Part I: Preliminary material

Before the lecture and the practical, please work through **Part I** of the R source file available from

http://www.maths.dur.ac.uk/~dma0je/PG/Mix/MSc/CodeMSc19.r

The following commands will become useful for the programming tasks in the practical:

if/then

This command performs an *action* if the *condition* is met. One can specify an alternative *action2* if an alternative *condition2* is met, and a further alternative *action3* if not any condition was met.

```
if (condition){
      action
} else if (condition2){
      action2
} else {
      action3
}
```

Both the parts commencing with `else` and `else if` are optional. For instance,

```
if (log(10)<pi){
      pi
} else {
      log(10)
}
```

gives the value of `pi`.

for

A `for` loop repeats an *action* for all elements of a *set*. Formally,

```
for (i in set ){
      action
}
```

For instance,

```
for (i in 1:10){
      cat('This is loop', i, '\n')
}
```

will produce 10 rows of text which report the number of the loop (The string '`\ n`' is borrowed from the C language and means to start a new line).

while

A `while` loop works similar as `for`, but instead of working though a *set*, it checks in every iteration whether a *condition* is met:

```
while (condition){
      action
}
```

Functions

Functions allow to prepare some code which can be used later with different function arguments. For instance,

```
testlog <- function(x){
    if (x>0){
       log(x)
    } else {
       cat("log not defined for non-positive argument.")
    }
}
```

will give the logarithm of `x` if `x` is positive, and an error message otherwise.
Functions can also have more than one argument, which are then separated by commas. Default values can be given behind a `=` symbol, for instance

```
max1<- function(a,b=1){
   result<- max(a,b)
   return(result)
}

max1(0.5)
[1] 1
max1(0.5,0)
[1] 0.5
```

apply

This function allows to carry out some operation onto all rows or columns of a matrix. For instance, if `W` is a $n \times p$ matrix, then

$$\texttt{apply}(\texttt{W}, 1, \texttt{sum})$$

would give a $n \times 1$ vector which contains the sums over each row, and

$$\texttt{apply}(\texttt{W}, 2, \texttt{mean})$$

would give the column means. Useful variants are `tapply` (carries out operations on the elements of `W` grouped by a factor, the name of which is given as second argument), and `lapply` (for operations on each element of a list `W`; here the second argument is not needed).

Part II: The EM Algorithm for Finite Gaussian Mixtures

The theoretical aspects of this part are covered in the lecture. The slides can be downloaded from

http://www.maths.dur.ac.uk/~dma0je/PG/Mix/MSc/SlidesMSc19.pdf

In the practical, the task is to implement a univariate version of the EM algorithm for Gaussian mixtures. Please follow the tasks given in the code file in order to do this.

Part III: Supplementary topics

Simulation from Gaussian mixtures

Given a set of parameters θ, data are simulated from a Gaussian mixture in two steps: Firstly we draw a $k \in \{1, \ldots, K\}$, then we simulate from a Gaussian:

- Draw a value x from a uniform distribution on $[0, 1]$ (using `runif`). If

$$x \in \left] \sum_{j=1}^{k-1} \pi_j, \sum_{j=1}^{k} \pi_j \right],$$

  we decide for component k.

- Draw a value y from a normal distribution with mean μ_k and variance σ^2 (using `rnorm`).

Likelihood and Disparity

We wish to compute the likelihood $L(\hat{\theta}|y_1, \ldots, y_n)$ (this is *not* the complete likelihood used in EM) of the fitted model. One has

$$L(\hat{\theta}|y_1, \ldots, y_n) = \prod_{i=1}^{n} f(y_i|\hat{\theta}) = \prod_{i=1}^{n} \left(\sum_{k=1}^{K} \hat{\pi}_k \phi_{\hat{\mu}_k, \hat{\sigma}^2}(y_i) \right) \tag{1}$$

so that the log-likelihood is given by

$$\ell(\hat{\theta}|y_1, \ldots, y_n) = \sum_{i=1}^{n} \log f(y_i|\hat{\theta}) = \sum_{i=1}^{n} \log \left(\sum_{k=1}^{K} \hat{\pi}_k \phi_{\hat{\mu}_k, \hat{\sigma}^2}(y_i) \right) \tag{2}$$

An alternative quantity which is often more convenient to use and interpret (for instance, in conjunction with likelihood ratio tests, see below), is the *disparity*

$$D(\hat{\theta}|y_1, \ldots, y_n) = -2 \log L(\hat{\theta}|y_1, \ldots, y_n) = -2\ell(\hat{\theta}|y_1, \ldots, y_n).$$

For the computation of either of these, we will need to compute all entries of the $n \times K$ matrix, say F, which is defined by the values of

$$\hat{\pi}_k \phi_{\hat{\mu}_k, \hat{\sigma}^2}(y_i), \qquad 1 \leq i \leq n, 1 \leq k \leq K$$

Note that, with $\mathtt{y} = (y_1, \ldots y_n)$, the command

$$\mathtt{pi}[\mathtt{k}] * \mathtt{dnorm}(\mathtt{y}, \mathtt{mu}[\mathtt{k}], \mathtt{sigma})$$

provides immediately the $k-$th column of F.

Likelihood ratio test for K

We wish to test

$$H_0 : K = K_0 \quad vs. \quad H_1 : K = K_0 + 1.$$

Denote by $\hat{\theta}_K$ the estimate of θ when K mixture components are used.
Wilk's likelihood ratio statistics:

$$\begin{aligned} W &= -2 \log \frac{L(\hat{\theta}_{K_0}|y_1, \ldots, y_n)}{L(\hat{\theta}_{K_0+1}|y_1, \ldots, y_n)} = \\ &= D(\hat{\theta}_{K_0}|y_1, \ldots, y_n) - D(\hat{\theta}_{K_0+1}|y_1, \ldots, y_n) \end{aligned}$$

The actual test is implemented through the bootstrap:

(i) Compute W as above. Call this value W_0.

(ii) From the model with K_0 components, simulate, say, 99 data sets of size n.

(iii) For each of these 99 data sets, recalculate $\hat{\theta}_{K_0}$ and $\hat{\theta}_{K_0+1}$, and compute the corresponding values of W.

(iv) Find the position P of W_0 within all the other values of W. The $p-$value is given by $1 - P/100$.

Part IV: Solutions

Full solutions will be made available, following the practical, under

http://www.maths.dur.ac.uk/˜dma0je/PG/Mix/MSc/SolutionsMSc19.r