Project 2 EM Algorithm

Problem 4: Responsibilities and prior for biased coins

In this problem, we are basically performing EM algorithm *manually*.

The responsibility of component i (coin i) can be calculated by:

$$\gamma_{ij} = \frac{P(C_i)P(D_j|C_i)}{\sum_i P(C_i)P(D_j|C_i)}$$

Given mixture weights λ and model parameter (the probability of obtaining heads for coins) θ, the expected hidden log likelihood is:

$$E(\ell_{\text{hid}}(\theta, \lambda)) = \sum_{i,k} \gamma_{ik} \log \lambda_k + \sum_{i,k} \gamma_{ik} \log P(X^{(i)}|\theta_k)$$

Since we're only updating λ (according to the description), we only need to maximize the "left sum". And the optimal model mixture weights $\hat{\lambda}_k$ is given by

$$\hat{\lambda}_k = \frac{1}{N} \sum_{i=1}^{N} \gamma_{ik}$$

Re-write into the notation of this problem, we should update the mixture weights by:

$$\hat{P}(C_A) = \frac{\gamma_{A1} + \gamma_{A2}}{2}$$

and

$$\hat{P}(C_B) = \frac{\gamma_{B1} + \gamma_{B2}}{2}$$

The responsibilites are calcualted below:

$$\begin{aligned}
\gamma_{A1} &= \frac{P(C_A)P(D_1|C_A)}{P(C_A)P(D_1|C_A) + P(C_B)P(D_1|C_B)} \\
&= \frac{0.6 \cdot (0.7^4 \cdot 0.3^6)}{0.6 \cdot (0.7^4 \cdot 0.3^6) + 0.4 \cdot (0.4^4 \cdot 0.6^6)} \\
&= 0.1802056
\end{aligned}$$

```
P_CA_D1 <- 0.6 * 0.7^4 * 0.3^6
P_CB_D1 <- 0.4 * 0.4^4 * 0.6^6
gamma_A1 <- P_CA_D1 / (P_CA_D1 + P_CB_D1)
gamma_A1
```

```
## [1] 0.1802056
```

$$\gamma_{B1} = 1 - \gamma_{A1} = 0.8197944$$

```r
gamma_B1 <- P_CB_D1 / (P_CA_D1 + P_CB_D1)
gamma_B1
```

```
## [1] 0.8197944
```

$$\begin{aligned}\gamma_{A2} &= \frac{P(C_A)P(D_2|C_A)}{P(C_A)P(D_2|C_A) + P(C_B)P(D_2|C_B)} \\ &= \frac{0.6 \cdot (0.7^8 \cdot 0.3^2)}{0.6 \cdot (0.7^8 \cdot 0.3^2) + 0.4 \cdot (0.4^8 \cdot 0.6^2)} \\ &= 0.9705765\end{aligned}$$

```r
P_CA_D2 <- 0.6 * 0.7^8 * 0.3^2
P_CB_D2 <- 0.4 * 0.4^8 * 0.6^2
gamma_A2 <- P_CA_D2 / (P_CA_D2 + P_CB_D2)
gamma_A2
```

```
## [1] 0.9705765
```

$$\gamma_{B2} = 1 - \gamma_{A2} = 0.0294235$$

```r
gamma_B2 <- P_CB_D2 / (P_CA_D2 + P_CB_D2)
gamma_B2
```

```
## [1] 0.02942349
```

The updated $P(C_A)$ would be

$$\begin{aligned}\hat{P}(C_A) &= \frac{\gamma_{A1} + \gamma_{A2}}{2} \\ &\approx 0.575\end{aligned}$$

```r
(gamma_A1 + gamma_A2) / 2
```

```
## [1] 0.5753911
```

$$\begin{aligned}\hat{P}(C_B) &= \frac{\gamma_{B1} + \gamma_{B2}}{2} \\ &\approx 0.425\end{aligned}$$

```r
(gamma_B1 + gamma_B2) / 2
```

```
## [1] 0.4246089
```

Problem 5: Learning a mixture model for two biased coins

In this problem, you will implement the EM algorithm for the coin toss problem in R.

Below we provide you with a skeleton of the algorithm. You can either fill this skeleton with the required functions or write your own version of the EM algorithm. If you choose to do the latter, please also present your results using Rmarkdown in a clear fashion.

```
set.seed(2023)
```

(a) Load data

We first read the data stored in the file "coinflip.csv".

```
# read the data into D
D <- read.csv("coinflip.csv")
# check the dimension of D
all(dim(D) == c(200, 100))
```

```
## [1] TRUE
```

(b) Initialize parameters

Next, we will need to initialize the mixture weights and the probabilities of obtaining heads. You can choose your own values as long as they make sense.

```
# Number of coins
k <- 2
# Mixture weights (a vector of length k)
lambda <- rep(0.5, k) ## YOUR CODE ##
# Probabilities of obtaining heads (a vector of length k)
theta <- c(0.7, 0.3) ## YOUR CODE ##
```

(c) The EM algorithm

Now we try to implement the EM algorithm. Please write your code in the indicated blocks.

```
##' This function implements the EM algorithm for the coin toss problem
##' @param D Data matrix of dimensions 100-by-N, where N is the number of observations
##' @param k Number of coins
##' @param lambda Vector of mixture weights
##' @param theta Vector of probabilities of obtaining heads
##' @param tolerance A threshold used to check convergence
coin_EM <- function(D, k, lambda, theta, tolerance = 1e-2) {

  # expected complete-data (hidden) log-likelihood
  ll_hid <- -Inf
  # observed log-likelihood
  ll_obs <- -Inf
  # difference between two iterations
```

```r
diff <- Inf
# number of observations
N <- nrow(D)
# responsibilities
gamma <- matrix(0, nrow = k, ncol = N)

# run the E-step and M-step until convergence
while (diff > tolerance) {

  # store old likelihood
  ll_obs_old <- ll_obs

  ############# E-step #############

  ### YOUR CODE STARTS ###

  # Compute the responsibilities
  L <- dim(D)[2] # number of coin tosses in each obs

  likelihoods <- rbind(theta[1]^rowSums(D) * (1-theta[1])^(L - rowSums(D)),
                       theta[2]^rowSums(D) * (1-theta[2])^(L - rowSums(D)))

  posterior_1 <- lambda[1] * likelihoods[1, ]
  posterior_2 <- lambda[2] * likelihoods[2, ]

  gamma[1, ] <- posterior_1 / (posterior_1 + posterior_2)
  gamma[2, ] <- posterior_2 / (posterior_1 + posterior_2)

  # Update expected complete-data (hidden) log-likelihood
  ll_hid <- sum(gamma * log(lambda * likelihoods))

  # Update observed log-likelihood
  ll_obs <- sum(log(colSums(lambda * likelihoods)))

  # Recompute difference between two iterations
  diff <- abs(ll_obs - ll_obs_old)

  ### YOUR CODE ENDS ###

  ############# M-step #############

  ### YOUR CODE STARTS ###

  # Recompute priors (mixture weights)
  lambda <- rowSums(gamma) / N

  # Recompute probability of heads for each coin
  heads <- rowSums(D)
  count_heads <- gamma %*% heads
  count_tot <- rowSums(gamma) * L

  theta <- t(count_heads / count_tot)
  ### YOUR CODE ENDS ###
```

```
  }

  return(list(ll_hid = ll_hid,
              ll_obs = ll_obs,
              lambda = lambda,
              theta = theta,
              gamma = gamma))

}
```

Run the EM algorithm:

```
res <- coin_EM(D, k, lambda, theta)
```

(d) Results

Probability of heads:

```
## YOUR CODE ##
res$theta
```

```
##           [,1]      [,2]
## [1,] 0.5702153 0.4503906
```

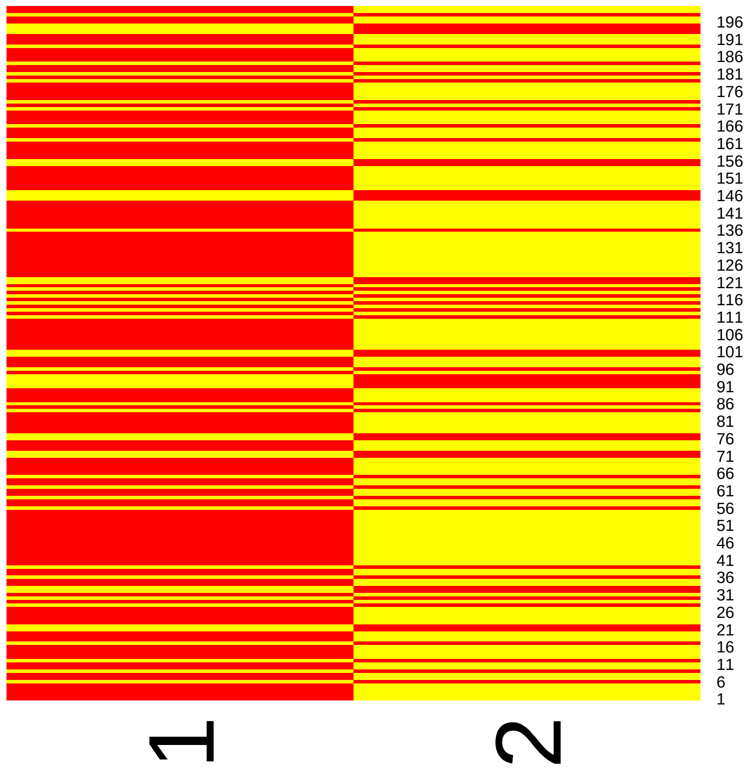
Mixture weights:

```
## YOUR CODE ##
res$lambda
```

```
## [1] 0.1669887 0.8330113
```

Heatmap of responsibilities:

```
## YOUR CODE ##
coul <- heat.colors(2, 1)
heatmap(t(res$gamma), Colv=NA, Rowv=NA, scale="column", col=coul)
```



How many observations belong to each coin?

```
## YOUR CODE ##
coin_1 <- sum(res$gamma[1, ] > res$gamma[2, ])
coin_1
```

```
## [1] 29
```

```
coin_2 <- sum(res$gamma[2, ] > res$gamma[1, ])
coin_2
```

```
## [1] 171
```