Basic programming (Part I)

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 and can be omitted. For instance, the command

```
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
}

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).

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
```

Finite Gaussian Mixtures (Part II)

Finite Gaussian mixtures

Assume we are given K univariate normal distributions $N(\mu_k, \sigma_k^2)$, $k = 1, \ldots, K$. A finite Gaussian mixture is a distribution which draws with probability p_k from the $k-$th normal distribution. Formally, the density of a finite Gaussian mixture is given by

$$f(y|\theta) = \sum_{k=1}^{K} p_k \phi(y|\mu_k, \sigma_k^2) \tag{1}$$

where $K < \infty$ is the number of mixture components, $\theta = (p_1, \ldots, p_{K-1}, \mu_1, \ldots, \mu_K, \sigma_1, \ldots, \sigma_K)^T$ is the vector of parameters, and $\phi(y|\mu_k, \sigma_k^2) = \frac{1}{\sqrt{2\pi\sigma_k^2}} \exp\left\{-\frac{1}{2}\left(\frac{y-\mu_k}{\sigma_k}\right)^2\right\}$ is the probability density function of a normal distribution with mean μ_k and variance σ_k^2, evaluated at y. Note that $p_K = 1 - \sum_{k=1}^{K-1} p_k$.

Of course, this could be generalized to mixtures of multivariate Gaussian distributions, or mixtures of other distributions than Gaussians (such as Poisson), but we will not do this in this course.

Estimation of Gaussian mixtures

Given some data $y_i, i = 1, \ldots, n$, we wish to obtain an estimator, $\hat{\theta}$, of θ. This is done by the **EM algorithm** (**E**xpectation - **M**aximization). Based on a vector of starting values, say θ_0, the EM algorithm iterates between....

E-step Update membership probabilities $w_{ik} = P(\text{obs. } i \text{ belongs to comp. } k)$ via

$$w_{ik} = \frac{p_k \phi(y|\mu_k, \sigma_k^2)}{\sum_{\ell=1}^{K} p_\ell \phi(y|\mu_\ell, \sigma_\ell^2)} \tag{2}$$

M-Step Update parameter estimates via

$$\hat{p}_k = \frac{1}{n}\sum_{i=1}^{n} w_{ik} \tag{3}$$

$$\hat{\mu}_k = \frac{\sum_{i=1}^{n} w_{ik} y_i}{\sum_{i=1}^{n} w_{ik}} \tag{4}$$

$$\hat{\sigma}_k^2 = \frac{\sum_{i=1}^{n} w_{ik}(y_i - \mu_k)^2}{\sum_{i=1}^{n} w_{ik}} \tag{5}$$

...until convergence is reached.

Derivation of EM algorithm for Gaussian mixtures

Complete Likelihood. Given some data $y_i, i = 1, \ldots, n$, we wish to obtain an estimator, $\hat{\theta}$, of θ. Let G be the random vector which draws a class $k \in \{1, \ldots, K\}$. We know that $P(G = k) = p_k$. Denoting $f_{ik} \equiv P(y_i|G = k) = \phi(Y|\mu_k, \sigma_k^2)$, then we also know that

$$P(y_i, G = k) = P(y_i|G = k)P(G = k) = f_{ik}p_k \tag{6}$$

The key idea is now as follows. Assume that, for an observation y_i, *the value of G is known*, i.e. we know to which of the K components the $i-$th observation belongs. We can express this knowledge through an indicator variable,

$$G_{ik} = \begin{cases} 1 & \text{if observation} \quad i \quad \text{belongs to component} \quad k \\ 0 & \text{otherwise.} \end{cases}$$

This gives "complete" data $(y_i, G_{i1}, \ldots G_{iK})$, $i = 1, \ldots, n$, with probability

$$P(y_i, G_{i1}, \ldots, G_{iK}) = \prod_{k=1}^{K} (f_{ik}p_k)^{G_{ik}}$$

(this follows from (6) since only one of the G_{ik}'s is true). The corresponding likelihood function, called *complete likelihood*, is

$$L^*(\theta|y_1, \ldots, y_n) = \prod_{i=1}^{n} \prod_{k=1}^{K} (p_k f_{ik})^{G_{ik}}. \tag{7}$$

One obtains the log-likelihood

$$\ell^* = \log L^* = \sum_{i=1}^{n} \sum_{k=1}^{K} G_{ik} \log p_k + G_{ik} \log f_{ik} \tag{8}$$

E-step. As the G_{ik} are in fact unknown, we replace them by their conditional expectations

$$w_{ik} \equiv E(G_{ik}|y_i) = P(G_{ik} = 1|y_i) = P(G = k|y_i)$$

Using Bayes' theorem, one has

$$w_{ik} = P(G = k|y_i) = \frac{P(G = k)P(y_i|G = k)}{\sum_\ell P(G = \ell)P(y_i|G = \ell)} = \frac{p_k f_{ik}}{\sum_\ell p_\ell f_{i\ell}}$$

which is equivalent to the expression provided in (2).

M-step. Setting $\partial\ell^*/\partial\mu_k = 0$ and $\partial\ell^*/\partial\sigma_k = 0$ for $k = 1, \ldots, K$, one obtains exactly the estimates which are given for μ_k and σ_k in (4) and (5), respectively. For the p_k, one needs to apply a Lagrange multiplier since $\sum_{k=1}^{K} p_k = 1$. Setting

$$\partial \left(\ell^* - \lambda(\sum_{k=1}^{K} p_k - 1) \right) / \partial p_k = 0, \qquad k = 1, \ldots, K$$

one obtains the updated formula for p_k given in (3).

Convergence was proven in Dempster et al. (1977), Wu (1983).

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

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

  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

$$L(\hat{\theta}|y_1, \ldots, y_n) = \prod_{i=1}^{n} f(y_i|\hat{\theta}) = \prod_{i=1}^{n} \left(\sum_{k=1}^{K} \hat{p}_k \phi(y_i|\hat{\mu}_k, \hat{\sigma}_k^2) \right) \tag{9}$$

so that the log-likelihood is given by

$$\ell(\hat{\theta}|y_1, \ldots, y_n) = \sum_{i=1}^{n} \log f(y_i|\hat{\theta}) = \sum_{i=1}^{n} \log \left(\sum_{k=1}^{K} \hat{p}_k \phi(y_i|\hat{\mu}_k, \hat{\sigma}_k^2) \right). \tag{10}$$

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

$$\hat{p}_k \phi(y_i|\hat{\mu}_k, \hat{\sigma}_k^2), \qquad 1 \le i \le n, 1 \le k \le K.$$

Note that, with `y` $=(y_1, \ldots y_n)$, the command

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

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 $1 - P/100$.

For details, see Aitkin, Francis, Hinde, and Darnell (2009), Statistical Modelling in R, page 442.