Applied Bayesian Analysis Assignment 2

Faizan Khalid Mohsin

February 23, 2019

Contents

1 Question 1 **2**

1.1 Question 1 a: 2

1.2 Question 1 b: 2

1.3 Question 1 c: 4

1.4 Question 1 d: 6

2 Question 2 **7**

2.1 Question 2 a 7

2.2 Question 2 b. 13

2.3 Question 2 c. 16

3 Question 3 **16**

3.1 Question 3 a 16

1 Question 1

http://rmarkdown.rstudio.com.

1.1 Question 1 a:

Done in the attached paper by hand.

1.2 Question 1 b:

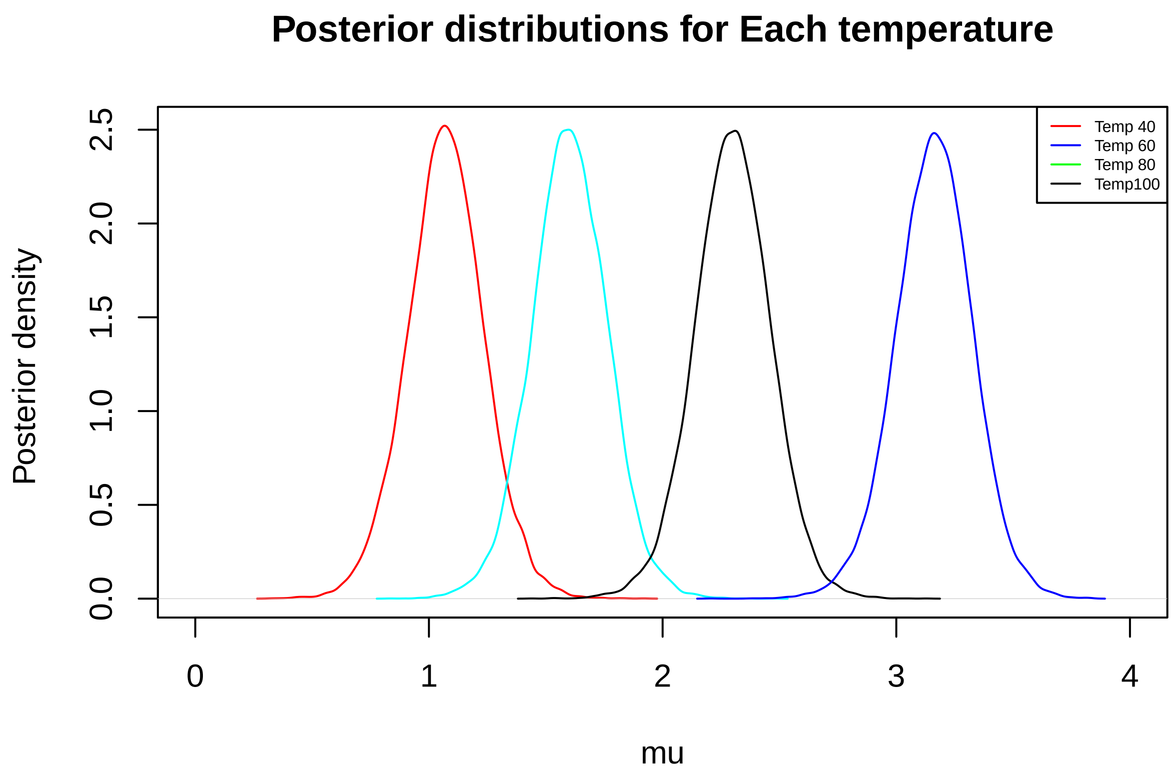
```
# Data

set.seed(25)
nbig<-20000

data = matrix( c( 1.13, 1.75, 2.30, 3.18,
                  1.20, 1.45, 2.15, 3.10,
                  1.00, 1.55 ,2.25, 3.28,
                  0.91, 1.64, 2.40, 3.35,
                  1.05, 1.60, 2.49, 3.12), nrow = 5, byrow = T)

data = as.data.frame(data)
colnames(data) = c("temp40", "temp60", "temp80", "temp100")
attach(data)

Y1<- mean(temp40)
Y2<- mean(temp60)
Y3<- mean(temp80)
Y4<- mean(temp100)

#MCMC Algorithm

mu00<-0
tau00<-1
alp<-1
beta<-1
alp0<-1
beta0<-1
mu1<-rep(0,nbig)
mu2<-rep(0,nbig)
mu3<-rep(0,nbig)
mu4<-rep(0,nbig)
tau<-rep(0,nbig)
mu0post1<-rep(0,nbig)
tau0post1<-rep(0,nbig)
mu0post2<-rep(0,nbig)
tau0post2<-rep(0,nbig)
mu0post3<-rep(0,nbig)
tau0post3<-rep(0,nbig)
mu0post4<-rep(0,nbig)
tau0post4<-rep(0,nbig)
```

```r
mu0<-rep(0,nbig)
tau0<-rep(0,nbig)
mupost00<-rep(0,nbig)
taupost00<-rep(0,nbig)
betapost<-rep(0,nbig)
betapost0<-rep(0,nbig)
mu0[1]<-1
mu1[1]<-0
mu2[1]<-0
mu3[1]<-0
mu4[1]<-0
tau[1]<-1
tau0[1]<-1

for(i in 2:nbig) {
  mu0post1[i]<-(tau0[i-1]*mu0[i-1]+5*tau[i-1]*Y1)/(tau0[i-1]+5*tau[i-1])
  tau0post1[i]<-tau0[i-1]+tau[i-1]*5
  mu1[i]<-rnorm(1,mu0post1[i],sd=1/sqrt(tau0post1[i]))
  mu0post2[i]<-(tau0[i-1]*mu0[i-1]+5*tau[i-1]*Y2)/(tau0[i-1]+5*tau[i-1])
  tau0post2[i]<-tau0[i-1]+tau[i-1]*5
  mu2[i]<-rnorm(1,mu0post2[i],sd=1/sqrt(tau0post2[i]))
  mu0post3[i]<-(tau0[i-1]*mu0[i-1]+5*tau[i-1]*Y3)/(tau0[i-1]+5*tau[i-1])
  tau0post3[i]<-tau0[i-1]+tau[i-1]*5
  mu3[i]<-rnorm(1,mu0post3[i],sd=1/sqrt(tau0post3[i]))
  mu0post4[i]<-(tau0[i-1]*mu0[i-1]+5*tau[i-1]*Y4)/(tau0[i-1]+5*tau[i-1])
  tau0post4[i]<-tau0[i-1]+tau[i-1]*5
  mu4[i]<-rnorm(1,mu0post4[i],sd=1/sqrt(tau0post4[i]))
  mupost00[i]<-(4*tau0[i-1]*((mu1[i-1]+mu2[i-1]+mu3[i-1]+mu4[i-1])/4)+tau00*mu00)/(4*tau0[i-1]+tau00)
  taupost00[i]<-4*tau0[i-1]+ tau00
  mu0[i]<-rnorm(1,mupost00[i],1/sqrt(taupost00[i]))
  alppost0<-alp0+(4/2)
  betapost0[i]<-beta0+(1/2)*(sum((mu1[i]-mu0[i])^2)+sum((mu2[i]-mu0[i])^2)+
                             sum((mu3[i]-mu0[i])^2)+sum((mu4[i]-mu0[i])^2))
  tau0[i]<-rgamma(1,alppost0,betapost0[i])
  alppost<-alp+(5+5+5+5)/2
  betapost[i]<- beta+(0.5*(sum((temp40-mu1[i])^2)+sum((temp60-mu2[i])^2)+
                           sum((temp80-mu3[i])^2)+sum((temp100-mu4[i])^2)))
  tau[i]<-rgamma(1,alppost,betapost[i])
}

cbind(mu0,mu1,mu2,mu3,mu4,tau,tau0)[1:10,]
```

```
##               mu0       mu1      mu2      mu3      mu4      tau      tau0
##  [1,]  1.00000000 0.0000000 0.000000 0.000000 0.000000 1.000000 1.0000000
##  [2,] -0.67087848 0.9618526 1.073106 1.627497 2.969598 3.567329 0.1406228
##  [3,] -0.21803713 0.7974946 1.723503 2.438722 3.146576 7.909098 0.1382672
##  [4,] -0.08856949 1.2008003 1.477891 2.461938 3.439468 7.995753 0.2759665
##  [5,]  1.74787069 1.0746105 1.959628 2.051584 3.009483 4.157769 1.2478098
##  [6,]  2.19847692 0.8599432 1.453477 2.384100 3.335221 4.848577 0.5409701
##  [7,]  0.49204091 1.0647587 1.817526 2.134459 3.103938 7.224207 0.5538967
##  [8,]  0.98798373 1.0414805 1.628266 1.903217 3.223159 7.829289 0.6663553
##  [9,]  0.99478290 0.8430916 1.528940 2.620348 3.005581 7.833863 0.2846369
## [10,]  1.67957313 0.9282185 1.471686 1.941989 3.366871 7.140161 1.0701806
```

1.3 Question 1 c:

```
plot(density(mu1[-(1:5)]),type="l", xlab="mu", ylab="Posterior density ",
cex=1.5, xlim=c(0,4), col="red", lty=1, main="Posterior distributions for Each temperature")
lines(density(mu2[-(1:5)]), col="cyan",lty=1)
lines(density(mu3[-(1:5)]),col="black",lty=1)
lines(density(mu4[-(1:5)]),col="blue", lty=1)
legend("topright", c("Temp 40", "Temp 60", "Temp 80", "Temp100"),lty=c(1, 1, 1,1), col=c("red","blue","g
```



We now give summary statistics.

```
#some summary statistics
#for temperature of 40
temp1<-rnorm(nbig,mean=mu1,sd=1/sqrt(tau))
mean(temp1)
```

```
## [1] 1.075937
```

```
sd(temp1)
```

```
## [1] 0.4081899
```

```
quantile(temp1, probs=seq(0,1,0.025)) # can use this for quantiles
```

```
##          0%        2.5%          5%        7.5%         10%       12.5%
## -1.1881030  0.2735042  0.4093063  0.4954395  0.5623692  0.6164876
##         15%       17.5%         20%       22.5%         25%       27.5%
##  0.6626517  0.7047026  0.7431994  0.7795541  0.8121028  0.8431112
```

```
##        30%     32.5%       35%     37.5%       40%     42.5% 
## 0.8721747 0.8989782 0.9270925 0.9536297 0.9789093 1.0031758 
##       45%     47.5%       50%     52.5%       55%     57.5% 
## 1.0280829 1.0542737 1.0776098 1.1017753 1.1270784 1.1522966 
##       60%     62.5%       65%     67.5%       70%     72.5% 
## 1.1768720 1.1998321 1.2244478 1.2517127 1.2795345 1.3090601 
##       75%     77.5%       80%     82.5%       85%     87.5% 
## 1.3409364 1.3741297 1.4096922 1.4452089 1.4866712 1.5288213 
##       90%     92.5%       95%     97.5%      100% 
## 1.5825283 1.6540241 1.7430060 1.8905469 2.8397865
```

```r
#for temperature of 60
temp2<-rnorm(nbig,mean=mu2,sd=1/sqrt(tau))
mean(temp2)
```

```
## [1] 1.599299
```

```r
sd(temp2)
```

```
## [1] 0.407251
```

```r
quantile(temp2, probs=seq(0,1,0.025)) # can use this for quantiles
```

```
##         0%       2.5%         5%       7.5%        10%      12.5% 
## -0.9018925  0.8057017  0.9404629  1.0217177  1.0885434  1.1407553 
##        15%      17.5%        20%      22.5%        25%      27.5% 
##  1.1866114  1.2282171  1.2672095  1.3042119  1.3368791  1.3648965 
##        30%      32.5%        35%      37.5%        40%      42.5% 
##  1.3926090  1.4197036  1.4464276  1.4699231  1.4970850  1.5230949 
##        45%      47.5%        50%      52.5%        55%      57.5% 
##  1.5482864  1.5719386  1.5954392  1.6215553  1.6472966  1.6714743 
##        60%      62.5%        65%      67.5%        70%      72.5% 
##  1.6980285  1.7221561  1.7488720  1.7758051  1.8018586  1.8317326 
##        75%      77.5%        80%      82.5%        85%      87.5% 
##  1.8614844  1.8976021  1.9315419  1.9679328  2.0100466  2.0525874 
##        90%      92.5%        95%      97.5%       100% 
##  2.1086761  2.1765653  2.2696366  2.4221266  3.6349408
```

```r
#for temperature of 80
temp3<-rnorm(nbig,mean=mu3,sd=1/sqrt(tau))
mean(temp3)
```

```
## [1] 2.298067
```

```r
sd(temp3)
```

```
## [1] 0.4100596
```

```r
quantile(temp3, probs=seq(0,1,0.025)) # can use this for quantiles
```

```
##         0%       2.5%         5%       7.5%        10%      12.5% 
## -0.1734029  1.4762093  1.6194603  1.7168903  1.7848460  1.8394244 
##        15%      17.5%        20%      22.5%        25%      27.5% 
##  1.8848909  1.9315313  1.9706710  2.0044349  2.0367474  2.0661966 
##        30%      32.5%        35%      37.5%        40%      42.5% 
##  2.0959162  2.1235632  2.1495094  2.1753845  2.2012886  2.2269243 
##        45%      47.5%        50%      52.5%        55%      57.5% 
##  2.2517509  2.2771972  2.3004862  2.3246684  2.3492710  2.3738857
```

```
##        60%      62.5%        65%      67.5%        70%      72.5%
##  2.4000962  2.4250823  2.4508861  2.4778945  2.5052257  2.5352723
##        75%      77.5%        80%      82.5%        85%      87.5%
##  2.5665385  2.5951755  2.6292515  2.6684601  2.7086424  2.7535407
##        90%      92.5%        95%      97.5%       100%
##  2.8096892  2.8715103  2.9561591  3.1040431  4.0449722
```

```r
#for temperature of 100
temp4<-rnorm(nbig,mean=mu4,sd=1/sqrt(tau))
mean(temp4)
```

```
## [1] 3.165984
```

```r
quantile(temp4, probs=seq(0,1,0.025)) # can use this for quantiles
```

```
##        0%      2.5%        5%      7.5%       10%     12.5%       15%
## 0.8356794 2.3569307 2.5046330 2.5944238 2.6557061 2.7067012 2.7538854
##     17.5%       20%     22.5%       25%     27.5%       30%     32.5%
## 2.7968554 2.8350830 2.8692147 2.9007361 2.9330345 2.9631431 2.9915510
##       35%     37.5%       40%     42.5%       45%     47.5%       50%
## 3.0180532 3.0430122 3.0707183 3.0956220 3.1199232 3.1445478 3.1676430
##     52.5%       55%     57.5%       60%     62.5%       65%     67.5%
## 3.1927804 3.2174989 3.2418744 3.2686004 3.2948031 3.3214517 3.3485636
##       70%     72.5%       75%     77.5%       80%     82.5%       85%
## 3.3750916 3.4036304 3.4338766 3.4666124 3.5001802 3.5386364 3.5781617
##     87.5%       90%     92.5%       95%     97.5%      100%
## 3.6223528 3.6741157 3.7343435 3.8217921 3.9505292 5.0653461
```

```r
sd(temp4)
```

```
## [1] 0.4071478
```

1.4 Question 1 d:

What is the posterior distribution of the difference in number of cells grown at a temperature of 40 versus 80? What is the posterior probability that there will be more cells grown at a temperature of 40 versus 80?

Below we calculate the posterior distribution and probability.

Posterior probability will be 0, since temp3-temp1 is always less than 0 at all occasions, as indicated by the quantiles and the plot below

```r
newmu<-mu1-mu4
mean(temp1-temp3)
```

```
## [1] -1.22213
```

```r
sd(temp1-temp3)
```

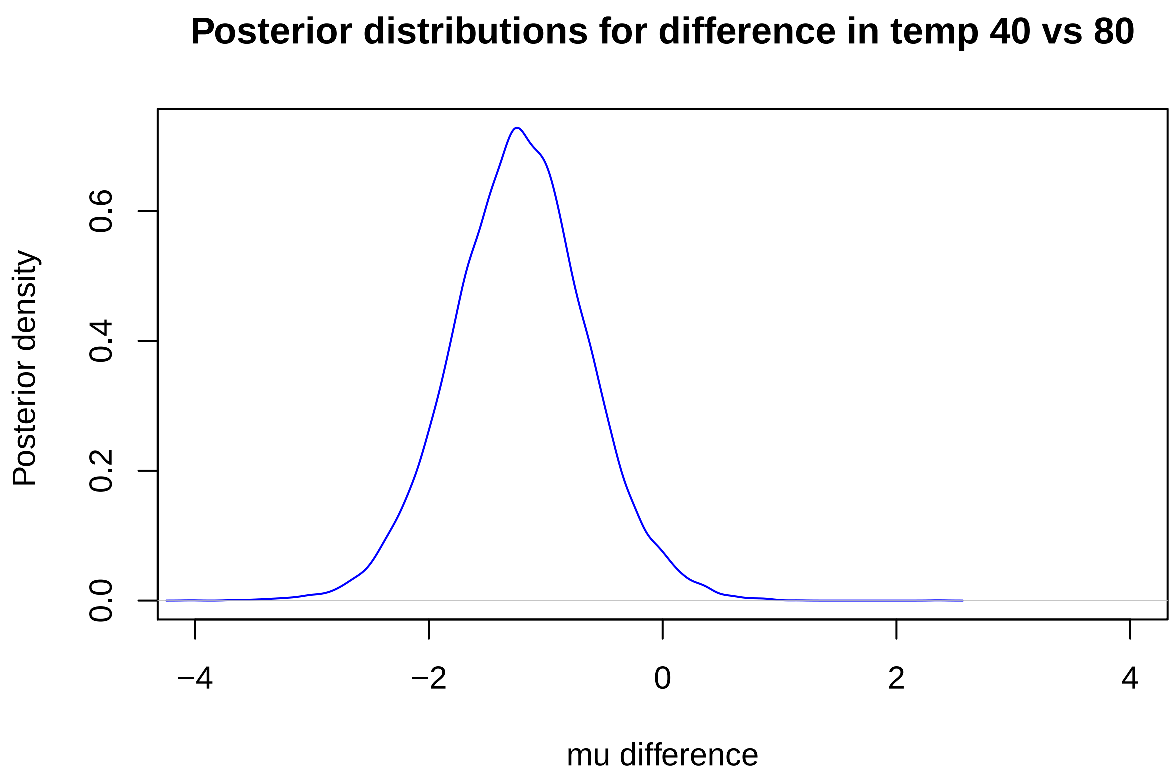
```
## [1] 0.5759653
```

```r
quantile(temp1-temp3, probs=seq(0,1,0.025))
```

```
##          0%        2.5%          5%        7.5%         10%       12.5%
## -4.03877161 -2.36003500 -2.16764043 -2.04007871 -1.94704866 -1.86734188
##         15%       17.5%         20%       22.5%         25%       27.5%
## -1.80317326 -1.74417089 -1.69316202 -1.64572156 -1.59937602 -1.55496531
##         30%       32.5%         35%       37.5%         40%       42.5%
## -1.51062791 -1.47206997 -1.43357653 -1.39568819 -1.35622523 -1.32117807
```

```
##         45%       47.5%         50%       52.5%         55%       57.5%
## -1.28843222 -1.25556285 -1.22109112 -1.18818651 -1.15139648 -1.11640377
##         60%       62.5%         65%       67.5%         70%       72.5%
## -1.08078435 -1.04330689 -1.00711812 -0.97026889 -0.93329153 -0.89189007
##         75%       77.5%         80%       82.5%         85%       87.5%
## -0.85098015 -0.80512530 -0.75434778 -0.70148875 -0.64090741 -0.57990318
##         90%       92.5%         95%       97.5%        100%
## -0.50300421 -0.41192743 -0.27819896 -0.06210936  2.35986813
```

```
plot(density(temp1-temp3),type="l", xlab="mu difference", ylab="Posterior density ",    cex=1.5, xlim=c
```



2 Question 2

2.1 Question 2 a

```
smokeDat=read.csv("SmokeAgeDeath.csv")
library(R2OpenBUGS)
```

```
## Warning: package 'R2OpenBUGS' was built under R version 3.4.4
```

```
library(coda)
```

```
## Warning: package 'coda' was built under R version 3.4.4
```

```
#Code for OpenBUGS model given below
cat("
model{
    for(i in 1:20)
    {
    #smoke   age   death  pyears

    death[i]~dpois(lam[i])
    log(lam[i]) <- log(pyears[i]) + beta0 + beta.s[smoke[i]] + beta.c[age[i]] + b[i]
    b[i] ~dnorm(0,tau)
    b.adj[i] <- b[i] - mean(b[])
    }
    for(is in 1:4){
    beta.s[is]~dnorm(0,tau.s)
    beta.s.adj[is] <- beta.s[is] -mean(beta.s[])
    }
    for(ic in 1:5){
    beta.c[ic]~dnorm(0,tau.c)
    beta.c.adj[ic] <- beta.c[ic] - mean(beta.c[])
    }
    #  Note: total person-years per categories is less than 115,000
    #  ln(115000) ~  = 11.65....
    #  so rate has to be bigger than 1/115,000 and log(rate) > -11.65...
    #  so log( base rate) should be between about -12 and 12.
    # 1/12/12 is about .00694
    beta0 ~ dnorm(0, .00694)
    beta0.adj <- beta0 + mean(b[]) + mean(beta.s[])+ mean(beta.c[])
    # for the <extra poisson variation> ...
    # assume bounded by very big number... say 1000 times...
    # so log(1000) is about 2.3*4 which is about 9.2
    std ~ dunif(0, 9)
    tau <- 1/std/std
    # for the relative risk between groups... a very large number would be 100 times,
    #  so, log(100) is about 2.3*2 or about 4.6
    #  also, note that 1/5/5 is 0.04
    #
    std.s ~dunif(0, 5)
    tau.s <- 1/std.s/std.s
    std.c ~ dunif(0,5)
    tau.c <- 1/std.c/std.c
    beta.o~dnorm(0, .04)
    }", file="smokemod.txt")

#defining parameters and data for the bugs function

params=c("beta.s.adj",  "std.s", "beta.c.adj", "std.c", "beta0.adj", "std")
attach(smokeDat)

bug.dat=list("smoke","age","death", "pyears")
init.fun=function(){list(
  beta.s=rnorm(4), std.s=runif(1,1,2),
  beta.c=rnorm(5), std.c=runif(1,1,2),
  std=runif(1,1,2), beta0=rnorm(1),
```

```r
  b=rnorm(20,0,.1))}

#using openBUGS to run our model, this code also gives the posterior which we need for 2a in OpenBUGS

smokeBug0=bugs(bug.dat, init.fun, params, model.file="smokemod.txt",
              n.chains=5, n.iter=8000, n.burnin=1000, debug=TRUE  #for production
              # n.chains=5, n.iter=6000, n.burnin=1000, debug=TRUE   #for testing
)
```

openBUGS Results output showing the posterior distributions:

```r
library(knitr)
results = read.csv("question2a_results.csv")
kable(results, caption = "Summary Statistics")
```

Table 1: Summary Statistics

X	mean	sd	val2.5pc	median	val97.5pc	sample
beta.c.adj[1]	-1.34600	0.24290	-1.69100	-1.33700	-1.0070	35000
beta.c.adj[2]	-0.72810	0.20150	-1.01300	-0.71950	-0.4425	35000
beta.c.adj[3]	-0.02599	0.16030	-0.27570	-0.02343	0.2166	35000
beta.c.adj[4]	0.69470	0.18180	0.44330	0.68340	0.9175	35000
beta.c.adj[5]	1.40600	0.12360	1.20800	1.40400	1.6000	35000
beta.s.adj[1]	-0.43050	0.31510	-0.64100	-0.46010	-0.2773	35000
beta.s.adj[2]	-0.53190	0.11560	-0.73930	-0.53460	-0.3289	35000
beta.s.adj[3]	0.05386	0.20450	-0.21570	0.06957	0.3238	35000
beta.s.adj[4]	0.90860	0.25420	0.69020	0.92740	1.1540	35000
beta0.adj	-7.37400	0.07059	-7.51300	-7.37300	-7.2400	35000
deviance	102.20000	4.35500	95.67000	101.50000	112.5000	35000
std	0.12080	0.35690	0.00356	0.07111	3.4030	35000
std.c	1.72100	0.89100	0.71550	1.43600	4.1400	35000
std.s	1.41600	0.97450	0.43360	1.03600	4.2980	35000

```r
#Deviance Information
DevianceInformation = read.csv("DevianceInformation.csv")
kable(DevianceInformation, caption = "Deviance Information")
```

Table 2: Deviance Information

X	Dbar	Dhat	DIC	pD
death	102.2	92.66	111.7	9.544
total	102.2	92.66	111.7	9.544

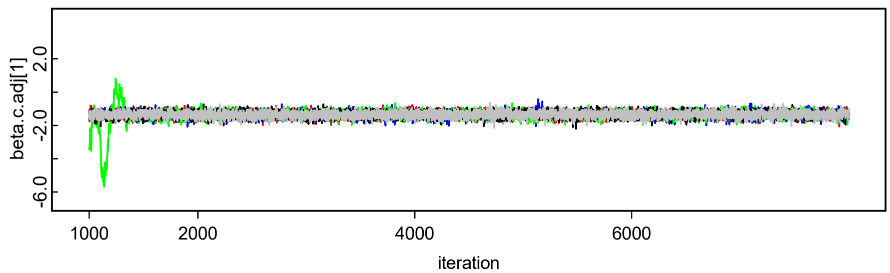
model is syntactically correct
data loaded
model compiled
initial values loaded but chain contains uninitialized variables
initial values loaded but chain contains uninitialized variables
initial values loaded but chain contains uninitialized variables
initial values loaded but chain contains uninitialized variables
initial values loaded but chain contains uninitialized variables
initial values generated, model initialized
model is updating
1000 updates took 1 s
model is updating
7000 updates took 9 s
CODA files written
Summary statistics

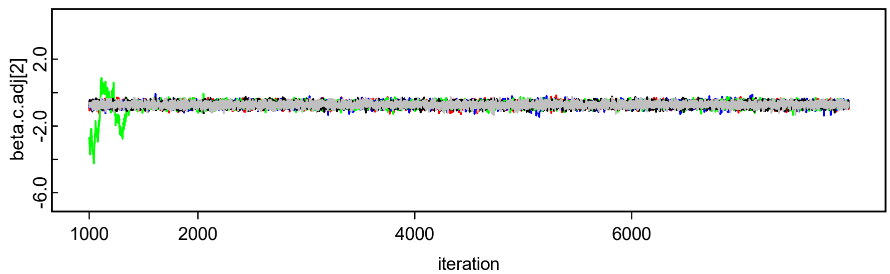
	mean	sd	val2.5pc	median	val97.5pc	sample
beta.c.adj[1]	-1.346	0.2429	-1.691	-1.337	-1.007	35000
beta.c.adj[2]	-0.7281	0.2015	-1.013	-0.7195	-0.4425	35000
beta.c.adj[3]	-0.02599	0.1603	-0.2757	-0.02343	0.2166	35000
beta.c.adj[4]	0.6947	0.1818	0.4433	0.6834	0.9175	35000
beta.c.adj[5]	1.406	0.1236	1.208	1.404	1.6	35000
beta.s.adj[1]	-0.4305	0.3151	-0.641	-0.4601	-0.2773	35000
beta.s.adj[2]	-0.5319	0.1156	-0.7393	-0.5346	-0.3289	35000
beta.s.adj[3]	0.05386	0.2045	-0.2157	0.06957	0.3238	35000
beta.s.adj[4]	0.9086	0.2542	0.6902	0.9274	1.154	35000
beta0.adj	-7.374	0.07059	-7.513	-7.373	-7.24	35000
deviance	102.2	4.355	95.67	101.5	112.5	35000
std	0.1208	0.3569	0.00356	0.07111	3.403	35000
std.c	1.721	0.891	0.7155	1.436	4.14	35000
std.s	1.416	0.9745	0.4336	1.036	4.298	35000

Deviance information

	Dbar	Dhat	DIC	pD
death	102.2	92.66	111.7	9.544
total	102.2	92.66	111.7	9.544

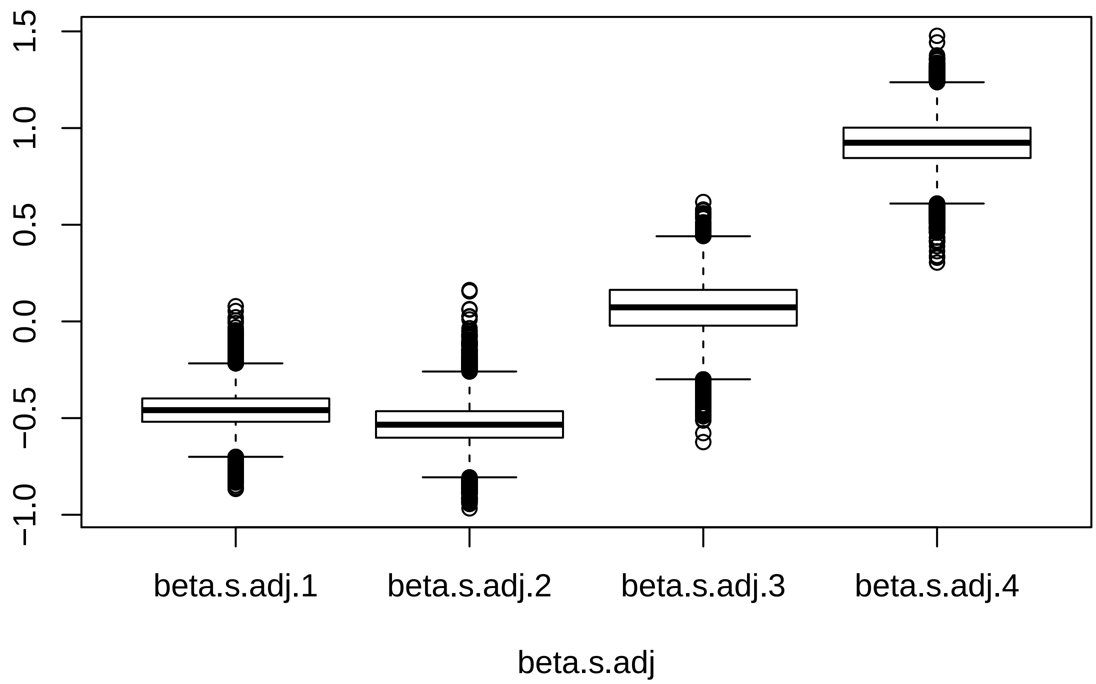
History



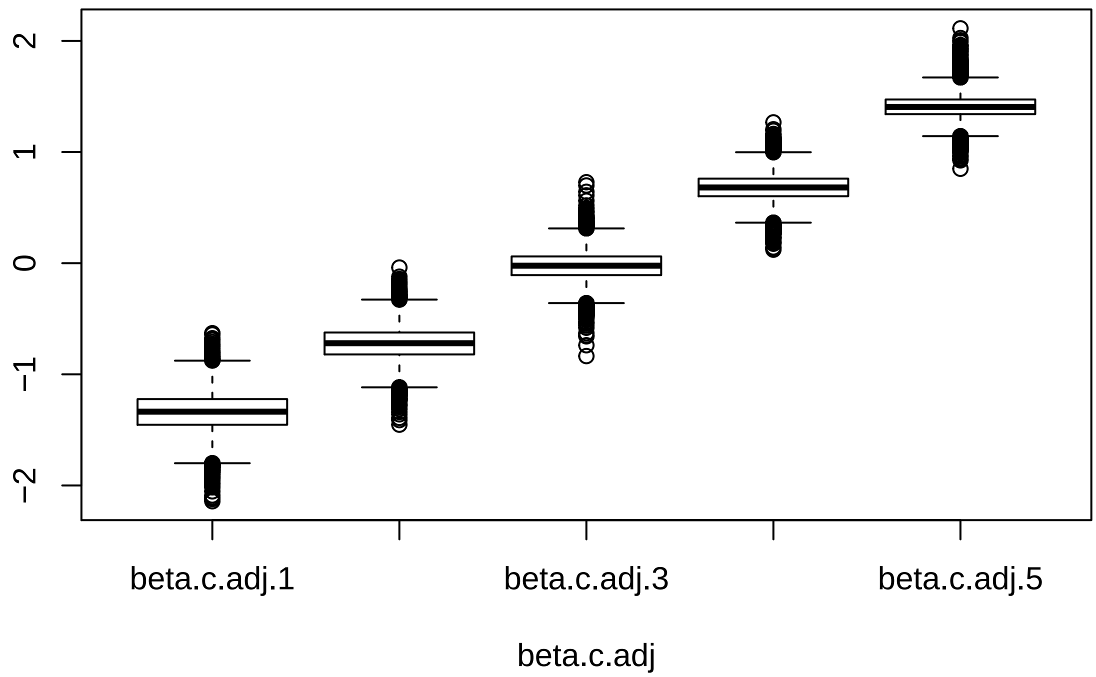


The above plots show the convergence in the chains after 8000 iterations

```r
#boxplots for smoking categories

boxplot(data.frame( (smokeBug0$sims.list)["beta.s.adj"]),
        xlab="beta.s.adj")
```




```r
#boxplots for age categories

boxplot(data.frame( (smokeBug0$sims.list)["beta.c.adj"]),
        xlab="beta.c.adj")
```



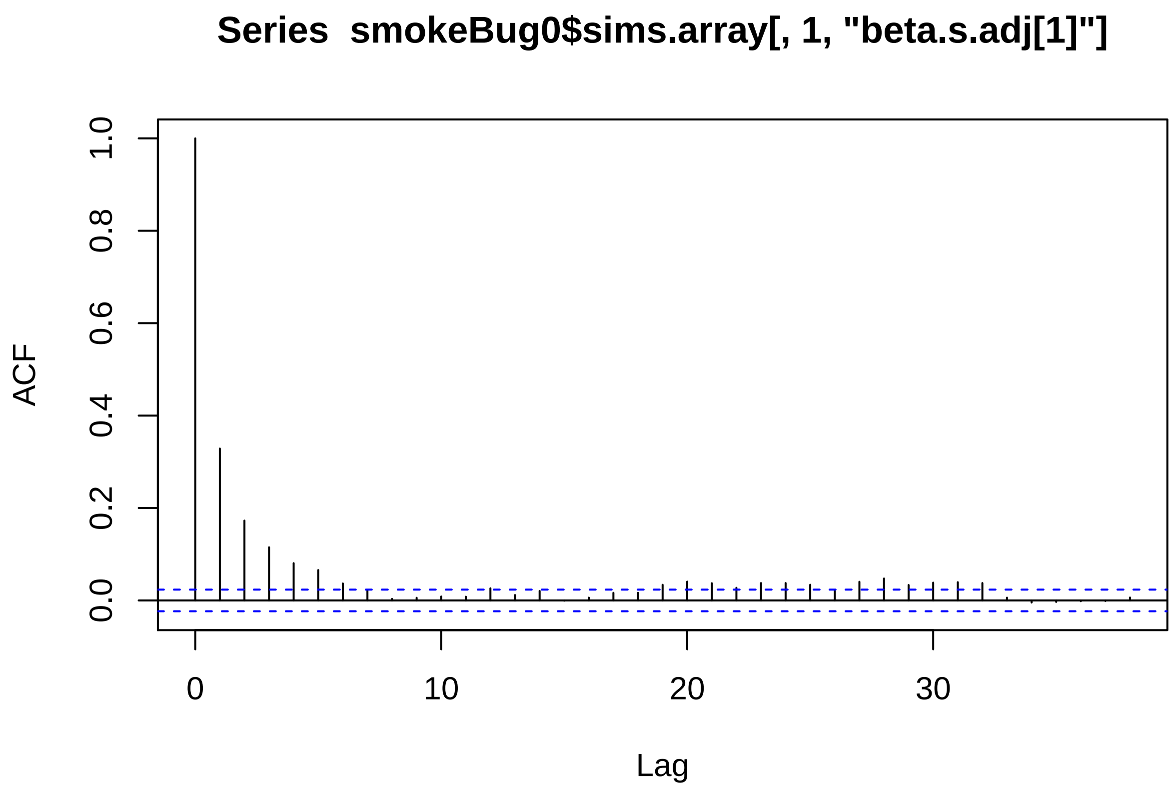
2.2 Question 2 b.

The following plots show that our model converges.

Trace plots have been shown in the output given above that shows the convergence in chains after 8000 iterations.

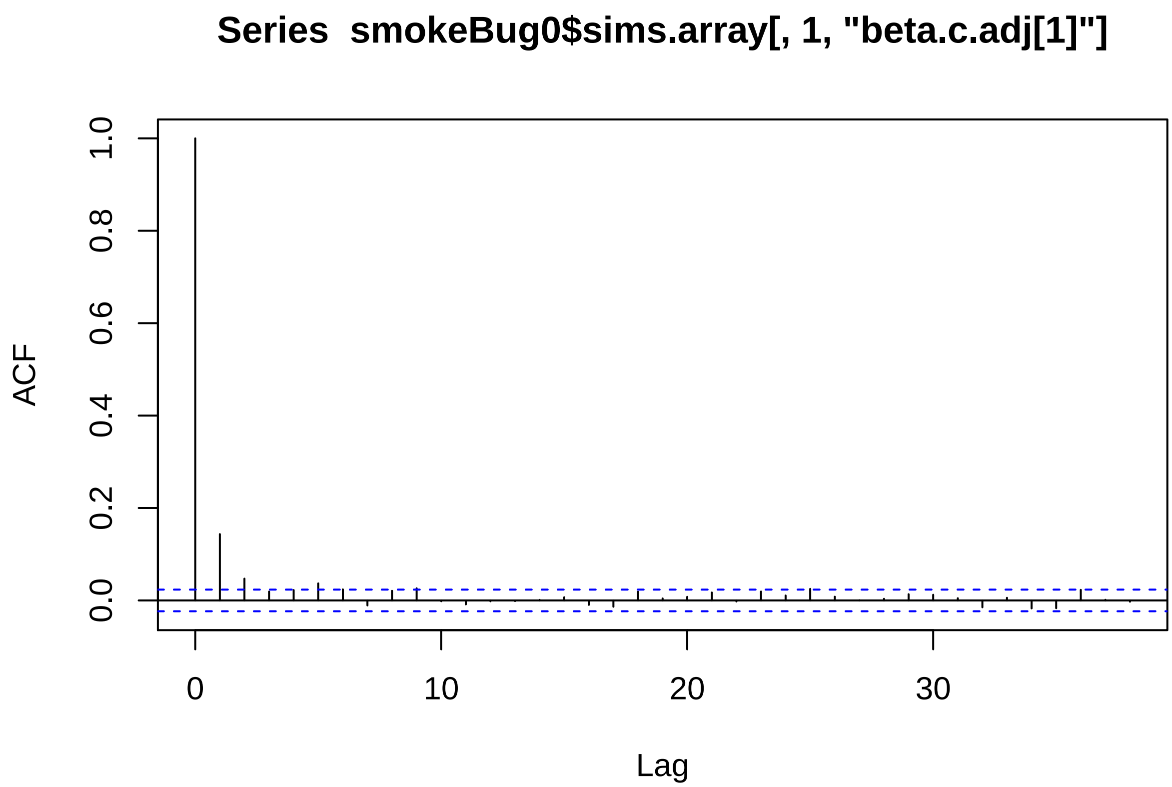
The autocorrelation functions plots are given below.

```
# The following plots show that our model converges
#autocorrelation functions given below
#trace plots have been shown in the output given above that shows the convergence in chains after 8000
acf(smokeBug0$sims.array[,1,"beta.s.adj[1]"])
```

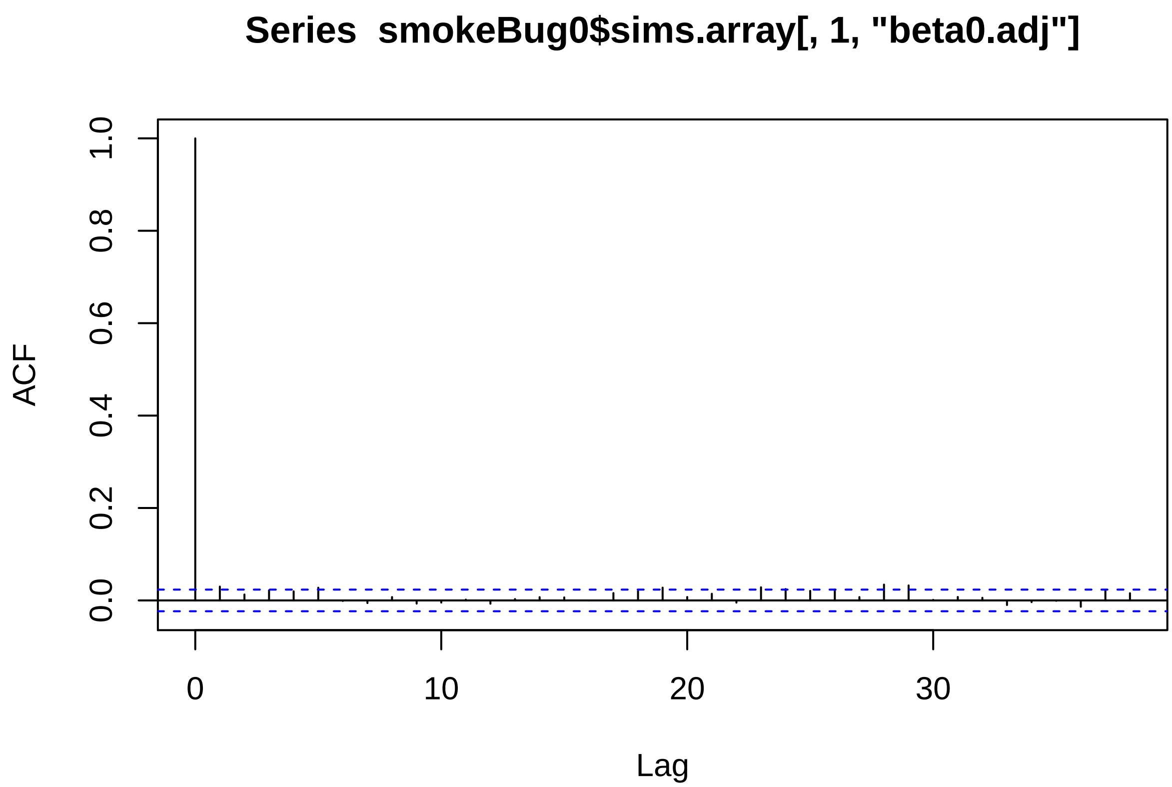


After 10 lags the plot seems to have converged. We observe the same for age which converges twice as #fast as shown below

```
#After 10 lags the plot seems to have converged. We observe the same for age which converges twice as #
acf(smokeBug0$sims.array[,1,"beta.c.adj[1]"])
```




```r
acf(smokeBug0$sims.array[,1,"beta0.adj"])
```



2.3 Question 2 c.

```
risk20<-smokeBug0$sims.list$beta.s.adj[,4]-smokeBug0$sims.list$beta.s.adj[,1]
mean(risk20)
```

```
## [1] 1.380155
```

```
exp(mean(risk20))
```

```
## [1] 3.975519
```

The above shows those who smoke >20 cigarettes per day versus nonsmoker have approximately 4 times (=3.98) the risk. We take the exponential in the last line since the model gives us the log of the risks.

```
quantile(exp(risk20), c(.025, .975)) # gives the 95% Cis for above risk
```

```
##     2.5%    97.5%
## 2.842539 5.443937
```

3 Question 3

3.1 Question 3 a

R-code given below for defining data parameters, model and priors

```
library(R2OpenBUGS)

diabetes = read.csv("DiabetesDrugEffect.csv")
attach(diabetes)

## Model 1

cat("
model{
  for( i in 1:12){
   diff[i] ~ dnorm(mu[i], Sediff[i])
   mu[i] ~ dnorm(theta, sd)
   }
  theta ~ dnorm(0, .1) # so, sd =5.  exp(5) ~ 148 which is huge
  sd ~ dnorm(0, .1)
  }
  ", file="diabetesMod3.txt")

bugM3.dat=list( "diff", "Sediff")  # what variable you need in the model

initM3.fun=function(){ list( theta=rnorm(1) ,
                             sd = rnorm(1),
                             mu = rnorm(1)

    ) }


paramsM3=c("mu", "theta", "sd")
     ### what variables you want to monitor
```

```
#### Could change the code below...
diabetesBaseM3=bugs(bugM3.dat, initM3.fun, paramsM3, model.file="diabetesMod3.txt",
    n.chains=3, n.iter=3000, n.burnin=500,
    n.thin=1 , debug=TRUE
        )

print(diabetesBaseM3,dig=3)


SArray= diabetesBaseM3$sims.array
vname=attr(SArray,"dimnames")[3][[1]]
chainL=attr(SArray,"dim")[1][[1]]
for(i in 1:length(vname)){
    nn=vname[i]
    plot(density(SArray[,,nn]), main=nn)
    xnul=locator(1)
    acf( SArray[,1,nn], main=nn)  #note: this is only for 1st chain
    xnul=locator(1)
    matplot(1:chainL,SArray[,,nn], main=nn,xlab="index",type="l")
    xnul=locator(1)
}
```

```
##  Model 2
```

```
cat("
model{
  for( i in 1:12){
   diff[i] ~ dnorm(theta, Sediff[i] + sd)
   }
  theta ~ dnorm(0, .1)
  sd ~ dnorm(0, .1)
  }
  ", file="diabetesMod2.txt")

bugM3.dat=list( "diff", "Sediff")  # what variable you need in the model

initM2.fun=function(){ list( theta=rnorm(1) ,
                             sd = rnorm(1)
    ) }


paramsM2=c( "theta", "sd")
     ### what variables you want to monitor
```

```
#### Could change the code below...
diabetesBaseM2=bugs(bugM3.dat, initM2.fun, paramsM2, model.file="diabetesMod2.txt",
    n.chains=3, n.iter=3000, n.burnin=500,
    n.thin=1 , debug=TRUE
        )

print(diabetesBaseM2,dig=3)

SArray= diabetesBaseM2$sims.array
vname=attr(SArray,"dimnames")[3][[1]]
chainL=attr(SArray,"dim")[1][[1]]
for(i in 1:length(vname)){
    nn=vname[i]
    plot(density(SArray[,,nn]), main=nn)
    xnul=locator(1)
    acf( SArray[,1,nn], main=nn)  #note: this is only for 1st chain
    xnul=locator(1)
    matplot(1:chainL,SArray[,,nn], main=nn,xlab="index",type="l")
    xnul=locator(1)
}
```