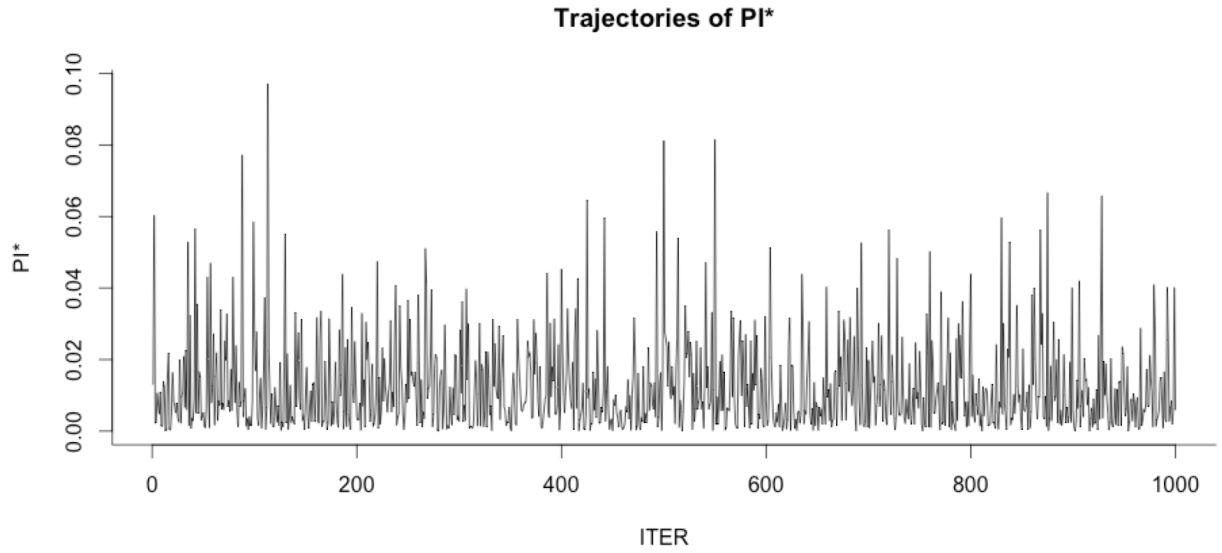
MCMC M1.F

(a) Trajectories of simulated values of pistar(t) against iteration / convergence diagnostics used.

The trajectories are spastic (not smooth), and appear to jump up and down.

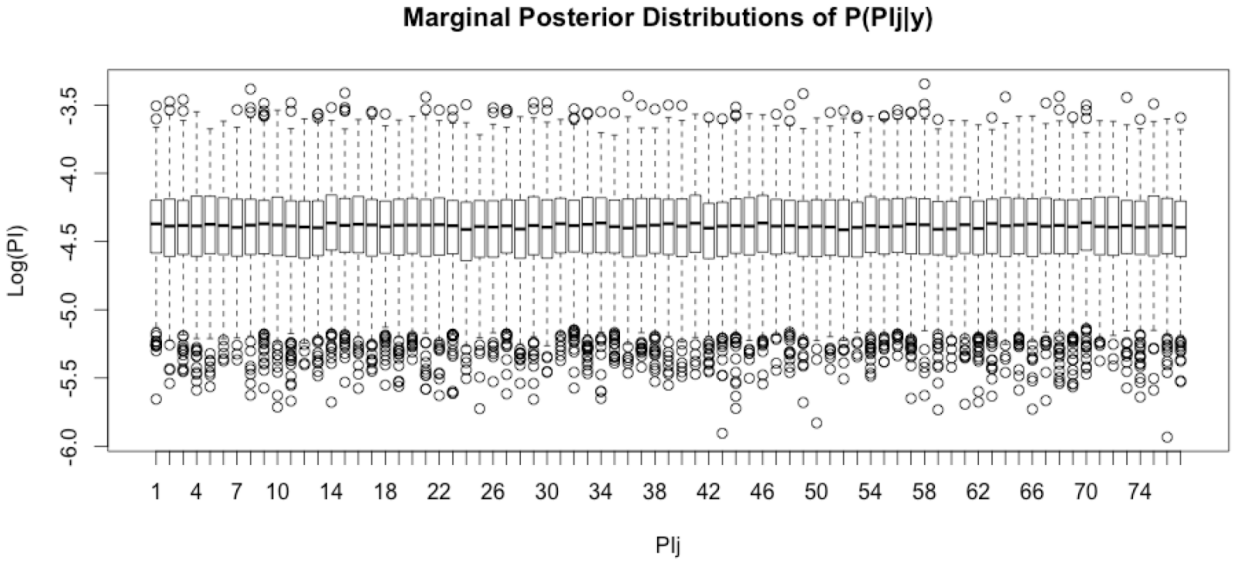


(b) Convergence diagnostics used.

As part of convergence diagnostics, one checks the mixing of the variable over many iterations (here, there is decent mixing with `n=1000`), the stationarity of the values (here, the values hover around one value, that of ~`0.01`), and whether the beginning portion of the chain appears highly dependent and moving in a noticeable direction (here, it seems there is no significant burn-in required).

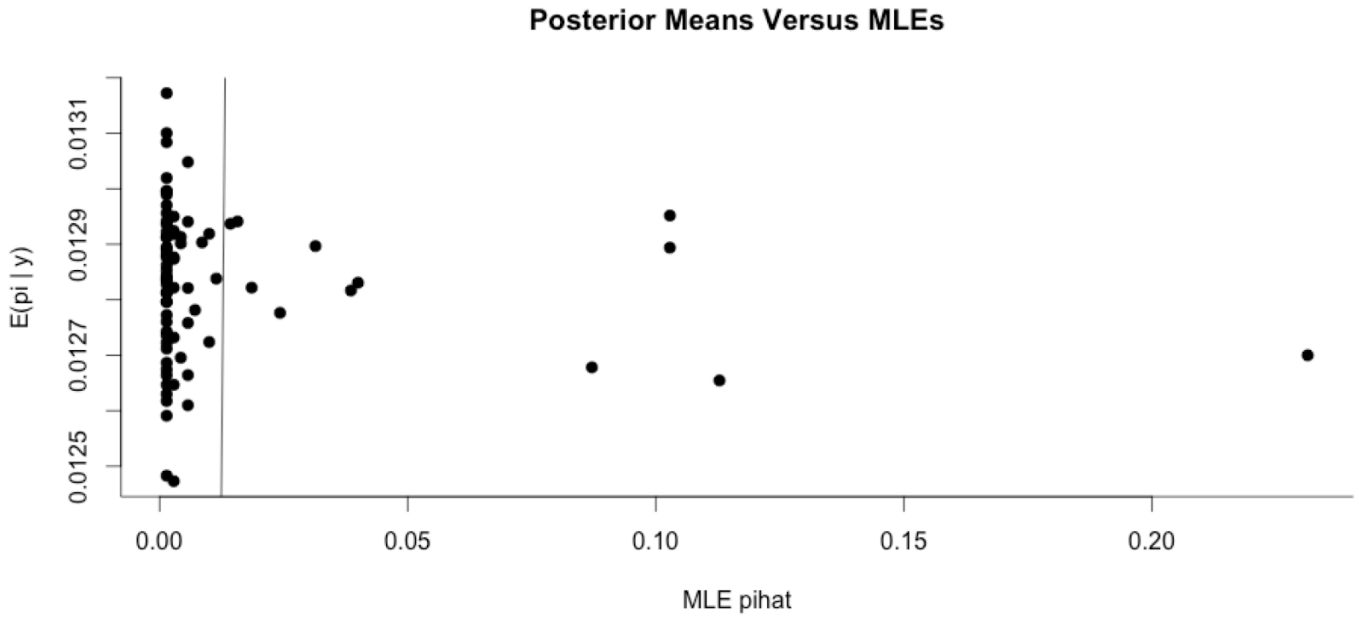
(c) Boxplot of the (marginal) posterior distributions p(πj |y).

The values for each `PIj` vary between ~`exp(-6)` to ~`exp(-3.5)`, centered at ~`exp(-4.4)` or ~`0.012`. There is a narrow interquartile range, and many lower outliers, a potentially concerning detail that requires further investigation.

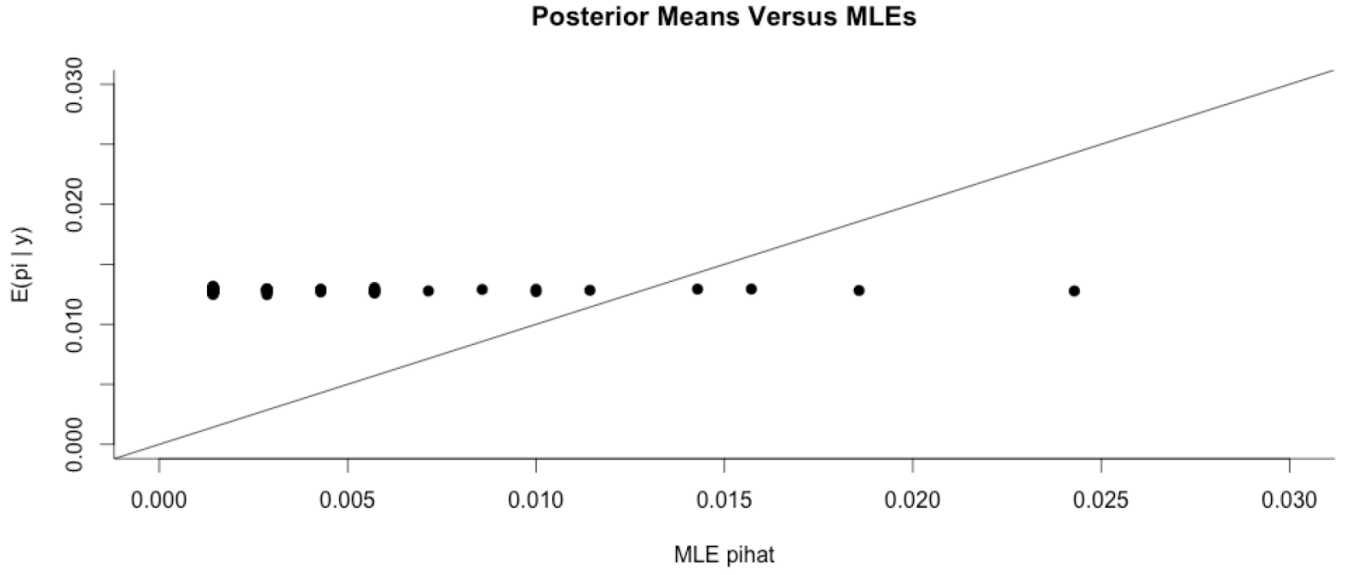


(d) Posterior means, MLEs, and shrinkage.

Here the line drawn is the 45 degree line, y=x.



Here, the graph is zoomed into the left corner, and compared against the 45 degree, y=x line. The y-axis of the upper graph shows that the posterior mean values were very concentrated in the range of about 0.0125 and 0.0130. On the equally dimensioned axes of the lower graph, it is evident that the slope of the posterior means is significantly lower than the 45 degree y=x line. This indicates a large amount of shrinkage occurred, where the model “borrowed” information from the prior mean, due to small bucket sizes in the data likelihood.



CODE

```
require(gtools)  # I found a Dir r.v. generator in this package
## if needed install it with
## install.packages("gtools")

## DATA
setwd("~/Google Drive/2. SPRING 2015/MCMC - Prof Mueller/M1")
x <- scan("sage.dta")                # raw data
y <- table(x)                        # counts
N <- length(y)
names(y) <- 1:N
n <- length(x)

## HYPERPARS:
rho <- 0.1
as <- 0.9
bs <- 0.1
a1 <- .1
a0 <- 10

## initialize
## this function creates a list with
##   z=(z1,.. zN); pistar=pi*, r=(r~[1],..r~[M0])
##   q=(q~[1],..q~[M1])
## You can use it to initialize the state of the MC
init <- function() { # initialize parameters
  ## z
  z <- ifelse(y<10,0,1)
  ## pi*: empirical frquency
  A0 <- which(z==0);     A1 <- which(z==1) # Set of indices where z=0 / z=1.
  M0 <- length(A0);      M1 <- length(A1) # Num of indices where z=0 / z=1.
  Y0 <- sum(y[A0]);      Y1 <- sum(y[A1]) # Num of X's with z=0 / z=1.
  pistar <- sum(Y1)/n
  ## q and r: empirical fequencies
  q <- y[A1]/Y1  # this is the q~ of the text
  r <- y[A0]/Y0  # this is the r~ of the text
  return(th=list(z=z,pistar=pistar,q=q,r=r))
}

## main function for MCMC
gibbs <- function(n.iter, verbose) {
  TH <- NULL # initialize - will save pi*,z here
             ##             for each iteration
  PI <- NULL # similar - will save (pi1,.., piN) here
  th <-  init()
  pistar <- th$pistar        # initialize pistar = pi*
  z <- th$z            # initialize z

  # Vector of probabilities for when z=1, and zeros otherwise.
  q <- rep(0, N)
  q.probs <- th$q
  q[which(z==1)] <- q.probs

  # Vector of probabilities for when z=0, and zeros otherwise.
  r <- rep(0, N)
  r.probs <- th$r
  r[which(z==0)] <- r.probs

  for (i in 1:n.iter) { # loop over iterations
    z <- sample.z(pistar,z, q, r)       # 1. z ~ p(z | pistar, y)
```

```
    q <- sample.q(pistar,z, q)       # 2. q ~ p(q | pistar,z,y)
    r <- sample.r(pistar,z, r)       # 3. r ~ p(r | pistar,z,y)
    pistar <- sample.pistar(z)    # 4. pi
    if (verbose > 0){
      if (i %% 10 == 0) {
        # print short summary, for every 10th iteration
        print(paste("Iteration: ", i))
        print(round(z, 2))
        print(round(q, 2))
        print(round(r, 2))
        print(round(pistar, 2))
      }
    }
    ## save iteration
    TH <- rbind(TH, c(pistar,z))

    pi <- rep(0,N)
    pi[z==1] <- pistar*q[z==1]
    pi[z==0] <- (1-pistar)*r[z==0]
    PI <- rbind(PI, pi)
  }
  return(list(TH=TH, PI=PI))
}

ex <- function() {
  ## RUN these lines to get the plots
  n.iter <- 1000; verbose=0
  gbs <- gibbs(n.iter, verbose)
  ## assume gbs returns a list with elements
  ## TH = (niter x p) matrix with each row being the
  ##      state (pi, z)
  ## PI = (niter x 1) vector with pi
  TH <- gbs$TH
  PI <- gbs$PI
  its <- 1:n.iter

  ## trajectory plot of Z
  plot(its, TH[,1],xlab="ITER",ylab="PI*",bty="l",type="l",
       main="Trajectories of PI*")

  ## boxplot
  boxplot(log(PI), main="Marginal Posterior Distributions of P(PIj|y)",
          xlab="PIj", ylab="Log(PI)")

  ## plotting posterior means vs. mle's
  pibar <- apply(PI,2,mean) # posterior means
  pihat <- as.numeric(y)/n

  plot(pihat, pibar, type="p",
       pch=19, bty="l",xlab="MLE pihat", ylab="E(pi | y)",
       main="Posterior Means Versus MLEs")
  abline(0,1)

  ## same thing, zoom in to left lower corner
  plot(pihat, pibar, type="p", xlim=c(0,0.03), ylim=c(0,0.03),
       pch=19, bty="l",xlab="MLE pihat", ylab="E(pi | y)",
       main="Posterior Means Versus MLEs")
  abline(0,1)

}
```

```
##### aux functions #####

sample.z <- function(pistar, z, q, r) {
  for (i in 1:length(z)) {
    if (z[i]==1) {
      pr <- q[i]^a1*pistar*rho
    } else if (z[i]==0) {
      pr <- r[i]^a0*(1-pistar)*(1-rho)
    }
    z[i] <- ifelse(runif(1)<pr, 1, 0)
  }
  return (z)
}

sample.q <- function(pistar, z, q) {
  M1 <- length(which(z==1))
  if (M1>0) {
    new.q <- rdirichlet(1, rep(a1+1, M1))
    q[which(z==1)] <- new.q
  } else if (M1==0) {
    q <- rep(0, N)
  }
  return (q)
}

sample.r <- function(pistar, z, r) {
  M0 <- length(which(z==0))
  if (M0>0) {
    new.r <- rdirichlet(1, rep(a0+1, M0))
    r[which(z==0)] <- new.r
  } else if (M0==0) {
    r <- rep(0, N)
  }
  return (r)
}

sample.pistar <- function(z) {
  M0 <- length(which(z==0))
  M1 <- length(which(z==1))
  new.pistar <- rbeta(1, M1+as, M0+bs)
  return (new.pistar)
}
```