Lab 8, STA 360/602

Rebecca C. Steorts

Agenda

Consider a three component mixture of normal distribution with a common prior on the mixture component means, the error variance and the variance within mixture component means. The prior on the mixture weights w is a three component Dirichlet distribution. (The data for this problem can be found in "Lab6Mixture.csv").

$$\begin{aligned} p(Y_i|\mu_1,\mu_2,\mu_3,w_1,w_2,w_3,\varepsilon^2) &= \sum_{j=1}^{3} w_i N(\mu_j,\varepsilon^2) \\ \mu_j|\mu_0,\sigma_0^2 &\sim N(\mu_0,\sigma_0^2) \\ \mu_0 &\sim N(0,3) \\ \sigma_0^2 &\sim IG(2,2) \\ (w_1,w_2,w_3) &\sim Dirichlet(\mathbf{1}) \\ \varepsilon^2 &\sim IG(2,2), \end{aligned}$$

for $i = 1, \ldots n$.

Specifically,

- w_1, w_2 and w_3 are the mixture weight of mixture components 1,2 and 3 respectively
- μ_1, μ_2 and μ_3 are the means of the mixture components
- ε^2 is the variance parameter of the error term around the mixture components.

Since we're building a hierarchical model for the means of the individual component, we have a common hyperprior, where, μ_0 is the mean parameter of this hyperprior, σ_0^2 is its variance parameter. Both of these have priors as well, but the parameters of those priors are fixed, where μ_0 has a Normal prior with mean 0 and variance 3, σ_0^2 has an Inverse-Gamma prior with shape and rate parameter of (2,2) respectively. Similarly, ε^2 has an Inverse-Gamma prior with shape and rate parameter of (2,2) respectively. While they have the same parametrisation, they do not share a prior. The mixture weights w_1, w_2, w_3 jointly come from a Dirichlet distribution, with parameter vector $(1, 1, 1)$. $w_1, w_2, w_3, \mu_1, \mu_2, \mu_3, \varepsilon^2, \mu_0$ and σ_0^2 are all random variables that we will estimate when we fit the model.

Task 1

Derive the joint posterior $p(w_1, w_2, w_3, \mu_1, \mu_2, \mu_3, \varepsilon^2, \mu_0, \sigma_0^2|Y_1, ..., Y_N)$ up to a normalizing constant.

This will be done in your lab interactively.

Task 2

Derive the full conditionals for all the parameters up to a normalizing constant.

- $p(w_1, w_2, w_3|\mu_1, \mu_2, \mu_3, \varepsilon^2, Y_1, ..., Y_N) \propto$
- $p(\mu_1|\mu_2, \mu_3, w_1, w_2, w_3, Y_1, ..., Y_N, \varepsilon^2, \mu_0, \sigma_0^2) \propto$
- $p(\mu_2|\mu_1, \mu_3, w_1, w_2, w_3, Y_1, ..., Y_N, \varepsilon^2, \mu_0, \sigma_0^2) \propto$
- $p(\mu_3|\mu_1, \mu_2, w_1, w_2, w_3, Y_1, ..., Y_N, \varepsilon^2, \mu_0, \sigma_0^2) \propto$
- $p(\varepsilon^2|\mu_1, \mu_2, \mu_3, w_1, w_2, w_3, Y_1, ..., Y_N) \propto$
- $p(\mu_0|\mu_1, \mu_2, \mu_3, \sigma_0^2) \propto$
- $p(\sigma_0^2|\mu_0, \mu_1, \mu_2, \mu_3) \propto$

This will be done in your lab interactively.

Task 3

Where necessary, (re)-derive the full conditionals under the data augmentation scheme.

Since neither the joint posterior nor any of the full conditionals involving the likelihood are of a form that's easy to sample, we introduce a data augmentation scheme. A common solution is to introduce an additional set of random variables $\{Z_i\}_{i=1}^N$ that assign each observation to one of the mixture components with the proabilitiy of assignment being the respective mixture weight. If we condition on Z_i we can then write the likelihood of Y_i as

$$p(Y_i|Z_i, \mu_1, \mu_2, \mu_3, \varepsilon^2 = \sum_{j=1}^{N} N(\mu_j, \varepsilon^2)\delta_j(Z_i) = N(\mu_{Z_i}, \varepsilon^2)$$
$$P(Z_i = j) = w_j.$$

This means that conditional on Z_i we no longer have a sum of Normal pdfs in our likelihood, resulting in a significant simplification. Conditional on the $\{Z_i\}$ updates will be straightforward, only depending on the mixture component that any given Y_i is currently assigned to. The drawback is that we also have to update $\{Z_i\}_{i=1}^N$ as well, introducing extra steps into our sampler. Also note that the Dirichlet distribution is a conjugate prior for categorical variables.

The latent variable model can be written as:

$$\begin{aligned}
Y_i \mid Z_i, \mu_1, \mu_2, \mu_3, \epsilon^2 &\sim N(\mu_{Z_i}, \epsilon^2) \\
\mu_j \mid \mu_0, \sigma_0^2 &\sim N(\mu_0, \sigma_0^2) \\
\mu_0 &\sim N(0, 3) \\
\sigma_0^2 &\sim IG(2, 2) \\
(w_1, w_2, w_3) &\sim Dirichlet(3, \mathbf{1}) \\
\epsilon^2 &\sim IG(2, 2) \\
Z_i \mid w_1, w_2, w_3 &\sim Cat(3, \boldsymbol{w}).
\end{aligned}$$

The full conditionals can be written as follows:

$$
\begin{aligned}
p(\mu_0 \mid \ldots) &= N\left(\frac{\sigma_0^2 \sum_{i=1}^3 \mu_i}{1/3 + 3\sigma_0^{-2}}, (1/3 + 3\sigma_0^{-2})^{-1}\right), \\
p(\sigma_0^2 \mid \ldots) &= IG\left(2 + 3/2, 2 + (1/2)\sum_{i=1}^3 (\mu_i - \mu_0)^2\right), \\
p(\epsilon^2 \mid \ldots) &= IG\left(2 + n/2, 2 + (1/2)\sum_{i=1}^n (Y_i - \mu_{Z_i})^2\right), \\
p(\boldsymbol{w} \mid \ldots) &= Dir(3, (1 + N_1, 1 + N_2, 1 + N_3)), \\
p(\mu_j \mid \ldots) &= N\left(\left(\mu_0\sigma_0^{-2} + \epsilon^{-2} \sum_{i:Z_i=j} y_i\right)(\sigma_0^{-2} + N_j\epsilon^{-2})^{-1}, \left(\sigma_0^{-2} + N_j\epsilon^{-2}\right)^{-1}\right), \\
P(Z_i = j) &= \frac{wjN(y_i \mid \mu_j, \epsilon^2)}{\sum_{k=1}^3 w_k N(y_i \mid \mu_k, \epsilon^2)}.
\end{aligned}
$$

Task 4

In task 3 you derived all the full conditionals, and due to data augmentation scheme they are all in a form that is easy to sample. Use these full conditionals to implement Gibbs sampling using the data from "Lab6Mixture.csv".

The code for part of Gibbs' sampler is given below. (You will need to finish the code for your homework this week).

One thing to note about the sampler is that we do *not* save the sampled values for Z_i. One reason is that these variables were added to the model for convenience, so they are not necessarily of interest. In addition, the distribution of these variables is sufficiently summarized by $\boldsymbol{w}$, which we will have a distribution for. There is also a practical reason to discard these samples. We must sampler 500 values at each iteration. If we run only 1000 iterations (and we may reasonably desire to run 10 times this many iterations), we must store 500,000 values. This is a large amount of memory and could drastically affect the speed of the sampler.

```r
library(xtable)
library(MCMCpack)
```

```
## Loading required package: coda

## Loading required package: MASS

## ##
## ## Markov Chain Monte Carlo Package (MCMCpack)

## ## Copyright (C) 2003-2020 Andrew D. Martin, Kevin M. Quinn, and Jong Hee Park

## ##
## ## Support provided by the U.S. National Science Foundation

## ## (Grants SES-0350646 and SES-0350613)
## ##
```

```r
categSampler <- function(probs){
  # this samples from a categorical distribution
  # the probabilites are given by probs
  cdf <- cumsum(probs)
  x <- runif(1)
  samp <- which(x <= cdf)[1]
  return(samp)
}
```

```
# set prior parameters
mu.mu <- 0
mu.v <- 3
s.a <- 2
s.b <- 2
e.a <- 2
e.b <- 2
```

```
nnUpdate <- function(prior.mu, prior.s, like.s, data){
  # computes the posterior parameters for a normal-normal model
  # prior.mu is prior mean
  # prior.s is prior variance
  # like.s is variance of likelihood
  # data is data observations
  x <- sum(data)
  n <- length(data)
  var <- 1/(1/prior.s + n/like.s)
  mu <- ((prior.mu/prior.s) + (x/like.s))*var
  return(c(mu,sqrt(var)))
}
```

```
nIGUpdate <- function(a, b, mu, data){
  # computes parameters for IG (shape, rate)
  # a is prior shape
  # b is prior rate
  # mu is mean of likelihood
  # data is data
  n <- length(data)
  a.post <- a + n/2
  b.post <- b + 1/2*sum((data-mu)^2)
  return(c(a.post,b.post))
}
```

```
counter <- function(k, zs){
  # counts the number of z_i = j for j = 1,..,k
  counts <- vector(mode = "numeric", length = k)
  for (i in 1:k){
    counts[i] <- sum(zs == i)
  }
  return(counts)
}
```

```
catProbs <- function(w, mus, epsilon, y.i){
  # calculates the probabilites that define the categorical distribution
  # for Z in our sampler
  unnormed <- w*dnorm(y.i, mean = mus, sd = sqrt(epsilon))
  probs <- unnormed/sum(unnormed)
  return(probs)
}
```

```
augSampler <- function(ys, zs, mus, m0, s0, w, epsilon, n.iter,
                       burnin = 1){
```

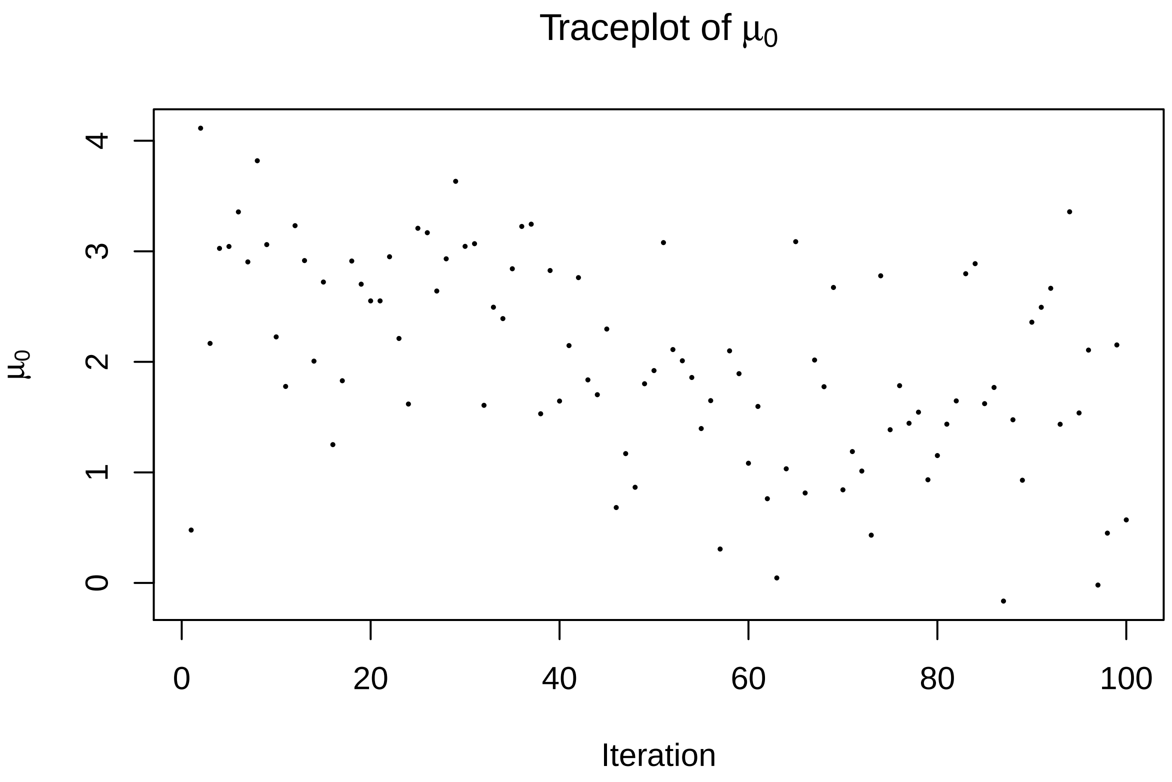
```r
  # Gibbs sampler with augmented data model
  # ys is data
  # zs is initial values of z
  # mus is (mu_1,mu_2,mu_3) initial values
  # m0 is initial value for mu_0
  # s0 is initial value for sigma_0^2
  # w is vector (w_1,w_2,w_3) that sums to one, initial value
  # epsilon is epsilon^2 initial value
  # n.iter is number of iterations
  # burnin is number of sampler to drop for burnin
  n <- length(ys)
  m <- length(mus)+length(w)+3
  k <- length(mus)
  res <- matrix(NA, nrow = n.iter, ncol = m)
  for (i in 1:n.iter){
    # update parameters, then sample
    m0.params <- nnUpdate(mu.mu, mu.v, s0, mus)
    m0 <- rnorm(1, mean = m0.params[1], m0.params[2])
    s0.params <- nIGUpdate(s.a,s.b, m0, mus)
    s0 <- 1/rgamma(1, shape = s0.params[1], rate = s0.params[2])
    ep.params <- nIGUpdate(e.a,e.b, mus[zs], ys)
    epsilon <- 1/rgamma(1, shape = ep.params[1], rate = ep.params[2])
    # calculate number in each category
    Ns <- counter(k, zs)
    w <- rdirichlet(1, 1 + Ns)
    for (j in 1:k){
      data.j <- ys[zs == j]
      mu.j.params <- nnUpdate(m0,s0, epsilon, data.j)
      mus[j] <- rnorm(1, mean = mu.j.params[1], mu.j.params[2])
    }
    # compute probabilites for each dist of each z_i
    z.probs <- sapply(1:n, function(x) {catProbs(w, mus,
                                          epsilon, ys[x])})
    # sampler z_i
    zs <- apply(z.probs, 2, function(x) {categSampler(x)})
    # store value
    res[i,] <- c(m0,s0,epsilon,mus,w)
  }
  return(res[burnin:n.iter,])
}
```

```r
# read in data and set parameters
y.data <- read.csv("Lab8Mixture.csv", header = FALSE)$V1
mus <- rnorm(3)
m0 <- 1
s0 <- 1
w <- rdirichlet(1, c(1,1,1))
epsilon <- 5
zs <- replicate(length(y.data), categSampler(w))
n.iter <- 100
```

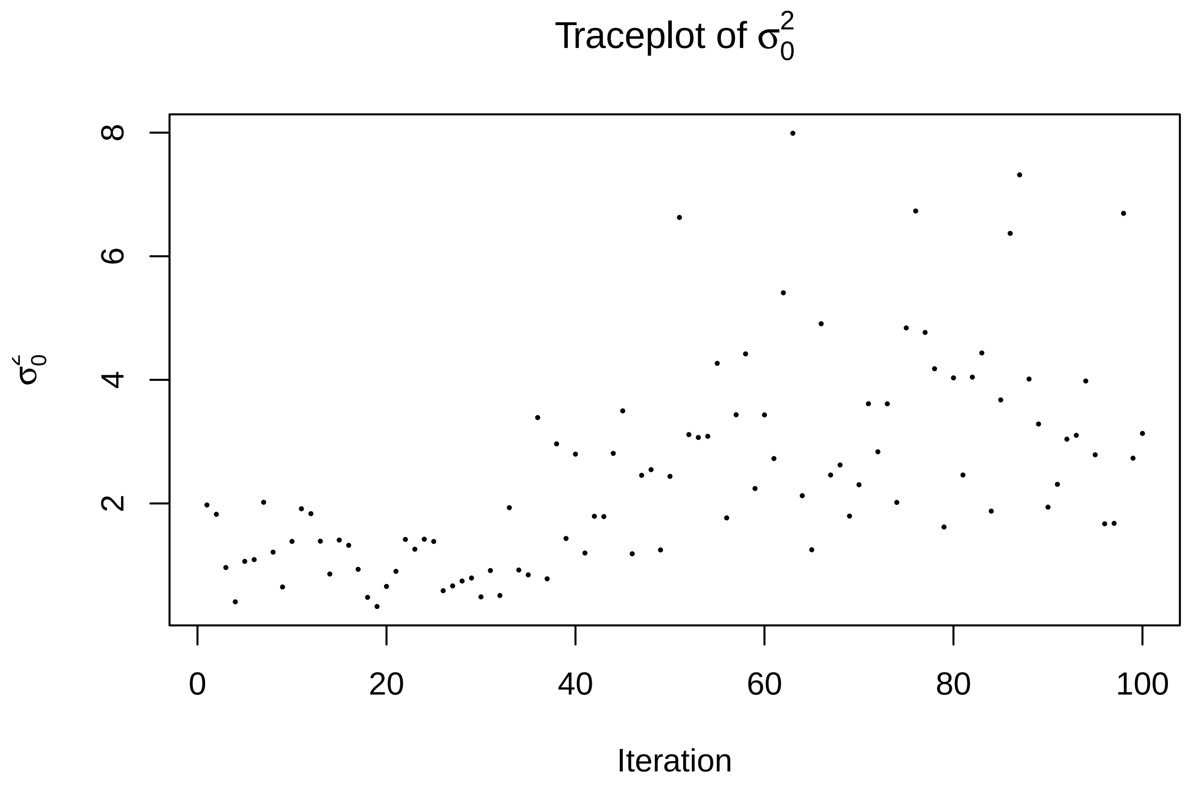
```r
# run sampler
post.samps <- augSampler(y.data, zs, mus, m0, s0, w, epsilon, n.iter)
# name parameters
```

```
param.names <- c("$\\mu_0$", "$\\sigma_0^2$","$\\epsilon^2$","$\\mu_1","$\\mu_2$","$\\mu_3$","$w_1$",
```

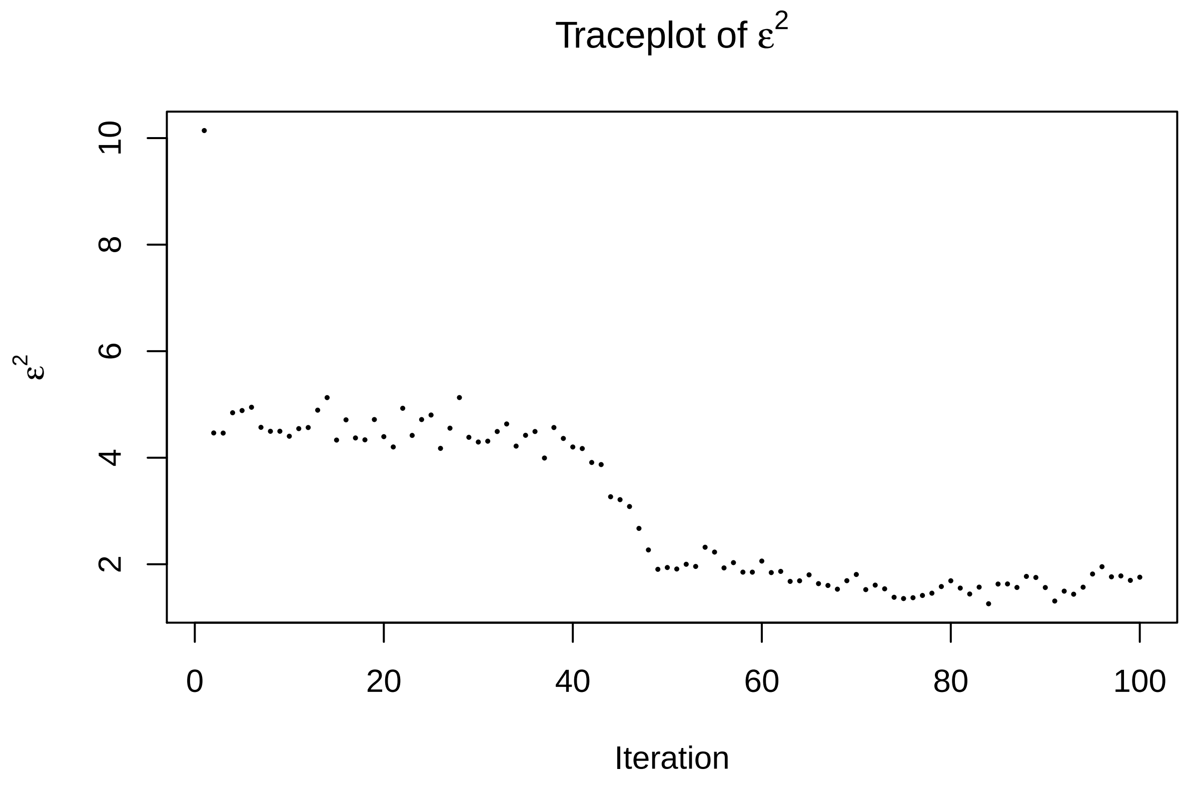
```
plot(1:n.iter, post.samps[,1], pch = 16, cex = .35,
     xlab = "Iteration", ylab = expression(mu[0]),
     main = expression(paste("Traceplot of ", mu[0])))
```



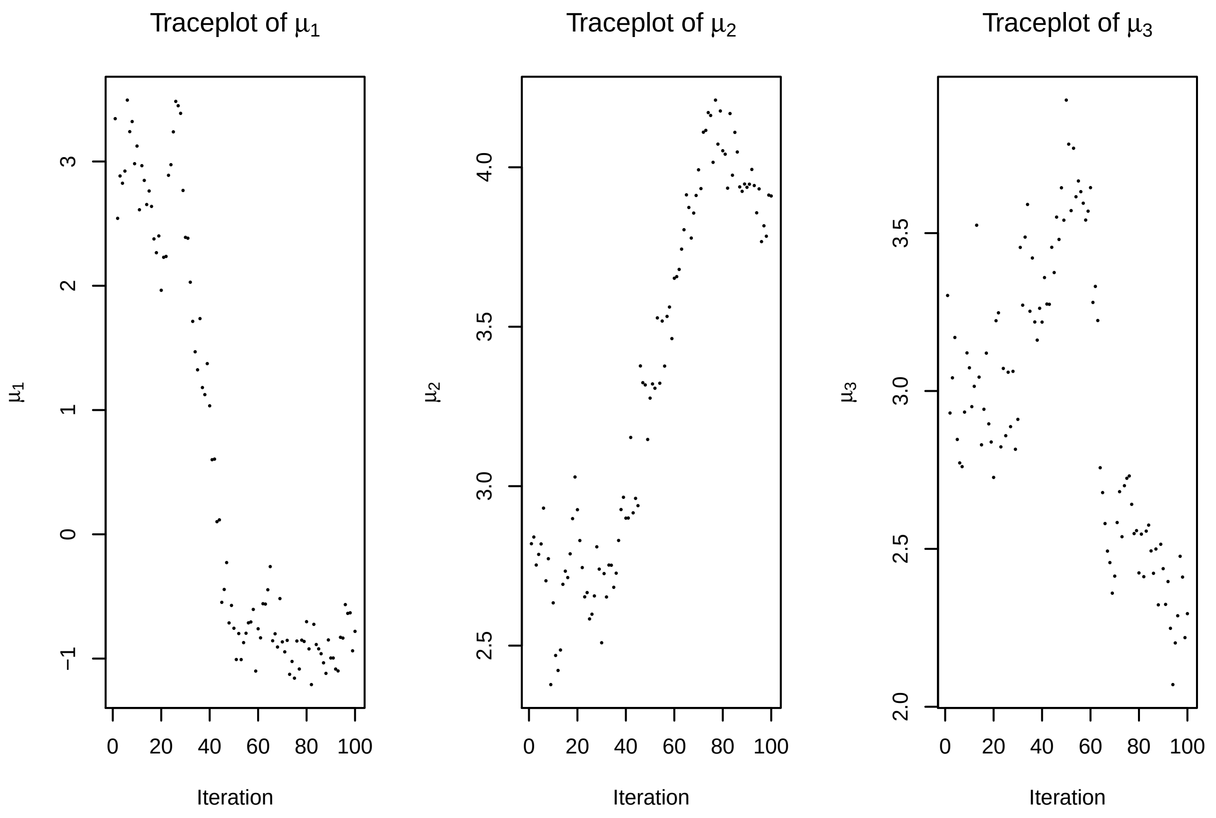
```
plot(1:n.iter, post.samps[,2], pch = 16, cex = .35,
     xlab = "Iteration", ylab = expression(sigma[0]^2),
     main = expression(paste("Traceplot of ", sigma[0]^2)))
```



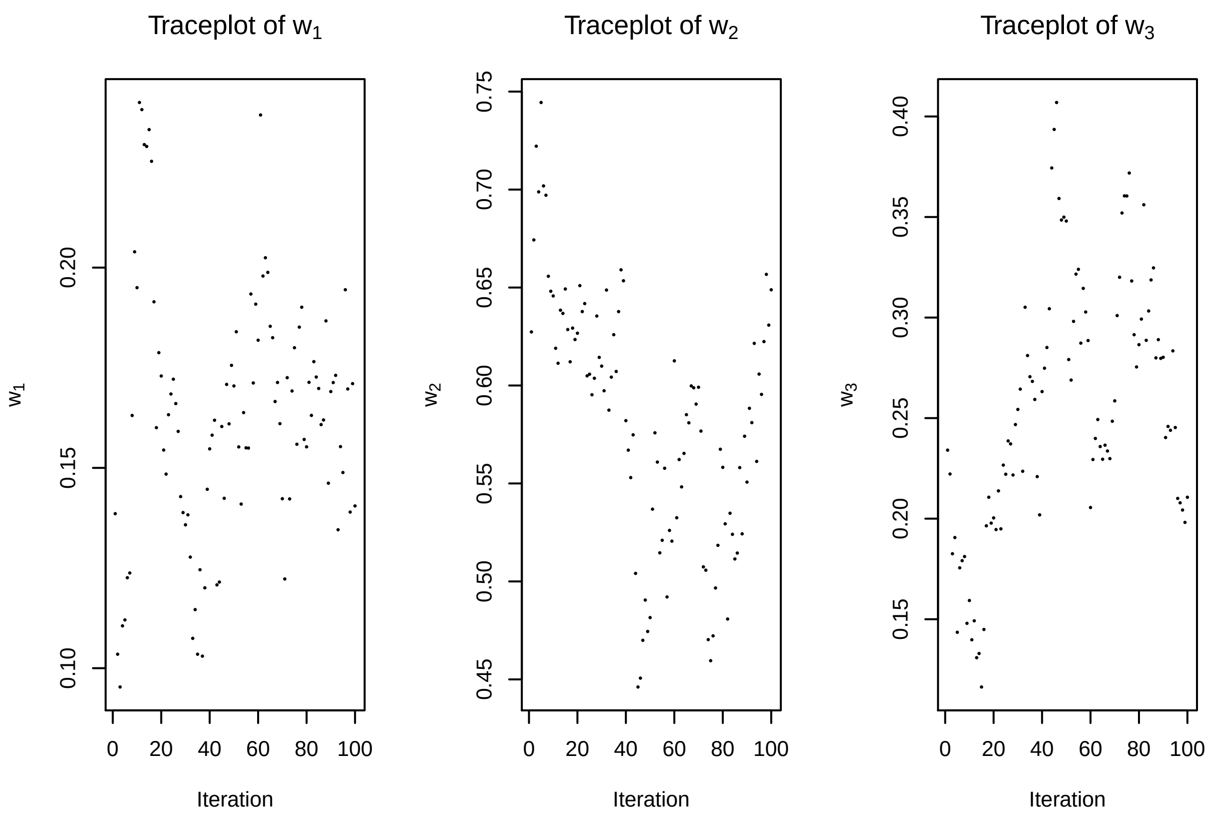
```
plot(1:n.iter, post.samps[,3], pch = 16, cex = .35,
     xlab = "Iteration", ylab = expression(epsilon^2),
     main = expression(paste("Traceplot of ", epsilon^2)))
```



```
par(mfrow=c(1,3))
for (ind in 1:3){
  x.lab <- bquote(mu[.(ind)])
  plot(1:n.iter, post.samps[,3+ind], pch = 16, cex = .35,
       xlab = "Iteration", ylab = x.lab,
       main = bquote(paste("Traceplot of ", .(x.lab))))
}
```



```
par(mfrow=c(1,3))
for (ind in 1:3){
  x.lab <- bquote(w[.(ind)])
  plot(1:n.iter, post.samps[,6+ind], pch = 16, cex = .35,
       xlab = "Iteration", ylab = x.lab,
       main = bquote(paste("Traceplot of ", .(x.lab))))
}
```



```
# calculate summary statistics and display using xtable
ints <- apply(post.samps, 2, function(x) {quantile(x, c(.025,.975))})
means <- apply(post.samps, 2, mean)
sum.data <- cbind(means, t(ints))
row.names(sum.data) <- param.names
xtable(sum.data, sanitize.colnames.function = identity)
```

```
## % latex table generated in R 3.6.2 by xtable 1.8-4 package
## % Wed Aug 26 20:01:46 2020
## \begin{table}[ht]
## \centering
## \begin{tabular}{rrrr}
##   \hline
##  & means & 2.5\% & 97.5\% \\
##   \hline
## \$$\backslash$mu\_0\$ & 2.01 & 0.17 & 3.50 \\
##   \$$\backslash$sigma\_0\verb|^|2\$ & 2.49 & 0.48 & 6.71 \\
##   \$$\backslash$epsilon\verb|^|2\$ & 3.02 & 1.36 & 5.04 \\
##   \$$\backslash$mu\_1 & 0.56 & -1.12 & 3.42 \\
##   \$$\backslash$mu\_2\$ & 3.33 & 2.48 & 4.17 \\
##   \$$\backslash$mu\_3\$ & 2.96 & 2.23 & 3.72 \\
##   \$w\_1\$ & 0.16 & 0.10 & 0.24 \\
##   \$w\_2\$ & 0.58 & 0.46 & 0.70 \\
##   \$w\_3\$ & 0.26 & 0.14 & 0.37 \\
##    \hline
## \end{tabular}
## \end{table}
```

Task 5

Given tasks 1-4 and the provided solutions,

- Show traceplots for all estimated parameters
- Show means and 95% credible intervals for the marginal posterior distributions of all the parameters

Now suppose you re-run the sampler using 3 different starting values, are your results in a,b the same? Justify your reasoning by with visualizations.

Finish task 5 for homework. Be sure to include all other parts of other tasks for completeness.