# TP1 – Algorithme EM

Dans ce premier TP, on cherche à estimer les paramètres d'une ellipse passant « au plus près » d'un ensemble de points, ces points formant un nuage dont la forme rappelle effectivement celle d'une ellipse. Le script `donnees` affiche une ellipse caractérisée par cinq paramètres réels tirés aléatoirement : la demi-longueur $a$ du grand axe ; l'excentricité $e$ ; les coordonnées $(x_C, y_C)$ du centre ; enfin, l'angle polaire $\theta$ du grand axe. Ce script affiche également $n_{\text{app}}$ points $P_i$ situés au voisinage de l'ellipse, qui forment un ensemble d'apprentissage $\mathcal{D}_{\text{app}}$.

## Exercice 1 : estimation par le maximum de vraisemblance

Une méthode d'estimation classique, déjà vue en 1A, est celle du *maximum de vraisemblance*. Cette méthode très intuitive revient à répéter, pour un certain nombre d'ellipses tirées aléatoirement, le calcul d'une valeur, appelée *vraisemblance*, qui mesure l'adéquation des données d'apprentissage à l'ellipse. Une fois les vraisemblances calculées pour toutes les ellipses tirées aléatoirement, l'ellipse qui maximise la vraisemblance est retenue.

La vraisemblance $L_{\mathbf{p}}(\mathcal{D}_{\text{app}})$ d'une ellipse de paramètres $\mathbf{p} = [a, e, x_C, y_C, \theta]$, relativement à $\mathcal{D}_{\text{app}}$, s'écrit :

$$L_{\mathbf{p}}(\mathcal{D}_{\text{app}}) = \prod_{i=1}^{n_{\text{app}}} f_{\mathbf{p}}(P_i) \tag{1}$$

où la densité de probabilité $f_{\mathbf{p}}(P_i)$ des points $P_i \in \mathcal{D}_{\text{app}}$ peut être modélisée par une loi normale :

$$f_{\mathbf{p}}(P_i) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left\{-\frac{r_{\mathbf{p}}(P_i)^2}{2\,\sigma^2}\right\} \tag{2}$$

Dans cette expression, $r_{\mathbf{p}}(P_i)$ mesure l'adéquation du point $P_i$ à l'ellipse de paramètres $\mathbf{p}$. Si $C$ désigne le centre de l'ellipse, $r_{\mathbf{p}}(P_i)$ est égal à la distance algébrique, sur la droite $(CP_i)$, de $P_i$ au point d'intersection $Q_i$ entre l'ellipse et la droite $(CP_i)$ :

$$r_{\mathbf{p}}(P_i) = \overline{Q_i P_i} \tag{3}$$

Quant à $\sigma$, bien que ce paramètre soit également inconnu, nous supposons pour simplifier qu'il est connu et égal à l'écart-type du bruit sur les données d'apprentissage.

Comme un produit est plus difficile à maximiser qu'une somme, et que la fonction logarithme est strictement croissante, il est préférable de maximiser la *log-vraisemblance* $\ln L_{\mathbf{p}}(\mathcal{D}_{\text{app}})$. Il faut donc résoudre le problème :

$$\max_{\mathbf{p}} \left\{\ln \prod_{i=1}^{n_{\text{app}}} f_{\mathbf{p}}(P_i)\right\} = \min_{\mathbf{p}} \left\{\sum_{i=1}^{n_{\text{app}}} r_{\mathbf{p}}(P_i)^2\right\} \tag{4}$$

Dans (4), l'exponentielle de (2) a disparu grâce au logarithme, la maximisation est devenue une minimisation à cause du signe $-$ dans l'argument de l'exponentielle, et les coefficients de normalisation ont été supprimés car ils n'influent pas sur l'optimisation.

Écrivez la fonction `max_vraisemblance`, appelée par le script `exercice_1`, qui résout le problème (4) en calculant la somme $\sum_{i=1}^{n_{\text{app}}} r_{\mathbf{p}}(P_i)^2$ pour `nb_tirages` valeurs du vecteur de paramètres $\mathbf{p}$ tirées aléatoirement. Remarque importante : il est inutile de coder le calcul de $r_{\mathbf{p}}(P_i)$, car ce calcul est effectué par la fonction `calcul_r`, qui vous est fournie. La fonction `calcul_score`, également fournie, sert à mesurer la précision de l'estimation. Pour une paire d'ellipses passées en entrée, cette fonction calcule le quotient entre l'aire de l'intersection et l'aire de l'union. Le score calculé est compris entre 0 (ellipses disjointes) et 1 (ellipses identiques).

## Exercice 2 : estimation par résolution d'un système linéaire

Pour estimer les paramètres recherchés, on peut manipuler l'équation cartésienne d'une ellipse :

$$\alpha\, x^2 + \beta\, x\, y + \gamma\, y^2 + \delta\, x + \epsilon\, y + \phi = 0 \tag{5}$$

car cette équation est linéaire vis-à-vis du sextuplet de coefficients $(\alpha, \beta, \gamma, \delta, \epsilon, \phi)$. Comme l'ellipse ne change pas s'ils sont tous multipliés par un même coefficient, ces six paramètres ne sont pas indépendants. Cela est cohérent avec le fait que le vecteur $\mathbf{p} = [a, e, x_C, y_C, \theta]$ contient cinq paramètres, et non six. Les équations (5), écrites pour les $n_{\text{app}}$ points de $\mathcal{D}_{\text{app}}$, forment un système linéaire, mais ce système est *homogène* :

$$\mathbf{A}\,\mathbf{X} = \mathbf{O}_{n_{\text{app}}} \tag{6}$$

où $\mathbf{X} = [\alpha, \beta, \gamma, \delta, \epsilon, \phi]^\top$, $\mathbf{A}$ est une matrice de taille $n_{\text{app}} \times 6$, et $\mathbf{O}_{n_{\text{app}}}$ désigne le vecteur nul de $\mathbb{R}^{n_{\text{app}}}$. Afin d'éviter la solution triviale $\widehat{\mathbf{X}} = \mathbf{O}_6$, il faut ajouter une contrainte. On peut par exemple imposer $\alpha + \gamma = 1$, car une ellipse vérifie forcément $\beta^2 - 4\,\alpha\,\gamma < 0$, donc il est impossible que $\alpha + \gamma = 0$. Cette contrainte ajoute une équation linéaire *non homogène* au système (6), qui n'admet donc plus la solution triviale $\widehat{\mathbf{X}} = \mathbf{O}_6$.

Écrivez la fonction `moindres_carres`, appelée par le script `exercice_2`, visant à résoudre en moindres carrés le système linéaire homogène (6), complété par la contrainte $\alpha + \gamma = 1$.

Vérifiez que le score obtenu par `exercice_2` est maximal, c'est-à-dire égal à 1, lorsque l'écart-type $\sigma$ du bruit sur les données d'apprentissage est égal à 0. Vérifiez également que cela n'est pas le cas pour le script `exercice_1`.

En fait, la précision de l'estimation par le maximum de vraisemblance dépend de la valeur de `nb_tirages`. Néanmoins, pour une valeur de `nb_tirages` induisant un temps de calcul « raisonnable », cette estimation est généralement moins précise que celle en moindres carrés. Vous êtes en droit de vous demander l'utilité de cette estimation. Vous allez voir qu'elle permet de contribuer à la résolution d'un problème apparemment proche, mais plus compliqué, à savoir l'estimation des paramètres d'une paire d'ellipses.

## Exercice 3 : estimation des paramètres d'une paire d'ellipses

Le script `donnees_2` affiche deux ellipses dont les paramètres sont tirés aléatoirement, ainsi que $n_{\text{app}}$ points tirés aléatoirement au voisinage de chacune de ces ellipses, qui forment donc un ensemble d'apprentissage $\mathcal{D}_{\text{app}}$ comportant $2\,n_{\text{app}}$ points. Si les données d'apprentissage étaient déjà regroupées en deux classes, c'est-à-dire si les croix étaient affichées en deux couleurs différentes, ce nouveau problème d'estimation ne serait pas plus compliqué que le précédent. Cela n'étant pas le cas, le problème devient nettement plus compliqué, car on ne sait pas quels points $P_i$ choisir pour constituer un système linéaire tel que (6).

En revanche, il est facile de généraliser l'estimation par le maximum de vraisemblance des paramètres d'une paire d'ellipses. En effet, nous pouvons modéliser la nouvelle densité de probabilité $f_{\mathbf{p}_1,\mathbf{p}_2}(P_i)$ par un mélange de deux lois normales, à hauteur de proportions $\pi_1$ et $\pi_2$ telles que $\pi_1 + \pi_2 = 1$ :

$$f_{\mathbf{p}_1,\mathbf{p}_2}(P_i) = \frac{\pi_1}{\sigma_1\sqrt{2\pi}} \exp\left\{-\frac{r_{\mathbf{p}_1}(P_i)^2}{2\,\sigma_1{}^2}\right\} + \frac{\pi_2}{\sigma_2\sqrt{2\pi}} \exp\left\{-\frac{r_{\mathbf{p}_2}(P_i)^2}{2\,\sigma_2{}^2}\right\} \tag{7}$$

Nous continuons de supposer, pour simplifier, que les écarts-types $\sigma_1$ et $\sigma_2$ sont connus et égaux à l'écart-type du bruit sur les données d'apprentissage. D'autre part, nous choisissons $\pi_1 = \pi_2 = 0,5$ car les données d'apprentissage ont été produites dans ces proportions. La maximisation de la log-vraisemblance $\ln L_{\mathbf{p}_1,\mathbf{p}_2}(\mathcal{D}_{\text{app}})$ s'écrit, d'après (1) et (7) :

$$\max_{\mathbf{p}_1,\mathbf{p}_2} \left\{ \ln \prod_{i=1}^{n_{\text{app}}} f_{\mathbf{p}_1,\mathbf{p}_2}(P_i) \right\} = \max_{\mathbf{p}_1,\mathbf{p}_2} \left\{ \sum_{i=1}^{n_{\text{app}}} \ln\left[\frac{\pi_1}{\sigma_1} \exp\left\{-\frac{r_{\mathbf{p}_1}(P_i)^2}{2\,\sigma_1{}^2}\right\} + \frac{\pi_2}{\sigma_2} \exp\left\{-\frac{r_{\mathbf{p}_2}(P_i)^2}{2\,\sigma_2{}^2}\right\}\right] \right\} \tag{8}$$

Faites une copie de la fonction `max_vraisemblance`, de nom `max_vraisemblance_2`, que vous modifierez de manière à permettre au script `exercice_3` de résoudre le problème (8). À moins de fixer `nb_tirages` à une valeur tellement élevée que cela rendrait le temps de calcul prohibitif, vous constatez que les résultats sont décevants. Néanmoins, une fois le problème (8) résolu, on peut séparer les données en deux classes $k = 1$ et $k = 2$, en associant chaque donnée d'apprentissage $P_i \in \mathcal{D}_{\text{app}}$ à l'ellipse la plus proche, ce qui revient à calculer :

$$\widehat{k}(P_i) = \underset{k=1,2}{\arg\max} \left\{ \frac{\pi_k}{\sigma_k} \exp\left\{ -\frac{r_{\mathbf{P}_k}(P_i)^2}{2\,\sigma_k{}^2} \right\} \right\} \tag{9}$$

Écrivez la fonction `probabilites`, appelée par `exercice_3`, qui calcule les deux valeurs de l'objectif de (9), pour chaque point $P_i \in \mathcal{D}_{\text{app}}$, et retourne ces valeurs dans une matrice de taille $2 \times n_{\text{app}}$. Une fois la partition effectuée, le script `exercice_3` estime en moindres carrés les paramètres de l'ellipse associée à chaque classe, à l'aide la fonction `moindres_carres` de l'exercice 1. Vous constatez que la nouvelle paire d'ellipses estimées « colle » mieux aux données que la paire d'ellipses estimées par le maximum de vraisemblance, ce que confirment les scores. Or, cette estimation peut être encore grandement améliorée en utilisant l'algorithme EM.

## Exercice 4 : estimation par l'algorithme EM

La figure du milieu affichée par le script `exercice_3` indique quel mélange de lois maximise la vraisemblance, parmi un ensemble fini de lois de mélange tirées aléatoirement. Or, la probabilité de « tomber pile » sur les paramètres optimaux est quasiment nulle. En revanche, les ellipses de la figure de droite sont généralement plus proches des données d'apprentissage que celles de la figure du milieu. Une idée consiste donc à « boucler », c'est-à-dire à utiliser les ellipses de la figure de droite pour définir un nouveau mélange de lois et mettre à jour la partition. L'algorithme EM (Espérance-Maximisation) est un algorithme très général. Il s'inspire de cette idée, à ceci près qu'il n'effectue pas une partition stricte des données. Les paramètres à estimer sont initialisés par le script `exercice_3`, puis l'algorithme EM répète en boucle les deux étapes suivantes :

- **Étape E** – Calcul des probabilités d'appartenance des points d'apprentissage $P_i \in \mathcal{D}_{\text{app}}$ aux deux classes :

$$\mathcal{P}_k(P_i) = \frac{\frac{\pi_k}{\sigma_k} \exp\left\{ -\frac{r_{\mathbf{P}_k}(P_i)^2}{2\,\sigma_k{}^2} \right\}}{\frac{\pi_1}{\sigma_1} \exp\left\{ -\frac{r_{\mathbf{P}_1}(P_i)^2}{2\,\sigma_1{}^2} \right\} + \frac{\pi_2}{\sigma_2} \exp\left\{ -\frac{r_{\mathbf{P}_2}(P_i)^2}{2\,\sigma_2{}^2} \right\}}, \qquad k = 1,2 \tag{10}$$

- **Étape M** – Mise à jour des proportions du mélange :

$$\pi_k = \frac{1}{n_{\text{app}}} \sum_{i=1}^{n_{\text{app}}} \mathcal{P}_k(P_i), \qquad k = 1,2 \tag{11}$$

  et estimation des paramètres des deux ellipses par résolution en *moindres carrés pondérés* de deux systèmes linéaires comportant chacun $n_{\text{app}}$ équations du type :

$$\mathcal{P}_k(P_i) \times \left( \alpha\, x_i^2 + \beta\, x_i\, y_i + \gamma\, y_i^2 + \delta\, x_i + \epsilon\, y_i + \phi \right) = 0 \tag{12}$$

Faites une copie de la fonction `probabilites`, de nom `probabilites_EM`, et une copie de la fonction `moindres_carres`, de nom `moindres_carres_ponderes`, que vous modifierez de manière à réaliser, respectivement, l'étape E et la deuxième partie de l'étape M de cet algorithme.