  -3.0988  -0.2582  -1.5154
##       nu
##  -0.9332
##
##
## [[3]]
##
## Fraction in 1st window = 0.1
## Fraction in 2nd window = 0.5
##
## alpha[1] alpha[2] alpha[3]    beta0    beta1    beta2    beta3 deviance
##  -0.1551   1.3001   1.9096  -2.1476   1.1436   1.9011   1.1835   1.8762
##       nu
##  -1.6897
```

Geweke statistic larger than 2.5 or smaller than -2.5 is a warning sign for nonstationarity samples. One can see only `beta2` from 2nd chain suffers such.

```
xyplot(ber.mcmc, layout=c(3,3), aspect="fill",
       main = "Divergence Plot")
```

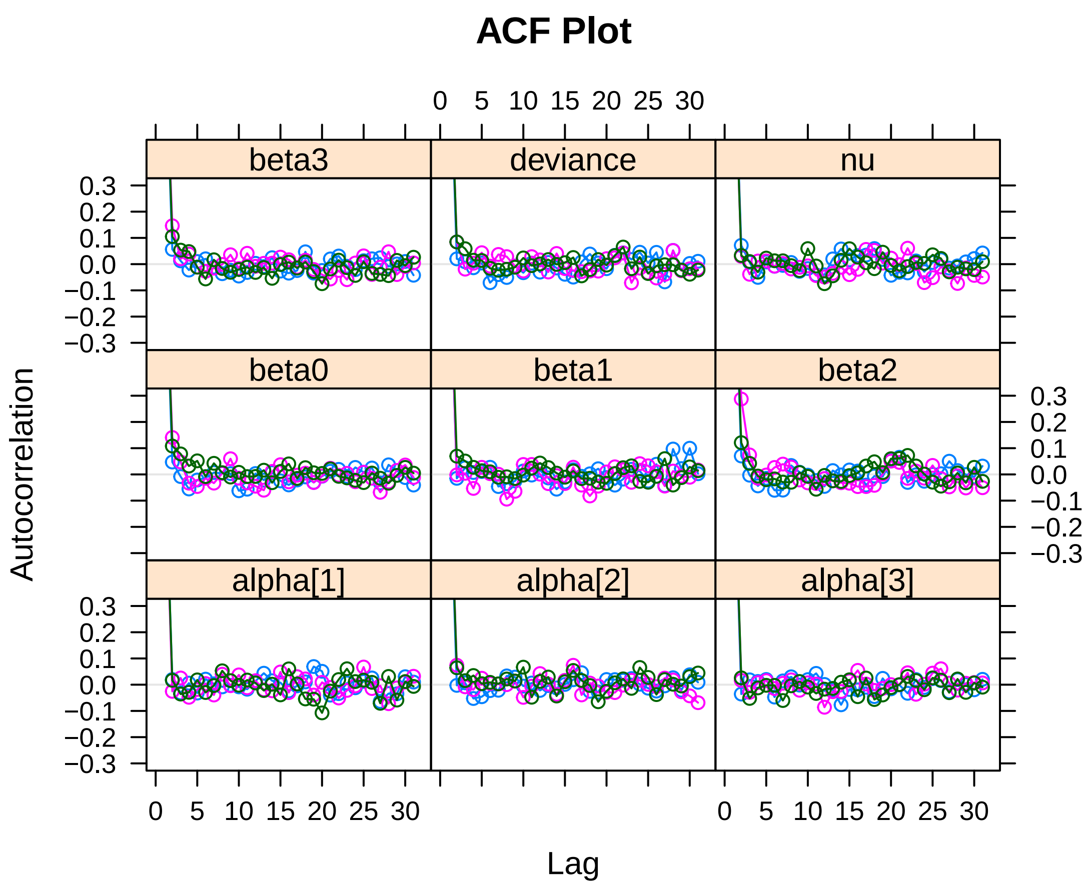


The Divergence plot also indicate that the chains were OK for convergence.

ACF

And Autocorrelation plot were plotted below. One can see the autocorrelation after 5 lags are mostly between -.1 to .1. It support the samples were uncorrelated.

```
acfplot(ber.mcmc, main = "ACF Plot", layout=c(3,3), aspect="fill")
```


ACF Plot
beta3
deviance
nu
beta0
beta1
beta2
alpha[1]
alpha[2]
alpha[3]
Autocorrelation
Lag