library(coda)

# Data

w = c(1.6907, 1.7242, 1.7552, 1.7842, 1.8113, 1.8369, 1.8610, 1.8839)

n = c(59, 60, 62, 56, 63, 59, 62, 60)

y = c(6, 13, 18, 28, 52, 53, 61, 60)

# posterior density

f = function(mu,th2,th3) {

# settings for priors

a0=0.25; b0=0.25; c0=2; d0=10; e0=2.004; f0=0.001

V = exp(th3)

m1 = exp(th2)

sig = sqrt(V)

x = (w-mu)/sig

xt = exp(x)/(1+exp(x))

h = xt^m1;

loglike = y\*log(h)+(n-y)\*log(1-h)

# prior ordinates

logpriorm1 = a0\*th2-m1\*b0

logpriorV = -e0\*th3-f0/V

logpriormu = -0.5\*((mu-c0)/d0)^2-log(d0)

logprior = logpriormu+logpriorV+logpriorm1

# log posterior

f = sum(loglike)+logprior}

# main MCMC loop

runMCMC = function(samp,mu,th2,th3,T,sd) {

for (i in 2:T+1) {

# canddiates for mu

mucand = mu[i-1]+sd[1]\*rnorm(1,0,1)

f.cand = f(mucand,th2[i-1],th3[i-1])

f.curr = f(mu[i-1], th2[i-1],th3[i-1])

if (log(runif(1)) <= f.cand-f.curr) mu[i] = mucand else

{mu[i] = mu[i-1] }

# candidates for log(m1)

th2cand = th2[i-1]+sd[2]\*rnorm(1,0,1)

f.cand = f(mu[i],th2cand,th3[i-1])

f.curr = f(mu[i],th2[i-1], th3[i-1])

if (log(runif(1)) <= f.cand-f.curr) th2[i] = th2cand else

{th2[i] = th2[i-1]}

# candidates for log(V)

th3cand = th3[i-1]+sd[3]\*rnorm(1,0,1)

f.cand = f(mu[i],th2[i],th3cand)

f.curr = f(mu[i],th2[i],th3[i-1])

if (log(runif(1)) <= f.cand-f.curr) th3[i] = th3cand else

{th3[i] = th3[i-1]}

samp[i-1,1] = mu[i]; samp[i-1,2] = exp(th2[i]); samp[i-1,3] = exp(th3[i]) }

return(samp)}

# number of iterations

T=100000

# warm-up samples

B=50000

B1=B+1

R=T-B

mu=th3=th2=numeric(T)

sd=acc=numeric(3)

# metropolis proposal standard devns

sd[1] = 0.01; sd[2] = 0.2; sd[3] = 0.4

# accumulate samples

samp = matrix(,T,3)

# initial parameter values

mu[1] = 0; th2[1]= 0; th3[1] =0

samp[1,1] = mu[1]; samp[1,2] = exp(th2[1]); samp[1,3] = exp(th3[1])

# first chain

chain1=runMCMC(samp,mu,th2,th3,T,sd)

chain1=chain1[B1:T,]

# posterior summary

quantile(chain1[1:R,1], probs=c(.025,0.5,0.975))

quantile(chain1[1:R,2], probs=c(.025,0.5,0.975))

quantile(chain1[1:R,3], probs=c(.025,0.5,0.975))

# second chain

chain2=runMCMC(samp,mu,th2,th3,T,sd)

chain2=chain2[B1:T,]

# posterior summary

quantile(chain2[1:R,1], probs=c(.025,0.5,0.975))

quantile(chain2[1:R,2], probs=c(.025,0.5,0.975))

quantile(chain2[1:R,3], probs=c(.025,0.5,0.975))

# combine chains

chain1=as.mcmc(chain1)

chain2=as.mcmc(chain2)

combchains = mcmc.list(chain1, chain2)

gelman.diag(combchains)

crosscorr(combchains)

accsum = "Acceptance rates: mu, m1, and sigma^2"

print(accsum)

1 - rejectionRate(combchains)

effectiveSize(combchains)

autocorr.diag(combchains)