— Mise en œuvre de la classification hiérarchique ascendante (CHA) et K-Means sur des données synthétiques et réelles

Objectif

Le but du TP est de mettre en oeuvre la classification hiérarchique ascendante (CHA) et la méthode de K-Means et de comparer les résultats de ces deux approches.

1 CHA

Pour illustrer les ultra-métriques *single linkage* et *complete linkage*, on va se baser sur des données synthétiques en 2D. Ces données sont générées selon des distributions gaussiennes $\mathcal{N}(\mu_1, \Sigma)$ pour la première classe et $\mathcal{N}(\mu_2, \Sigma)$ pour la 2e (voir sur Moodle le script `gendata.m` qui utilise la fonction `mvnrnd`).

1. Télécharger l'archive `excha.zip` sur Moodle. Le programme `scriptcha_cha.m` illustre l'application de la CHA sur les données synthétiques. Expliquer brièvement comment fonctionne la fonction `aggclust.m`
2. Faites tourner le script. Que constatez-vous ?
3. Tester maintenant votre programme sur les données `ds2.dat` et `george.dat`. Représenter les clusters obtenus et commenter. Le nombre de clusters est laissé à votre libre choix.

ds2.dat

george.dat

Remarque :

— La structure `level` contient les informations sur les clusters à différents niveaux de l'arbre. Ainsi `level(N-1)` fait référence à une solution à 2 clusters. Les indices des points qui sont dans le cluster j peuvent être récupérés via `level(N-1).cluster{j}`.

— les fonctions `aggclust` et `distance` pouvant être lentes, vous pouvez sous-échantillonner les données (voir la fonction `mydownsampling.m`) sur Moodle.

2 K-Means

2.1 Codage

1. Ecrire une fonction `[clusters, Jw] = affectation(X, C, K)`

   Entrées : − matrice des données $X = \begin{pmatrix} x_1^\top \\ \vdots \\ x_N^\top \end{pmatrix} \in \mathbb{R}^{N \times d}$ avec $x_i \in \mathbb{R}^d$

   − les K centres de gravité dans la matrice $C = \begin{pmatrix} \mu_1^\top \\ \vdots \\ \mu_K^\top \end{pmatrix} \in \mathbb{R}^{K \times d}$ avec $\mu_k \in \mathbb{R}^d$.

   Sorties : − la liste des clusters auxquels appartiennent les N points
   − le critère d'inertie intra-classe $J_w = \sum_{k=1}^{K} \sum_{i \in \mathcal{C}_k} \|x_i - \mu_k\|^2$.

   Le principe de cette fonction est résumé par l'algorithme suivant

```
[clusters, Jw] ← affectation(X, C, K)
N ← size(X, 1)
Jw = 0
Pour i = 1...N faire
    Pour k = 1...K faire
        distance(k) ← ||X(i,:) − C(k,:)||²
    Fin Pour
    [val, indice] = min(distance)
    clusters(i) ← indice
    Jw ← Jw + val
Fin Pour
```

2. Ecrire une fonction `C = nouveaux_centres(X, clusters)` qui calcule les nouveaux centres des clusters à partir des données X et de leur affectation aux clusters.

3. Ecrire une fonction `[C, clusters, JwIter] = Kmoyennes(X, K, C0)` qui part de K centres C_0 initialisés aléatoirement et retourne les centres des clusters découverts, l'affectation des données à ces clusters et les valeurs de J_w au fil des itérations. Le critère d'arrêt peut être un nombre maximal d'itérations atteint ou la variation du critère Jw.

4. Tester votre programme K-means sur les données synthétiques (voir script `gendata.m`). Vérifier que J_w décroit de façon monotone. Représenter les clusters.

2.2 Comparaison n'est pas raison

1. Appliquer votre programme aux données `ds2.dat` et `george.dat` et comparer les résultats obtenus par rapport à CHA.
2. Tester l'algorithme K-means sur `george.dat` pour plusieurs initialisations différentes. Que constate-t-on ?

2.3 C'est chaud mais c'est fun

On considère l'image 128×128 `bird_small.tiff` disponible sur Moodle. Chaque pixel est caractérisé par les valeurs RGB. L'objectif est de réaliser le clustering de ces pixels dans l'espace des couleurs. Le script `birdie.m` vous est fourni et est à compléter.

1. Télécharger le script et compléter le pour réaliser le clustering des pixels. Le nombre de clusters est à votre choix. Visualiser les couleurs encodées par les centres obtenus (voir dans le script)
2. Connaissant les centres, on veut encoder l'image `bird_large.tiff` en remplaçant la valeur de chaque pixel par le centre du cluster le plus proche. Compléter alors le script.
3. Visualiser le résultat pour différentes valeurs de K. Commenter les résultats obtenus. Quel est le taux de compression obtenu en faisant le clustering ?