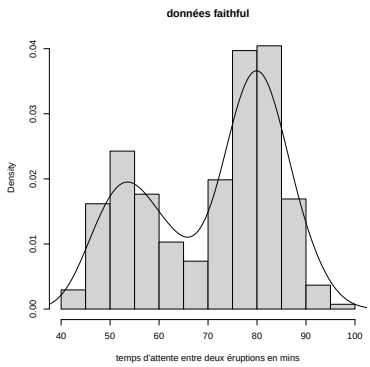
Modélisation en biologie. Modèles à variables latentes.

Les modèles de mélanges de lois de probabilités sont un outil statistique permettant de modéliser des phénomènes biologiques hétérogènes. On retrouve leur utilisation en modélisation des données omics : génomique, chip-seq, RNA-seq... Les modèles de mélange peuvent être utilisés pour effectuer des classifications. A partir d'une représentation graphique des données, on peut détecter une distribution bimodale (avec deux modes, voir plus) et proposer une modélisation de type "mélange de lois de probabilités".

```
library(datasets)
data(faithful)
par(mfrow=c(1,1))
hist(faithful$waiting, main="données faithful",
     freq=FALSE,
     xlab="temps d'attente entre deux éruptions en mins")
points(density(faithful$waiting),type="l")
```



Cadre statistique pour les mélanges de lois de probabilités.

On considère n observations $(y_1,\dots,y_n)$ indexées par $i \in 1,\dots,n$, indépendantes les unes des autres. On suppose que ces observations sont la réalisation d'un modèle de mélange de lois gaussiennes (mais on pourrait prendre une autre loi si l'on voulait) :

— Z_i est une variable catégorielle à K classes suivant une loi multinomiale. Les Z_i sont i.i.d. Conditionnellement à la valeur prise par la variable aléatoire latente Z_i, les Y_i sont indépendants et suivent une loi normale de paramètres μ_k et σ_k :

$$Z_i \sim \mathcal{M}(1,(\pi_1,\dots,\pi_K))$$
$$Y_i \,|\, (Z_i = k) \sim \mathcal{N}(\mu_k,\sigma_k)$$

— On note $\pi = (\pi_1,\dots,\pi_K)$, $\mu = (\mu_1,\dots,\mu_K)$ et $\sigma = (\sigma_1,\dots,\sigma_K)$. On note $\theta = (\pi,\mu,\sigma)$ le vecteur regroupant l'ensemble des paramètres du modèle.

— Dans ce modèle, les variables $(Y_1,\dots,Y_n)$ sont les **variables dites "observées"** du modèle de mélange, et $(Z_1,\dots,Z_n)$ sont les **variables latentes (ou cachées)** du modèle.

Remarques :

- La loi multinomiale généralise la loi biomiale dans le cas d'une expérience à plus de deux issues possibles (voir la page wikipedia sur la loi multinomiale). Ici, on a une loi multinomiale de paramètres 1 et $(\pi_1,\ldots,\pi_K)$. Ce qui revient simplement à dire que $P(Z_i = k) = \pi_k$ pour tout $k \in 1,\ldots,K$ et tout $i \in 1,\ldots,n$.
- Question : compter le nombre de paramètres à estimer dans le cas d'un modèle de mélange de deux gaussiennes univariées. (réponse : 5)
- Question : compter le nombre de paramètres à estimer dans le cas d'un modèle de mélange de 3 gaussiennes bivariées. (réponse : 18)

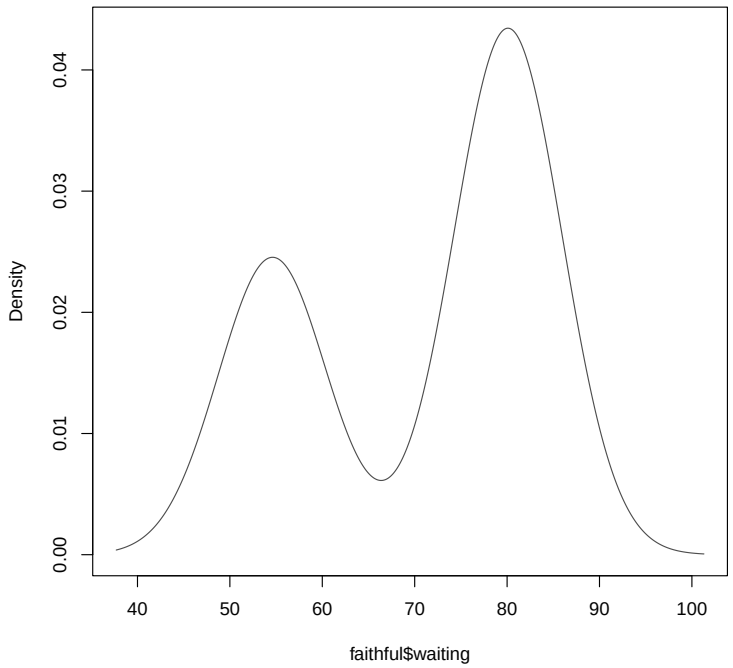
Mise en oeuvre sous R

A partir d'un vecteur de données, on peut utiliser le package `mclust` pour inférer les valeurs des paramètres de notre modèle de mélange.

```
library(mclust)

## Package 'mclust' version 6.0.1
## Type 'citation("mclust")' for citing this R package in publications.

res=Mclust(faithful$waiting)
par(mfrow=c(1,2))
res$parameters

## $pro
## [1] 0.3609461 0.6390539
##
## $mean
##        1        2
## 54.61675 80.09239
##
## $variance
## $variance$modelName
## [1] "E"
##
## $variance$d
## [1] 1
##
## $variance$G
## [1] 2
##
## $variance$sigmasq
## [1] 34.44093
```

Le vecteur (π_1,π_2) est (0.36, 0.64), la moyenne μ_1 dans le premier groupe est 54.6167498, la moyenne μ_2 dans le premier groupe est 80.0923933, la variance dans les deux groupes est la même $\sigma_1^2 = \sigma_2^2$ et vaut 34.4409276.

```
plot(res,what = "density")
```



Estimations des paramètres dans un modèle de mélange

La densité des données s'écrit :

$$f(y_i) = \sum_{k=1}^{K} \pi_k \phi(y_i; \mu_k, \sigma_k^2)$$

où $\phi(y_i; \mu_k, \sigma_k^2) = \frac{1}{\sqrt{2\pi\sigma_k^2}} \exp(-\frac{(x-\mu_k)^2}{2\sigma_k^2})$ est la densité de la loi gaussienne.
La log-vraisemblance du modèle s'écrit (les observations étant i.i.d.) :

$$L(y_1, \dots, y_n; \theta) = \sum_{i=1}^{n} \log\left(\sum_{k=1}^{K} \pi_k \phi(y_i; \mu_k, \sigma_k^2) \right)$$

On remarque la présence de la somme dans le logarithme, ce qui empêche d'écrire simplement les estimateurs du maximum de vraisemblance ou même de résoudre simplement le problème d'optimisation. On pourrait utiliser des méthodes numériques à la place, mais celles-ci peuvent devenir rapidement très coûteuses.
Le problème de résolution serait plus simple si l'on connaissait les valeurs prises par les variables aléatoires $(Z_1, \dots, Z_n)$. En pratique, on observe uniquement des réalisations de $(Y_1, \dots, Y_n)$: $(y_1, \dots, y_n)$ sans observer les valeurs de $(Z_1, \dots, Z_n)$. Les variables $(Y_1, \dots, Y_n)$ sont les variables "observées" du modèle de mélange, et $(Z_1, \dots, Z_n)$ sont les variables latentes (ou cachées) du modèle. Les couples (Y_i, Z_i) sont indépendants. On note $z_{ik} = \mathbf{1}_{Z_i = k}$. On peut écrire la log-vraisemblance observée y_i complétée par les variables cachées z_i :

$$L((y_1, z_1), \dots, (y_n, z_n); \theta) = \sum_{i=1}^{n} \sum_{k=1}^{K} z_{ik} \log\big(\pi_k \phi(y_i; \mu_k, \sigma_k^2)\big)$$

Cette log-vraisemblance est plus facile à maximiser que la log-vraisemblance $L(y_1, \dots, y_n; \theta)$. Cependant, en pratique, on ne connait pas les valeurs de z_{ik} pour tout i et tout k. On peut cependant faire "comme si" on connaissait ces valeurs, puis chercher à trouver des "bons" estimateurs de θ, puis réaffecter des valeurs "probables" aux z_{ik}, puis recalculer des estimateurs de θ, et ceci jusqu'à convergence. C'est l'idée sous-jacente à l'algorithme EM.

L'algorithme Espérance-Maximisation (Dempster et al. 1977)

Cet algorithme est assez général. Il s'agit plutôt d'un principe général d'estimation.

- On initialise les valeurs des paramètres $\theta^{(0)}$. On va mettre à jour la valeur des paramètres à chaque étape c et on note $\theta^{(c)}$ à chaque étape.
- **Initialisation :** On choisit une valeur $\theta^{(0)}$ (si possible, pas trop mauvaise).
- **Etape E :** On calcule la loi des variables cachées Z_i sachant les paramètres courants et les données observées $y_1, \ldots, y_n$.
- **Etape M :** On calcule une mise à jour des paramètres $\theta^{(c+1)}$en maximisant l'espèrance conditionnelle de la vraisemblance des données complétée sachant les données observées.
- On alterne étapes E et M jusqu'à convergence de l'algorithme.
- On peut montrer que chaque itération augmente la vraisemblance des données : pour tout c

$$L(y_1, \ldots, y_n; \theta^{(c+1)}) \geq L(y_1, \ldots, y_n; \theta^{(c)})$$

L'algo EM pour l'estimation de paramètes pour le cas du modèle de mélange de lois gaussiennes

- L'étape E revient à calculer les quantités suivantes :

$$t_{ij} = \mathbb{E}\left(z_{ij} \mid (y_1, \ldots, y_n); \theta^{(c)}\right)$$

$$t_{ij} = \frac{\pi_j^{(c)} \phi(y_i; \mu_j^{(c)}, \sigma_j^{(c)})}{\sum_{k=1}^{K} \pi_k^{(c)} \phi(y_i; \mu_k^{(c)}, \sigma_k^{(c)})}$$

- L'étape M revient à calculer les quantités suivantes :

$$\pi_k = \frac{1}{n} \sum_{i=1}^{n} t_{ik}$$

$$\mu_k = \frac{\sum_{i=1}^{n} t_{ik} y_i}{\sum_{i=1}^{n} t_{ik}}$$

$$\sigma_k^2 = \frac{\sum_{i=1}^{n} t_{ik} (y_i - \mu_k)^2}{\sum_{i=1}^{n} t_{ik}}$$

  Ces estimateurs correspondent aux estimateurs du maximum de vraisemblance trouvés à partir de la quantité $L((y_1, z_1), \ldots, (y_n, z_n); \theta) = \sum_{i=1}^{n} \sum_{k=1}^{K} z_{ik} \log\left(\pi_k \phi(y_i; \mu_k, \sigma_k^2)\right)$ où l'on a remplacé les valeurs z_{ik} par les t_{ik} estimés à l'étape E, pour tout i et tout k.

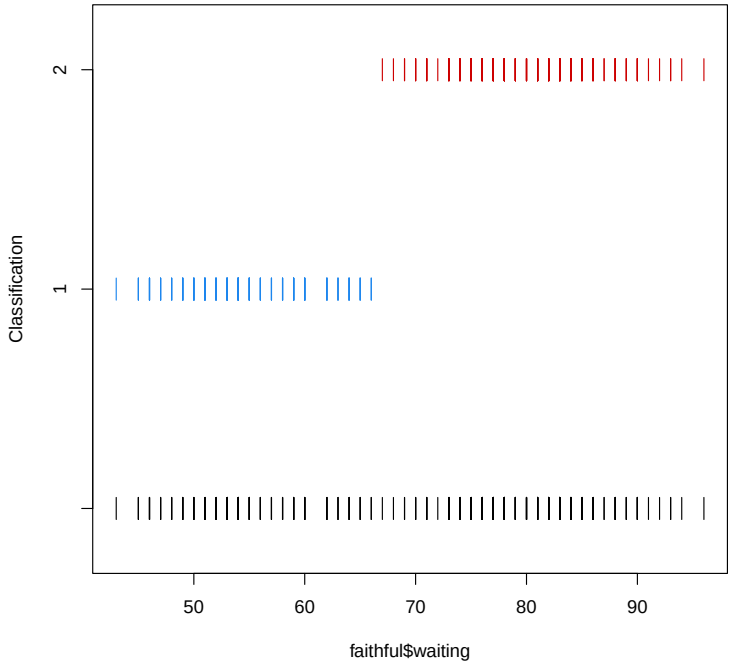
Quelques variantes de l'algorithme EM pour la classification.

- Algorithme CEM (Classification EM) : à chaque étape E, on affecte chaque observation i à sa classe la plus probable. C'est la règle du Maximum A Posteriori (MAP).
- Algorithme SEM (Stochastique EM) : à chaque étape E, on tire la valeur z_{ik} sous une loi $\mathscr{M}\left(1, t_{i1}^{(c)}, \ldots, t_{iK}^{(c)}\right)$.

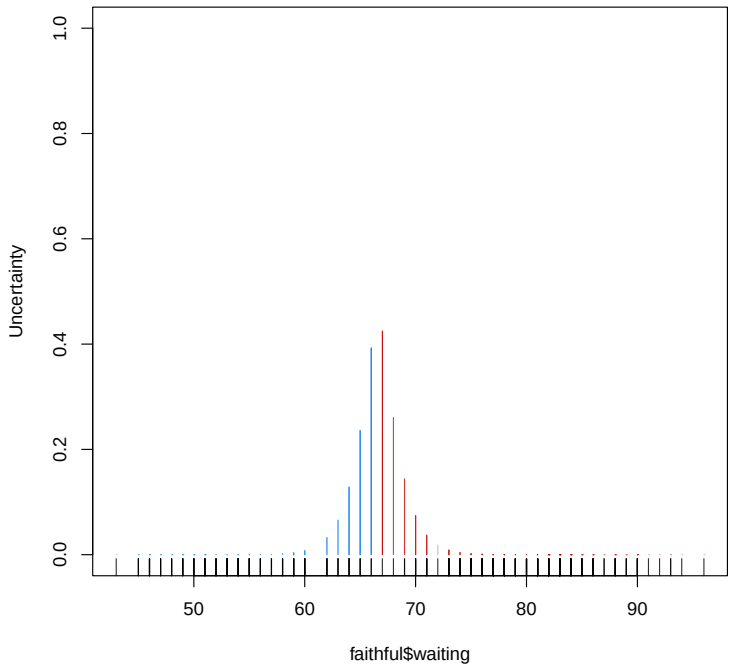
Classification des observations à l'aide d'un modèle de mélange.

A l'aide d'un modèle de mélange et des valeurs t_{ik} calculées pour tout individus i et toute composante k, on peut affecter un individu à un groupe.

```
table(res$classification)

##
##   1   2
##  99 173

plot(res,what = "classification")
```




```
plot(res,what = "uncertainty")
```



Les valeurs des t_{ik} sont accessibles à partir de l'objet suivant :

```
head(res$z)

##              [,1]         [,2]
## [1,] 1.030748e-04 0.9998969252
## [2,] 9.999098e-01 0.0000901813
## [3,] 4.146885e-03 0.9958531153
## [4,] 9.675687e-01 0.0324313138
## [5,] 1.217871e-06 0.9999987821
## [6,] 9.998111e-01 0.0001889468

head(res$classification)

## [1] 2 1 2 1 2 1
```

Choix du nombre de classes

— Le choix du nombre de classes K dans un modèle de mélange peut être effectué à l'aide du critère BIC ou du critère ICL (Integrated Completed Likelihood).
On sélectionne le nombre de classes $K = 1, \ldots, K^{max}$ qui maximise l'un des critères suivant, où ν_K est le nombre de paramètres du modèle

— **Bayesian Information Criterion (BIC)** (Schwarz, 1978)

$$\text{BIC}(K) = \log L(y_1, \ldots, y_n | \theta^{(c)}) - \frac{\nu_K}{2} \log(n)$$

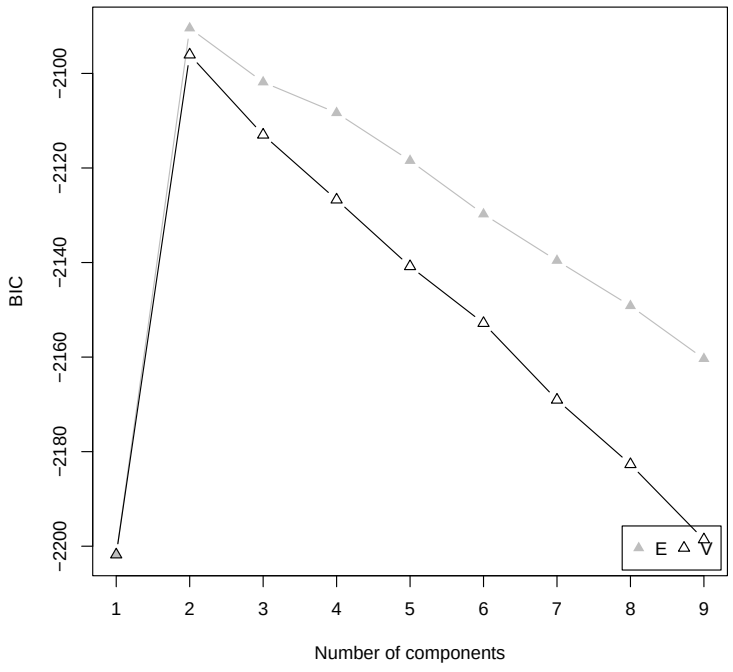
— **Integrated Completed Likelihood (ICL)** (Biernacki et al., 2000)

$$\text{ICL}(K) = \text{BIC}(K) - \sum_{i,k} t_{ik} \log(t_{ik})$$

— Le critère ICL prend en compte l'incertitude de classification à travers le calcul de la quantité $-\sum_{i=1}^{n}\sum_{k=1}^{K} t_{ik} \log t_{ik}$. Le critère ICL va favoriser un nombre de classes optimal de sorte que les classes détectées ne soient pas chevauchantes.

Sur les données `faithful`, le nombre de classes sélectionnées par le critère BIC est 2. On remarque également que, à nombre de classes fixé, on peut comparer deux modèles différents, un modèle E où les variances des différentes composantes sont égales, un modèle V où chaque composante du modèle de mélange a sa propre variance. Ici, le modèle E est privilégié.

```
plot(res,what = "BIC")
```



Autres remarques

— Pour un nombre de classes K fixé, il y a $K!$ définitions équivalentes possibles d'un modèle de mélange : on peut permuter les labels des classes.

— Nous avons détaillé les formules pour le cas des modèles de mélange de lois gaussiennes mais tout type de lois peut être utilisé : loi de Poisson, loi multinomiale, loi normale multidimensionnelle, etc... Dans le cas des lois normales multidimensionnelles, de nombreuses variantes de paramétrisations des matrices de variance covariance sont possibles (voir notices du package `mclust`) (voir TP2).