2c

Benjamin Moran

8 May 2017

(b) Use WinBUGS for a Bayesian analysis of (a), and find reasonable point and interval estimates for σ_1^2 and σ_2^2. Include graphs, including one of the joint posterior. [6 marks]

```
(BT3 <- summary(aov(Sample~(I/J/K),df3)))
(m1 <- BT3[[1]]$`Mean Sq`[3])
(m2 <- (BT3[[1]]$`Mean Sq`[2]-BT3[[1]]$`Mean Sq`[3])/2)
(m3 <- (BT3[[1]]$`Mean Sq`[1]-BT3[[1]]$`Mean Sq`[2])/4)
(mAll <- m1+m2+m3)
(m1/mAll)
(m2/mAll)
(m3/mAll)
```

```
#define the model
library(R2OpenBUGS)
boxtiao3model <- function(){
  for(i in 1 : I) {
    m[i] ~ dnorm(theta,  tau.I)
    for(j in 1 : J) {
      y[i , j] ~ dnorm(m[i], tau.J)
      for(k in 1: K){
        z[i,j,k] ~ dnorm(y[i,j], tau.K)
      }
    }
  }
  sigma2.I <- 1/tau.I
  tau.I ~ dgamma(0.001, 0.001)
  sigma2.J <- 1/tau.J
  tau.J ~ dgamma(0.001, 0.001)
  sigma2.K <- 1/tau.K
  tau.K ~ dgamma(0.001, 0.001)
  theta ~ dflat()
}
# write the model code out to a file
write.model(boxtiao3model, "boxtiao3model.txt")
model.file1 = paste(getwd(),"boxtiao3model.txt", sep="/")
#prepare the data for input into OpenBUGS
I <- 10
J <- 2
K <- 2
y <- as.matrix(df3)

data <- list ("I","J","K","y", .Dim = c(10,2,2))

#initialization of variables
inits <- function(){
  list(theta=1, tau.I=1, tau.J=4, tau.K=2)
}
```

```
WINE="/opt/local/bin/wine"
WINEPATH="/opt/local/bin/winepath"
OpenBUGS.pgm=paste0("/Users/benjamin/Applications/wine/",
                    "drive_c/ProgramFiles/OpenBUGS/OpenBUGS323/OpenBUGS.exe")

#these are the parameters to save
parameters = c("sigma2.I","sigma2.J","sigma2.K")

#run the model
bt3.sim <- bugs(data,
                inits,
                model.file = model.file1,
                parameters=parameters,
                n.burnin = 2000,
                n.chains = 5,
                n.iter = 10000,
                OpenBUGS.pgm=OpenBUGS.pgm,
                WINE=WINE,
                WINEPATH=WINEPATH,
                useWINE=T,
                codaPkg = T,
                debug = T)
```

```
out.coda <- read.bugs(dyes.sim)
plot(out.coda)
HPDinterval(out.coda)
densityplot(out.coda)
plot(out.coda)
(out.summary <- summary(out.coda, q=c(0.025, 0.975)))

sigma2_with_stat <- c(out.summary$statistics[2,1],
out.summary$q["sigma2.btw",])
sigma2_btw_stat <- c(out.summary$statistics[3,1],
out.summary$q["sigma2.with",])

summary_stats <- rbind(sigma2.with = sigma2_with_stat,sigma2.btw = sigma2_btw_stat)
colnames(summary_stats)[1] <- "Mean"
summary_stats
```

```
dfI <-  rep(seq(1,10),4)
dfJ <- c(rep(1,20),rep(2,20))
dfK <- c(rep(1,10),rep(2,10),rep(1,10),rep(2,10))
df <- cbind(dfI,dfJ,dfK)

dfJ1K1 <- c(2.004,4.342,0.869,3.531,2.579,-1.404,-1.676,1.670,2.141,-1.229)
dfJ1K2 <- c(2.713,4.229,-2.621,4.185,4.271,-1.003,-0.208,2.426,3.527,-0.596)
dfJ2K1 <- c(0.602,3.344,-3.896,1.722,-2.101,-0.755,-9.139,1.834,0.462,4.471)
dfJ2K2 <- c(0.252,3.057,-3.696,0.380,0.651,-2.202,-8.653,1.200,0.665,1.606)
df2 <- c(dfJ1K1,dfJ1K2,dfJ2K1,dfJ2K2)
df <- NA
for(i in 1:length(dfJ1K1)){
  test <- c(dfJ1K1[i],dfJ1K2[i],dfJ2K1[i],dfJ2K2[i])
  df <- c(df,test)
}
```

```
# Model

  model{
    for(i in 1: I){
      a[i] ~ dnorm(0.0, tauI)
      for(j in 1:J){
        b[i,j] ~ dnorm(0.0, tauJ)
        for(k in 1:K){
          e[i,j,k] <- theta + a[i]  + b[i,j]
          y[i,j,k] ~ dnorm(e[i,j,k], tauK)
        }
      }
    }

# Priors

  theta ~ dflat()
  tauI ~ dgamma(0.001, 0.001)
  tauJ ~ dgamma(0.001, 0.001)
  tauK ~ dgamma(0.001, 0.001)

  sigma2I <-1/tauI
  sigma2J <-1/tauJ
  sigma2K <-1/tauK

  denom <- sigma2I + sigma2J + sigma2K
  r1 <- sigma2I/denom
  r2 <- sigma2J/denom
  r3 <- sigma2K/denom
}

# Initial values

  INITS

  list(
       theta=0,
       a = c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0),
       b = structure(
            .Data = c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                      0, 0, 0, 0, 0, 0, 0, 0, 0, 0),
            .Dim = c(10, 2)
            ),
       tauI = 1.0,
       tauJ = 1.0,
       tauK = 1.0
       )

# Data (in tabular format)

  DATA

  list(
```

```
      I=10,
      J=2,
      K=2,
      y = structure(
           .Data=c(2.004,   2.713,  0.602,  0.252,
                   4.342,   4.229,  3.344,  3.057,
                   0.869,  -2.621, -3.896, -3.696,
                   3.531,   4.185,  1.722,  0.380,
                   2.579,   4.271, -2.101,  0.651,
                   -1.404, -1.003, -0.755, -2.202,
                   -1.676, -0.208, -9.139, -8.653,
                   1.670,   2.426,  1.834,  1.200,
                   2.141,   3.527,  0.462,  0.665,
                   -1.229, -0.596,  4.471,  1.606),
           .Dim=c(10, 2, 2)
           )
      )
```

```
#define the model
library(R2OpenBUGS)
threecompmodel <- function(){
    for( i in 1 : I ) {
        a[i] ~ dnorm(theta, tauI)
        for( j in 1 : J ) {
            b[i , j] ~ dnorm(a[i], tauJ)
            for( k in 1 : K ) {
                e[i , j , k] <- mu0 + a[i] + b[i , j]
                y[i , j , k] ~ dnorm(e[i , j , k], tauK)
            }
        }
    }

    ## Priors

  theta ~ dnorm(0.0,1.00E-04) # Batch Mean
    mu0 ~ dnorm(0.0,1.00E-04) # Sample Mean
    tauI ~ dgamma(0.001, 0.001) # Between Batch variation
    tauJ ~ dgamma(0.001, 0.001) # Between Sample variation
    tauK ~ dgamma(0.001, 0.001) # Between Test variation

    sigma2I <- 1 / tauI
    sigma2J <- 1 / tauJ
    sigma2K <- 1 / tauK

    sigma2JK <- sigma2J / sigma2K

    denom <- sigma2I + sigma2J + sigma2K
    r1 <- sigma2I / denom
    r2 <- sigma2J / denom
    r3 <- sigma2K / denom
}
# write the model code out to a file
write.model(threecompmodel, "threecompmodelmodel.txt")
model.file.3comp = paste(getwd(),"threecompmodelmodel.txt", sep="/")
```

```
#prepare the data for input into OpenBUGS

data <-
  list(I=10,J=2,K=2,y = structure(
          .Data=c(2.004,   2.713,  0.602,  0.252,
                    4.342,   4.229,  3.344,  3.057,
                    0.869,  -2.621, -3.896, -3.696,
                    3.531,   4.185,  1.722,  0.380,
                    2.579,   4.271, -2.101,  0.651,
                    -1.404, -1.003, -0.755, -2.202,
                    -1.676, -0.208, -9.139, -8.653,
                    1.670,   2.426,  1.834,  1.200,
                    2.141,   3.527,  0.462,  0.665,
                    -1.229, -0.596,  4.471,  1.606),
          .Dim=c(10,2,2)
))

#initialization of variables
inits <- function(){
  list(
    mu0 = 0,
  theta=0,
  a = c(0,0,0,0,0,0,0,0,0,0),
    b = structure(
          .Data = c(0,0,0,0,0,0,0,0,0,0,
                    0,0,0,0,0,0,0,0,0,0),
          .Dim = c(10,2)
          ),
  tauI = 1.0,
  tauJ = 1.0,
  tauK = 1.0
)
}

WINE="/opt/local/bin/wine"
WINEPATH="/opt/local/bin/winepath"
OpenBUGS.pgm=paste0("/Users/benjamin/Applications/wine/",
                    "drive_c/ProgramFiles/OpenBUGS/OpenBUGS323/OpenBUGS.exe")

#these are the parameters to save
parameters = c("sigma2I","sigma2J","sigma2K", "sigma2JK", "r1", "r2", "r3")

#run the model
threecomp.sim <- bugs(data,
                  inits,
                  model.file = model.file.3comp,
                  parameters=parameters,
                  n.burnin = 5000,
                  n.chains = 5,
                  n.iter = 50000,
                  OpenBUGS.pgm=OpenBUGS.pgm,
                  WINE=WINE,
                  WINEPATH=WINEPATH,
```

```
                    useWINE=T,
                    codaPkg = F,
                    debug = F)
```

```
library(ggplot2)
library(ggtern)
attach.bugs(threecomp.sim)

threecompsimlist <- tbl_df(threecomp.sim$sims.list)

# densityplot(threecompsimlist$sigma2I)
# densityplot(threecompsimlist$sigma2J)
# densityplot(threecompsimlist$sigma2K)

# sig2I <- threecompsimlist$sigma2I
# sig2J <- threecompsimlist$sigma2J
# sig2K <- threecompsimlist$sigma2K

df <- tbl_df(threecompsimlist$r1)
df <- tbl_df(cbind(rBatches = threecompsimlist$sigma2I, rSamples = threecompsimlist$sigma2J, rTests = tl
denom <- sum(threecomp.sim$summary[1:3,1])

rI <- threecomp.sim$summary[1,1]/denom
rJ <- threecomp.sim$summary[2,1]/denom
rK <- threecomp.sim$summary[3,1]/denom

ggtern(data = df, aes(x=rBatches,z=rSamples,y=rTests)) +
  theme_rgbw() +
  geom_point(size=0.1, colour = "red") +
  geom_confidence_tern(breaks = c(0.95)) +
  theme_clockwise() +
  labs(x="Batches",z="Samples",y="Tests",title="Contours of the posterior distribution of (r1, r2, r3)"
```