Lab 5

Senan Hogan-H.

19 April 2018

Completed with James Kinney, around 50/50 work completed between us.

2 Normal Data

a.

b.

c.

```
schools <- read.table ("schools.dat", header = T)
J <- nrow(schools)
y <- schools$estimate
sigma.y <- schools$sd
data <- list ("J", "y", "sigma.y")
inits <- function(){
  list (theta=rnorm(J,0,100), mu.theta=rnorm(1,0,100), sigma.theta=runif(1,0,100))
}
parameters <- c("theta", "mu.theta", "sigma.theta")

theta.update <- function () {
  theta.hat <-
    (mu / tau ^ 2 + y / sigma.y ^ 2) / (1 / tau ^ 2 + 1 / sigma.y ^
                                          2)
  V.theta <- 1 / (1 / tau ^ 2 + 1 / sigma.y ^ 2)
  rnorm(J, theta.hat, sqrt(V.theta))
}
mu.update <- function () {
  rnorm(1, mean(theta), tau / sqrt(J))
}
tau.update <- function () {
  sqrt(sum((theta - mu) ^ 2) / rchisq(1, J - 1))
}
n.chains <- 5
n.iter <- 1000

sims <- array (NA, c(n.iter, n.chains, J+2))
dimnames(sims) <- list(NULL, NULL, c(paste("theta[", 1:8, "]", sep=""), "mu", "tau"))


for (m in 1:n.chains) {
  mu <- rnorm (1, mean(y), sd(y))
  tau <- runif (1, 0, sd(y))
  for (t in 1:n.iter) {
    theta <- theta.update()
    mu <- mu.update()
    tau <- tau.update()
    sims[t, m,] <- c(theta, mu, tau)
  }
}

big <- c()
for (i in 1:1000) {
  big[i] <- sum(sims[i,1,1:8]>28)
}
mean(big)
```

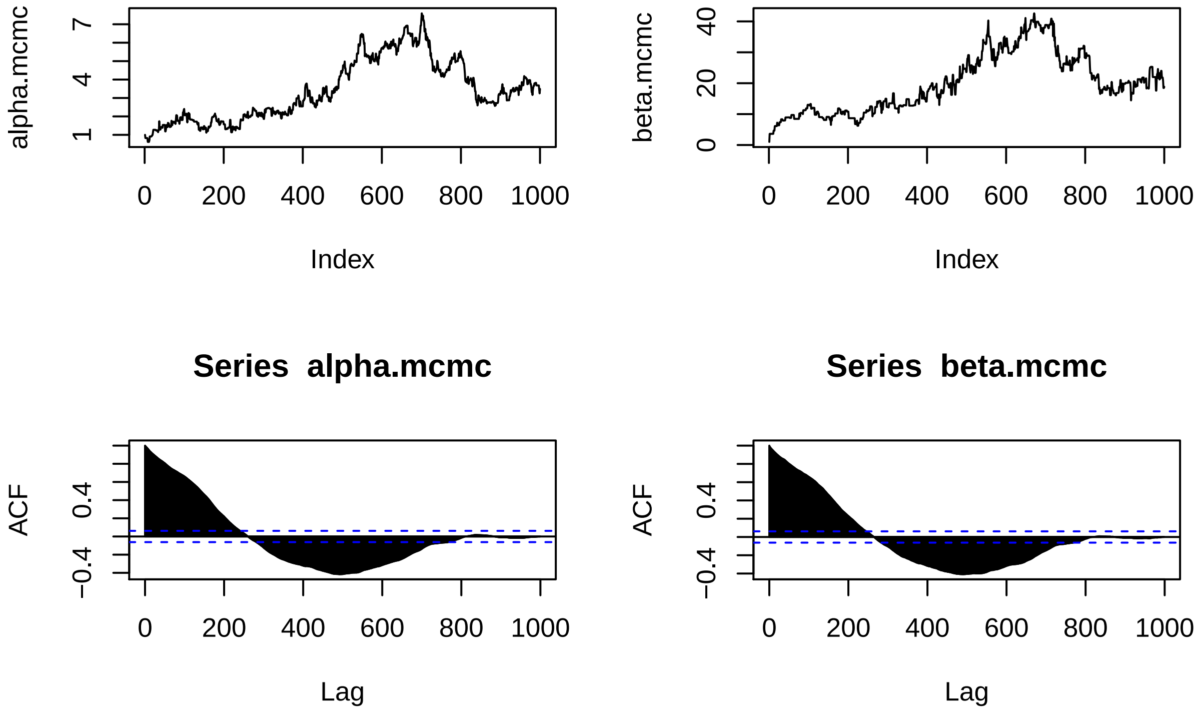
```
## [1] 0.074
```

3 Binomial Data

```
data <- read.table("rat-tumors.txt",header=T)
# Adjust this data to fit for some baseball players where p is known
# rbinom(50, 1, prob = 0.306)

y <- data$y
n <- data$N
J <- length(y)

log.prior <- function(alpha,beta) {
  {-2.5}*log(alpha + beta)
}

draw.thetas <- function(alpha,beta) {
  return(rbeta(J,alpha+y,beta+n-y))
}

draw.alpha <- function(alpha,beta,theta,prop.sd) {
  alpha.star <- rnorm(1,alpha,prop.sd)
  num <- J*(lgamma(alpha.star+beta) - lgamma(alpha.star)) +
    alpha.star*sum(log(theta)) + log.prior(alpha.star,beta)
  den <- J*(lgamma(alpha+beta)      - lgamma(alpha)) +
    alpha     *sum(log(theta)) + log.prior(alpha,beta)
# print(c(alpha,alpha.star,num,den))
  acc <- ifelse((log(runif(1))<=num - den)&&(alpha.star>0),1,0)
  alpha.acc <<- alpha.acc + acc
  return(ifelse(acc,alpha.star,alpha))
}

draw.beta <- function(alpha,beta,theta,prop.sd) {
  beta.star <- rnorm(1,beta,prop.sd)
  num <- J*(lgamma(alpha+beta.star) - lgamma(beta.star)) +
    beta.star*sum(log(1-theta)) + log.prior(alpha,beta.star)
  den <- J*(lgamma(alpha+beta)      - lgamma(beta)) +
    beta     *sum(log(1-theta)) + log.prior(alpha,beta)
# print(c(beta,beta.star,num,den))
  acc <- ifelse((log(runif(1))<=num - den)&&(beta.star>0),1,0)
  beta.acc <<- beta.acc + acc
  return(ifelse(acc,beta.star,beta))
}

################################################################

B <- 0
M <- 1000

MM <- B + M

alpha <- matrix(NA,MM)
beta <- alpha
theta <- matrix(NA,nrow=MM,ncol=J)

# Metropolis tuning parameters
alpha.prop.sd <-  0.25
beta.prop.sd <-   3

# Initial values for the chain
alpha[1] <- 1
beta[1] <- 1
theta[1,] <- draw.thetas(alpha[1],beta[1]) # or theta[1,] <- (y+.5/(n+.5)

# Monitor acceptance frequency
alpha.acc <- 0
beta.acc <- 0

# MCMC simulation
for (m in 2:MM) {
  alpha[m] <- draw.alpha(alpha[m-1],beta[m-1],theta[m-1,],alpha.prop.sd)
  beta[m] <- draw.beta(alpha[m],beta[m-1],theta[m-1,],beta.prop.sd)
  theta[m,] <- draw.thetas(alpha[m],beta[m])
}
```

```
## Warning in log(alpha + beta): NaNs produced
```

```
good <- (B+1):MM

alpha.mcmc <- alpha[good]
beta.mcmc <- beta[good]
theta.mcmc <- theta[good,]


par(mfrow=c(2,2))
plot(alpha.mcmc,type="l")
plot(beta.mcmc,type="l")
acf(alpha.mcmc,1000)
acf(beta.mcmc,1000)
```



```r
print(round(c(alpha.rate=alpha.acc/MM,beta.rate=beta.acc/MM),2))
```

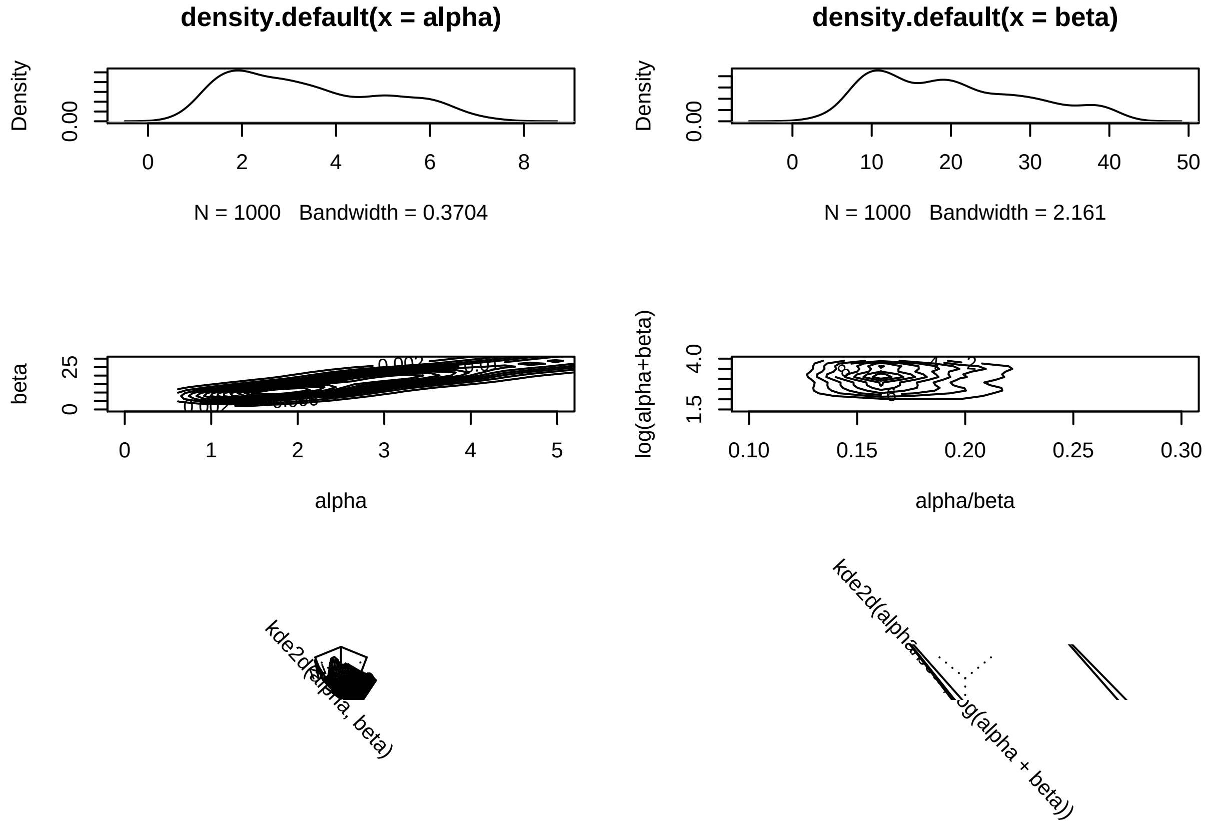
```
## alpha.rate  beta.rate
##       0.64       0.45
```

```r
########################################################################

par(mfrow=c(3,2))

plot(density(alpha))
plot(density(beta))

contour(kde2d(alpha,beta),xlim=c(0,5),ylim=c(0,30),xlab="alpha",ylab="beta")
contour(kde2d(alpha/beta,log(alpha+beta)),xlim=c(0.1,0.3),ylim=c(1.5,4),
        xlab="alpha/beta",ylab="log(alpha+beta)")

persp(kde2d(alpha,beta),theta=45,phi=45,xlim=c(0,5),ylim=c(0,30))
```

```
## Warning in persp.default(kde2d(alpha, beta), theta = 45, phi = 45, xlim =
## c(0, : surface extends beyond the box
```

```r
persp(kde2d(alpha/beta,log(alpha+beta)),theta=45,phi=45,xlim=c(0.1,0.3),ylim=c(1.5,4))
```

```
## Warning in persp.default(kde2d(alpha/beta, log(alpha + beta)), theta =
## 45, : surface extends beyond the box
```


density.default(x = alpha)
Density
0.00
0
2
4
6
8
N = 1000 Bandwidth = 0.3704
density.default(x = beta)
Density
0.00
0
10
20
30
40
50
N = 1000 Bandwidth = 2.161
beta
25
0
0
1
2
3
4
5
alpha
log(alpha+beta)
4.0
1.5
0.10
0.15
0.20
0.25
0.30
alpha/beta
kde2d(alpha, beta)
kde2d(alpha/beta, log(alpha + beta))

```
########################################################################
```

e.