TP 8

ALGORITHME DES k MOYENNES

On appelle «points» des listes Python formées de d flottants et on les identifie à des vecteurs de $\mathbb{R}^d$.
On dispose d'une liste `observations` formée de n points que l'on souhaite répartir en k groupes de la façon la plus satisfaisante possible.

▷ Récupérer le fichier Python sur cahier prépa. Ce fichier contient les fonctions suivantes :

- `creerDonnees` engendre de façon aléatoire des données bidimensionnelles (dans $[0,1]^2$) réparties en groupes. Nous partons donc de données «artificielles» pour visualiser et tester la qualité de l'algorithme. On pourrait également partir de données concrètes en dimension d quelconque ;
- `afficheCentres` interface graphique appelée pour marquer des points particuliers : «centres» successifs calculés par l'algorithme ;
- `afficheClasses` interface graphique appelée pour marquer par des couleurs les nuages de données et leur classe de rattachement ;
- `kMoyAfficheEtapes` et `kMoy` ne sont que partiellement renseignées, pour appeler l'interface graphique et disposer correctement des «subplots» à l'écran, c'est là que vous rédigerez l'algorithme des k moyennes.

Il y a des lignes de code introduites par `#` qui sont à décommenter pour les exécuter le moment venu.
Enfin la ligne de code suivante initialise les données de référence :

```
centres, observations, groupe = creerDonnees()
```

Programmer et visualiser les étapes

1. Rédiger le code de `distCarre(X1,X2)` : étant donnés deux points de $\mathbb{R}^d$, cette fonction calcule et renvoie le carré de la distance euclidienne entre ces deux points.

On rappelle le principe : on dispose d'une liste de k points appelée *centres*, que l'on fait évoluer à travers des étapes successives association/recentrage. On va rédiger les fonctions d'initialisation, association et recentrage.

2. Compléter la fonction `initialisationForgy(observations,k)` qui tire et renvoie un échantillon de k observations prises au hasard parmi les n données (utiliser par exemple `rd.sample`).

3. Rédiger l'étape d'association de l'algorithme : on répartit les observations pour les rattacher chacune au centre le plus proche. La réponse renvoyée est une liste de n entiers qui donne, pour chaque observation, l'indice du centre dont elle est le plus proche.

4. Rédiger l'étape de recentrage `newCentres` : on calcule et on renvoie une liste de k nouveaux centres. Pour le calcul, pour chaque indice j on calcule l'isobarycentre des points associés au centre numéro j : c'est là le nouveau centre numéro j.

5. Compléter la fonction `kMoyAfficheEtapes` qui applique l'algorithme sur 6 étapes (sans regarder donc de condition d'arrêt) en faisant appel aux fonctions précédentes.

6. Tester en décommentant la ligne `reponse1=kMoyAfficheEtapes(observations,5)`

Finalisation : condition d'arrêt

7. Compléter la fonction `listesIdentiques` : on lui fournit deux listes d'entiers de même longueur et elle renvoie `True` si et seulement si les deux listes ont le même contenu.

8. Rédiger la fonction `kMoy` qui implémente l'algorithme des k-moyennes avec son test d'arrêt ordinaire, lorsque les affectations n'évoluent plus.

Les fonctions d'affichage ont été renseignées, respecter la demande pour les zones à compléter.

Faire quelques tests. L'affichage a été prévu pour faire apparaître l'influence du choix des premiers centres. Commenter.