StatMod2 - Hierarchical Models and Shrinkage - Genes

Maurice Diesendruck

April 15, 2015

1 Model

First, take average of technical replicates to generate one time series for each gene. Consider the following model for gene $j = 1, \cdots, 14$, in group $i = 1, 2, 3$.

$$y_{ij}(t) = \beta_{0j} + \beta_{1j}t + \epsilon_{ij}(t), \text{ where } \epsilon_{ij} \sim \ N(\mathbf{0}, \sigma^2 \boldsymbol{I})$$

This implies the following structure:

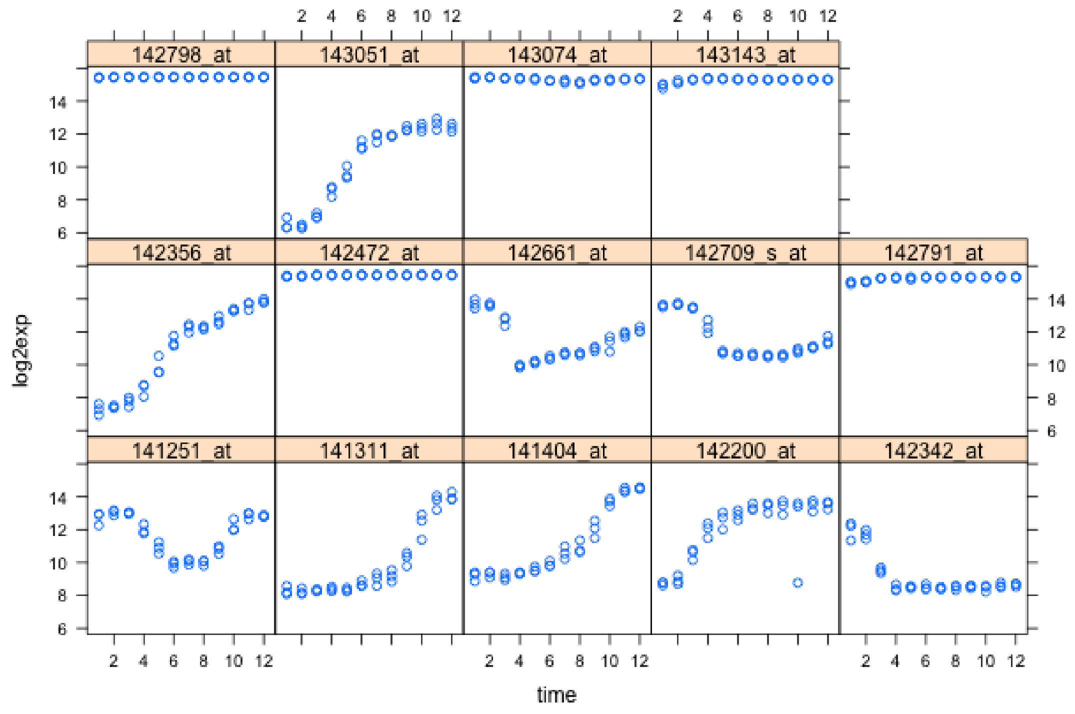
$$\begin{aligned} y_j(t) &\sim \ N(X\boldsymbol{\beta_j}, \sigma^2 I) && \text{where } \beta_j = (\beta_{0j}, \beta_{1j})' \\ \boldsymbol{\beta_j} &\sim \ N(\boldsymbol{\mu_i}, \tau_i^2 I) && \text{Gene-level priors} \\ \boldsymbol{\mu_i} &\sim \ N(\boldsymbol{m_i}, v_i I) && \text{Group-level priors} \\ \tau_i^2 &\sim \ InvGamma(a_i, b_i) && \text{Group-level priors} \end{aligned}$$

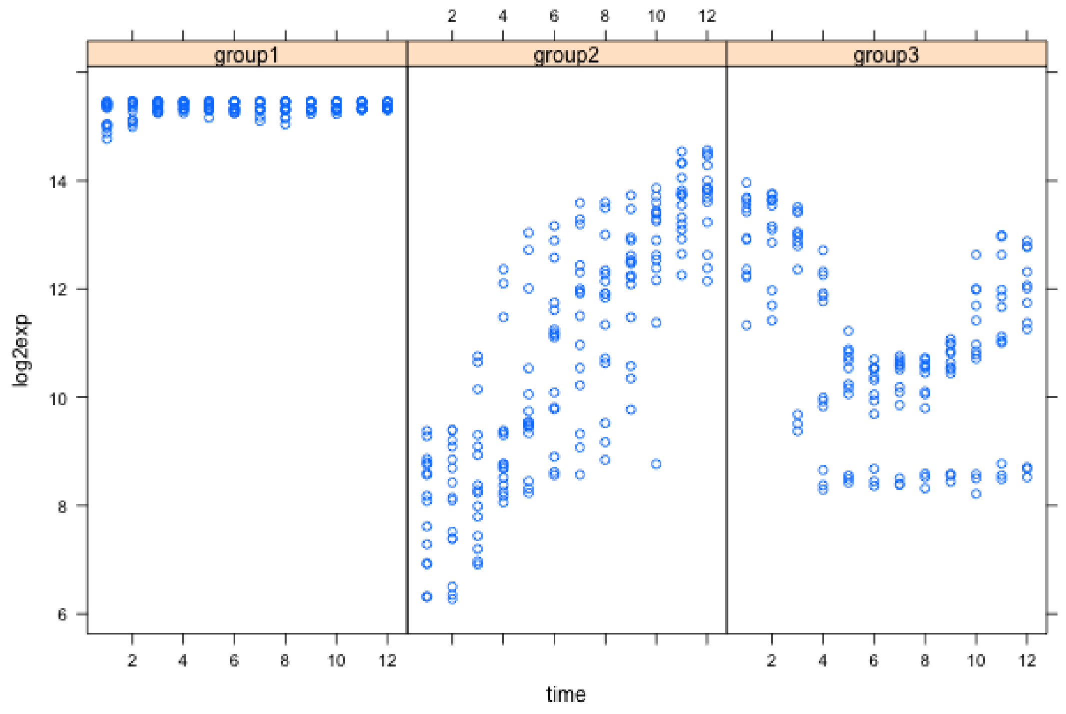
The joint posterior is as follows, and can be written group-wise. Here is it written for group 1.

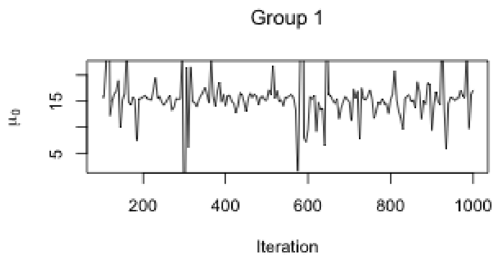
$$\begin{aligned} P(\boldsymbol{\beta_1}, \boldsymbol{\mu_1}, \tau_1^2 | \boldsymbol{y_1}) &\propto \ P(\boldsymbol{y_1}|\boldsymbol{\beta_1})P(\boldsymbol{\beta_1}|\boldsymbol{\mu_1}, \tau_1^2)P(\boldsymbol{\mu_1})P(\tau_1^2) \\ &\propto \ N(\boldsymbol{y_1}|X\boldsymbol{\beta_1}, \sigma^2 I)N(\boldsymbol{\beta_1}|\boldsymbol{\mu_1}, \tau_1^2)N(\boldsymbol{\mu_1}|\boldsymbol{m_1}, v_i I)InvGamma(\tau_1^2|a_1, b_1) \end{aligned}$$

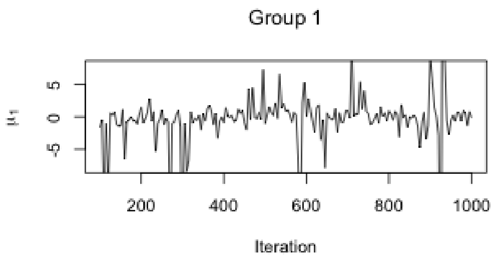
$$\begin{aligned} P(\boldsymbol{\beta_1}|\cdots) &\sim \ N\left(\Big(\frac{X'X}{\sigma^2} + \frac{I}{\tau_1^2}\Big)^{-1}\Big(\frac{X'\boldsymbol{y_1}}{\sigma^2} + \frac{I}{\tau_1^2}\Big), \Big(\frac{X'X}{\sigma^2} + \frac{I}{\tau_1^2}\Big)^{-1}\right) \\ P(\boldsymbol{\mu_1}|\cdots) &\sim \ N\left(\Big(\frac{I}{v_1} + \frac{I}{\tau_1^2}\Big)^{-1}\Big(\frac{I}{v_1}\boldsymbol{m_1} + \frac{I}{\tau_1^2}\boldsymbol{\beta_1}\Big), \Big(\frac{I}{v_1} + \frac{I}{\tau_1^2}\Big)^{-1}\right) \\ P(\frac{1}{\tau_1^2}) &\sim \ Ga(a_1 + 1, \frac{(\boldsymbol{\beta_1} - \boldsymbol{\mu_1})'(\boldsymbol{\beta_1} - \boldsymbol{\mu_1})}{2} + b1) \end{aligned}$$

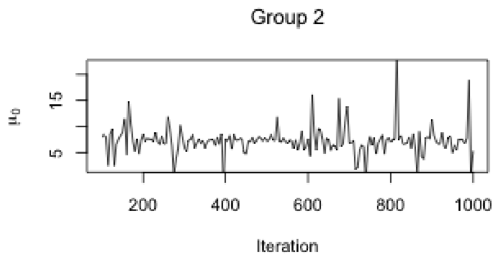
2 Results

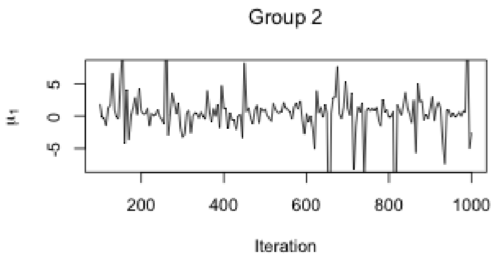

142798_at
143051_at
143074_at
143143_at
142356_at
142472_at
142661_at
142709_s_at
142791_at
141251_at
141311_at
141404_at
142200_at
142342_at
log2exp
time

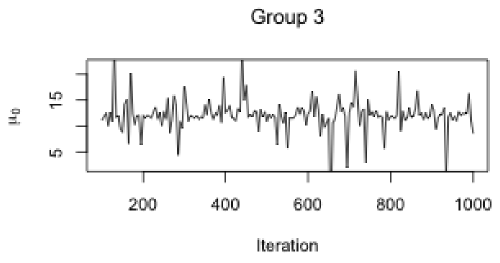

group1
group2
group3
log2exp
time

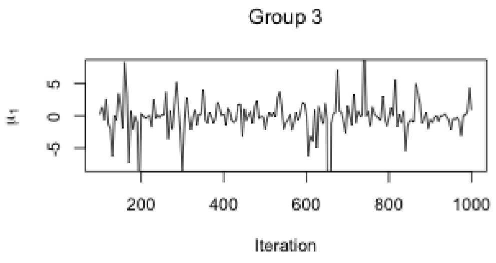

Group 1
μ_0
5
15
200
400
600
800
1000
Iteration

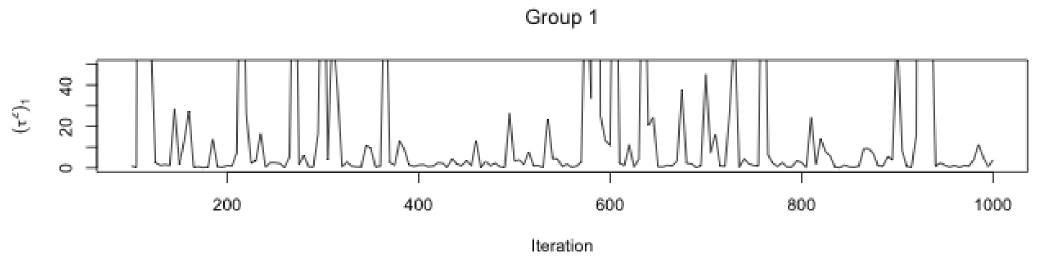

Group 1
μ_1
-5
0
5
200
400
600
800
1000
Iteration

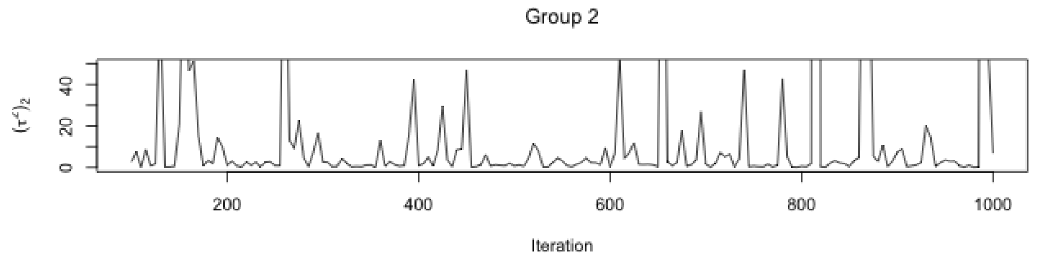

Group 2
μ_0
5
15
200
400
600
800
1000
Iteration

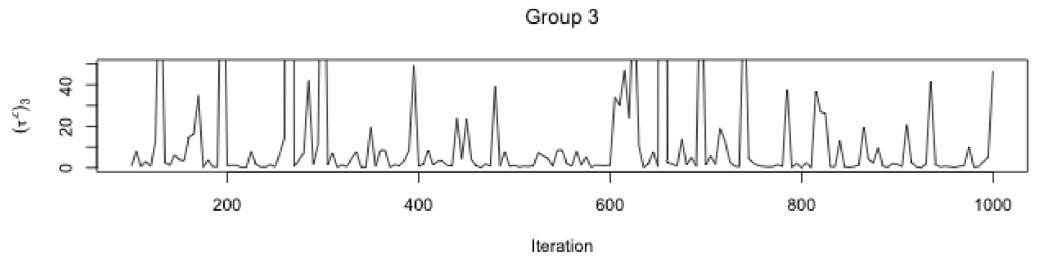

Group 2
μ_1
-5
0
5
200
400
600
800
1000
Iteration

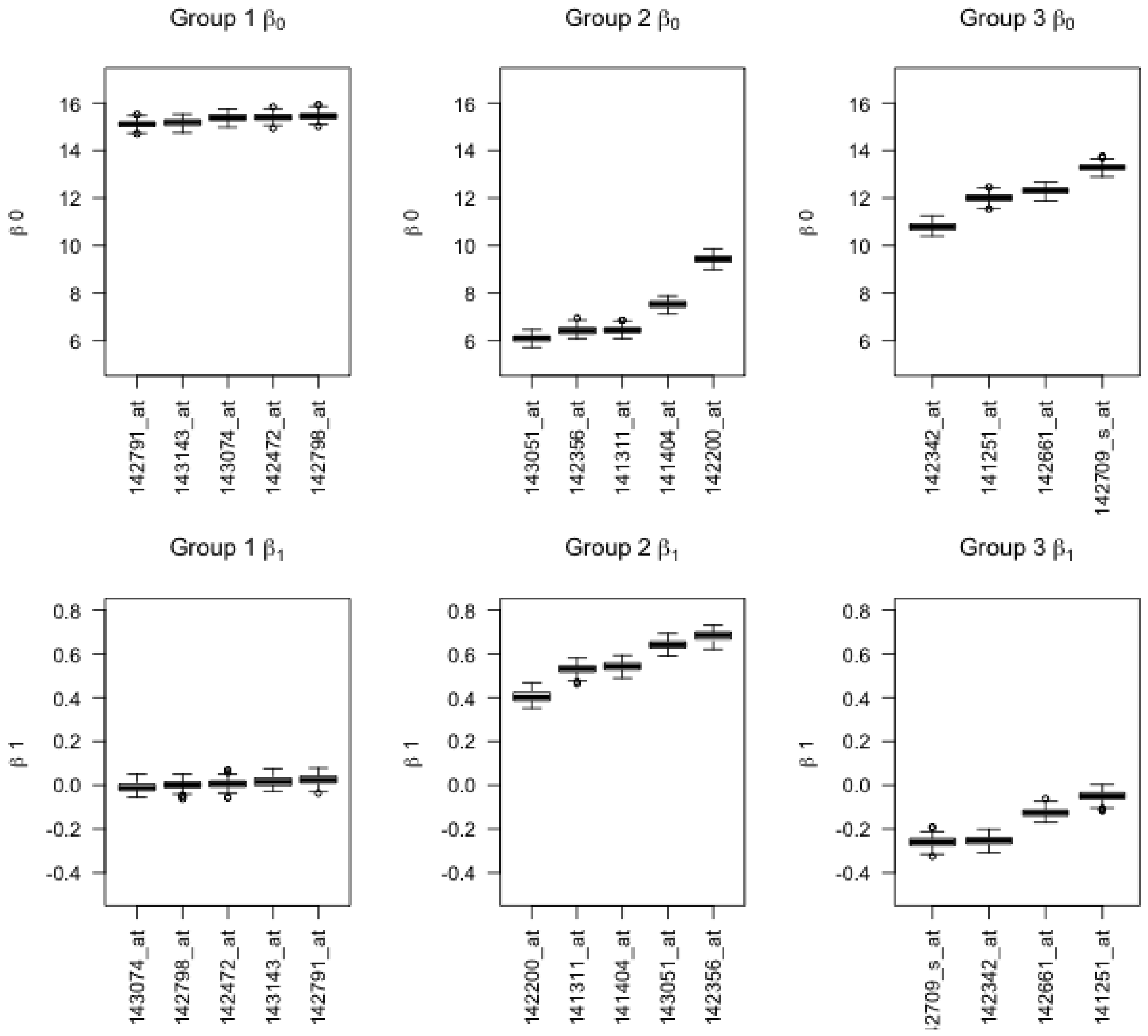

Group 3
μ_0
5
15
200
400
600
800
1000
Iteration


Group 3
μ_1
-5
0
5
200
400
600
800
1000
Iteration


Group 1
$(\tau^2)_1$
0
20
40
200
400
600
800
1000
Iteration


Group 2
$(\tau^2)_2$
0
20
40
200
400
600
800
1000
Iteration


Group 3
$(\tau^2)_3$
0
20
40
200
400
600
800
1000
Iteration



Group 1 β_0
Group 2 β_0
Group 3 β_0
β 0
142791_at
143143_at
143074_at
142472_at
142798_at
143051_at
142356_at
141311_at
141404_at
142200_at
142342_at
141251_at
142661_at
142709_s_at
Group 1 β_1
Group 2 β_1
Group 3 β_1
β 1

3 Full R Code

```r
# StatMod2 - Hierarchical Models and Shrinkage - Genes

library(lattice)
library(MASS)

# Get data.
path <- paste("~/Google Drive/2. SPRING 2015/STAT MOD 2 - Prof Scott/",
              "hierarch-shrinkage/genes", sep="")
setwd(path)
data <- read.csv("droslong.csv")
data <- data[order(data[,1], data[,2]),]
head(data, 100)

# Quickly plot.
xyplot(log2exp~time | gene, data=data)
xyplot(log2exp~time | group, data=data)

##########################################
# PRELIMINARY DATA MANIPULATION

# Average technical replicants, and run frequentist linear model fit for each,
# to get estimates for prior on group mean and group variance.
DATA.AVG.TR <- NULL
lm.coefs <- NULL
gene.names <- as.vector(unique(data$gene))
num.names <- length(gene.names)
for (i in 1:num.names) {
  # Subset of data for each gene.
  name <- gene.names[i]
  d <- data[which(data$gene==name),]
  gene.time.avgs <- aggregate(d$log2exp~d$time, d, mean)
  d <- cbind(d$gene[1], d$group[1], gene.time.avgs)
  names(d) <- c("gene", "group", "time", "log2exp")
  DATA.AVG.TR <- rbind(DATA.AVG.TR, d)

  # Also do a Frequentist exploration and store lm coefs.
  fit.coefs <- lm(d$log2exp~d$time)$coefficients
  lm.coefs <- rbind(lm.coefs, c(toString(name), fit.coefs,
                              toString(d$group[1])))
}
# Got linear model coefficients for each gene. Now average them for the prior
# on group mean and group variance.
just.coefs <- cbind(as.numeric(lm.coefs[,2]), as.numeric(lm.coefs[,3]))
lm.coefs <- as.data.frame(lm.coefs)
names(lm.coefs) <- c("gene", "b0", "b1", "group")
```

```r
group.mean.coefs <- NULL
group.indices <- list()
for (i in 1:3) {
  group.name <- paste("group",i, sep="")
  ind <- which(lm.coefs$group==group.name)
  group.indices[[i]]  <- ind
  group.coefs <- just.coefs[which(lm.coefs$group==group.name),]
  group.mean.coefs <- rbind(group.mean.coefs, colMeans(group.coefs))
}

# Aim for brevity.
D <- DATA.AVG.TR
xyplot(log2exp~time | group, data=D)

##########################################
# BEGINNING OF REAL GIBBS SAMPLER

sigma2 <- 1/rgamma(1, 0.5, 0.5)
a1=0.5; b1=0.5; # Hyper params for Tau's, which are ~ InvGamma.
a2=0.5; b2=0.5;
a3=0.5; b3=0.5;
m1=group.mean.coefs[1,]; v1=1000; # Hyper params for Mu's, which are ~ Normal.
m2=group.mean.coefs[2,]; v2=1000;
m3=group.mean.coefs[3,]; v3=1000;

Gibbs <- function(D, n.iter=1000) {
  t1sq <- 1/rgamma(1, a1, b1) # Initialize Tau's.
  t2sq <- 1/rgamma(1, a2, b2)
  t3sq <- 1/rgamma(1, a3, b3)
  mu1 <- mvrnorm(1, mu=m1, Sigma=diag(x=v1, 2)) # Initialize Mu's.
  mu2 <- mvrnorm(1, mu=m2, Sigma=diag(x=v2, 2))
  mu3 <- mvrnorm(1, mu=m3, Sigma=diag(x=v3, 2))

  # Prepare per-store data.
  gene.names <- unique(D$gene)
  num.genes <- length(gene.names)

  # Create containers for chains and variables.
  BETAS.BY.GENE <- array(NA, c(num.genes, 2, n.iter))
  num.groups <- 3
  length.mu <- length(mu1)
  MU <- array(NA, c(num.groups, length.mu, n.iter+1))
  MU[,,1] <- rbind(mu1, mu2, mu3)
  TAUSQ <- matrix(NA, nrow=n.iter+1, ncol=num.groups)
  TAUSQ[1,] <- c(t1sq, t2sq, t3sq)

  # Do Gibbs many times.
  for (iter in 1:n.iter) {

    # Sample Beta's (be0, be1) for each gene.
    # Use Normal-Normal full conditional posterior.
    for (i in 1:num.genes) {
      name <- toString(gene.names[i])
      gene.data <- data[which(D$gene==name),]
      group.num <- as.numeric(gene.data$group[1])
      y <- gene.data$log2exp
      X <- as.matrix(cbind(1, gene.data[,c("time")]))
      XtX <- t(X)%*%X
      Xty <- t(X)%*%y
      # Sample new Betas using group-specific Mu and Tausq.
      latest.mu <- MU[group.num,,iter]
      latest.tausq <- TAUSQ[iter,group.num]
      diag.tausqs <- diag(x=latest.tausq, 2)
      b <- sample.beta(sigma2, diag.tausqs, XtX, Xty, latest.mu)
      BETAS.BY.GENE[i,,iter] <- b
    }

    # Use Normal-Normal full conditional posterior to update Mu's, given Beta's.
    gr1.beta.mean <- colMeans(BETAS.BY.GENE[group.indices[[1]],,iter])
    gr2.beta.mean <- colMeans(BETAS.BY.GENE[group.indices[[2]],,iter])
    gr3.beta.mean <- colMeans(BETAS.BY.GENE[group.indices[[3]],,iter])
    mu1 <- sample.mu(m1, v1, num.genes, gr1.beta.mean, t1sq)
    mu2 <- sample.mu(m2, v2, num.genes, gr2.beta.mean, t2sq)
    mu3 <- sample.mu(m3, v3, num.genes, gr3.beta.mean, t2sq)
    MU[,,iter+1] <- rbind(mu1, mu2, mu3)

    # Use Normal-InvGamma full conditional posterior to update Tau's, given Beta's.
    t1sq <- sample.tausq(gr1.beta.mean, MU[1,,iter+1], a1, b1)
    t2sq <- sample.tausq(gr2.beta.mean, MU[2,,iter+1], a2, b2)
    t3sq <- sample.tausq(gr3.beta.mean, MU[3,,iter+1], a3, b3)
    TAUSQ[iter+1,] <- c(t1sq, t2sq, t3sq)
  }

  return (list(BETAS.BY.GENE=BETAS.BY.GENE, MU=MU, TAUSQ=TAUSQ))
}

sample.beta <- function(sigma2, diag.tausqs, XtX, Xty, latest.mu) {
  cov <- ginv(XtX/sigma2 + solve(diag.tausqs))
  mean <- cov%*%(Xty/sigma2 + solve(diag.tausqs)%*%latest.mu)
  beta <- mvrnorm(1, mu=mean, Sigma=cov)
  return (beta)
}
sample.mu <- function(m1, v1, num.genes, gr1.beta.mean, t1sq) {
  I.p <- diag(length(m1))
  cov <- ginv(I.p/v1 + I.p/t1sq)
  mean <- cov%*%((I.p/v1)%*%m1 + (I.p/t1sq)%*%gr1.beta.mean)
  mu1 <- mvrnorm(1, mu=mean, Sigma=cov)
  return (mu1)
}
sample.tausq <- function(gr1.beta.mean, mu1, a1, b1) {
  shape <- a1+1
  rate <- t(gr1.beta.mean-mu1)%*%(gr1.beta.mean-mu1)/2 + b1
  tausq.inv <- rgamma(1, shape=shape, rate=rate)
  tausq <- 1/tausq.inv
  return (tausq)
}

substrRight <- function(x, n){
  substr(x, nchar(x)-n+1, nchar(x))
}

##############################################
# RUN EVERYTHING AND PLOT RESULTS

results <- Gibbs(data, n.iter=1000)
P <- results$BETAS.BY.GENE
MU <- results$MU
TAUSQ <- results$TAUSQ
gene.names <- unique(D$gene)

# Show traceplots for Mu's.
par(mfrow=c(3, 2))
n <- dim(MU)[3]
# Burn first 10th and thin by 5.
it <- seq(floor(0.1*n), n, 5)
count <- dim(MU)[1]
for (p in 1:count) {
  plot(its, MU[p,1,it], xlab="Iteration", type="l",
       ylab=bquote(mu[0]), ylim=c(2, 22),
       main=bquote("Group"~.(p)))
  plot(its, MU[p,2,it], xlab="Iteration", type="l",
       ylab=bquote(mu[1]), ylim=c(-8,8),
       main=bquote("Group"~.(p)))
}

# Show traceplots for Tau's.
par(mfrow=c(3,1))
for (p in 1:count) {
  plot(its, TAUSQ[it,p], xlab="Iteration", type="l",
       ylab=bquote((tau^2)[.(p)]), ylim=c(0,50),
       main=bquote("Group"~.(p)))
```

```
}

# Show boxplots for betas of all genes together.
par(mfrow=c(2,3))
num.groups <- 3
n <- dim(P)[3]
# Burn first 10th and thin by 5.
it <- seq(floor(0.1*n), n, 5)
count <- dim(P)[2]
for (p in 1:count) {                               # Do one at a time: b0, b1.
  for (g in 1:num.groups) {                        # Find the b_i groupwise.
    ind <- group.indices[[g]]                      # Get indices of this group.
    data.to.plot <- t(P[ind,p,it])                 # Fetch b_i's for that group.
    m <- melt(data.to.plot)[c(2, 3)]               # Melt into boxplottable form.
    names(m) <- c("gene", "value")                 # Rename for reference.
    gp.gene.names <- gene.names[ind]               # Get gene names in this group.
    for (i in 1:length(gp.gene.names)) {           # Replace numbers with gene names.
      m$gene[m$gene==i] <- gp.gene.names[i]
    }
    bymedian <- with(m, reorder(m$gene, m$value, median))
    if (p==1) {
      boxplot(m$value ~ bymedian, data=m,
              ylab=bquote(beta~.(p-1)),
              main=bquote("Group"~.(g)~beta[.(p-1)]),
              ylim=c(5,17), las=2)
    } else if (p==2) {
      boxplot(m$value ~ bymedian, data=m,
              ylab=bquote(beta~.(p-1)),
              main=bquote("Group"~.(g)~beta[.(p-1)]),
              ylim=c(-0.5,0.8), las=2)
    }
  }
}
```