EM algorithm for GMM

Matthew Stephens and Heejung Shim

Simulate data

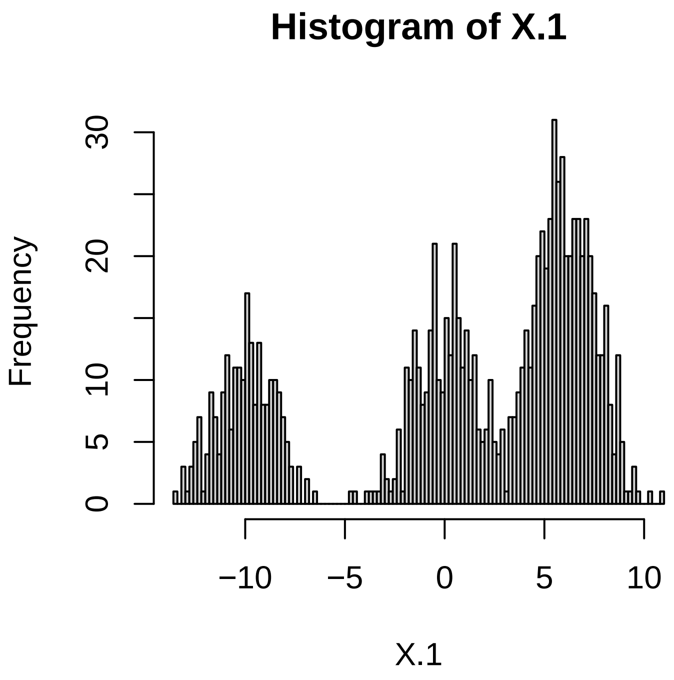
- Simulate 1000 samples from the following mixture model. For $i = 1 \ldots, 1000$,

$$Z_i \sim \text{categorical } (0.2, 0.3, 0.5),$$

$$X_i|Z_i = 1 \sim \text{Normal}(\mu_1 = -10, \sigma_1^2 = 2),$$
$$X_i|Z_i = 2 \sim \text{Normal}(\mu_2 = 0, \sigma_2^2 = 2),$$
$$X_i|Z_i = 3 \sim \text{Normal}(\mu_3 = 6, \sigma_3^2 = 2).$$

Please set a seed using 'set.seed(30027)'. Make a histogram of the simulated samples.

```r
set.seed(30027)
# mixture components
mu.true.1     = c(-10, 0, 6)
sigma2.true.1 = c(2, 2, 2)
pi.true.1 = c(0.2, 0.3, 0.5)
NUM.SAMPLES <- 1000
# simulate Z
Z = sample(1:3, NUM.SAMPLES, replace=TRUE, prob = pi.true.1)
# sample from mixture model
X.1 <- rnorm(NUM.SAMPLES, mean=mu.true.1[Z], sd=sqrt(sigma2.true.1[Z]))
hist(X.1, breaks=100)
```



b) Let's simulate 200 samples from the following mixture model. For $i = 1 \dots, 200$,

$$Z_i \sim \text{categorical } (0.2, 0.3, 0.5),$$

$$X_i|Z_i = 1 \sim \text{Normal}(\mu_1 = -2.5, \sigma_1^2 = 2),$$
$$X_i|Z_i = 2 \sim \text{Normal}(\mu_2 = 0, \sigma_2^2 = 2),$$
$$X_i|Z_i = 3 \sim \text{Normal}(\mu_3 = 2.5, \sigma_3^2 = 2).$$

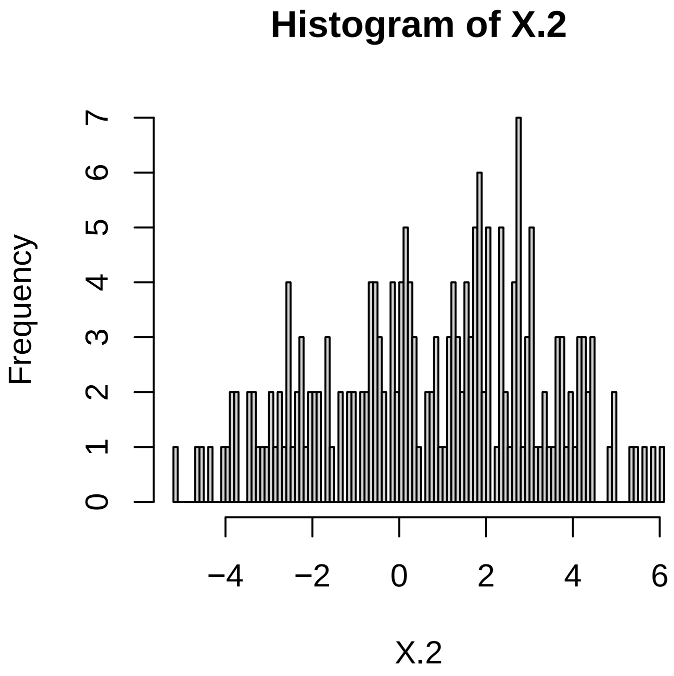
Please set a seed using 'set.seed(30027)'. Make a histogram of the simulated samples.

```r
set.seed(30027)

# mixture components
mu.true.2     = c(-2.5, 0, 2.5)
sigma2.true.2 = c(2, 2, 2)
pi.true.2 = c(0.2, 0.3, 0.5)

NUM.SAMPLES <- 200
# simulate Z
Z = sample(1:3, NUM.SAMPLES, replace=TRUE, prob = pi.true.2)

# sample from mixture model
X.2 <- rnorm(NUM.SAMPLES, mean=mu.true.2[Z], sd=sqrt(sigma2.true.2[Z]))
hist(X.2, breaks=100)
```



Implementation of the EM algorithm for GMM

We will assume that the observed data follow a mixture of three Normal distributions with known variance ($\sigma^2 = 2$). Specifically, for $i = 1 \dots, n$,

$$Z_i \sim \text{categorical } (\pi_1, \pi_2, 1 - \pi_1 - \pi_2),$$

$$X_i|Z_i = 1 \sim \text{Normal}(\mu_1, 2),$$
$$X_i|Z_i = 2 \sim \text{Normal}(\mu_2, 2),$$
$$X_i|Z_i = 3 \sim \text{Normal}(\mu_3, 2).$$

We aim to obtain MLE of parameters $\theta = (\pi_1, \pi_2, \mu_1, \mu_2, \mu_3)$ using the EM algorithm.

We implement the E and M step in the `EM.iter` function below. The `compute.log.lik` function below compute the incomplete log-likelihood, assuming the parameters are known. We will check that the incomplete log-likelihoods increases at each step by plotting them. The `mixture.EM` function is the main function which runs multiple EM steps and checks for convergence by computing the incomplete log-likelihoods at each step.

```r
# w.init : initial value for pi
# mu.init : initial value for mu
# epsilon : If the incomplete log-likelihood has changed by less than epsilon,
# EM will stop.
# max.iter : maximum number of EM-iterations
mixture.EM <- function(X, w.init, mu.init, epsilon=1e-5, max.iter=100) {

  w.curr = w.init
  mu.curr = mu.init

  # store incomplete log-likehoods for each iteration
  log_liks = c()

  # compute incomplete log-likehoods using initial values of parameters.
  log_liks = c(log_liks, compute.log.lik(X, w.curr, mu.curr)$ill)

  # set the change in incomplete log-likelihood with 1
  delta.ll = 1

  # number of iteration
  n.iter = 1

  # If the log-likelihood has changed by less than epsilon, EM will stop.
  while((delta.ll > epsilon) & (n.iter <= max.iter)){

    # run EM step
    EM.out = EM.iter(X, w.curr, mu.curr)

    # replace the current value with the new parameter estimate
    w.curr = EM.out$w.new
    mu.curr = EM.out$mu.new

    # incomplete log-likehoods with new parameter estimate
    log_liks = c(log_liks, compute.log.lik(X, w.curr, mu.curr)$ill)

    # compute the change in incomplete log-likelihood
    delta.ll = log_liks[length(log_liks)]  - log_liks[length(log_liks)-1]

    # increase the number of iteration
    n.iter = n.iter + 1
  }
  return(list(w.curr=w.curr, mu.curr=mu.curr, log_liks=log_liks))
}
```

```r
EM.iter <- function(X, w.curr, mu.curr) {

  # E-step: compute E_{Z|X,\theta_0}[I(Z_i = k)]

  # for each sample $X_i$, compute $P(X_i, Z_i=k)$
  prob.x.z = compute.prob.x.z(X, w.curr, mu.curr)$prob.x.z

  # compute P(Z_i=k | X_i)
  P_ik = prob.x.z / rowSums(prob.x.z)

  # M-step
  w.new = colSums(P_ik)/sum(P_ik)  # sum(P_ik) is equivalent to sample size
  mu.new = colSums(P_ik*X)/colSums(P_ik)

  return(list(w.new=w.new, mu.new=mu.new))
}
```

Now we write a function to compute the incomplete log-likelihood, assuming the parameters are known.

$$\ell(\theta) = \sum_{i=1}^{n} \log \left(\sum_{k=1}^{3} \pi_k f(x_i; \mu_k, \sigma_k^2 = 2) \right)$$

```r
# Compute incomplete log-likehoods
compute.log.lik <- function(X, w.curr, mu.curr) {

  # for each sample $X_i$, compute $P(X_i, Z_i=k)$
  prob.x.z = compute.prob.x.z(X, w.curr, mu.curr)$prob.x.z

  # incomplete log-likehoods
  ill = sum(log(rowSums(prob.x.z)))

  return(list(ill=ill))
}

# for each sample $X_i$, compute $P(X_i, Z_i=k)$
compute.prob.x.z <- function(X, w.curr, mu.curr) {

  # for each sample $X_i$, compute $P(X_i, Z_i=k)$. Store these values in the columns of L:
  L = matrix(NA, nrow=length(X), ncol= length(w.curr))
  for(k in seq_len(ncol(L))) {
    L[, k] = dnorm(X, mean=mu.curr[k], sd=sqrt(2))*w.curr[k]
  }

  return(list(prob.x.z=L))
}
```

Apply the EM algorithm to the first simulated data set

Run EM algorithm with different initial values and check that the incomplete log-likelihoods increases at each step by plotting them.

```r
EM1 <- mixture.EM(X.1, w.init=c(0.2,0.3, 0.5), mu.init=c(-4, 1, 3), epsilon=1e-5, max.iter=100)
ee = EM1
```

```
print(paste("Estimate pi = (", round(ee$w.curr[1],2), ",",
            round(ee$w.curr[2],2), ",",
            round(ee$w.curr[3],2), ")", sep=""))
```

```
## [1] "Estimate pi = (0.22,0.29,0.49)"
```

```
print(paste("True pi = (", pi.true.1[1], ",", pi.true.1[2], ",", pi.true.1[3],")", sep=""))
```

```
## [1] "True pi = (0.2,0.3,0.5)"
```

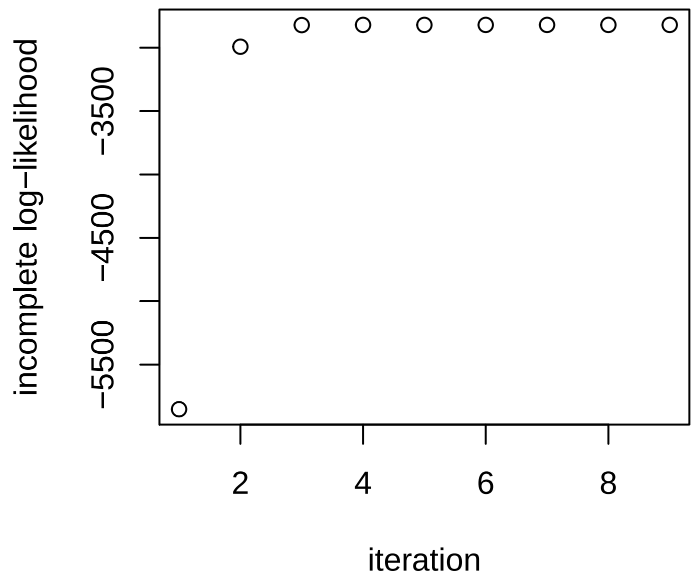
```
print(paste("Estimate mu = (", round(ee$mu.curr[1],2), ",",
            round(ee$mu.curr[2],2), ",",
            round(ee$mu.curr[3],2), ")", sep=""))
```

```
## [1] "Estimate mu = (-10,-0.03,6.06)"
```

```
print(paste("True mu = (", mu.true.1[1], ",", mu.true.1[2], ",", mu.true.1[3],")", sep=""))
```

```
## [1] "True mu = (-10,0,6)"
```

```
plot(ee$log_liks, ylab='incomplete log-likelihood', xlab='iteration')
```



```
EM2 <- mixture.EM(X.1, w.init=c(0.9,0.05, 0.05), mu.init=c(-4, 1, 3), epsilon=1e-5, max.iter=100)
ee = EM2
print(paste("Estimate pi = (", round(ee$w.curr[1],2), ",",
            round(ee$w.curr[2],2), ",",
            round(ee$w.curr[3],2), ")", sep=""))
```

```
## [1] "Estimate pi = (0.22,0.29,0.49)"
```

```
print(paste("True pi = (", pi.true.1[1], ",", pi.true.1[2], ",", pi.true.1[3],")", sep=""))
```

```
## [1] "True pi = (0.2,0.3,0.5)"
```

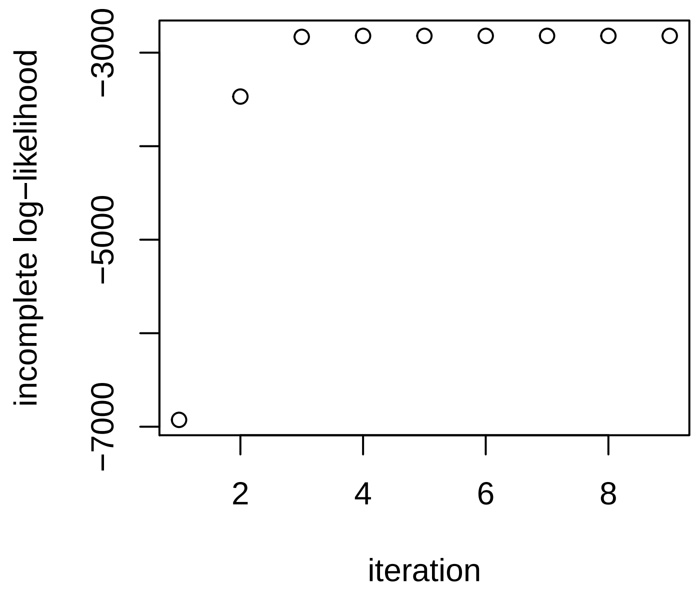
```
print(paste("Estimate mu = (", round(ee$mu.curr[1],2), ",",
            round(ee$mu.curr[2],2), ",",
            round(ee$mu.curr[3],2), ")", sep=""))
```

```
## [1] "Estimate mu = (-10,-0.03,6.06)"
```

```r
print(paste("True mu = (", mu.true.1[1], ",", mu.true.1[2], ",", mu.true.1[3],")", sep=""))
```

```
## [1] "True mu = (-10,0,6)"
```

```r
plot(ee$log_liks, ylab='incomplete log-likelihood', xlab='iteration')
```



```r
EM3 <- mixture.EM(X.1, w.init=c(0.9,0.05, 0.05), mu.init=c(10, 4, 1), epsilon=1e-5, max.iter=100)
ee = EM3
print(paste("Estimate pi = (", round(ee$w.curr[1],2), ",",
            round(ee$w.curr[2],2), ",",
            round(ee$w.curr[3],2), ")", sep=""))
```

```
## [1] "Estimate pi = (0.49,0.29,0.22)"
```

```r
print(paste("True pi = (", pi.true.1[1], ",", pi.true.1[2], ",", pi.true.1[3],")", sep=""))
```

```
## [1] "True pi = (0.2,0.3,0.5)"
```

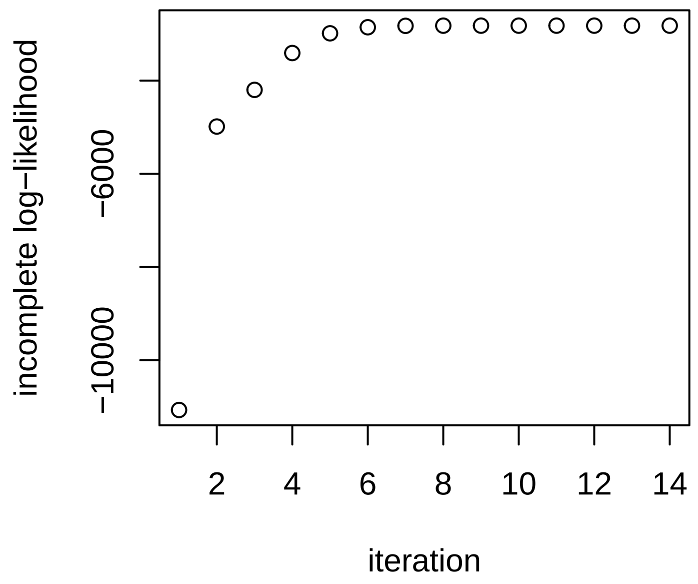
```r
print(paste("Estimate mu = (", round(ee$mu.curr[1],2), ",",
            round(ee$mu.curr[2],2), ",",
            round(ee$mu.curr[3],2), ")", sep=""))
```

```
## [1] "Estimate mu = (6.06,-0.03,-10)"
```

```r
print(paste("True mu = (", mu.true.1[1], ",", mu.true.1[2], ",", mu.true.1[3],")", sep=""))
```

```
## [1] "True mu = (-10,0,6)"
```

```r
plot(ee$log_liks, ylab='incomplete log-likelihood', xlab='iteration')
```



Check which estimators have the highest incomplete log-likelihood.

```r
EM1$log_liks[length(EM1$log_liks)]
```

```
## [1] -2820.215
```

```r
EM2$log_liks[length(EM2$log_liks)]
```

```
## [1] -2820.215
```

```r
EM3$log_liks[length(EM3$log_liks)]
```

```
## [1] -2820.215
```

Estimators from three EM runs have (equally) highest incomplete log-likelihoods. You can see that the estimators from the EM runs are the same if labels for clusters are switched. So it doesn't matter which estimators we choose. I will choose the estimators from the first EM run - $\hat{\pi}_1 = 0.2211658, \hat{\pi}_2 = 0.2854424, \hat{\mu}_1 = -9.99961961, \hat{\mu}_2 = -0.03233427, \hat{\mu}_3 = 6.05589298$ which are similar to true values of the parameters.

Apply the EM algorithm to the second simulated data set

Run EM algorithm with different initial values and check that the incomplete log-likelihoods increases at each step by plotting them.

```r
EM1 <- mixture.EM(X.2, w.init=c(0.2,0.3, 0.5), mu.init=c(-4, 1, 3), epsilon=1e-5, max.iter=100)
ee = EM1
print(paste("Estimate pi = (", round(ee$w.curr[1],2), ",",
            round(ee$w.curr[2],2), ",",
            round(ee$w.curr[3],2), ")", sep=""))
```

```
## [1] "Estimate pi = (0.26,0.34,0.4)"
```

```r
print(paste("True pi = (", pi.true.2[1], ",", pi.true.2[2], ",", pi.true.2[3],")", sep=""))
```

```
## [1] "True pi = (0.2,0.3,0.5)"
```

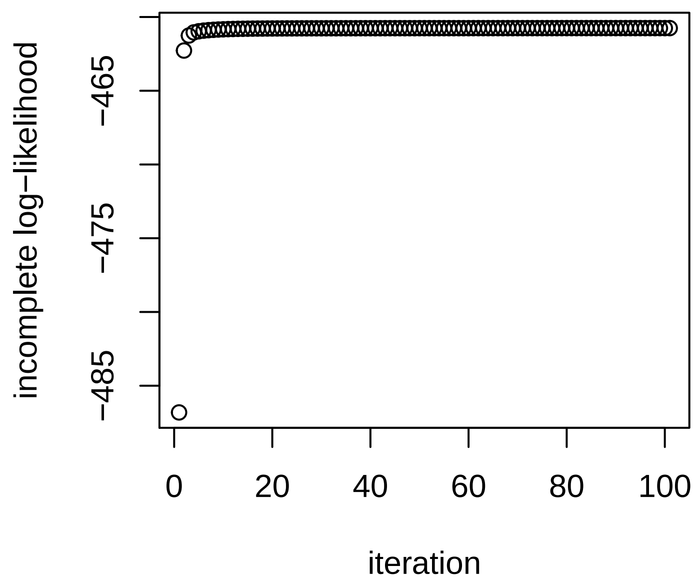
```r
print(paste("Estimate mu = (", round(ee$mu.curr[1],2), ",",
            round(ee$mu.curr[2],2), ",",
            round(ee$mu.curr[3],2), ")", sep=""))
```

```
## [1] "Estimate mu = (-2.33,0.66,2.91)"
```

```
print(paste("True mu = (", mu.true.2[1], ",", mu.true.2[2], ",", mu.true.2[3],")", sep=""))
```

```
## [1] "True mu = (-2.5,0,2.5)"
```

```
plot(ee$log_liks, ylab='incomplete log-likelihood', xlab='iteration')
```



```
EM2 <- mixture.EM(X.2, w.init=c(0.9,0.05, 0.05), mu.init=c(-4, 1, 3), epsilon=1e-5, max.iter=100)
ee = EM2
print(paste("Estimate pi = (", round(ee$w.curr[1],2), ",",
            round(ee$w.curr[2],2), ",",
            round(ee$w.curr[3],2), ")", sep=""))
```

```
## [1] "Estimate pi = (0.27,0.35,0.38)"
```

```
print(paste("True pi = (", pi.true.2[1], ",", pi.true.2[2], ",", pi.true.2[3],")", sep=""))
```

```
## [1] "True pi = (0.2,0.3,0.5)"
```

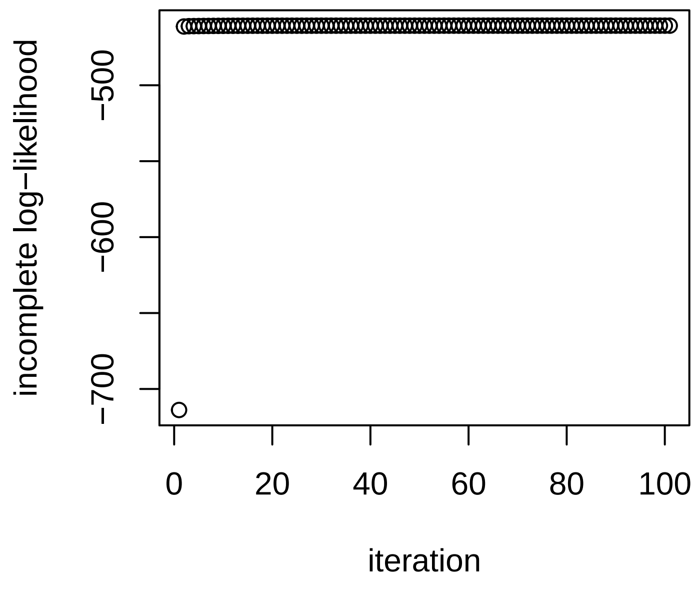
```
print(paste("Estimate mu = (", round(ee$mu.curr[1],2), ",",
            round(ee$mu.curr[2],2), ",",
            round(ee$mu.curr[3],2), ")", sep=""))
```

```
## [1] "Estimate mu = (-2.31,0.74,2.94)"
```

```
print(paste("True mu = (", mu.true.2[1], ",", mu.true.2[2], ",", mu.true.2[3],")", sep=""))
```

```
## [1] "True mu = (-2.5,0,2.5)"
```

```
plot(ee$log_liks, ylab='incomplete log-likelihood', xlab='iteration')
```



```r
EM3 <- mixture.EM(X.2, w.init=c(0.9,0.05, 0.05), mu.init=c(10, 4, 1), epsilon=1e-5, max.iter=100)
ee = EM3
print(paste("Estimate pi = (", round(ee$w.curr[1],2), ",",
            round(ee$w.curr[2],2), ",",
            round(ee$w.curr[3],2), ")", sep=""))
```

```
## [1] "Estimate pi = (0.2,0.49,0.31)"
```

```r
print(paste("True pi = (", pi.true.2[1], ",", pi.true.2[2], ",", pi.true.2[3],")", sep=""))
```

```
## [1] "True pi = (0.2,0.3,0.5)"
```

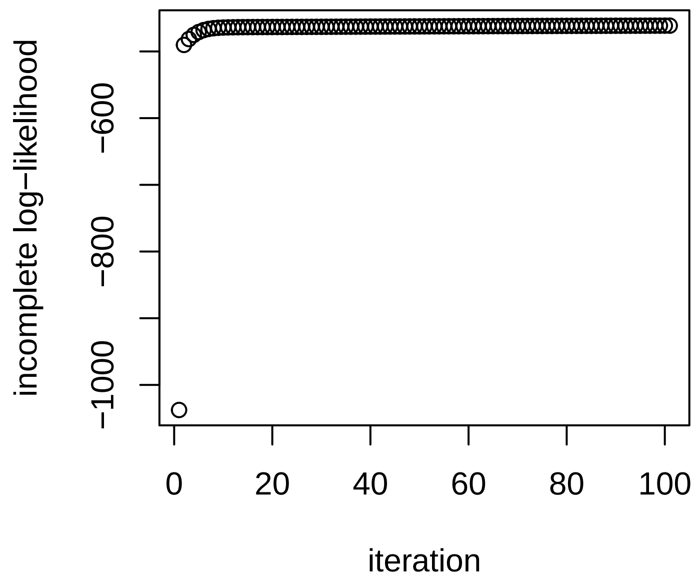
```r
print(paste("Estimate mu = (", round(ee$mu.curr[1],2), ",",
            round(ee$mu.curr[2],2), ",",
            round(ee$mu.curr[3],2), ")", sep=""))
```

```
## [1] "Estimate mu = (3.46,1.5,-2.11)"
```

```r
print(paste("True mu = (", mu.true.2[1], ",", mu.true.2[2], ",", mu.true.2[3],")", sep=""))
```

```
## [1] "True mu = (-2.5,0,2.5)"
```

```r
plot(ee$log_liks, ylab='incomplete log-likelihood', xlab='iteration')
```



Check which estimators have the highest incomplete log-likelihood.

```
EM1$log_liks[length(EM1$log_liks)]
```

```
## [1] -460.7521
```

```
EM2$log_liks[length(EM2$log_liks)]
```

```
## [1] -460.7533
```

```
EM3$log_liks[length(EM3$log_liks)]
```

```
## [1] -461.3282
```

Estimators from the first EM run has the highest incomplete log-likelihood. We can see that the estimators from the first EM run are more similar to the true parameters. I will choose the estimators from the first EM run - $\hat{\pi}_1 = 0.2640038, \hat{\pi}_2 = 0.3362927, \hat{\mu}_1 = -2.3301134, \hat{\mu}_2 = 0.6604183, \hat{\mu}_3 = 2.9051919$.