TP2 – Estimations jointes de deux droites de régression

Afin que vous n'ayez qu'un seul fichier à rendre pour ce TP, au lieu de créer un fichier pour chaque fonction que vous aurez à écrire, un fichier nommé `fonctions_TP2_stat` vous est fourni pour que vous complétiez les différentes fonctions qui y sont présentes.

Lancez le script `donnees` qui génère et affiche deux droites ainsi que des points P_i autour de cette dernière, simulant du bruit sur les données. À partir de ces données, on va chercher à estimer les paramètres de ces deux droites de trois manières différentes à partir de l'équation paramétrique. Ces estimations seront effectuée par :

- le maximum de vraisemblance dans l'exercice 2,
- les moindres carrés dans l'exercice 3,
- l'algorithme EM (Espérance-Maximisation) dans l'exercice 4.

Exercice 1 : modèles issus de l'estimation d'une droite

Dans un premier temps on va constater les limites des estimateurs du TP1 de Statistiques lorsqu'il s'agit de généraliser à deux droites. Pour cela, recopiez les corps des premières fonctions que vous aviez écrites dans le TP1, à savoir `centrage_des_donnees`, `tirages_aleatoires_uniformes`, `estimation_Dyx_MV` et `estimation_Dyx_MC`. La fonction de tirages aléatoires comporte une sortie supplémentaire par rapport au TP1 qu'il faut laisser à 0 pour le moment (elle sera utile pour l'exercice 2). Une fois cela fait, lancez le script `exercice_1` pour constater que les estimations qui fonctionnaient précédemment ne sont plus bonnes pour le cas de deux droites.

Exercice 2 : estimation par le maximum de vraisemblance

Si les points étaient déjà « partitionnés », 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 problème précédent. Cela n'étant pas le cas, le problème devient nettement plus compliqué, car on ne sait quels points P_i choisir pour chaque droite.

En revanche, il est facile de généraliser l'estimation par le maximum de vraisemblance des paramètres (a_1, b_1) et (a_2, b_2) de deux droites. En effet, nous pouvons modéliser la nouvelle densité de probabilité par un mélange de deux lois normales, avec des proportions π_1 et π_2 telles que $\pi_1 + \pi_2 = 1$:

$$f_{(\sigma_1,a_1,b_1,\sigma_2,a_2,b_2)}(P_i) = \frac{\pi_1}{\sigma_1\sqrt{2\pi}} \exp\left\{-\frac{r_{(a_1,b_1)}(P_i)^2}{2\,{\sigma_1}^2}\right\} + \frac{\pi_2}{\sigma_2\sqrt{2\pi}} \exp\left\{-\frac{r_{(a_2,b_2)}(P_i)^2}{2\,{\sigma_2}^2}\right\} \tag{1}$$

Nous supposons, pour simplifier, que les écarts-types σ_1 et σ_2 sont connus et égaux à l'écart-type σ du bruit pour créer les points. D'autre part, nous choisissons les proportions π_1 et π_2 égales à $1/2$ car les points ont été produits, à partir des deux droites, dans ces mêmes proportions. Dans ce cas, la maximisation de la log-vraisemblance s'écrit alors :

$$\underset{(a_1,b_1,a_2,b_2)\in\mathbb{R}^4}{\arg\max} \left\{\ln \prod_{i=1}^{n} f_{(a_1,b_1,a_2,b_2)}(P_i)\right\} = \underset{(a_1,b_1,a_2,b_2)\in\mathbb{R}^4}{\arg\max} \left\{\sum_{i=1}^{n} \ln\left[\exp\left\{-\frac{r_{(a_1,b_1)}(P_i)^2}{2\,\sigma^2}\right\} + \exp\left\{-\frac{r_{(a_2,b_2)}(P_i)^2}{2\,\sigma^2}\right\}\right]\right\} \tag{2}$$

Il n'est pas possible de simplifier davantage par rapport au cas à une droite. D'autre part, pour résoudre (2), il n'est plus possible d'utiliser le centre de gravité des points comme faisant partie de la droite, vu qu'elles sont deux. C'est pourquoi, en plus des tirages aléatoires des angles ψ, il va aussi falloir faire des tirages aléatoires de centres de gravité G entre -20 et 20 pour chacune des coordonnées (à noter que n'importe quel point sur la droite fait l'affaire en vérité). Le résidu reste alors sous la forme :

$$r_{(a,b)}(P_i) = y_i - a\,x_i - b = (y_i - y_G) - \tan\psi\,(x_i - x_G) \tag{3}$$

Complétez la fonction `tirages_aleatoires_uniformes` afin d'y ajouter les tirages des centres de gravité G. Ecrivez ensuite la fonction `estimation_Dyx_MV_2droites` qui prend deux jeux de tirages aléatoires en entrée afin d'estimer l'équation des deux droites par résolution de (2). En lançant le script `exercice_2`, vous constaterez que, à moins de fixer `n_tirages` à une valeur tellement élevée que cela rendrait le temps de calcul prohibitif, les résultats sont décevants.

Exercice 3 : estimation en moindres carrés

Néanmoins, une fois le problème (2) résolu, on peut partitionner les données en deux classes $k = 1$ et $k = 2$, en associant chaque point P_i à la droite estimée la plus proche, ce qui revient à calculer :

$$\widehat{k}(P_i) = \underset{k=1,2}{\arg\max} \left\{ \pi_k \, \exp\left\{ -\frac{r_{(a_k,b_k)}(P_i)^2}{2\,\sigma^2} \right\} \right\} \tag{4}$$

Écrivez la fonction `probabilites_classe`, appelée par `exercice_3` qui calcule les probabilités de chaque point P_i d'appartenir aux droites à partir de (4), et retourne ces valeurs dans deux vecteurs (un pour chaque droite). Complétez ensuite la fonction `classification_points` pour partitionner les points en deux classes. Le script `exercice_3` estime par la suite en moindres carrés les paramètres de la droite associée à chaque classe, à l'aide la fonction `estimation_Dyx_MC` de l'exercice 1. En lançant le script `exercice_3`, vous constatez que les deux nouvelles droites estimées « collent » mieux aux données que celles estimées par le maximum de vraisemblance. Or, cette estimation peut encore être améliorée en utilisant l'algorithme EM.

Exercice 4 : estimation par l'algorithme EM

L'estimation par maximum de vraisemblance indique quel mélange de lois maximise la vraisemblance, parmi un ensemble fini de lois de mélange tirées aléatoirement. Cependant, la probabilité de « tomber pile » sur les paramètres optimaux est quasiment nulle. En revanche, les droites estimées par les moindres carrés (à partir des classes issues du maximum de vraisemblance) 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 nouvelles droites estimées pour définir un nouveau mélange de lois et mettre à jour la partition. L'algorithme EM (Espérance-Maximisation) 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 les exercices précédents, puis l'algorithme EM répète en boucle les trois étapes suivantes :

- **Étape E** – Calcul des probabilités d'appartenance aux deux classes des points d'apprentissage P_i :

$$\mathbb{P}_k(P_i) = \frac{\pi_k \, \exp\left\{ -\frac{r_{(a_k,b_k)}(P_i)^2}{2\,\sigma^2} \right\}}{\pi_1 \, \exp\left\{ -\frac{r_{(a_1,b_1)}(P_i)^2}{2\,\sigma^2} \right\} + \pi_2 \, \exp\left\{ -\frac{r_{(a_2,b_2)}(P_i)^2}{2\,\sigma^2} \right\}}, \qquad k = 1, 2 \tag{5}$$

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

$$\pi_k = \frac{1}{n} \sum_{i=1}^{n} \mathbb{P}_k(P_i), \qquad k = 1, 2 \tag{6}$$

- Estimation des paramètres des deux droites par résolution en moindres carrés pondérés de deux systèmes linéaires comportant chacun les n équations du type :

$$\mathbb{P}_k(P_i) \times (a_k \, x_i + b_k) = \mathbb{P}_k(P_i) \times y_i \tag{7}$$

Les étapes **E** et **M** ont déjà été effectuées. Complétez la fonction `estimation_Dyx_MCP` afin de réaliser l'estimation en moindres carrés pondérées à l'aide de (7). Complétez ensuite la fonction `iteration_estimation_Dyx_EM` qui réalise une itération de chacune des trois étapes de l'algorithme mentionnées ci-dessus. Lancez enfin le script `exercice_4` pour parvenir à trouver de meilleures droites que les méthodes précédentes.