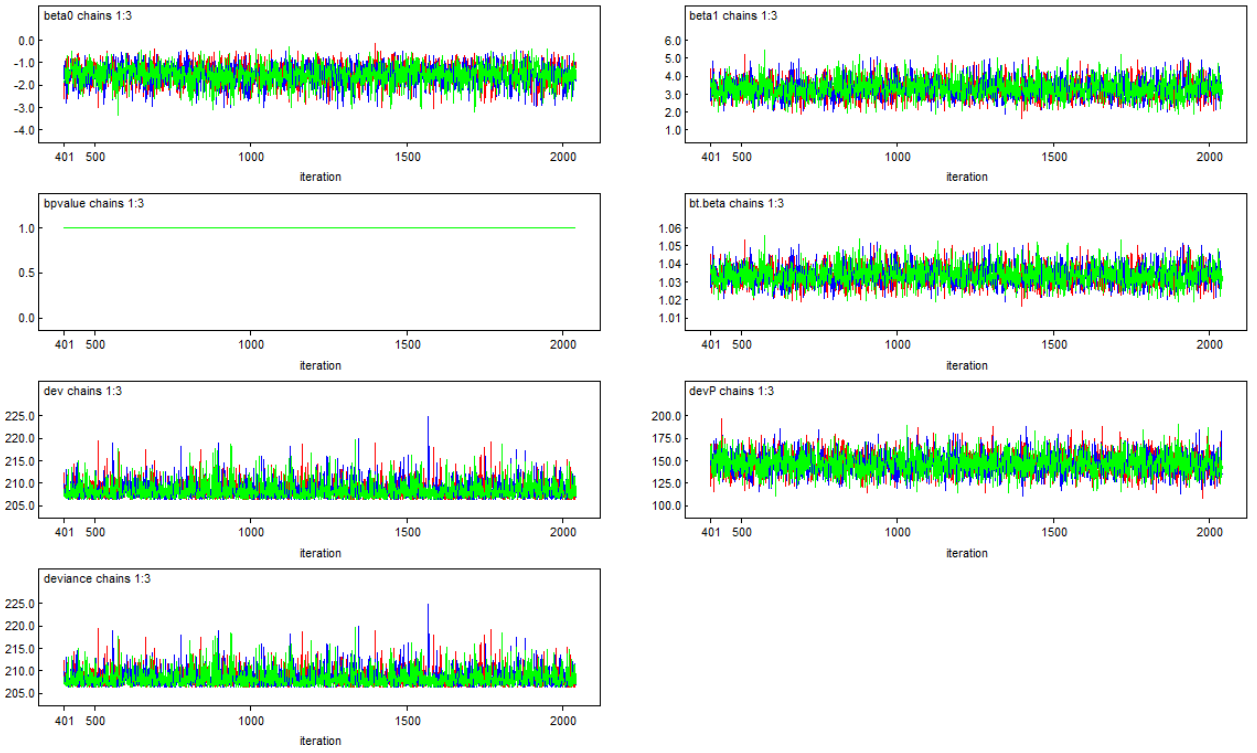
Christian Hower
Stat 410
1A)

```
sink("hw5.txt")
cat("
    model {

    # Priors
    beta0 ~ dnorm(0, .00001)
    beta1 ~ dnorm(0, .00001)

    # Likelihood    for (i in 1:n){
    counts[i] ~ dpois(mu[i])
    log(mu[i]) <- beta0 +  beta1*PctCover[i]
    }
    # Deviance Observed
    for (i in 1:n){
    LL[i] <- -1*mu[i] + counts[i]*log(mu[i]) -
    logfact(counts[i])
    }
    # Deviance ideal
    for (i in 1:n){
    CountPred[i] ~ dpois(mu[i])
    LLP[i] <- -1*mu[i] + CountPred[i]*log(mu[i]) -
    logfact(CountPred[i])
    }
    # Monitoring
    dev <- -2*sum(LL[])
    devP <- -2*sum(LLP[])
    test <- step(dev-devP)
    bpvalue <- mean(test)
    #derived parameter
    bt.beta <- exp(.01*beta1)
    }
    ",fill = TRUE)
sink()

# Bundle data
win.data <- list(n = as.numeric(length(sal$Count)),
                 counts = as.numeric(sal$Count),
                 PctCover = as.numeric(sal$PctCover))

# Initial values
inits <- function() list(beta0 = runif(1, -2, 2), beta1 = runif(1, -
2, 2))
# Parameters monitored
```

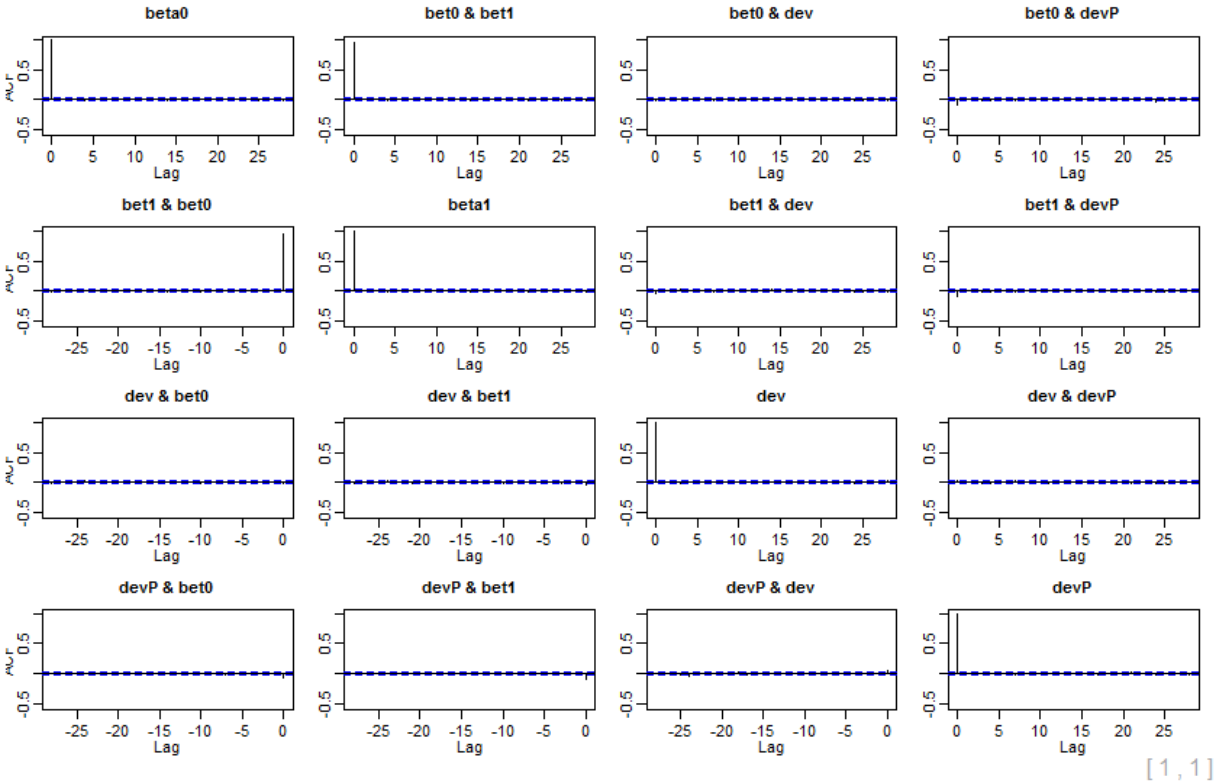
```
params <- c("beta0", "beta1", "dev", "devP", "bpvalue", "bt.beta")

# MCMC settings
ni <- 102000
nt <- 50
nb <- 20000
nc <- 3

# Call WinBUGS from R
out <- bugs(win.data, inits, params, "hw5.txt", n.chains = nc,
            n.thin = nt, n.iter = ni, n.burnin = nb, debug = TRUE,
            bugs.directory = bugs.dir, working.directory = getwd())
```

1B)

```
# Autocorrelation, deviance plot, summary
acf(out$sims.matrix)
print(out , dig = 3)
dev.obs <- out$sims.matrix[,3]
dev.fit <- out$sims.matrix[,4]
plot(dev.fit ~ dev.obs , ylim=c(100, 240))
abline(a = 0 , b = 1 , col="red" , lwd = 3 )
```

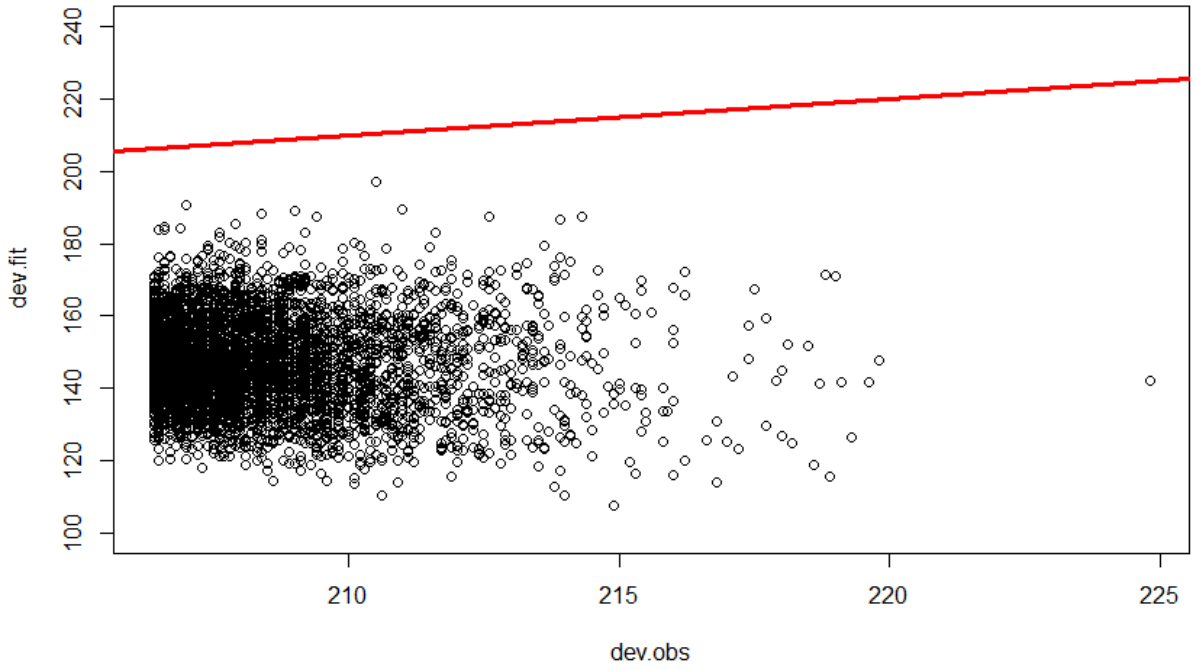


MCMC Chains have converged converged.



[1 , 1]

Beta1 and Beta0 do not appear auto correlated.



The deviance plot shows a problem. The data appear overdispersed. Observed deviance is much larger than deviance expected in an ideal scenario.

```
> print(out , dig = 3)
Inference for Bugs model at "hw5.txt", fit using WinBUGS,
 3 chains, each with 102000 iterations (first 20000 discarded), n.thin = 50
 n.sims = 4920 iterations saved
            mean     sd    2.5%     25%     50%     75%   97.5%  Rhat n.eff
beta0     -1.550  0.463  -2.541  -1.848  -1.525  -1.229  -0.705 1.002  1600
beta1      3.312  0.543   2.317   2.935   3.285   3.666   4.447 1.002  1800
dev      208.384  1.965 206.400 207.000 207.800 209.200 213.502 1.001  4900
devP     146.638 12.032 124.395 138.200 146.200 154.500 170.902 1.001  3600
bpvalue    1.000  0.000   1.000   1.000   1.000   1.000   1.000 1.000     1
bt.beta    1.034  0.006   1.023   1.030   1.033   1.037   1.045 1.002  1600
deviance 208.384  1.965 206.400 207.000 207.800 209.200 213.502 1.001  4900

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 = Dbar-Dhat)
pD = 2.0 and DIC = 210.4
DIC is an estimate of expected predictive error (lower deviance is better).
```

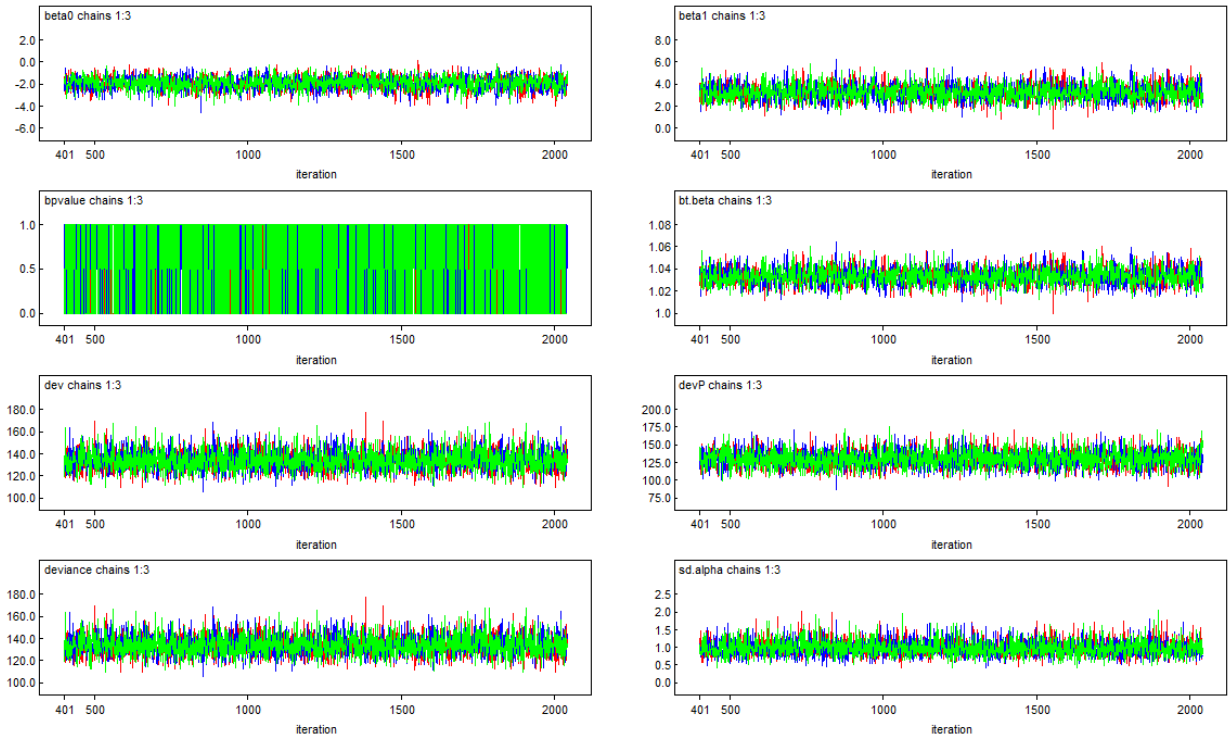
Rhat is good (< 1.1) but bpvalue of 1 indicates that the model is a poor fit.

1C)

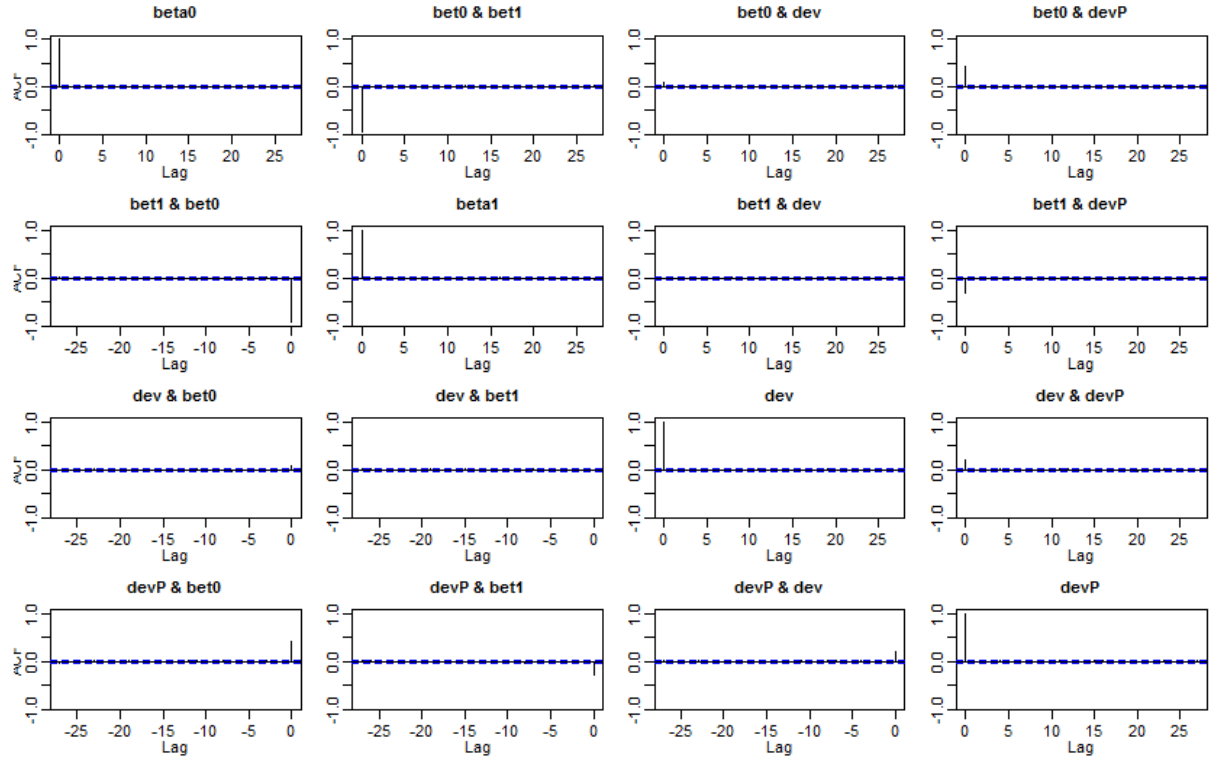
```
sink("hw5.2.txt")

cat("

    model {

    # Priors

    beta0 ~ dnorm(0, .00001)

    beta1 ~ dnorm(0, .00001)

    sd.alpha ~ dunif(0, 10)

    site.prec <- 1/(sd.alpha*sd.alpha)

    # Likelihood

    for (i in 1:n){

    counts[i] ~ dpois(mu[i])

    log(mu[i]) <- beta0 +  beta1*PctCover[i] + alpha[S[i]]

    }

    for (i in 1:n){

    alpha[i] ~ dnorm(0 , site.prec)

    }

    # Deviance for dataset

    for (i in 1:n){

    LL[i] <- -1*mu[i] + counts[i]*log(mu[i]) -
```

```
    logfact(counts[i])
    }
    # Deviance for ideal datasets
    for (i in 1:n){
    CountPred[i] ~ dpois(mu[i])
    LLP[i] <- -1*mu[i] + CountPred[i]*log(mu[i]) -
    logfact(CountPred[i])
    }
    # Objects for Bayesian P value
    dev <- -2*sum(LL[])
    devP <- -2*sum(LLP[])
    test <- step(dev-devP)
    bpvalue <- mean(test)
    #derived parameter
    bt.beta <- exp(.01*beta1)
    }
    ",fill = TRUE)
sink(
# Bundle data
win.data <- list(n = as.numeric(length(sal$Count)),
                 counts = as.numeric(sal$Count),
                 PctCover = as.numeric(sal$PctCover),
                 S = as.numeric(sal$Site))
# Initial values
inits <- function() list(beta0 = runif(1, -2, 2), beta1 = runif(1, -2,
2), alpha = rnorm(length(sal$Site), 0 , 2))
# Parameters monitored
params <- c("beta0", "beta1", "dev", "devP", "bpvalue",
            "bt.beta" , "sd.alpha")


# MCMC settings
```

```
ni <- 102000
nt <- 50
nb <- 20000
nc <- 3
# Call WinBUGS from R
out2 <- bugs(win.data, inits, params, "hw5.2.txt", n.chains = nc,
             n.thin = nt, n.iter = ni, n.burnin = nb, debug = TRUE,
             bugs.directory = bugs.dir, working.directory = getwd())
```

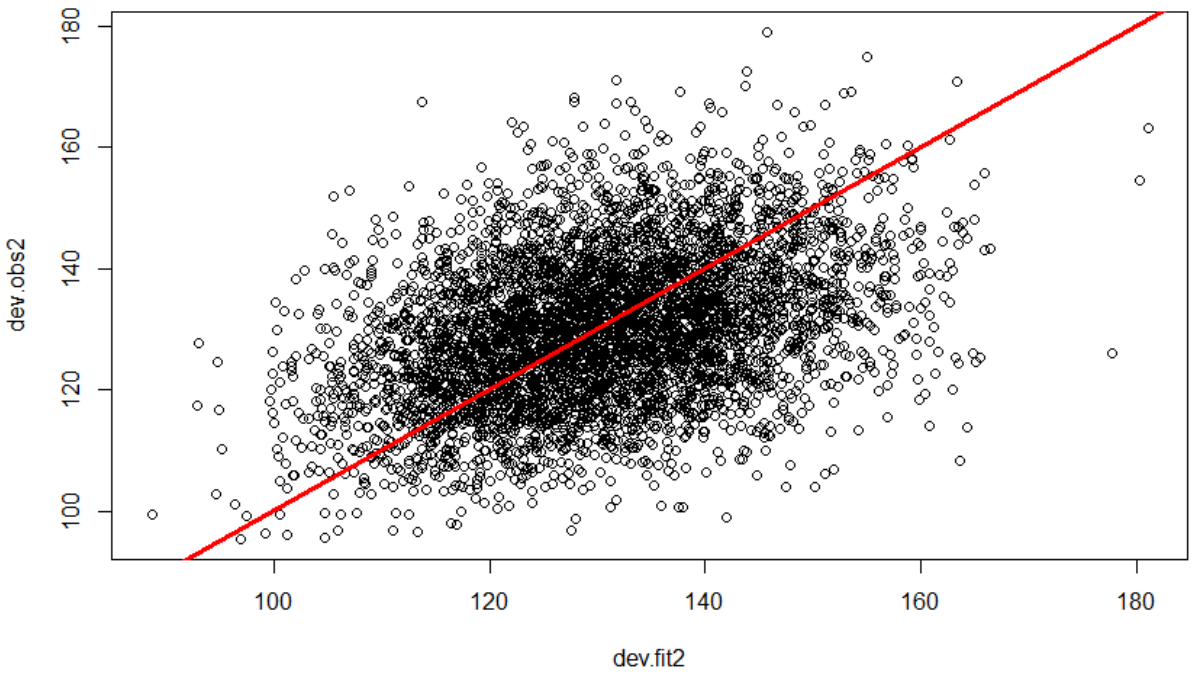


Chains look like they have converged.



There does not appear to be any autocorrelation issues. Values are within tolerance bars.

1D)



Deviance plot looks excellent.

```
> print(out2 , dig = 3)
Inference for Bugs model at "hw5.2.txt", fit using WinBUGS,
 3 chains, each with 102000 iterations (first 20000 discarded), n.thin = 50
 n.sims = 4920 iterations saved
            mean     sd    2.5%     25%     50%     75%   97.5%  Rhat n.eff
beta0     -1.912  0.594  -3.142  -2.295  -1.884  -1.506  -0.837 1.001  4900
beta1      3.267  0.737   1.892   2.760   3.232   3.750   4.734 1.001  4900
dev      133.649  9.211 117.500 127.100 132.800 139.600 153.702 1.002  1300
devP     129.932 12.109 107.600 121.500 129.400 137.900 155.305 1.001  4000
bpvalue    0.603  0.489   0.000   0.000   1.000   1.000   1.000 1.001  4900
bt.beta    1.033  0.008   1.019   1.028   1.033   1.038   1.048 1.001  4900
sd.alpha   0.976  0.217   0.616   0.824   0.950   1.101   1.472 1.001  4900
deviance 133.649  9.211 117.500 127.100 132.800 139.600 153.702 1.002  1300

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 = Dbar-Dhat)
pD = 24.4 and DIC = 158.1
DIC is an estimate of expected predictive error (lower deviance is better).
```

Bayesian P value of .603 indicates very good fit. Rhat values look good and indicate convergence.

1E) Salamander counts increase with canopy cover. For every percent increase in canopy conver there is 1.9 – 4.8 percent mean increase in salamander counts according to the 95% credible interval for these data.

2A)

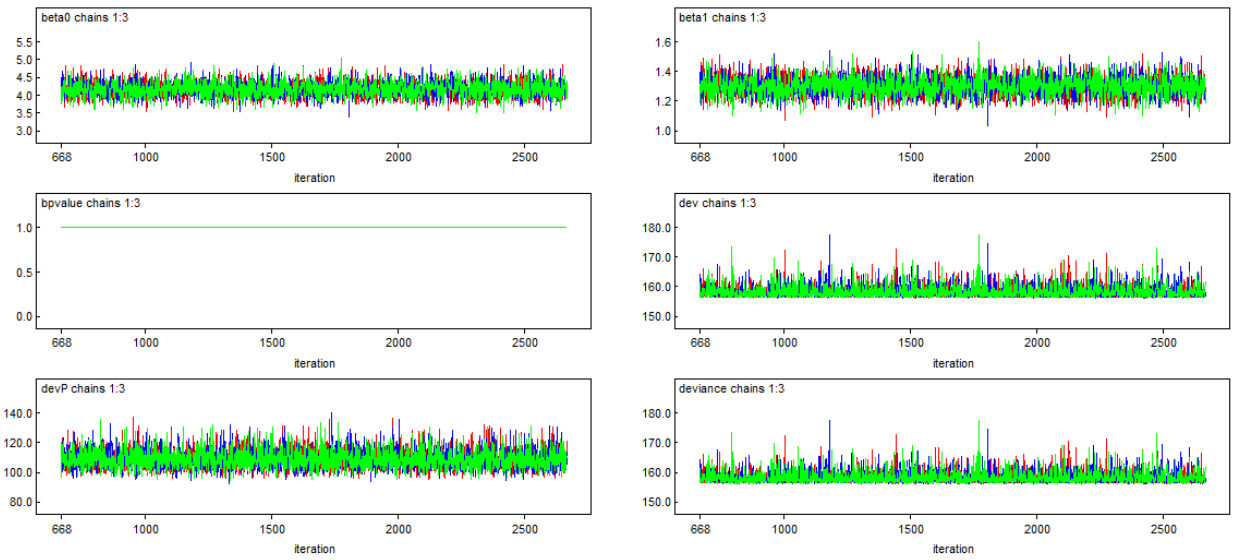
```
alf$N <- alf$Tumors + alf$Notumor

sink("hw5.3.txt")

cat("

    Model{

    #priors

    beta0~dnorm(0,0.00001)

    beta1~dnorm(0,0.0001)

    #likelihood

    for(i in 1:n){

    Tumors[i]~dbin(p[i],N[i])

    logit(p[i])<-beta0+beta1*lDose[i]

    }

    # Deviance Observed

    for(i in 1:n){

    LL[i]<- logfact(N[i]) - logfact(Tumors[i]) -
```

```
    logfact(N[i] - Tumors[i]) +
    Tumors[i]*log(p[i]) +
    (N[i] - Tumors[i])*log(1-p[i])
    }
    # Deviance Fit
    for(i in 1:n){
    TumorsPred[i]~dbin(p[i],N[i])
    LLP[i]<- logfact(N[i]) - logfact(TumorsPred[i]) -
    logfact(N[i] - TumorsPred[i]) +
    TumorsPred[i]*log(p[i]) +
    (N[i] - TumorsPred[i])*log(1-p[i])
    }
    dev<- -2*sum(LL[])
    devP<- -2*sum(LLP[])
    test<- step(dev - devP)
    bpvalue<-mean(test)
    }
    ", fill=TRUE)
sink()



# Bundle data
win.data <- list(Tumors = as.numeric(alf$Tumors),
                 n = as.numeric(length(alf$Tumors)),
                 N = as.numeric(alf$N),
                 lDose = as.numeric(alf$lDose))
# Initial values
inits <- function() list(beta0 = runif(1, -2, 2), beta1 = runif(1, -2,
2))
# Parameters monitored
params <- c("beta0", "beta1","dev","devP", "bpvalue")
```

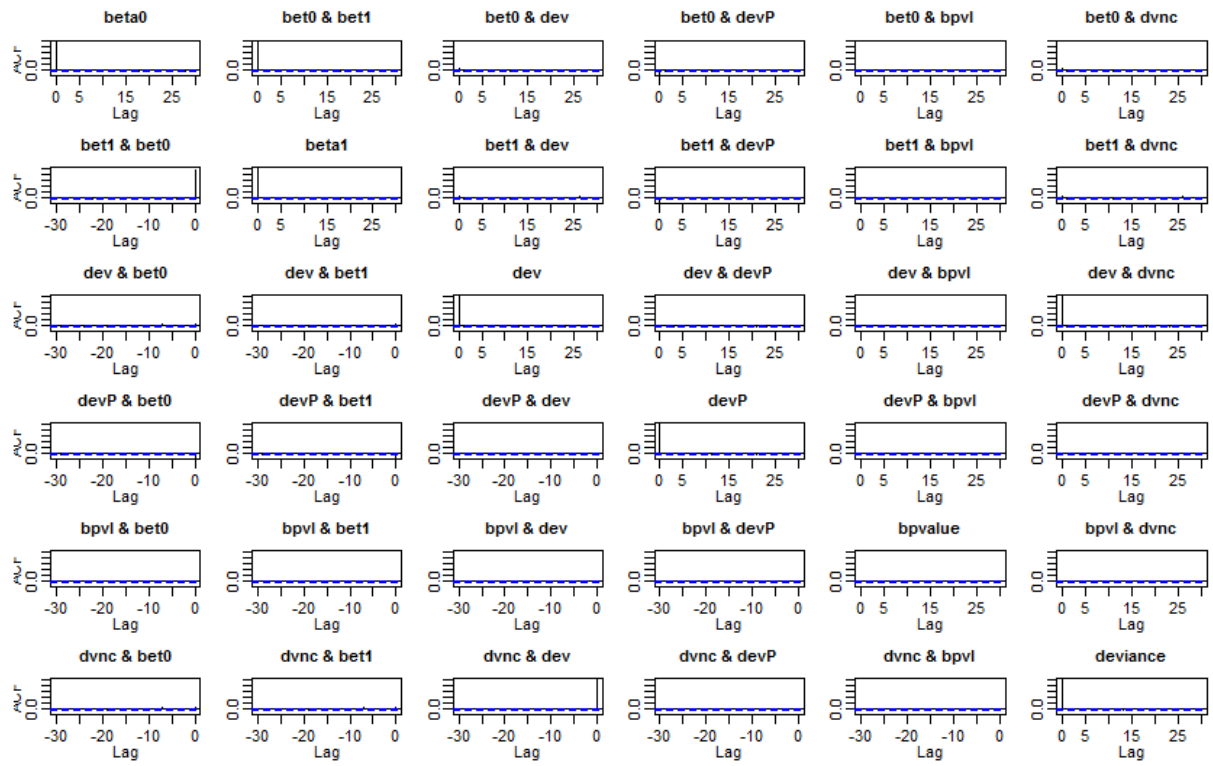
```
# MCMC settings
ni <- 80000
nt <- 30
nb <- 20000
nc <- 3
# Call WinBUGS
out3 <- bugs(win.data, inits, params, "hw5.3.txt", n.chains = nc,
             n.thin = nt, n.iter = ni, n.burnin = nb, debug = TRUE,
             bugs.directory = bugs.dir, working.directory = getwd())
```

2B)

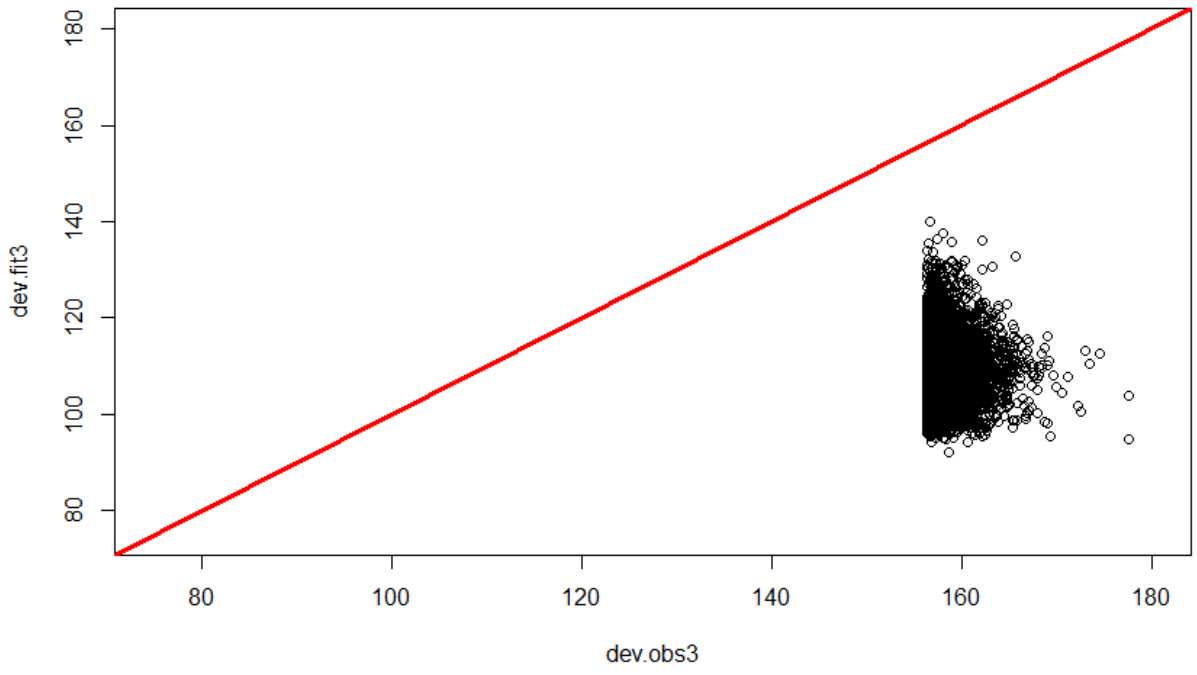
```
# ACF, Deviance Plot, Summary
acf(out3$sims.matrix)
dev.obs3 <- out3$sims.matrix[,3]
dev.fit3 <- out3$sims.matrix[,4]
plot(dev.fit3 ~ dev.obs3 , xlim = c(75,180) , ylim = c(75,180))
abline(a = 0 , b = 1 , col="red" , lwd = 3 )
print(out3 , dig = 3)
```



MCMC Chains look good but Bayesian p value looks problematic.



Autocorrelation appears to be fine. Within threshold bars.



Deviance plot shows overdispersion. Much more deviance observed than ideal.

```
> print(out3 , dig = 3)
Inference for Bugs model at "hw5.3.txt", fit using WinBUGS,
 3 chains, each with 80000 iterations (first 20000 discarded), n.thin = 30
 n.sims = 6000 iterations saved
            mean    sd    2.5%     25%     50%     75%   97.5%  Rhat n.eff
beta0      4.169 0.210   3.776   4.024   4.164   4.306   4.588 1.002  2500
beta1      1.300 0.065   1.177   1.256   1.298   1.342   1.429 1.002  2500
dev      158.328 2.092 156.300 156.900 157.700 159.100 164.000 1.001  6000
devP     108.941 6.454  98.260 104.300 108.300 112.700 123.400 1.001  6000
bpvalue    1.000 0.000   1.000   1.000   1.000   1.000   1.000 1.000     1
deviance 158.328 2.092 156.300 156.900 157.700 159.100 164.000 1.001  6000

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 = Dbar-Dhat)
pD = 2.0 and DIC = 160.4
DIC is an estimate of expected predictive error (lower deviance is better).
```

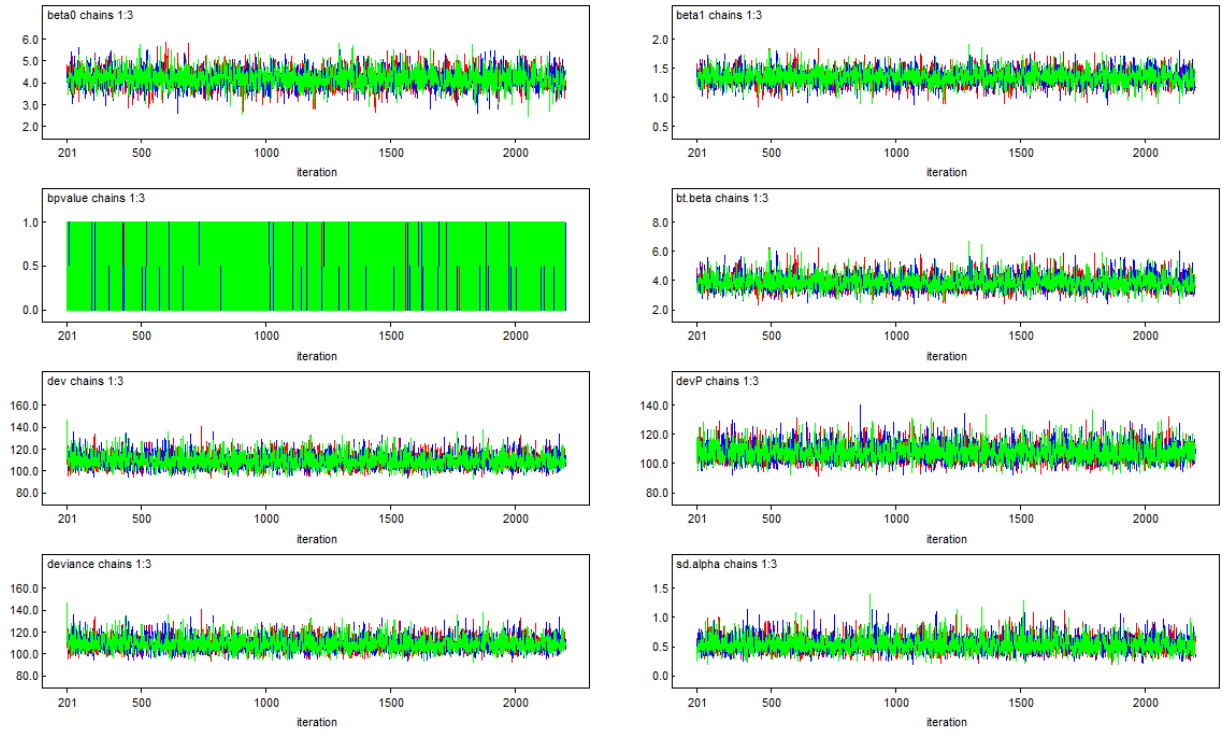
Rhat values look good. Bayesian P value of 1 consistent with overdispersion observed in the plot.

2C)

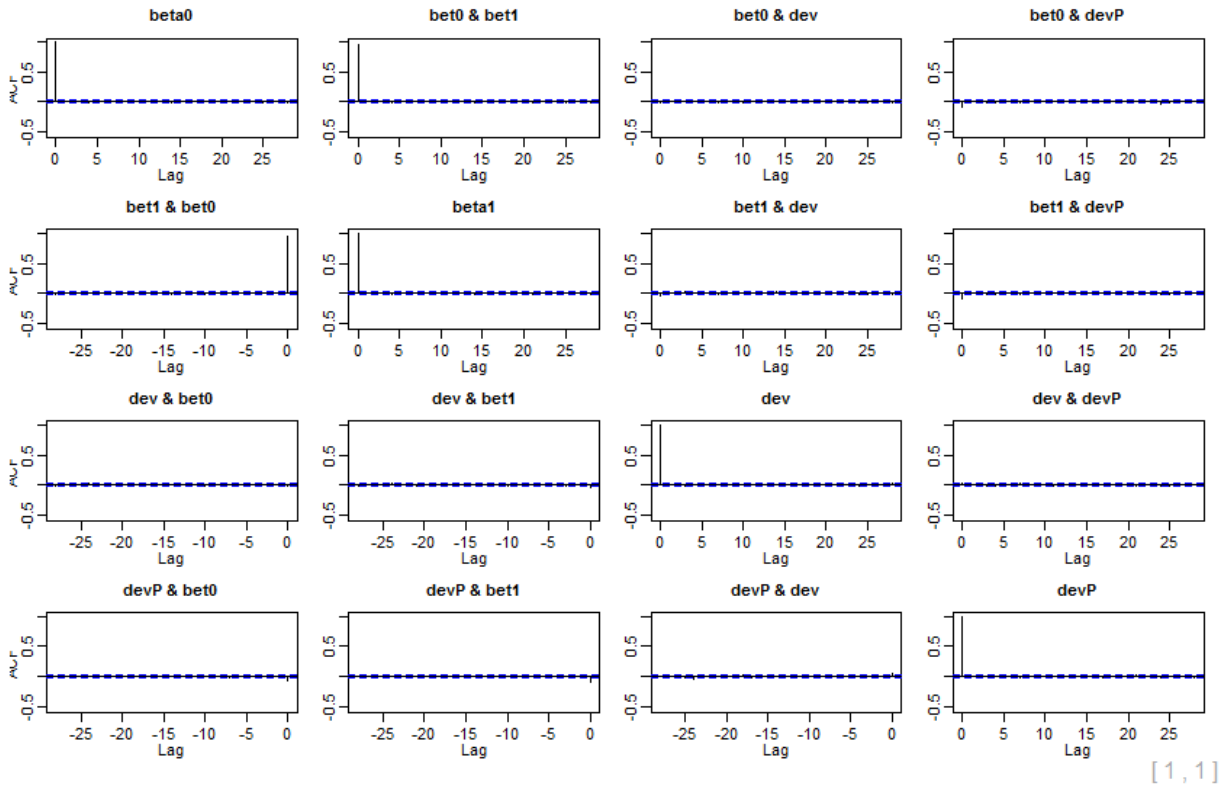
```
sink("hw5.4.txt")

cat("

    Model{

    #priors

    beta0~dnorm(0,0.00001)

    beta1~dnorm(0,0.0001)

    sd.alpha ~ dunif(0 , 10)

    alpha.prec <- 1/(sd.alpha*sd.alpha)


    for (i in 1:n){

    alpha[i] ~ dnorm(0 , alpha.prec)

    }

    #likelihood

    for(i in 1:n){

    Tumors[i]~dbin(p[i],N[i])

    logit(p[i])<-beta0+beta1*lDose[i]+ alpha[T[i]]

    }

    # Observed Deviance
```

```
    for(i in 1:n){
    LL[i]<- logfact(N[i]) - logfact(Tumors[i]) -
    logfact(N[i] - Tumors[i]) +
    Tumors[i]*log(p[i]) +
    (N[i] - Tumors[i])*log(1-p[i])
    }
    # Ideal Deviance
    for(i in 1:n){
    TumorsPred[i]~dbin(p[i],N[i])
    LLP[i]<- logfact(N[i]) - logfact(TumorsPred[i]) -
    logfact(N[i] - TumorsPred[i]) +
    TumorsPred[i]*log(p[i]) +
    (N[i] - TumorsPred[i])*log(1-p[i])
    }
    dev<- -2*sum(LL[])
    devP<- -2*sum(LLP[])
    test<- step(dev - devP)
    bpvalue<-mean(test)
    bt.beta <- exp(beta1)
    }
    ", fill=TRUE)
sink()
# Bundle data
win.data <- list(Tumors = as.numeric(alf$Tumors),
                 n = as.numeric(length(alf$Tumors)),
                 N = as.numeric(alf$N),
                 lDose = as.numeric(alf$lDose),
                 T = as.numeric(alf$Tank))
```

```
# Initial values

inits <- function() list(beta0 = runif(1, -2, 2), beta1 = runif(1, -2,
2), alpha = rnorm(length(alf$Tank), 0 , 2) )

# Parameters monitored

params <- c("beta0", "beta1","dev","devP", "bpvalue",
"sd.alpha","bt.beta")

# MCMC settings

ni <- 220000

nt <- 100

nb <- 20000

nc <- 3

# Call WinBUGS

out4 <- bugs(win.data, inits, params, "hw5.4.txt", n.chains = nc,

             n.thin = nt, n.iter = ni, n.burnin = nb, debug = TRUE,

             bugs.directory = bugs.dir, working.directory = getwd())
```



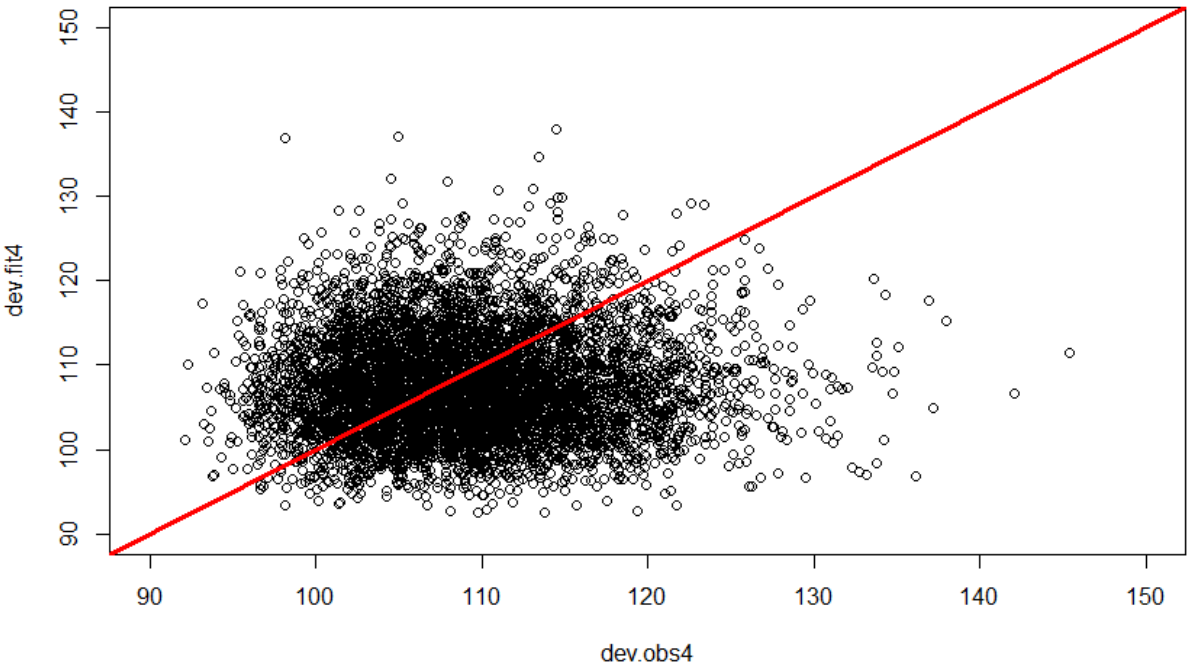
MCMC Chains look good.



[1, 1]

There does not appear to be any autocorrelation.

2D)



The plot looks almost ideal. The plot indicates model fits the data.

```
> print(out4 , dig = 3)
Inference for Bugs model at "hw5.4.txt", fit using WinBUGS,
 3 chains, each with 220000 iterations (first 20000 discarded), n.thin = 100
 n.sims = 6000 iterations saved
            mean    sd   2.5%     25%     50%     75%   97.5%  Rhat n.eff
beta0      4.199 0.421  3.396   3.917   4.198   4.462   5.061 1.001  6000
beta1      1.345 0.134  1.088   1.256   1.343   1.430   1.619 1.001  6000
dev      109.201 6.777 97.910 104.400 108.500 113.400 124.302 1.001  6000
devP     107.647 6.281 97.359 103.100 107.000 111.400 121.800 1.001  6000
bpvalue    0.565 0.496  0.000   0.000   1.000   1.000   1.000 1.002  1400
sd.alpha   0.532 0.135  0.315   0.438   0.517   0.608   0.849 1.001  6000
bt.beta    3.872 0.526  2.968   3.510   3.830   4.181   5.050 1.001  6000
deviance 109.201 6.777 97.910 104.400 108.500 113.400 124.302 1.001  6000

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 = Dbar-Dhat)
pD = 16.3 and DIC = 125.5
DIC is an estimate of expected predictive error (lower deviance is better).
```

Rhat values look good so the MCMC chains have converged on a distribution. Bpvalue of .565 is very close to ideal (.50).

2E) Increases in alfatoxicol dose are associated with more tumors. For every logdose increase in Alfatoxicol the odds of finding a tumor is 3 - 5 times higher according to the 95% credible interval for these data.