République Algérienne Démocratique et Populaire

Ministère de L'Enseignement Supérieur et de la Recherche Scientifique

Université 20 Août 1955 Skikda

Faculté des Sciences

Département d'Informatique

Mémoire de fin d'études en vue de l'obtention du diplôme De

Master

Option : Génie logiciel avancé et applications (GLAA)

Thème

Le clustering des pays basé sur des données de COVID-19

Réalisé par:

Zendouh Abdelhamid

Hamdi Nouh Abderraouf

Encadré par :

Dr.Ramdane Chafika

Session : Juin 2022

REMERCIEMENT

Nos remerciements vont en premier lieu à Allah le tout puissant et

Miséricordieux, qui nous a donné la force, le courage et la patience d'accomplir ce travail.

Nous tiendrons à remercier très chaleureusement Dr.Ramdane qui nous a permis de bénéficier de son encadrement. Ces conseils, Son œil critique, sa patience et son dévouement ont été déterminants dans la réalisation de ce modeste travail. Nos remerciements s'étendent également à tous nos enseignants qui ont contribué à notre formation tout au long de nos années d'étude.

Enfin, nous tiendrons à remercier tous ceux qui, de près ou de loin, ont contribué à la réalisation ce travail.

DEDICACE

A la mémoire de tous nos chers qui sont partie et sont sous terre

A nos parents les plus chères personnes aux mondes

Atout membre de nos familles soit femmes, fils, frères et sœurs...

Aucun langage ne saurait exprimer nos respects et nos considérations pour votre soutien et encouragements pour accomplir ce travail

A tous nos amis de la promotion

Résumé :

Dans ce travail de master, nous traitons le problème du clustering des données, qui est considéré comme une tâche fondamentale dans de nombreux domaines différents. Il est également modélisé comme un problème d'optimisation qui aide à comprendre les données analysées et pour lesquelles nous avons décidé de choisir l'algorithme de recherche de coucou.

Nous avons d'abord appliqué l'algorithme à un ensemble de données synthétiques et réelles. Ensuite, nous avons passé à son adaptation aux données épidémiologiques sur la COVID19 concernant différents critères caractérisant les pays.

Des expérimentations, des tests et des analyses de résultats ont été présenté pour enrichir le travail.

Mots-clés : clustering de données, recherche de coucou, métaheuristique, optimisation, Covid19.

Abstract :

In this master thesis, we deal with the problem of data clustering, which is considered a fundamental task in many different fields. It is also modeled as an optimization problem which helps to understand the analyzed data and for which we decided to choose the cuckoo search algorithm. We first applied the algorithm to a set of synthetic and real data. Then, we passed to its adaptation to the epidemiological data on COVID19 concerning different criteria characterizing the countries. Experiments, tests and analyzes of results were presented to enrich the work.

Keywords : data clustering, cuckoo search, metaheuristic, optimization, covid19.

Liste des matières

Chapitre 2 : La Recherche de Coucou

Chapitre 3 : La Conception de l'application

Chapitre 4 : Implémentation de l'approche

Chapitre 5 : Expérimentation : tests, résultats et analyse

Liste des Figures

Liste des Tableaux

Introduction Générale

Introduction générale

Dans la vie quotidienne, les humains font recours au regroupement comme l'une des activités mentales les plus primitives de l'organisation des données qu'ils reçoivent chaque jour, afin qu'ils puissent en tirer des conclusions importantes. Évidement, le clustering est un sujet de recherche active en analyse et en fouille de données , en statistiques, traitement d'image, intelligence artificielle, reconnaissance de formes, l'analyse du web, le marketing, le diagnostic médical, la biologie et beaucoup d'autres, il fait partie de tout un processus d'analyse exploratoire. Il consiste à diviser les données en un ensemble de groupes ou clusters. Ces clusters trouvés peuvent être étudiés et analysés pour réduire les données en un ensemble de représentants moins nombreux permettant une représentation simplifiée des données initiales.

A nos jours, il y a un effort mondial de la communauté de recherche pour explorer l'impact médical, économique et sociologique de la pandémie de COVID-19. De nombreuses disciplines différentes tentent de trouver des solutions et de conduire des stratégies à une grande variété de différents problèmes très cruciaux.

Dans ce sens, nous présentons une nouvelle analyse qui permet de regrouper les pays en utilisant des données épidémiologiques de Johns Hopkins en prenant en compte les cas déclarés, les décès, les personnes vaccinées et totalement vaccinées enregistrés dans la période allant du 04 avril 2020 au 04 avril 2022.

Pour ce faire, nous allons opter pour la modélisation de clustering comme un problème d'optimisation et sa résolution avec l'algorithme de la recherche de coucous inspiré du mode de reproduction de certaines espèces de coucous.

A la fin, nous présentons des expérimentations intensives, des tests, des résultats et des analyses qui pourraient être utiles à une variété de décideurs différents, tels que les médecins et les gestionnaires du secteur de la santé, les experts en économie/finance, les politiciens et même les sociologues.

Par conséquent, ce mémoire est reparti en cinq chapitres

Chapitre 1 :

Dans ce chapitre, nous présentons le domaine général de clustering des données et ces différentes techniques

Chapitre 2 :

Dans ce chapitre, nous donnons des idées sur la méthode de la recherche de coucou, en présentant l'habitat, le comportement de l'oiseau coucou, et aussi plus principalement les étapes de l'algorithme et son déroulement.

Chapitre 3 :

Dans ce chapitre, nous passons à la conception de notre système, où nous présentons la majorité de modules de ce système.

Chapitre 4 :

Le quatrième chapitre présente le déroulement et les différentes parties de notre application et les différents outils et l'environnement de développement utilisés.

Chapitre 5 :

Dans le dernier chapitre, nous présentons les différentes expérimentations, tests et résultats ainsi qu'une analyse détaillée des résultats.

Nous terminons par une conclusion.

Chapitre 1

Clustering des données

1.1. Introduction

La classification dans l'analyse de données sert a regrouper les données en classes homogènes. Dans lequel il y a deux types d'approches : supervisée et non supervisée (clustering)

La classification supervisée est basée sur un algorithme qui cherche des règles de classification pour pouvoir faire la prédiction la classe d'appartenance d'un objet dans un ensemble d'objets de classes prédéfinis. Par contre les classes de la classification non supervisée ne sont pas prédéfinies.

Dans ce chapitre nous allons voir les différents techniques et base du clustering de données.

1.2. Définition de clustering

Le clustering est généralement défini comme la tâche qui consiste à trouver des groupes naturels dans un ensemble de données multidimensionnelles. Le regroupement est fait tel que les données dans le même cluster ou groupe sont plus similaires que celles dans des clusters différents. Ainsi qu'il est une tâche d'apprentissage "non supervisée" car on ne dispose d'aucune autre information préalable que la description des données ce qui implique que les classes possibles ne sont pas connues à l'avance. [1]

1.3. Définition d'un cluster

La définition de ce que constitue un cluster n'est pas bien défini et le terme, cluster n'a pas de définition précise [2] [3]. Cependant, plusieurs définitions d'un cluster sont généralement utilisées.

A. Définition de cluster bien séparé

Un cluster est un ensemble de points tel que n'importe quel point dans le cluster est plus proche (ou plus similaire) de chaque autre point dans le cluster que de n'importe quel point qui n'est pas dans le cluster. Parfois un seuil est employé pour spécifier que tous les points dans un cluster doivent être suffisamment proches (ou similaires) l'un de l'autre (figure 1.1).

Figure 1.1 : Trois clusters bien séparés

B. Définition de cluster basé sur le centre

Un cluster est un ensemble de points tel qu'un point dans un cluster est plus proche (plus similaire) du " centre" de ce cluster, que du centre de n'importe quel autre cluster. Le centre d'un cluster est souvent un centroïde, la moyenne de tous les points dans le cluster, ou un médoïde, le point le plus représentatif d'un cluster, cependant beaucoup d'algorithmes de clustering utilisent cette définition (figure 1.2).

Figure 1.2 : Quatre clusters basés sur le centre

C. Définition de Cluster contiguë (le voisin le plus proche ou le clustering transitif):

Un cluster est un ensemble de points tel qu'un point dans un cluster est plus proche (ou plus similaire) d'un ou de plusieurs autres points dans le cluster que de n'importe quel point qui n'est pas dans le cluster (figure 1.3).

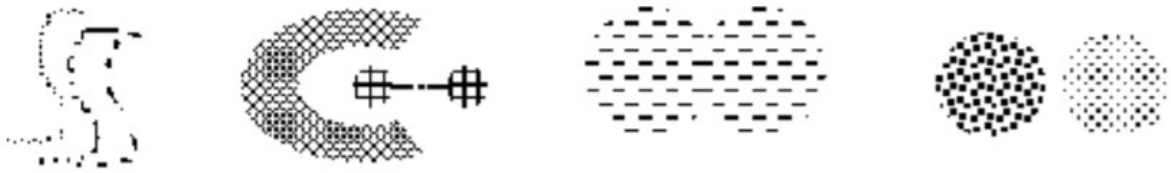


Figure 1.3 : Huit clusters contigus

D. Définition basée sur la densité

Un cluster est une région dense de points, qui est séparée des autres régions de haute densité par des régions de basse densité. Cette définition est souvent utilisée quand les clusters sont irréguliers ou entrelacés et quand les bruits sont présents (figure 1.4).

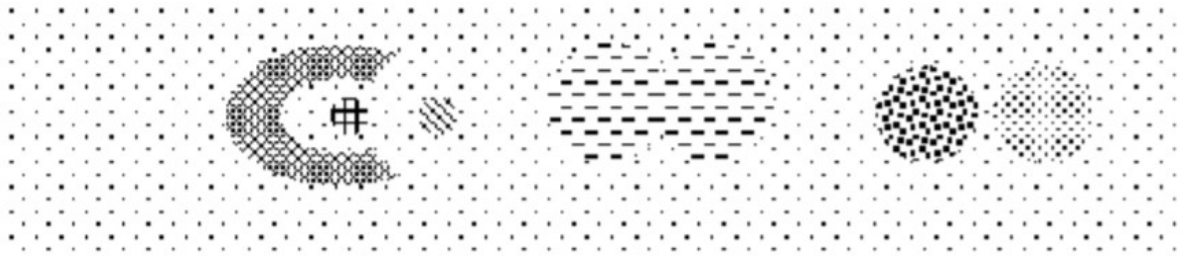


Figure 1.4 : Six clusters denses

1.4.Les concepts de clustering

1.4.1. La matrice de données

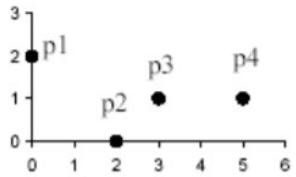
Les objets (échantillons, mesures, modèles, événements) sont habituellement représentés comme des points (vecteurs) dans un espace multidimensionnel, où chaque dimension représente un attribut distinct (variable, mesure) décrivant l'objet.

Ainsi, un ensemble d'objets est représenté comme une matrice m×n , avec m lignes , une pour chaque objet et n colonnes , une pour chaque attribut. Cette matrice est appelée matrice de données ou jeu de données. La figure 1.5, ci-dessous, fournit un exemple concret de quelques points et leur matrice de données correspondante.

1.4.2. La matrice de proximité

Plusieurs algorithmes de clustering utilisent la matrice de données originale et beaucoup d'autres emploient une matrice de similarité, ou une matrice de dissimilarité. Pour la convenance, les deux matrices sont généralement mentionnées comme une matrice de proximité, *P*. Une matrice de proximité, *P*, est une matrice $m \times m$ contenant toutes les dissimilarités ou les similarités entre les objets considérés. Si p_i et p_j sont le $i^{ème}$ et le $j^{ème}$ objets, respectivement, alors l'entrée à la i^{emme} ligne et la j^{emme} colonne de la matrice de proximité est la similarité, ou la dissimilarité, entre p_i et p_j

Les figures 1.5 montrent, respectivement, quatre points, leur matrice de données et leur matrice proximité correspondante



point	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

	p1	p2	p3	p4
p1	0.000	2.828	3.162	5.099
p2	2.828	0.000	1.414	3.162
p3	3.162	1.414	0.000	2.000
p4	5.099	3.162	2.000	0.000

Figure 1.5 : Quatre points, leur matrice de données et leur matrice de proximité.

1.4.3. Types et échelles de données

Le mesure d'approximation et le type de clustering utilisé dépend du type et de l'échelle d'Attributs de données [4]. Les trois types d'attributs sont montrés dans le tableau 1.1, les différentes échelles de données sont présentées dans le tableau 1.2.

Binaire	Deux valeurs possibles, vrai ou faux
Discret	Nombre de valeurs finie ou d'entiers
Continu	Nombre de valeurs infinie ou de réels

Tableau 1.1 Les différents types d'attributs

Qualitative	Nominal	les valeurs sont juste des noms différents. par exemple : les codes postaux, les couleurs, le sexe.

	Ordinal	les valeurs reflètent un ordre, rien plus. par exemple : bon, meilleur, mieux ou couleurs ordonnées par le spectre.
Quantitative	Intervalle	la différence entre les valeurs est significative par exemple, l'intervalle de température.
	Ratio	rapport entre deux grandeurs. par exemple : les quantités monétaires, comme le salaire et le bénéfice et beaucoup de quantités physiques comme courant électrique, pression, etc.

Tableau 1.2 Les différentes échelles de données

1.4.4. Distance et similarité

La plupart des algorithmes de clustering utilisent une mesure de la proximité des objets entre eux pour traiter. Cette notion de "proximité" est formalisée à l'aide d'une mesure de similarité et de dissimilarité ou encore par une distance.

La similarité permet de mesurer la ressemblance entre objets, elle doit vérifier la positivité ($S_{i,j} \geq 0$) et la symétrie ($S_{i,j} = S_{j,i}$) tel que : $S_{i,j}$ une similarité dans R^p entre deux objets x_i ,x_j . La distance est la mesure la plus utilisée parmi les types de mesures de similarité et de dissimilarité, elle permet de vérifier les propriétés suivantes [5] :

La positivité : $d_{i,j} = d(x_i, x_j) \geq 0$.

La réflexivité : $d_{i,j} = 0 \Leftrightarrow x_i = x_j$.

L'identité : $d_{i,j} = 0$

La symétrie : $d_{i,j} = d_{j,i}$.

L'inégalité triangulaire : $d_{i,k} + d_{k,j} \geq d_{i,j}$,Tel que $d_{i,j}$ une distance dans R^p entre deux objets x_i ,x_j

Les mesures de distance les plus courantes entre deux objets x_i et x_j sont :

Distance Euclidienne $$d(x_i, x_j) = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{n} (x_i - x_j)^2} \quad (1.1)$$

Distance de Hamming $$d(x_i, x_j) = \sum_{i=1}^{n} \sum_{j=1}^{n} |x_i - x_j| \quad (1.2)$$

Distance de chebyschev $$d(x_i, x_j) = max_{\substack{i=1,2\dots n \\ j=1,2\dots n}} |x_i - x_j| \quad (1.3)$$

Distance de Minkowski $$d(x_i, x_j) = \sqrt[p]{\sum_{i=1}^{n}\sum_{j=1}^{n}(x_i - x_j)^2}, P > 0 \quad (1.4)$$

Distance de Canberra $$d(x_i, x_j) = \sum_{i=1}^{n}\sum_{j=1}^{n}\frac{|x_i - x_j|}{x_i + x_j} \begin{matrix} x_j > 0 \\ et \\ x_j > 0 \end{matrix} \quad (1.5)$$

Séparation angulaire $$d(x_i, x_j) = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} x_i x_j}{\left[\sum_{i=1}^{n} x_i^2 \sum_{j=1}^{n} x_j^2\right]^{1/2}} \quad (1.6)$$

1.5. Les différentes techniques de clustering

Plusieurs techniques de clustering sont apparues dans la littérature afin de découvrir des groupes cohésifs. Elles peuvent être classifiées aux types suivants [6].

1.5.1. Clustering par partitionnement

Les techniques par partitionnement créent une décomposition d`un seul niveau des points de données [2], ces méthodes construisent K partitions de données, où chaque partition représente un cluster. Si k est le nombre de clusters souhaité alors les approches par partitionnement trouvent typiquement tous les k clusters immédiatement.

Il existe différents algorithmes à partitionnement, Nous allons décrire les deux algorithmes les plus connus : Kmeans et Kmedoid [5].

- **Kmeans**

  La technique de clustering de Kmeans [6] est basée sur la notion du centroïde qui est le point de la moyenne ou la médiane d'un groupe de points.

  L'algorithme de Kmeans est décrit comme suit :

  1. Choisir K points initiaux qui seront les centres de K groupe (centroïde).
  2. Attribuer chaque point au centroïde le plus proche.
  3. Recalculer le centroïde de chaque groupe.
  4. Répéter les étapes 2 et 3 jusqu'à ce que les centroïdes ne changent pas.

  La figure 1.6 montre les étapes de l'algorithme de Kmeans.

Cet algorithme converge toujours vers une solution, qui est typiquement un minimum local, car la variance à l'intérieur d'un cluster ne peut que décroître ou rester stationnaire entre deux étapes successives [5].

Kmeans a des propriétés importantes tels que : la complexité temporelle et l'efficacité de traitement de grand jeu de données, cependant il souffre de quelque limités et problèmes tels que :

- le résultat dépend fortement des controïdes initiaux.
- le nombre de cluster doit être fixé à l'avance.
- Il se termine souvent à un optimum local.
- Il est sensible au bruit.
- Il ne peut être utilisé que les données numériques.
- Il ne sait gérer des clusters proches dont les tailles sont très différentes, ni des clusters de forme allongée ou concave.

Une solution pour pallier à ces problèmes est de permettre de diviser ou concaténer les clusters résultants, typiquement un cluster est divisé quand sa variance est au-dessus d'un certain seuil pré spécifié, et deux clusters sont concaténés quand la distance entre leurs controïdes est en dessous d'un autre seuil pré spécifié [1].

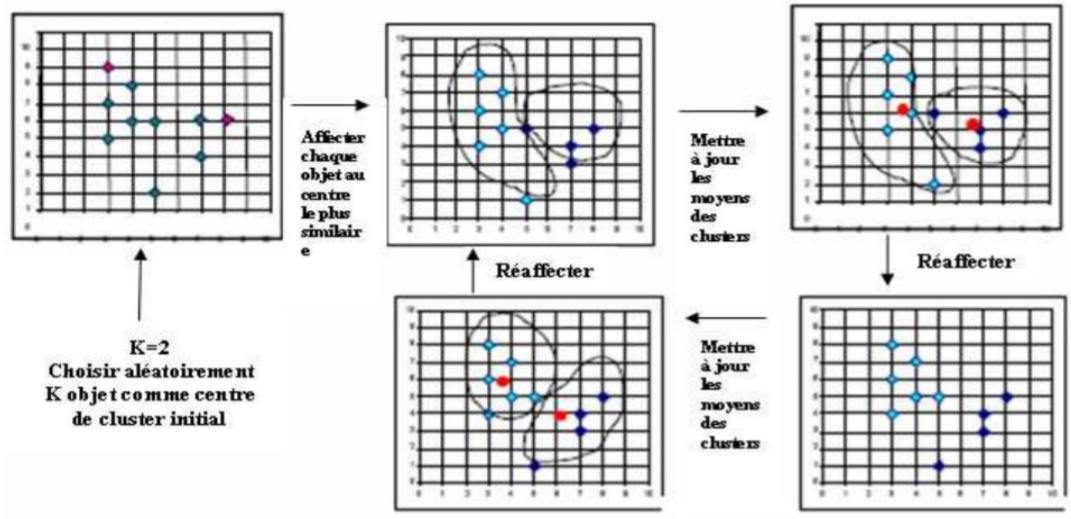


Figure 1.6 Les étapes de l'algorithme de Kmeans.

- **Kmedoid**

La méthode de Kmedoid partitionne l'espace de distance en k clusters. Un cluster est représenté par un médoïde, ce dernier est un point plus représentatif d'un groupe de points, qui est le plus placé au centre en tenant en compte quelques mesures, comme par exemple, la distance.les étapes de Kmedoid sont décrites comme suit [6] :

1. Choisir k points initiaux. Ces points sont les médoïdes candidats qui sont destinés à être les points les plus centraux de leurs clusters.
2. Considérer l'effet de remplacer un des points choisis (médoïdes) avec un des points non choisis. Conceptuellement, ceci est fait de la façon suivante:

On calcule la distance entre chaque point non choisi et le médoïde candidat le plus proche et on calcule la somme de toutes les distances, cette somme représente le "coût" de la configuration actuelle. Tous les échanges possibles d'un point non choisi par un autre choisi sont considérés, et le coût de chaque configuration est calculé.

3. Choisir la configuration avec le coût le plus bas. Si c'est une nouvelle configuration, alors répéter l'étape 2.
4. Sinon, associer chaque point non choisi au point choisi le plus proche (médoïde) et arrêter.

L'un des grands avantages de cette méthode est sa robustesse. L'utilisation des médoïdes pour définir des clusters rend cette méthode très résistante contre les bruits de données, elle est un peu moins sensible au bruit que Kmeans, mais elle a un certain nombre d'inconvénients, tels que [8] :

- Elle ne doit pas stocker une vaste quantité de l'information en plus des données originales dans la mémoire;
- En utilisant des médoïdes, cette méthode ne fournit aucune manière de décrire les clusters, autre que le détail d'adhésion.
- La recherche de médoïdes est plus coûteuse que le simple calcul de centroïdes.

1.5.2. Clustering hiérarchique

Dans le clustering hiérarchique, l'objectif est de créer une série hiérarchique de clusters. cette hiérarchie est également connu sous le nom de dendrogramme ou, en d`autres termes, un arbre de clusters, la racine de cet arbre est formée par le cluster **X** contenant l'ensemble des points de données et chaque nœud constitue un cluster $C_i \subset \boldsymbol{X}$ d'autre part les feuilles de l'arbre correspondent aux singletons $\{x_1\}, \dots \dots, \{x_n\}$.

Le dendrogramme permet une visualisation de l'organisation des données et du processus de clustering car c'est un arbre inversé qui décrit l'ordre dans lequel des points sont fusionnés (vue ascendante) ou les clusters sont dédoublés (vue de haut en bas), il est possible alors d'obtenir une partition de X en coupant l'arbre a un niveau l donné. Par exemple, le choix de l = 4 dans le dendrogramme de la (figure1.7) renvoie le partitionnement suivant :

$C = \{\{x_1, x_2, x_3\}, \{x_4, x_5, x_6\}\}$

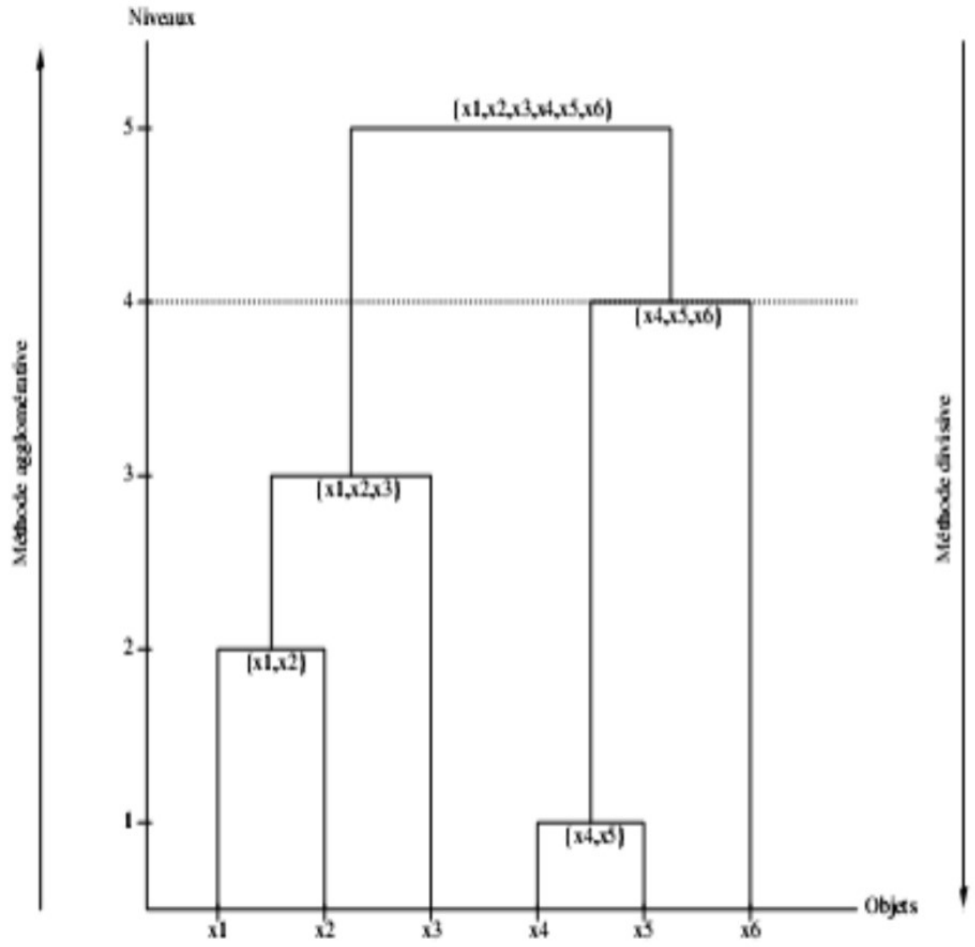


Figure 1.7 Exemple d'un dendrogramme.

Ce dernier paramètre l peut être choisi relativement au nombre de clusters désiré ou à l'aide d'une analyse statistique de la qualité des différentes partitions que l'on peut extraire de l'arbre.

On distingue deux approches de base de clustering hiérarchique pour parvenir à un tel arbre hiérarchique : les algorithmes agglomératifs et divisifs.

A. Algorithmes divisifs

Ces algorithmes commencent par une "racine" qui contient tous les objets, puis continue par divisions successives de chaque nœud jusqu'à obtenir des singletons (clusters d'un seul point).

Brièvement l'algorithme est comme suit: [7]

1. Choisir le cluster contenant la paire d'objets la plus éloignée. C'est le cluster avec le plus grand diamètre.
2. Dans ce cluster, enlever l'objet qui a la plus grande distance moyenne des autres objets. cet objet forme un nouveau cluster de singleton.
3. Pour l'objet h dans le cluster étant dédoublé, calculez la distance moyenne entre ce dernier et le cluster courant; et la distance moyenne entre l'objet et le nouveau cluster. Si la distance au nouveau cluster est inférieure à celle au cluster courant, déplacez l'objet h à un nouveau cluster. Faites une boucle au-dessus de tous les objets dans le cluster.
4. Si aucun objet ne se déplace, et le nombre courant de clusters est plus grand que k, alors aller à l'étape 1. Sinon arrêter.

B. Algorithmes agglomératifs

Un regroupement agglomératif construit l'arbre en partant des "feuilles" (singletons) et procède par fusions successives des plus proches clusters jusqu'a obtenir un unique cluster "racine", contenant l'ensemble des objets.

La méthode agglomérative est décrite comme suit: [7]

1. Considérer chaque objet comme un cluster et calculer la matrice de proximité.
2. Trouver le plus petit élément dans la matrice. Ceci correspond à la paire de clusters qui sont les plus semblables, et fusionnez ces deux clusters (soit i et h).
3. Calculer les distances entre le nouveau cluster formé et les clusters restants. Supprimer la ligne et la colonne du cluster i et recouvrer la ligne et la colonne du cluster h avec les nouvelles valeurs.
4. Si le nombre courant de clusters est plus grand que k, alors aller à l'étape 2. Sinon arrêter. Le clustering hiérarchique agglomératif a un certain nombre de limitations et problèmes tels que [2]:
    - aucune fonction objectif n'est optimisée.
    - Les décisions de fusionnement sont finales

- les bonnes décisions de fusionnement locales peuvent ne pas avoir comme de bons résultats globaux.
- Il a des problèmes avec le bruit, forme des clusters non convexes, et une tendance de diviser de grands clusters.

1.6. Métaheuristique pour le clustering

Le problème de clustering peut être modélisé comme un problème d'optimisation où l'espace de recherche grandit exponentiellement et ne peut pas être parcouru exhaustivement même pour des problèmes de taille moyenne. En effet, le problème de clustering est connu pour être NP-difficile [12].

Résoudre un problème d'optimisation, c'est trouver l'optimum d'une fonction, parmi un nombre fini de choix, souvent très grand, cela se fait par les métaheuristiques qui forment un ensemble de méthodes pour résoudre ces problèmes réputés difficiles. Parmi ces méthodes il existe la méthode de recherche coucou, les algorithmes génétiques, et les algorithmes évolutionnaires, les essaimes de particules et beaucoup d'autres qui ont contribué de façon remarquable pour résoudre le problème de clustering.

1.6.1. Clustering basé sure l'Algorithme génétique

1.6.1.1. Principe de base

La mesure de clustering qui a été adoptée est la somme des distances euclidiennes des points de leurs centres de clusters respectifs. Mathématiquement, la mesure M de clustering pour K clusters C_1 , C_2 ,.. , C_K est donnée par

$$M(C_1, C_2, \dots, C_K) = \sum_{i=1}^{K} \sum_{x_j \in C_i} \|x_j - z_i\|$$

La tâche de l'AG (Algorithme Génétique) est de rechercher les centres de clusters appropriés z_1 , z_2 ,..., z_K de sorte que la mesure de clusters M soit réduite au minimum.

1.6.1.2. Algorithme de clustering AG

Les étapes de base d'AG, sont présentées Figure 1.8.

```
Begin
1.  t=0
2.  initialize population P(t)
3.  compute fitness P(t)
4.  t = t+1
5.  if termination criterion achieved go to step 10
6.  select P(t) from P(t-1)
7.  crossover P(t)
8.  mutate P(t)
9.  go to step 3
10.  Output best and stop
End
```

Figure 1.8 : Étapes de base de l'AG

A. représentation de la chaîne de caractères

Chaque chaîne de caractères est une séquence de nombres réels représentant les centres du K-clusters. Pour un espace N-dimensionnel, la longueur d'un chromosome est N_K^* mots, où les premières positions de N (ou, gènes) représentent les N-dimensions de du premier centre du cluster, les N-positions suivants représentent ceux du deuxième centre du cluster, et ainsi de suite.

B. Initialisation de la population

Les centres du K clusters encodés dans chaque chromosome sont initialisés à K points choisis aléatoirement de l'ensemble de données. Ce processus est répété pour chacun des P chromosomes de la population, où P est la taille de la population.

C. Calcul du fitness

Le processus de calcul du fitness comprend deux phases. Dans la première phase, les clusters sont formés selon les centres codés dans le chromosome considéré. Ceci est fait en assignant chaque point x_i , i=1, 2, n, à l'un des clusters C_j avec le centre z_j tel que

$$\|x_i - z_j\| < \|x_i - z_p\|, p = 1, 2, \dots, K, \text{et } p \neq j$$

Après la mise de clusters, les centres des clusters encodés dans le chromosome sont remplacés par les points moyens des clusters respectives. En d'autres termes, pour le cluster C_i , le nouveau centre z_i^* est calculé comme suit :

$$z_i^* = \frac{1}{n_i} \sum_{x_j \in C_i} x_i \, , i = 1, 2, \dots, K.$$

Ces z_i^* remplacent maintenant les z_i précédents dans le chromosome

D. Sélection

un chromosome est attribué un certain nombre de copies, qui est proportionnelle à son aptitude dans la population, qui vont dans la piscine d'accouplement pour d'autres opérations génétiques. Roulette sélection est une technique commune qui met en œuvre la stratégie de sélection proportionnelle.

E. Croissement

Le croisement est un processus probabiliste qui échange de l'information entre deux chromosomes parents pour générer deux chromosomes enfants. un croisement à point unique avec une probabilité de croisement fixe de μ_c est utilisé. Pour les chromosomes de longueur l, un nombre entier aléatoire, appelé point de croisement, est généré dans la plage [1,*l*-1]. Les parties des chromosomes situées à droite du point de croisement sont échangées pour produire deux descendants

F. Mutation

Chaque chromosome subit une mutation avec une probabilité fixe μm . Pour la représentation binaire des chromosomes, une position de bit (ou gène) est mutée en changeant simplement sa valeur. Puisque nous considérons la représentation en virgule flottante,

nous utilisons la mutation suivante. Un nombre Δ dans l'intervalle [0, 1] est généré avec une distribution uniforme. If the value at a gene position is , after mutation it becomes

$$v \pm 2 * \delta * v, v \neq 0$$

$$v \pm 2 * \delta, v = 0$$

Le signe + ou - se produit avec une probabilité égale. Notez que nous aurions pu mettre en œuvre la mutation comme

$$v \pm \delta * v$$

Cependant, un problème avec cette forme est que si les valeurs à une position particulière dans tous les chromosomes d'une population deviennent positives (ou négatives), alors nous ne serons jamais en mesure de générer un nouveau chromosome ayant une valeur négative (ou positive) à cette position. Afin de surmonter cette limitation, nous avons incorporé un facteur de 2 tout en mettant en œuvre la mutation. D'autres formes comme

$$v \pm (\delta + \varepsilon) * v$$

Où $0 < \varepsilon < 1$ aurait également satisfait notre objectif. On peut noter dans ce contexte que des types similaires d'opérateurs de mutation pour l'encodage réel ont été utilisés principalement dans le domaine des stratégies évolutives [13]

Les processus de calcul de fitness, de sélection, de croisement et de mutation sont exécutés pour un nombre maximum d'itérations. La meilleure chaîne de caractères vue jusqu'à la dernière génération fournit la solution au problème de clustering. Nous avons mis en œuvre l'élitisme à chaque génération en préservant la meilleure chaîne de caractères vue jusqu'à cette génération dans un endroit en dehors de la population. Ainsi à la fin, cet emplacement contient les centres des clusters finals.

1.6.2. Clustering par essaim de particules

Particle Swarm Optimization (PSO) est une méthode utile pour l'optimisation continue de la fonction non linéaire qui simule les comportements sociaux. La méthodologie proposée s'inspire des bandes d'oiseaux, des bancs de poissons et de la théorie de l'essaimage en général, et c'est un algorithme extrêmement efficace mais simple pour optimiser un large éventail de fonctions [14]. L'idée principale de l'algorithme est de maintenir un ensemble de solutions potentielles, à savoir, les particules, où chacun représente une solution à un problème d'optimisation. Rappelant l'idée des troupeaux d'oiseaux, un exemple simple qui décrit l'intuition de l'algorithme est décrit dans [15] et supposons un groupe d'oiseaux, cherchant au hasard de la nourriture dans une zone où il n'y a qu'un seul morceau de nourriture. Tous les oiseaux ne savent pas où se trouve la nourriture, mais ils savent jusqu'où se trouve la nourriture à chaque étape. La stratégie d'PSO est basée sur l'idée que la meilleure façon de trouver la nourriture est de suivre l'oiseau qui est le plus proche de la nourriture.

En revenant au contexte du clustering, nous pouvons définir une solution comme un ensemble de n-coordonnées, où chacune correspond à la position c-dimensionnelle d'un centroïde du cluster. Dans le problème de PSOClustering, il s'ensuit que nous pouvons avoir plus d'une solution possible, dans laquelle chaque n solution se compose de positions de cluster c-dimensionnelles, c.-à-d. les centroïdes du cluster (voir les figures 1.9 et 1.10). Il est important de noter que l'algorithme lui-même peut être utilisé dans n'importe quel espace dimensionnel, même si dans ce travail, seuls les espaces 2D et 3D sont pris en compte à des fins de visualisation. Le but de l'algorithme proposé est alors de trouver la meilleure évaluation d'une fonction de forme physique donnée ou, dans notre cas, la meilleure configuration spatiale des centroïdes. Étant donné que chaque particule représente une position dans l'espace N_d, le but est alors d'ajuster sa position en fonction de

- la meilleure position de la particule trouvée jusqu'à present
- la meilleure position dans le voisinage de cette particule

Pour remplir les énoncés précédents, chaque particule stocke ces valeurs:

- x_i : sa position actuelle

- v_i : sa vitesse courante
- y_i : la meilleure position découverte par la particule

En utilisant la notation ci-dessus, intentionnellement conservée comme dans [16], la position d'une particule est ajustée selon :

$$v_{i,k}(t+1) = wv_{i,k} + c_1 r_{1,k}(t)\left(y_{i,k}(t) - x_{i,k}(t)\right) + c_2 r_{2,k}(t)\left(y(t) - x_{i,k}(t)\right) \quad (1)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (2)$$

Dans l'équation (1), w est appelé poids d'inertie, c_1 et c_2 sont les constantes d'accélération, et les deux $r_{1,j}$ (t) et $r_{2,j}$ (t) sont échantillonnés à partir d'une distribution uniforme U(0, 1). La vitesse de la particule est alors calculée en utilisant les contributions de (1) la vitesse précédente, (2) une composante cognitive liée à sa distance la mieux atteinte, et (3) la composante sociale qui tient compte de la distance la mieux atteinte sur toutes les particules de l'essaim. La meilleure position d'une particule est calculée à l'aide de l'équation triviale (3), qui actualise simplement la meilleure position si la valeur de fitness dans l'étape *i*-time actuelle est inférieure à la valeur de fitness précédente de la particule.

$$y_i(t+1) = \begin{cases} y_i(t) & if \quad f(x_i(t+1)) \geq f(y_i(t)) \\ x_i(t+1) & if \quad f(x_i(t+1)) < f(y_i(t)) \end{cases} \quad (3)$$

Le PSO est habituellement exécuté avec une itération continue de l'équation (1) et de l'équation (2), jusqu'à ce qu'un nombre spécifié d'itérations ait été atteint. Une solution alternative est d'arrêter lorsque les vitesses sont proches de zéro, ce qui signifie que l'algorithme a atteint un minimum dans le processus d'optimisation.

Une fois de plus, il est important de noter que même si dans [16] deux types d'approches PSO sont présentés, respectivement nommés g_{best} et l_{best} où les composantes sociales sont essentiellement limitées soit au voisinage actuel de la particule plutôt que l'essaim entier, dans ce travail, nous nous référons uniquement à la proposition g_{best} la plus basique.

L'équation (4) met en œuvre l'évaluation de la performance PSO à chaque étape de temps, où $|C_{i,j}|$ est le nombre de vecteurs de données appartenant au cluster C_{ij} , z_p est le vecteur des données d'entrée appartenant au cluster C_{ij}, m_j est le centroïde *j*-th de la particule *i*-th dans le cluster C_{ij} , N_c est le nombre de clusters, et il peut être décrit comme suit.

$$J_e = \frac{\sum_{j=1}^{N_c}\left[\sum_{\forall Z \in C_{ij}} d(z_p, m_j)/|C_{i,j}|\right]}{N_c} \qquad (4)$$

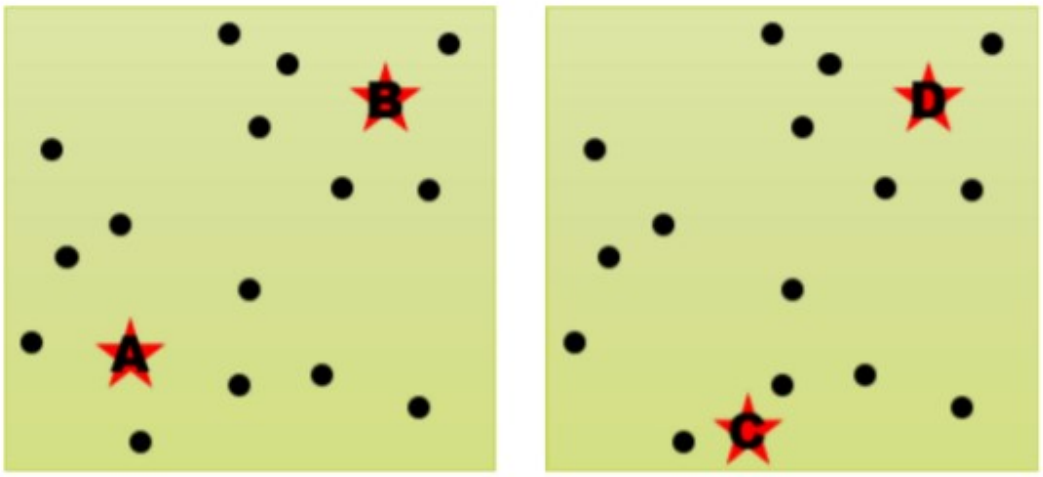


Figure 1.9: Dans cet exemple, nous utilisons deux particules pour regrouper les données en utilisant deux clusters en 2 classes (dimensions). Chaque particule est représentée à l'aide d'un carré vert, dans lequel nous pouvons détecter les deux centroïdes suivis (A et B dans la première particule, C et D dans la deuxième particule). Veuillez noter que les points noirs représentent les données, qui sont les mêmes dans chaque carré vert.

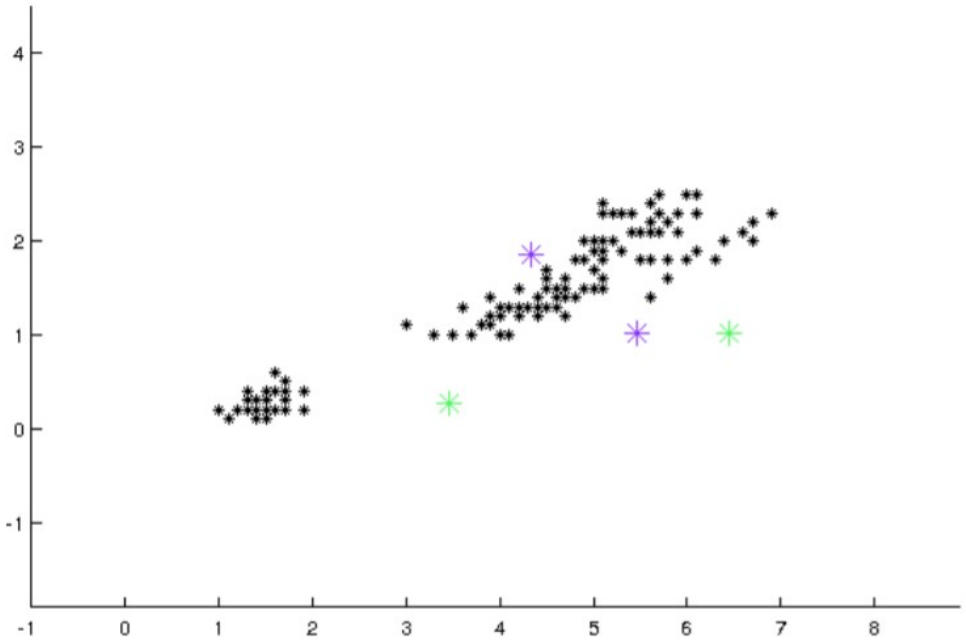


Figure 1.10: L'ensemble de données IRIS initialisé avec deux particules (vert et magenta dans cette image) chacune utilisant deux c-centroïdes.

1.6.3. Clustering par fourmis artificielles

La colonie de fourmis décrite ici suit les grandes lignes des principes couramment utilisés dans ce domaine. Néanmoins, nous introduisons également d'importantes différences liées au problème étudié.

Chaque donnée est un vecteur de n valeurs réelles et est symbolisée par un objet. Initialement, tous les objets sont répartis aléatoirement sur une grille 2D toroïdale et carrée dont la taille est automatiquement mise à l'échelle de la base de données. Pendant l'exécution de l'algorithme, les objets peuvent être empilés sur la même cellule, constituant des tas. Un tas représente donc une classe ou un cluster. La distance entre 2 objets X et Y peut être calculée par la distance euclidienne entre 2 points en R^n. Le centre d'une classe est déterminé par le centre de masse de ses points. Il n'y a pas de lien entre la position d'un objet sur la grille et la n valeur de ses attributs dans R^n.

Un nombre fixe de fourmis se déplacent sur la grille 2D et peuvent effectuer différentes actions. Chaque fourmi se déplace à chaque itération, et peut éventuellement déposer ou ramasser un objet en fonction de son état. Si une fourmi ne porte pas d'objet, elle peut :

- ramasser un seul objet dans une cellule voisine selon une probabilité fixe
- ramasser l'objet le plus différent d'un tas d'une cellule voisine (c'est-à-dire l'objet le plus éloigné du centre de masse du tas).

Si une fourmi porte un objet O, elle peut :

- déposer O sur une cellule vide voisine avec une probabilité fixe,
- déposer O sur un objet unique voisin OI si O et O~ sont suffisamment proches les uns des autres (selon le seuil de dissimilarité exprimé en pourcentage de la dissimilarité maximale dans la base de données),
- Déposer O sur un tas voisin H si O est suffisamment proche du centre de masse de H (à nouveau, selon un autre seuil de dissimilarité).

Initialement cet algorithme basé sur les fourmis est appliqué à la base de données parce qu'il a l'avantage suivant : il ne nécessite aucune information telle que le nombre de classes, ou une partition initiale. La partition créée est cependant composée de trop de clusters (mais qui sont assez homogènes) et avec quelques erreurs de classification évidentes, car on arrête l'algorithme avant convergence qui serait trop long à obtenir.

On utilise donc l'algorithme Kmeans pour supprimer les petites erreurs de classification, et aussi pour assigner des objets "libres", c-à-d. des objets laissés seuls sur le plateau mais aussi des objets encore portés par les fourmis quand l'algorithme s'arrête. Cela élimine vraiment les erreurs de classification, mais puisque les Kmeans sont localement optimaux seulement et que nous lui fournissons trop de clusters, la partition obtenue contient encore trop de clusters mais très homogènes.

Par conséquent, nous avons appliqué à nouveau l'algorithme basé sur les fourmis, mais sur des tas d'objets plutôt que sur des objets uniques : pendant cette deuxième partie, les tas créés précédemment peuvent être ramassés et déposés par les fourmis comme s'ils étaient des objets. Nous utilisons les mêmes algorithmes basés sur les fourmis, mais

adaptés aux tas. Par exemple, on peut définir une distance entre trop de tas comme la distance entre leur centre de masse. Cette partie d'AntClass peut être considérée comme un clustering hiérarchique : les fourmis travaillent d'abord sur les objets, constituant des tas petits mais très homogènes. Puis, travaillant directement sur ces tas comme s'ils étaient des objets, ils construiront hiérarchiquement des classes plus importantes. A la fin de cette étape, le nombre réel de classes est très bien approximatif, mais comme mentionné précédemment, il y a encore quelques tas qui ne sont pas assignés. Par conséquent, nous utilisons une fois de plus l'algorithme Kmeans pour obtenir la partition finale. Mais cette fois, puisque la partition d'entrée donnée aux Kmeans est très proche de la partition "optimale", la sortie est de haute qualité.

Ainsi AntClass se compose principalement en quatre étapes:

- algorithme basé sur colonies de fourmis pour les objets de clustering, suivi par
- les Kmeans algorithme en utilisant la partition initiale fournie par les fourmis, puis
- concentration à base de fourmis, mais sur des tas trouvés précédemment, et enfin
- l'algorithme Kmeans sur les objets une fois de plus.

1.7.Technique de validation de clustering

L'évaluation des résultats d'un clustering est un problème majeur, qui renvoie à la question suivante : qu'est-ce qu'un bon schéma de clustering?

L'objectif principal de la validation de clusters est d'évaluer le résultat de clustering afin de trouver le meilleur partitionnement du jeu de données. Il existe des approches de validité de cluster pour évaluer quantitativement le résultat d'un algorithme de clustering [17] mais dans la plupart des évaluations expérimentales des algorithmes, des jeux de données 2D sont employés pour que le lecteur soit capable de vérifier visuellement la validité des résultats (c-à-d, à quel point l'algorithme de clustering a découvert les clusters du jeu de données). Il est clair que la visualisation du jeu de données est une vérification cruciale des résultats de clustering. Dans le cas de grands jeux de données multidimensionnels (par exemple plus de trois dimensions) la visualisation efficace du jeu de données serait difficile. De plus la

perception de clusters à l'aide des outils de visualisation disponibles est une tâche difficile pour les gens qui ne sont pas habitués aux espaces d'un grand nombre de dimensions [18].

Il est commun de distinguer entre l'évaluation intrinsèque et extrinsèque. Les mesures de qualité externes emploient la connaissance externe. Beaucoup de ces derniers comparent le clustering à une autre partition. Les mesures de qualité internes n'emploient aucune connaissance externe, mais elles sont basées sur ce qui est disponible pour l'algorithme de clustering [19].

Il existe deux critères qui ont été largement considérés suffisants pour mesurer la qualité du partitionnement de données [20].

La compacité, les membres de chaque cluster doivent être proche l'un de l'autre autant que possible. Une mesure commune de compacité est la variance, qu'on doit minimiser. La séparation, les clusters eux-mêmes doivent être largement espacée. La distance euclidienne entre les centroïdes de clusters donne une indication de la séparation de clusters.

1.7.1 Mesures externes

Les mesures de qualité externes emploient la connaissance externe, elles s'agirent de confronter un schéma avec une classification prédéfinie. Ces mesures portent donc sur l'adéquation entre le schéma obtenu et une connaissance "externe" sur les données (schéma attendu) [21]. Les mesures externes que nous allons présenter, appliquent directement la connaissance des étiquettes de classes. Elles évaluent les clusters générés en prenant en compte les classes d'appartenances correctes.

- **La F-mesure :**

La F-mesure est une fonction utilisée souvent dans la littérature pour évaluer les algorithmes de clustering. La F-mesure adopte les idées de la précision et du rappel de la recherche documentaire. Elle compare la qualité de clustering en tenant compte des classes correctes connues pour un jeu de données. Soit $C = (C_1, C_2, \ldots, C_k)$ un clustering donné et $R = (R_1, R_2, \ldots, R_{k'})$ les classes correctes.

Chaque classe R_i contient N_i points de données, chaque cluster C_j (généré par l'algorithme) est considéré comme l'ensemble de N_j points de données. N_{ij} donne le nombre de points de la classe R_i dans le cluster C_j et N donne le nombre total des points du jeu de données. Pour chaque classe R_i et un cluster C_j , la précision et le rappel sont alors défini comme [6] :

$$Prec(R_i, C_j) = \frac{N_{ij}}{N_j} \; et \; Rep(R_i, C_j) = \frac{N_{ij}}{N_i} \qquad (1.15)$$

Et la valeur de F-mesure correspondante est :

$$Fmes(R_i, C_j) = \frac{(b^2+1)\cdot Prec(R_i,C_j)\cdot Rep(R_i,C_j)}{b^2\cdot Prec(R_i,C_j)+Rep(R_i,C_j)} \qquad (1.16)$$

Où des coefficients égaux de Prec(R_i ,C_j) et Rep(R_i ,C_j) sont obtenu si b=1. La valeur globale de F-mesure F pour toute la partition est calculée comme

$$F(C) = \sum_{i=1}^{k'} \max_{C_i \in C}(Fmes(R_i, C_j)) \qquad (1.17)$$

Elle est limitée à l'intervalle [0,1] et devrait être maximale

- **La pureté :**

La pureté de cluster $C_j \in C$ est définie comme le pourcentage du type de données prédominant selon la classe réelle connue $R_i \in R$, qui est :

$$Pur(C_j) = \max_{R_i \in R} \frac{N_{ij}}{N_j} \qquad (1.18)$$

Où N_j est la taille du cluster C_j et N_{ij} est le nombre des points de données de la classe R_i dans ce cluster. La pureté P(C) d'une partition entière est alors calculée comme la pureté moyenne de tous les clusters. Elle est limité à l'intervalle] 0,1] et devrait être maximale [22].

1.7.2 Mesures internes

Les mesures de qualité interne n'utilisent pas de connaissances externes mais uniquement les données d'entrées (matrice de (dis) similarité, descriptions des données etc.) comme référence [23]

Les mesures qui peuvent être appliquées dans ce type d'évaluation, essayent de capturer les deux objectifs d'analyse de cluster : la minimisation de la distance intra cluster (qui résulte aux clusters compacts) et la maximisation de la distance inter-cluster (qui résulte aux clusters bien-séparés).

- La variance intra cluster est basée sur le concept de la minimisation de la distance intra clusters elle est définie en fonction de la distance entre les points d'un même cluster ou en fonction de la distance entre les points et le centroïde d'un cluster. Elle est donnée par :

$$Var(C) = \sum_{C_i \in C} \sum_{p_j \in C_i} d(p_j - \mu_i)^2 \qquad (1.19)$$

Où p_j dénote un point de données, k dénote le nombre de clusters, μ_i représente le centroïde de cluster C_i , *d (.,.)* est la fonction de distance utilisée pour calculer la déviation entre le point de données p_j et le centroïde μ_i [24].

- **La connectivité :**

La mesure de connectivité de cluster évalue le degré auquel des points de données voisins ont été placés dans le même cluster. Elle est calculée par la formule suivante :

$$Conn = \sum_{i=1}^{m} \left(\sum_{l=1}^{L} p_{i,nn,(l)} \right), \text{où } p_{r,s} = \begin{cases} 1/1 & \text{if } \neg \exists c_j \\ 0 & \text{otherwise,} \end{cases} \qquad (1.20)$$

$nn_{i(l)}$ est l^{eme} le plus proche voisin du point de données p_i et L est le paramètre déterminant le nombre des voisins qui contribuent à la connectivité. La connectivité devrait être minimale [25].

- **SSE :**

La somme des erreurs au carré (*en anglais : The Sum of Squared Errors*) (SSE) mesure essentiellement la variation des erreurs de modélisation. En d'autres termes, il montre comment la variation de la variable dépendante dans un modèle de régression ne peut pas être expliquée par le modèle. En général, une somme résiduelle plus faible des carrés indique que le modèle de régression peut mieux expliquer les données, tandis qu'une somme résiduelle plus élevée des carrés indique que le modèle explique mal les données.

elle est calculé à l'aide de la formule ci-dessous :

$$SSE = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \qquad (1.21)$$

Où :

y_i : la valeur observée

$\hat{y}_i$: la valeur estimée par la ligne de régression

- **Les erreurs de quantification (Quantization error) :**

  Les erreurs de quantification (*en anglais : Quantization error*) (QE) n'ont jamais été prises en compte dans les approches antérieures des réseaux neuronaux quantifiés. Nous proposons une méthode de formation tenant compte des erreurs de quantification qui utilise le QE comme terme de régularisation. Un QE est défini par l'expression suivante :

  $$QE(w) = w - w^q \qquad (1.22)$$

  où w^q est les valeurs de poids quantifiées dont les expressions sont représentées par la quantification logarithmique, la quantification linéaire ou la binarisation. Par la suite, nous définissons le terme de régularisation par quantification-erreur (*en anglais : quantization-error-based regularization*) (QER) comme suit :

  $$QER_2(w) = \|w - w^q\|_2 \qquad (1.23)$$

  $$QER_1(w) = \|w - w^q\|_1 \qquad (1.24)$$

  Elles sont fondées sur la norme L2 et la norme LL. Le terme QER est annexé à la fonction objective comme suit :

  $$L(w) = E(w) + \lambda QER(w) \qquad (1.25)$$

  où E(w) est la fonction de perte et λ est le taux d'adaptation de la QER. Ainsi, les poids sont mis à jour via le problème d'optimisation suivant, ainsi que la formation générale du réseau neuronal.

  $$w^* = arg\ \min_w L(w) \qquad (1.26)$$

La décroissance pondérale à l'aide de la norme L2 et de la régularisation de la norme L1 est une technique courante dans la formation des réseaux neuronaux pour la généralisation des poids afin d'éviter le débordement [27]. Ces régularisations servent à limiter la divergence de poids. En revanche, la QER oblige les poids à être plus proches en valeurs de leurs valeurs quantifiées et les poids peuvent obtenir moins d'erreurs de quantification En mettant à jour les poids pour minimiser L(w), les erreurs de quantification et les pertes sont progressivement réduites. Parce que notre méthode est une technologie de formation, nous pouvons utiliser un réseau formé pour la prédiction sans modifier l'architecture matérielle existante telle que [26]. En outre, le traitement QER est limité par rapport à l'ensemble de l'étape de formation dans le code d'optimisation.

- **La Variance intra-cluster :**

La variance intra-cluster est basée sur le principe de minimisation de la distance intra-cluster qu'elle est définie comme une fonction de la distance entre les points du même cluster ou comme une fonction de la distance entre les points et le centroïde d'un cluster. Il est donné par :

$$Var(C) = \sum_{Ci \in C} \sum_{p_j \in C_i} d(p_j - \mu_i)^2 \qquad (1.27)$$

Où p_j désigne un point de données, k désigne le nombre de clusters, μ_i représente le groupe centroïde C_i , d (.,.) est la fonction de distance utilisée pour calculer l'écart entre le point de données pj et le centroïde μi [24].

- **La Variance inter-cluster :**

La variance inter-cluster (*en anglais : Sum of squares Between (SSB)*) est utilisée pour quantifier la séparation externe. Il est défini comme la somme de la distance au carré entre le point moyen global et chaque centroïde. Plus la valeur est élevée, plus la concentration est élevée.

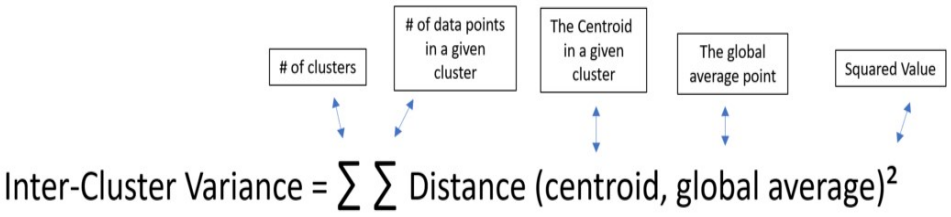


Figure 1.11 Formule pour calculer la variance inter-cluster

nous n'avons qu'à minimiser la variance intra-cluster parce que minimiser le SSW (within-cluster sums of squares) maximisera nécessairement le SSB (Between-cluster sums of squares).

1.8.Conclusion

Dans ce chapitre, nous avons examiné les concepts de base du clustering de données, ses différentes techniques et les critères utilisés pour valider et évaluer ces techniques. Dans le chapitre suivant, nous discuterons d'une métaheuristique récente qui est la recherche de coucou que nous utiliserons pour résoudre le problème de clustering.

Chapitre 2

La Recherche de Coucou

2.1 Introduction

L'optimisation joue un rôle très essentiel dans la recherche opérationnelle, informatique et mathématique. Elle consiste à ajuster des entrées selon des caractéristiques souhaitées. En fait, l'optimisation est un processus mathématique ou expérimental permettant d'aboutir à un résultat minimal ou maximal. Dans un problème d'optimisation, l'entrée est un ensemble de variables et de contraintes, le traitement est un ensemble d'opérations permettant la satisfaction d'une fonction objectif (connue aussi par: fonction fitness) et la sortie est une solution qui a une certaine qualité calculée selon la fonction objectif du problème traité. De nombreux algorithmes de résolution de problèmes d'optimisation ont été proposés dans la littérature.

Inspirées de la nature, les métaheuristiques utilisent des stratégies de recherche pour explorer l'espace de recherche et par la suite exploiter parfaitement les régions prometteuses de cet espace. Les algorithmes métaheuristiques sont souvent inspirés de la nature, et ils sont maintenantparmi les algorithmes les plus largement utilisés pour l'optimisation.

La puissance de presque toutes les métaheuristiques modernes vient du fait qu'elles imitent les meilleures caractéristiques de lanature, en particulier les systèmes biologiques évolués de la sélection naturelle sur des millions d'années. Ils ont de nombreux avantages par rapport aux algorithmes classiques. Ils sont très divers, notamment les algorithmes génétiques, le recuit simulé, l'évolution différentielle, lesalgorithmes de fourmis et d'abeilles, l'optimisation d'essaim de particules, la recherche de coucou et d'autres.

La recherche de coucou en anglais CuckooSearch (CS) est l'un des derniers algorithmes métaheuristiques inspirés de la nature, développés en 2009 par Xin-She Yang et Suash Deb.CS [33] est basé sur le parasitisme de couvaison de certaines espèces de coucou.

De plus, cet algorithme est renforcé par les soi-disant vols de Lévy, plutôt que par de simples randonnées aléatoires isotropes.

Dans ce chapitre, nous allons introduire la recherche de coucou en détail, en commençant par son comportement de reproduction intéressante, nous allons voir également les caractéristiques des vols de Lévy et son algorithme de base ainsi que son champ d'applications.

2.2 Le Coucou

Le coucou est un oiseau discret, longiligne, de taille moyenne et son chant marque le début de la belle saison (Figure 2.1). Parmi ses caractéristiques fascinantes, on cite aussi sa stratégie de reproduction. Le parasitisme des couvées chez un certain nombre d'espèces, est le plus étudie et discute. Les coucous femelles mettent un ou plusieurs œufs dans des nids, des autres espèces d'oiseaux, précédemment observes. Le but est de garantir un passage à la génération suivante en laissant les oiseaux hôtes guides par leur instinct naturel d'éclosion, de couvaison et d'apporter la nourriture aux petits coucous. Afin d'augmenter la probabilité d'avoir un nouveau coucou, la femelle gobe un œuf dans le nid parasite, avant d'y pondre le sien. Certaines espèces hôtes peuvent avoir un conflit avec le coucou intrus. Quand des oiseaux hôtes découvrent la présence d'œuf ne leur appartenant pas, grâce a une aire de peau sensible et dénudée qu'ils ont alors sous le ventre, soit ils s'en débarrassent, soit ils abandonnent le nid en construisant un nouveau ailleurs.

Figure 2.1 Oiseau coucou

2.3 L'habitat des coucous

L'habitat est le milieu de vie, de reproduction et de développement d'une population d'une espèce donnée. De même pour les coucous, il constitue une source de nourriture et un lieu de reproduction. Les Coucous se produisent dans une grande variété d'habitats. La majorité des espèces survivent dans les forêts et les bois, principalement dans les forêts tropicales à feuilles persistantes. En plus des forêts, certaines espèces de coucous occupent des environnements plus ouverts, ce qui peut inclure même les zones arides comme les déserts [31]. A titre d'exemple, le

Coucou plaintif fréquente une assez grande variété d'habitats tels que les bois ouverts, les forêts secondaires, arbustes et broussailles, les champs cultivés et aussi les jardins, aussi bien en milieu rural que citadin. On peut aussi le voir dans les herbages et les marais. Le Coucou gris fréquente les forêts de conifères ou de feuillus, et les zones boisées en général, les espaces boisés ouverts, les lisières de forêts et les clairières, les steppes arborées, les prairies, les marais et les roselières, ainsi que les zones cultivées avec des arbres et des buissons. Le Coucou geai fréquente des habitats semi-arides tels que les zones boisées ouvertes (surtout avec des acacias et des arbustes épineux), les contreforts rocheux des collines dans les savanes sèches, et les zones agricoles sèches avec des arbres et des buissons. On le voit souvent voler au-dessus des espaces découverts. Selon la distribution, et particulièrement en Europe, on peut le trouver dans les landes de bruyères avec du chêne-liège et des conifères du genre *Pinus pinea*. Il fréquente également les bosquets et les plantations d'oliviers.

2.4 Inspiration du comportement Coucou

En se basant sur le comportement des coucous ainsi que sur leur mode de vie et de reproduction, une nouvelle métaheuristique a été proposée récemment nommée: La Recherche Coucou La nature innove, invente, teste, valide, améliore et diversifie les systèmes vivants depuis des centaines de millions d'années, elle a été toujours une source d'inspiration. Plusieurs questions préoccupent les biologistes : dans un groupe d'oiseaux, pourquoi le groupe est-il souvent cohérent alors que chaque individu semble autonome? Comment les activités de tous les individus sont elles coordonnées sans supervision? Les éthologistes qui étudient le comportement de groupes d'oiseaux ou d'autres animaux ou même insectes observent que la coopération entre les éléments de groupe est auto-organisée: souvent, elle résulte d'interactions entre les individus. Bien que ces interactions puissent être simples, elles permettent à la collectivité de résoudre des problèmes difficiles.

Dans cette optique, un comportement attirant observé chez les oiseaux coucou a été investigué et exploité et il a mené à l'apparition de l'algorithme de la recherche coucou. Coucou sont des oiseaux fascinants, non seulement à cause des beaux sons qu'ils peuvent faire, mais aussi en raison de leur stratégie de reproduction agressive. Certaines espèces comme les coucou Ani et Guira pondent leurs œufs dans des nids communautaires, bien qu'ils peuvent retirer les œufs des autres pour augmenter la probabilité d'éclosion de leurs œufs [30].

Un certain nombre d'espèces s'engagent au parasitisme obligatoire des couvées en pondant leurs œufs dans les nids d'autres oiseaux hôtes (souvent d'autres espèces).Il existe trois types fondamentaux de parasitisme de couvée: le parasitisme intraspécifique decouvée, l'élevage en coopération et la prise de nid. Certains oiseaux hôtes peuvent entrer en conflit direct avec les coucous intrus. Si un oiseauhôte découvre que les œufs ne lui appartiennent pas, il jettera ces œufs ou abandonnera simplement son nid et construira un nouveau nid ailleurs.

Quelques espèces de coucou comme Tapera, ont évolué de telle manière que les coucous femelles sont souvent très spécialisés dans la mimique des couleurs et les motifs des œufs de quelques espèces hôtes choisies [30] .Cela réduit la probabilité que leurs œufs soient abandonnés et augmente ainsi leur reproductivité.

En outre, le moment de la ponte de certaines espèces est également étonnant. Les coucous parasites choisissent souvent un nid où l'oiseau hôte vient de pondre ses propres œufs.

En général, les œufs de coucou éclosent légèrement plus tôt que leurs œufs hôtes. Une fois que le premier poussin coucou est éclos, la première action instinctive qu'il prendra est d'expulser les œufs hôtes en propulsant aveuglément les œufs hors du nid, ce qui augmente la part de nourriture du poussin coucou fournie par son oiseau hôte. Des études montrent également qu'un poussin coucou peut également imiter l'appel des poussins hôtes pour accéder à plus de possibilités d'alimentation (Figure 2.2).

Figure 2.2 Un poussin coucou expulse les œufs

2.5 Vol de Lévy

En général, le processus de recherche de nourriture chez les animaux est effectivement aléatoire. En fait, leur déplacement est basé sur leur position actuelle ainsi qu'une probabilité du déplacement vers une autre position. Des études expérimentales sur le comportement de certains animaux et insectes ont montré que leur comportement peut être modélisé par un schéma mathématique nommé vol de Lévy (en anglais Lévy flight) [32].

Le comportement de parasitisme de couvées chez les coucous est combiné dans l'algorithme de la recherche coucou avec les vols de Lévy pour améliorer la recherche d'un nouveau nid. Les vols de Lévy (Figure 2.3), baptisées par le mathématicien français Paul Lévy, représente un modèle des marches aléatoires caractérisées par leurs longueurs de pas qui suivent une distribution de loi de puissance qui s'écrit de la forme suivante:[28]

$$N(s)=s^{-t}$$

Dans la nature, les animaux cherchent de la nourriture de manière aléatoire ou quasi-aléatoire. En général, la recherche de la nourriture est effectivement une marche aléatoire parce que le prochain mouvement est basé sur l'emplacement/état actuel et la probabilité de transition à l'emplacement suivant. La direction qu'ils choisissent dépend implicitement d'une probabilité qui peut être modélisée mathématiquement. Différentes études ont montré que le comportement de vol de nombreux oiseaux et insectes a les caractéristiques typiques des vols de Lévy [33].

D'autre part Reynolds et Frye [32] ont montré que les mouches de fruits "Drosophila melanogaster" explorent leur paysage en utilisant une série de trajectoires droites ponctuées par un brusque virage à 90°, ce qui conduit à un modèle de recherche intermittente sans échelle d'un style de vol de Lévy. Ce modèle est communément représenté par de petits pas aléatoires suivis à long terme par de grands sauts [29]. Un tel comportement de vol de Lévy a été appliqué à l'optimisation et les résultats ont montré une capacité prometteuse [33].

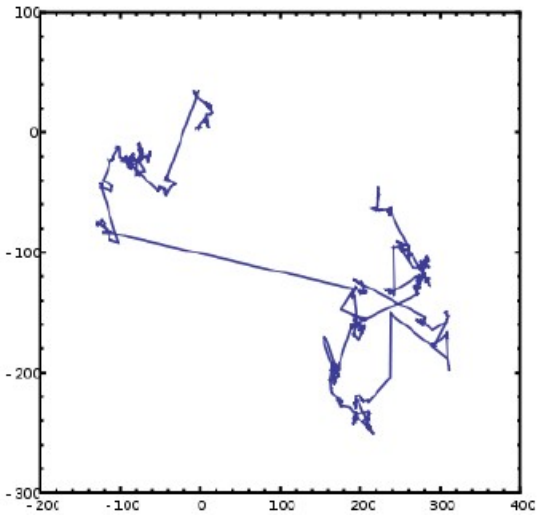


Figure 2.3 Exemple de 1000 pas par les vols de Lévy en 2 dimensions

2.6 Les principes de bases de l'algorithme de la recherche coucou (CS)

La métaheuristique de la recherche coucou CS développée par Xin-She Yang et Suash Deb en 2009 prend comme une base les idées suivantes:

- Chaque coucou pond un œuf à la fois et choisit un nid aléatoirement ;
- Un bon nid de bonne qualité peut passer vers une nouvelle génération ;
- Le nombre de nids hôtes est fixé, et un œuf posé par un coucou peut être découvert par l'oiseau hôte suivant une probabilité $p_\alpha \in [0, 1]$.

Dans ce cas, l'oiseau hôte peut jeter l'œuf ou abandonner le nid et construire un nid complètement nouveau. Pour simplifier, cette dernière hypothèse peut être approchée par la probabilité p_α de n nids d'être remplacés par de nouveaux nids. L'algorithme de la recherche du coucou (Figure 2.4) se résume autour des règles suivantes:

- Chaque œuf du coucou dans un nid représente une solution.
- Chaque oiseau de coucou pondra un seul œuf à la fois, et choisira son nid de façon "aléatoire". Donc, chaque individu de la population des coucous a le droit de générer aléatoirement une seule nouvelle solution.

- Les meilleurs nids de meilleure qualité d'œufs nous mèneront vers les nouvelles générations. Ici, on a introduit implicitement la notion d'intensification ou la recherche autour des meilleures solutions.
- Certaines nouvelles solutions doivent être générées par les vols du Lévy autour de la meilleure solution obtenue jusqu'ici. Cela accélérera la recherche locale.
- Le nombre de nids hôtes est fixe, et l'œuf pondu par l'oiseau est découvert par l'hôte avec une probabilité $p_\alpha \in [0, 1]$. Dans ce cas, l'oiseau hôte choisi de se débarrasser de l'œuf, ou d'abandonner le nid et de reconstruire un autre nid quelque part. Pour la simplification, cette dernière hypothèse sera approximée par la fraction p_α des n nids qui sont remplacés par des nouveaux (nouvelles solutions aléatoires).
- De nouvelles solutions doivent êtres générées par randomisation vers des régions lointaines et dont les emplacements doivent être assez loin de la meilleure solution actuelle, ce qui fera que le système ne sera pas pris au piège dans un optimum local.

Début
Initialiser une population de N nids
Tant que *critères d'arrêt* ***faire***
Obtenir un Coucou aléatoirement par les vols de Levy
Evaluer sa fitness F_i
Choisir un nid parmi les N aléatoirement
Remplacer le coucou qui a la meilleur fitness
Trouver le meilleur coucou de la population
Modifier les nouveaux coucous
Evaluer leurs qualités (fitness)
Sélectionner les coucous de la nouvelle génération
Trouver le meilleur nid
Fin Tant que
Retourner la meilleure solution
Fin

Figure 2.4 Algorithme de Recherche Coucou

2.7. Pseudo-code de l'algorithme de la Recherche Coucou (CS) par le vol de Lévy

Les étapes de base de la recherche Coucou (CS) peuvent être résumées par le code pseudo illustré ci-dessous (Figure 2.5) :

Début

Fonction objective f (x), x = (x 1, ..., X d) T

Générer la population initiale de n de nids hôtes xi (i = 1, 2, ..., n)

Tant que (T<MaxGeneration) ou (critère d'arrêt)

1. Obtenez un coucou (i) aléatoirement par le vol de Lévy

2. évaluer sa qualité/ *fitness Fi*

3. Choisissez un nid parmi n (par exemple, j) aléatoirement

4. *T=T+1*

si (*Fi >Fj*), // les nouvelles solutions de nid j dominent ceux de nid i

Remplacer j par la nouvelle solution;

Fin si

Une fraction (P_a) des nids les plus mauvais est abandonnée et de nouveaux sont construites;

Gardez les meilleures solutions (ou des nids avec des solutions de qualité);

Triez les solutions et trouver ci le meilleur courant

Fin tant que

Post-traitement des résultats et la visualisation

Fin

Figure 2.5 pseudo-code de l'algorithme de la recherche Coucou (CS)

2.8 Efficacité de la recherche coucou (comparaison avec d'autres métaheuristiques):

La faculté d'équilibrer la recherche entre l'exploitation des zones prometteuses et l'exploration de l'espace de recherche à l'échelle globale, est un indice de performance lié à toutes les métaheuristiques. La métaheuristique la plus robuste est celle qui balance parfaitement entre l'intensification et la diversification. CS a un meilleur contrôle d'équilibre de la stratégie de recherche intensive locale et une exploration plus efficace de l'ensemble de l'espace de recherche. Aussi, le nombre réduit de paramètres fait de CS un algorithme moins complexe et donc potentiellement plus générique.

CS utilise un nombre réduit de paramètres de contrôle. En effet, CS utilise deux paramètres, la taille de la population n et les portions des nids abandonnés p_a. En principe, n est fixé et p_a essentiellement contrôle l'élitisme et l'équilibre entre la randomisation et la recherche locale.

Le nombre réduit de paramètres rend l'algorithme moins complexe et donc potentiellement plus générique. Tout ceci est expliqué par la facilité de régler les deux paramètres de CS tout en gardant sa robustesse de résoudre une large gamme de problèmes d'optimisation même les problèmes qualifiés NP-difficiles.

La comparaison de CS et de PSO et GA a montré que CS surpasse les performances de ces métaheuristiques. Effectivement, une étude analytique [34] a montré que DE, PSO et SA sont des cas particuliers de la recherche de coucou et il a été aussi montré que la recherche de coucou a une convergence globale.

Des études théoriques sur l'optimisation des essaims de particules ont suggéré que la PSO peut converger rapidement vers la meilleure solution actuelle, mais pas nécessairement les meilleures solutions globales. En fait, certains analystes suggèrent que les équations de mise à jour des PSO ne satisferont pas aux conditions de convergence globale, et donc il n'y a aucune garantie pour la convergence globale. D'autre part, il a prouvé que la recherche du coucou satisfait aux exigences de convergence globale et a ainsi garantit des propriétés. Cela implique que pour l'optimisation multimodale, la PSO peut converger prématurément vers un optimum local, alors que la recherche de coucou peut généralement converger vers l'optimalité globale. Un autre avantage de

la recherche de coucou est que sa recherche globale utilise des vols Lévy, plutôt que des randonnées aléatoires standard. Comme les vols de Lévy ont une moyenne et une variance infinies, CS peut explorer l'espace de recherche plus efficacement que les algorithmes utilisant des processus gaussiens standard. Cet avantage, conjugué à la fois aux capacités locales et de recherche et à la convergence globale garantie, rend la recherche de coucou très efficace. En effet, diverses études et applications ont démontré que la recherche du coucou est très efficace [34].

2.9 Applications de la recherche coucou

La recherche de coucou et ses variantes ont été appliquées dans presque tous les domaines des sciences, de l'ingénierie et de l'industrie(Figure 2.6). Évidemment, l'optimisation de l'ingénierie fait partie des diverses applications. Certaines des études d'application sont résumées dans le Tableau ci-dessous [34] :

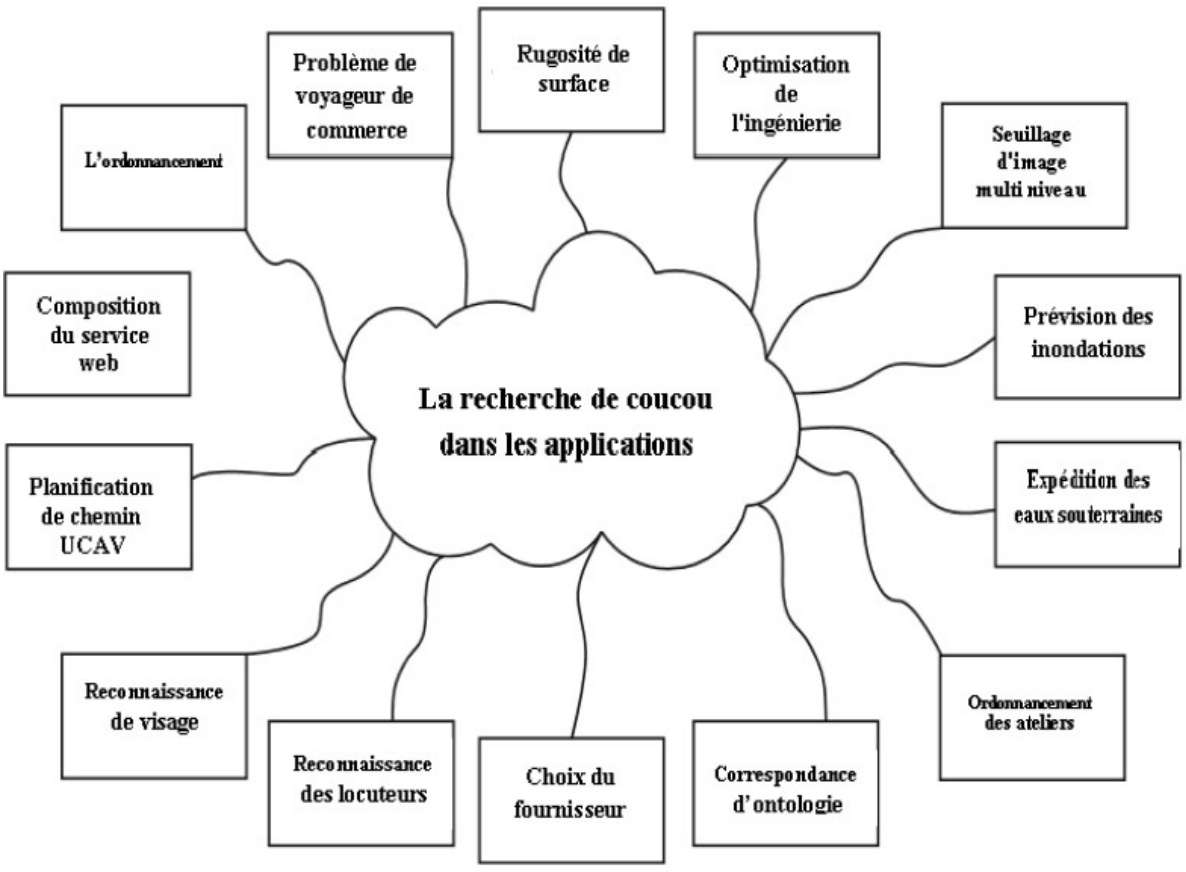


Figure 2.6 Applications de la Recherche de coucou

Application	Auteur	Année
Seuil d'image multi niveau	Brajevic et al.	2012
Prévision des inondations	Chaowanawatee and Heednacram	2012
Réseaux de capteurs sans fil	Dhivya and Sundarambal	2011
La fusion des données	Dhivya et al.	2011
Cluster dans les réseaux sans fil	Dhivya et al.	2011
Clustering	Goel et al.	2011
Expédition des eauxsouterraines	Gupta et al.	2011
Choix du fournisseur	Kanagaraj et al.	2012
Prévision de charge	Kavousi-Fard and Kavousi-Fard	2013
Rugosité de surface	Madic et al.	2013
Planification de magasin de flux	Marichelvam	2012
Remplacement optimal	Mellal et al.	2012
Allocation DG dans le réseau	Moravej and Akhlaghi	2013
Optimisation du filtreFloraison	Natarajan et al.	2012
Réseau neuronal BPNN	Nawi et al.	2013
Problème Voyageur de commerce	Ouaarab et al.	2013
Composition du service Web	Pop et al.	2011
Composition du service Web	Chifu et al.	2012
Correspondanceontologique	Ritze and Paulheim	2011
Reconnaissance des locuteurs	Sood and Kaur	2013

Tests automatisés de logiciels	Srivastava et al.	2012
Optimisation de fabrication	Syberfeldt and Lidberg	2012
Reconnaissance de visage	Tiwari	2011
Formation des modèlesneuronaux	Vázquez	2013
Expéditionéconomique non convexe	Vo et al.	2013
Planification des trajets UCAV	Wang et al.	2012
Optimisation des activités	Yang et al.	2012
Sélection des paramètresd'usinage	Yildiz	2013
Planification des travauxen grille	Prakash et al.	2012
Affectation quadratique	Dejam et al.	2012
Problèmed'emboîtement de feuilles	Elkeran	2013
Optimisation des requêtes	Joshi and Srivastava	2013
N-Queens puzzle	Sharma and Keswani	2013
Jeux d'ordinateur	Speed	2010

Tableau 2.1 : Applications de la recherche coucou

2.10. Conclusion

Nous avons vu dans ce chapitre la méthode de recherche de coucou en détail, nous avons également présenté le comportement de coucou, le vol de Lévy ainsi que les principes de base de l'algorithme de coucou en spécifiant le pseudo-code de cet algorithme par le vol de Lévy.

Chapitre 3

La conception de l’application

3.1. Introduction

Après avoir présenté dans le chapitre précédent les notions et les concepts de base de la recherche coucou, nous allons présenter dans ce chapitre son adaptation au problème de clustering.

Nous commençons par la description de cette adaptation, nous passons ensuite à la présentation de l'objectif de notre application ainsi que la mise en évidence de l'architecture générale de notre système et la description de ses modules qui constitue une étape fondamentale qui précède l'implémentation, nous détaillons, également, les différents diagrammes et scénarios à implémenter dans la phase suivante. Ceci permettra de mieux comprendre notre application.

Pour cela nous allons utiliser les diagrammes de flux de données pour la modélisation de notre application.

3.2. Clustering basé sur la Recherche Coucou

La recherche de Coucou (CS) est l'un des algorithmes metaheuristiques. Cet algorithme fonctionne sur la base de la stratégie de reproduction de l'oiseau coucou.

Notre travail représente un algorithme de clustering de données basé sur l'optimisation par la recherche de coucou. La recherche de coucou est générique et robuste pour de nombreux problèmes d'optimisation et dispose de fonctionnalités intéressantes.

Pour résoudre le problème de clustering de données, l'algorithme de recherche de coucou est adapté pour trouver les centroïdes des clusters. Pour faire cela, nous supposons que nous avons *n* objets (données) et chaque objet est défini par *m* attributs. Dans notre travail, l'objectif principale du CS est de trouver *k* centroïdes des clusters qui réduisent au maximum la fonction objectif dénotée par SSE qui représente une mesure de qualité interne, elle permet de calculer la distance intra cluster (la somme des distances entre les points et leur centroïde de cluster correspondant), elle est définie par l'équation (3.1). Sachant que l'ensemble de données (jeu de données) doit être représenté par une matrice (n,m), tel que *n* représente le nombre d'objets (le nombre de points de données) et *m* représente le nombre des attributs (le nombre de dimensions).

$$SSE = \sum_{i=1}^{k} \sum_{j=1}^{n} W_{ij} * \sqrt{\sum_{p=1}^{m} (o_{jp} - c_{ip})^2} \qquad (3.1)$$

Où $W_{ij} = 1$ si l'objet est dans le cluster et **0** sinon, **k** est nombre de cluster, **n** est le nombre d'objet, **m** est le nombre des attributs, et c_{ip} est la valeur de l'attribut numéro **p** du centroïdes de cluster numéro **i** [35].

Dans le mécanisme de recherche de coucou, les nids sont des solutions et dans notre travail une solution est un ensemble de centroïdes représenté par un vecteur de dimension **k*m** (où **k** est le nombre de centroïdes des clusters et **m** représentent le nombre d'attributs).

3.3. Présentation de notre de système

Notre système est composé des étapes suivantes :

- Déterminer un jeu de données à partitionner.
- Appliquer l'algorithme CS sur un jeu de données.
- Comparer les résultats en utilisant des mesures d'évaluation internes (la fonction objective SSE) et externes (la F-mesure).
- Evaluer l'indice FE qui représente le nombre d'évaluations de la fonction objectif que l'algorithme effectue jusqu'à l'obtention de la meilleure valeur de cette fonction.
- Représenter les résultats graphiquement.

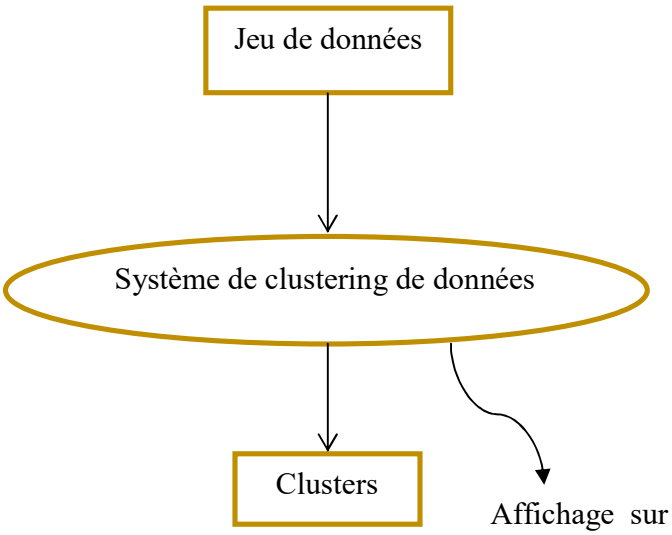


Figure 3.1 : Diagramme de flux de données (DFD)-niveau 1-schéma global

3.4. Module CS (Cuckoo Search)

Ce module permet d'appliquer l'algorithme CS à un jeu de données, cet algorithme est constitué de plusieurs étapes qui sont :

- Initialisation aléatoire des centroïdes.
- Affectation des points aux clusters.
- Evaluer les solutions en utilisant la fonction objectif
- Trier les solutions actuelles et trouver la meilleure .
- Générer des solutions en utilisant les vols de Lévy
- Abandonner les mauvaises solutions avec une probabilité P_a
- Remplacer la solution courante par une meilleure et calculer l'indice FE.
- Calculer la F-mesure.

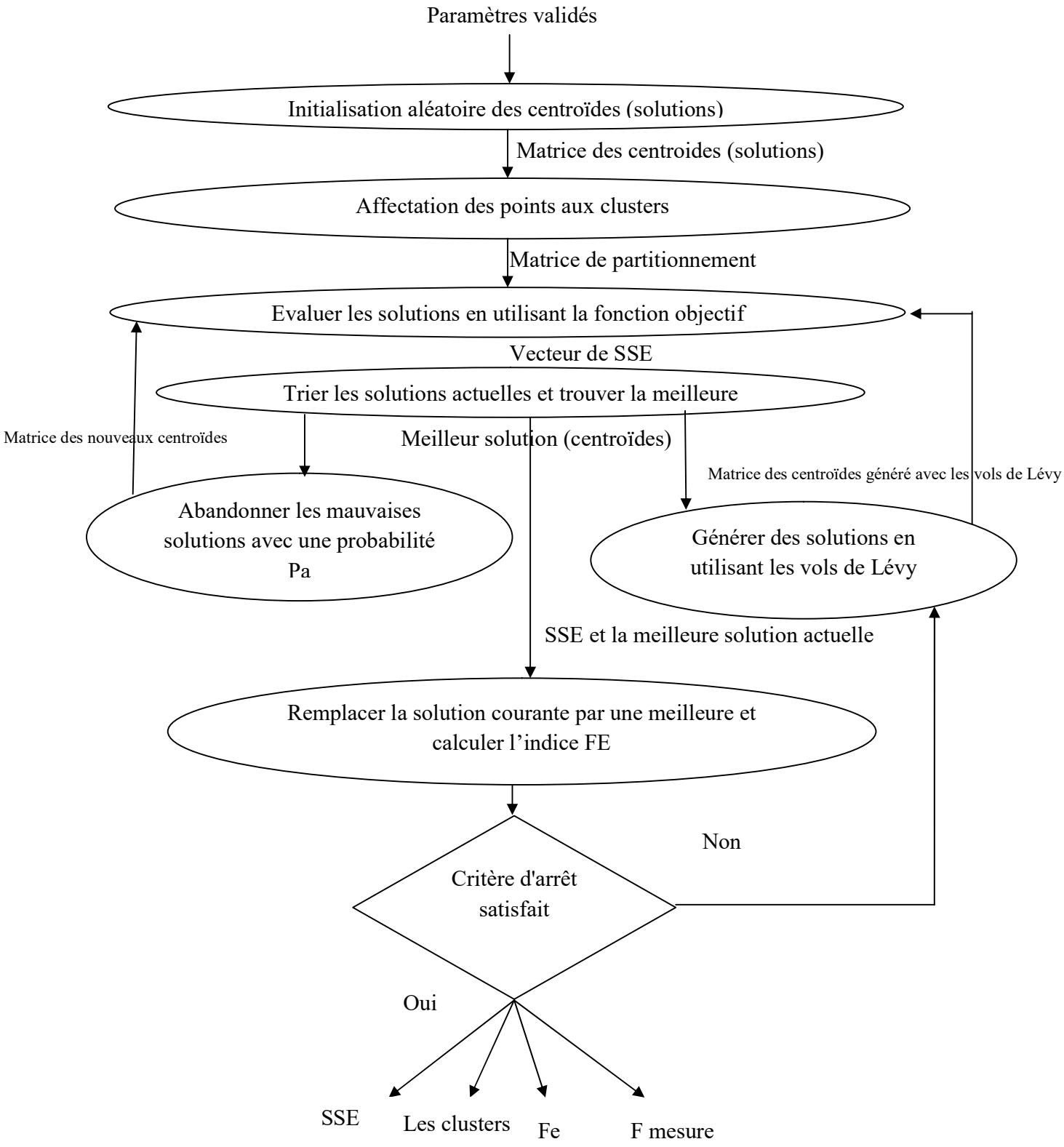


Figure 3.2 : Diagramme de flux de données(DFD)- niveau 3- Algorithme CS

3.4.1. Initialisation aléatoire

Cette étape permet de choisir aléatoirement les centroïdes à partir d'un jeu de données.

3.4.2. Affectation des points aux clusters

Après avoir initialisé les centroïdes, on affecte les points aux clusters en calculant la distance entre les points et les centroïdes en utilisant la formule suivante :

$$d = \sqrt{\sum_{p=1}^{m}(o_{jp} - c_{ip})^2} \qquad (3.2)$$

3.4.3. Evaluer les solutions en utilisant la fonction objectif

Cette étape permet d'évaluer les solutions (centroides) en minimisant la fonction objectif SSE

$$SSE = \sum_{i=1}^{k}\sum_{j=1}^{n} W_{ij} * \sqrt{\sum_{p=1}^{m}(o_{jp} - c_{ip})^2} \qquad (3.3)$$

3.4.4. Trier les solutions actuelles et trouver la meilleure

Après avoir évalué les solutions, on effectue le tri de celle-ci de façon croissante en utilisant ses fonctions objectif, ce qui permet de trouver la meilleure solution.

3.4.5. Calcul des centroides avec le vol de Lévy

En appliquant le vol de Lévy, cette étape permet de générer des solutions (calcul de centroïdes) en utilisant les formules suivantes

$$s = \frac{u}{|v|^{1/\beta}} \qquad (3.4)$$

$$u \sim N(0, \sigma_u^2)\ , v \sim N(0, \sigma_v^2) \qquad (3.5)$$

$$\sigma_u = \left\{\frac{\Gamma(1+\beta)\sin(\pi\beta/\ )}{\Gamma\left[\frac{1+\beta}{2}\right]\beta 2^{(\beta-1)/2}}\right\}^{1/\beta} \qquad (3.6)$$

Où S représente la longueur du pas par la distribution de vol de Lévy.

3.4.6. Abandonner les mauvaises solutions (centroïdes) avec une probabilité P_a

Cette étape permet d'abandonner les mauvaises solutions avec une probabilité p_a et de construire des nouveaux.

3.4.7. Remplacer la solution courante par une meilleure et calculer l'indice FE

Cette étape permet de remplacer la solution courante par une meilleure en utilisant un test de comparaison entre les valeurs de la fonction objectif courante et l'ancienne. Elle permet également de calculer le nombre d'évaluations (FE) de la fonction objectif que l'algorithme CS effectue pour obtenir la meilleure valeur de la fonction objectif SSE.

3.4.8. Critère d'arrêt

Le critère d'arrêt de cet algorithme est un nombre d'itération limité, si ce dernier atteint la fin des itérations l'algorithme s'arrête sinon il va continuer.

3.4.9. Calcul de la F-mesure

Le calcul de la F-Mesure est constitué d'une seule étape représentée par la figure (3.3)

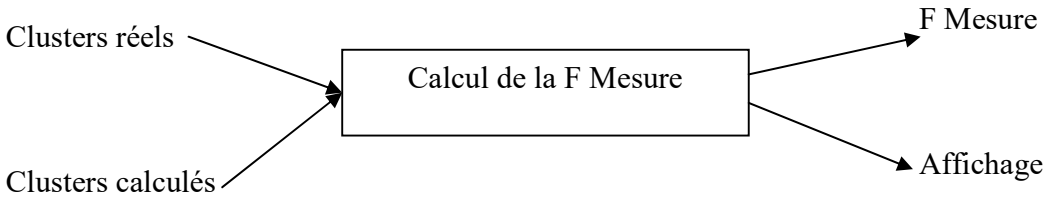


Figure 3.3 : Diagramme de flux de données(DFD)- niveau 4- F-Mesure

Dans cette étape, on compare les clusters calculés par l'algorithme CS avec les clusters réels importés à partir d'un fichier du jeu de données. Le calcul de la F-Mesure se fait par la formule suivante :

$$\boldsymbol{FMesure} = \sum_{i=1}^{k} \frac{N_i}{N} \max_{C_i \in C} (\boldsymbol{Fmes}(R_i, C_j)) \qquad (3.7)$$

On détermine la qualité de clustering selon la valeur de la F-mesure, elle est limitée à l'intervalle [0,1], si cette valeur tend vers 1 le partitionnement est meilleur sinon le partitionnement est mauvais.

3.5. Conclusion

Nous avons vu dans ce chapitre plus sur la méthode de recherche de coucou pour le clustering de données, suivi par la détermination de l'idée de base de l'approche et l'architecture générale du système.

Dans le chapitre suivant, nous allons présenter l'implémentation de l'approche avec les structures de données et le langage utilisé ainsi qu'une session de déroulement de notre application.

Chapitre 4
Implémentation de l'approche

4.1. Introduction

Dans ce chapitre, nous allons passer à la présentation de notre application. D'abord, nous commençons par les composants de l'environnement de travail et le langage utilisé, suivi par la présentation des différentes structures de données et les différentes fenêtres de notre application et nous présentons également le déroulement d'une session de travail.

4.2. Implémentation

4.2.1. Éléments de l'environnement de travail

- Système d'exploitation : Windows 10 Pro
- Machine utilisé : PC (Intel® Core™ i7-8650U CPU 1.90GHz 2.11GHz, 16GB RAM)
- Langage : MATLAB 2015a

4.2.2. Langage MATLAB

MATLAB (MATrixLABoratory) est un langage de programmation de quatrième génération et un environnement d'analyse numérique. MATLAB permet de faire du calcul matriciel, de développer et d'exécuter des algorithmes, de créer des interfaces utilisateur (IU) et de visualiser des données.

4.2.3. La structure des données

4.2.3.1. La notion des tableaux

Un tableau est représenté par une liste qui contient les éléments d'une manière contiguë et accessible.

4.2.3.2. Variables utilisés

- K : nombre de clusters.
- nd : nombre de dimensions des données.
- nbr : nombre des points de données.
- Data[*nbr, nd*] : Matrice des données avec *nd* dimensions et *nbr* points.
- partition_reelle[*nbr*] : partition (classification) des points au clusters correcte.
- best_vect[*nbr*] : partition trouvée après l'exécution.

4.2.4. Déroulement d'une session du travail

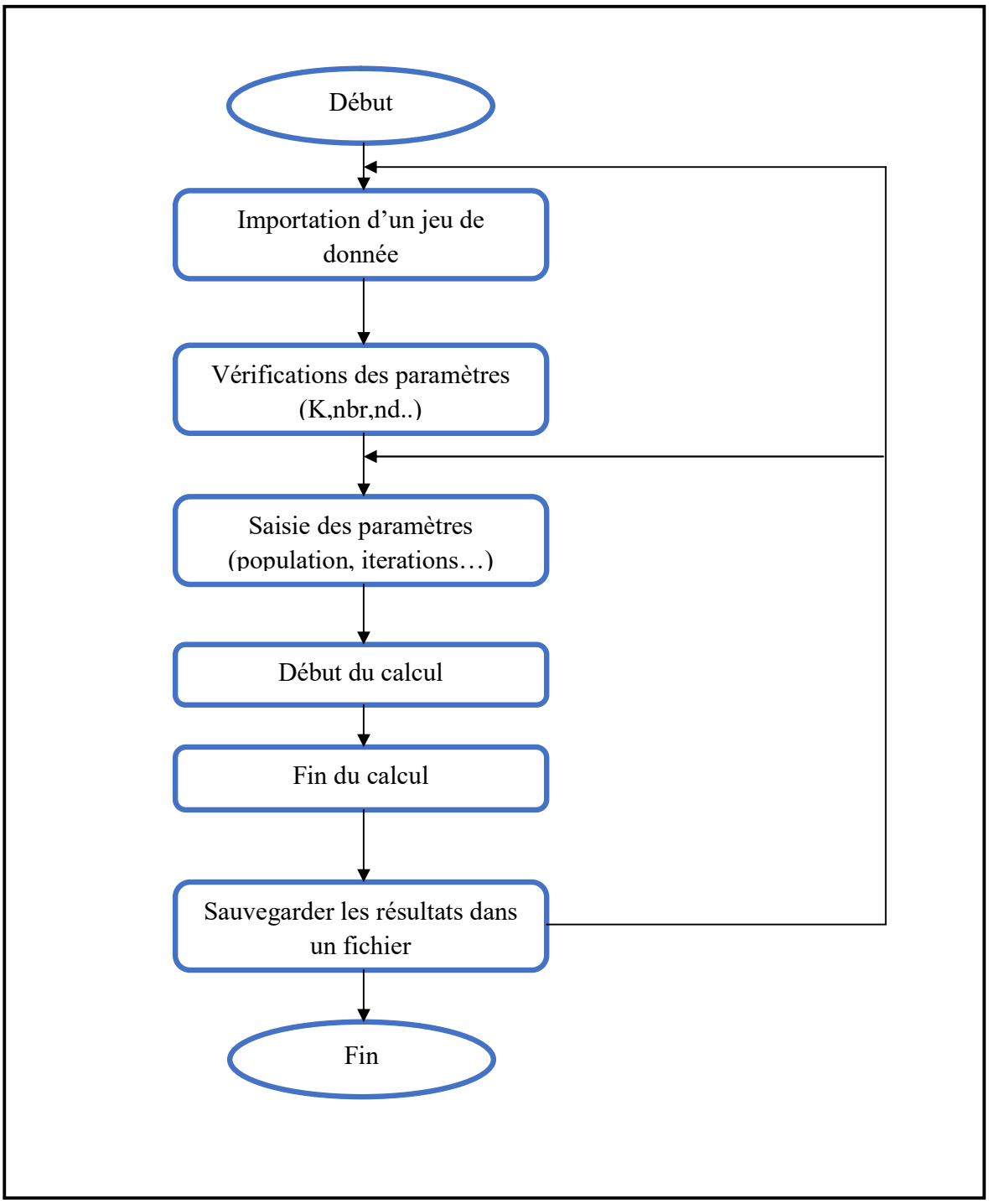


Figure 4.1 : Déroulement d'une session de travail

4.2.5. Présentation du logiciel

Notre application se compose d'une fenêtre principale (voir figure 4.2)qui est divisée en 5 sections.

Figure 4.2 : Fenêtre principal

1) Section 1 « Importer un jeu de données» : c'est là où l'utilisateur peut faire l'importation d'un jeu de donnée en cliquant sur le bouton  suivi par une boite de dialogue pour sélectionner un fichier d'une extension *.mat* parmi les jeux existant (voir figure 4.3).

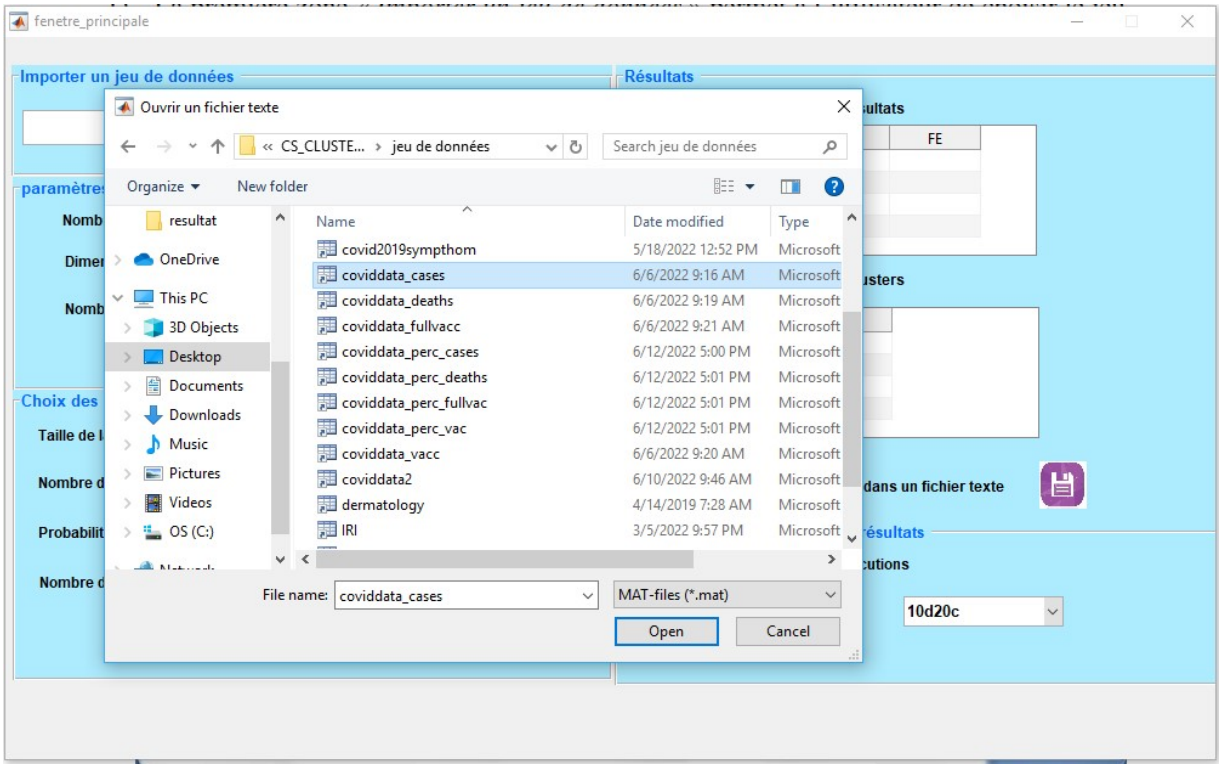


Figure 4.3: Boite de dialogue pour la sélection du jeu de donnée

2) Section 2 « Paramètre du jeu de données » : le logiciel import les paramètres constantes du jeu donnée qui sont : Le nombre de clusters **K**, le nombre de points **nbr** et nombre de dimensions **nd.** Qui peut être modifiée par l'utilisateur sans modifier le fichier original. du jeu de donnée. Le bouton  est pour actualiser les paramètres par défaut (champs vide) (voire Figure 4.4).

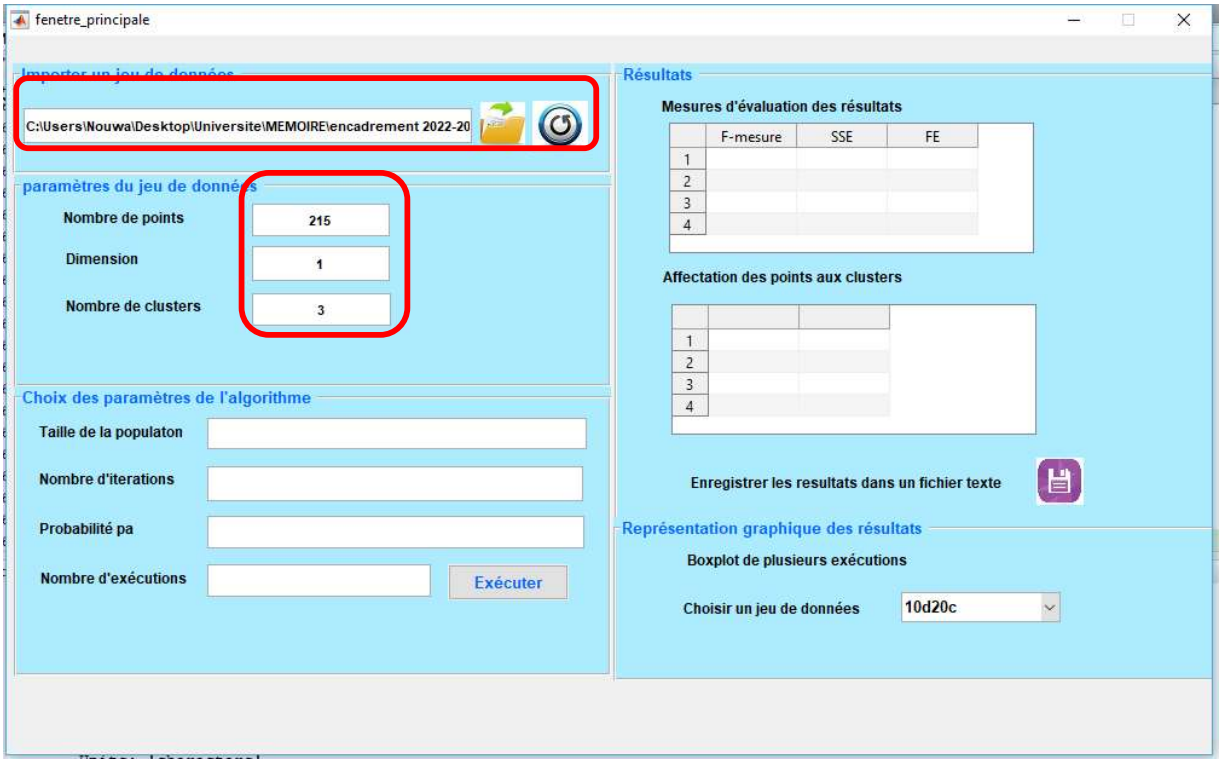


Figure 4.4 : Paramètres du jeu de données

3) Section 3 «Choix des paramètres de l'algorithme» : après la saisie de chaque paramètre tel que : la taille du population, le nombre d'itérations, probabilité et nombre d'exécutions, l'utilisateur clique sur le bouton «Exécuter» pour démarrer l'exécution. Une barre de progression s'affiche pour chaque exécution (voir Figure 4.5)

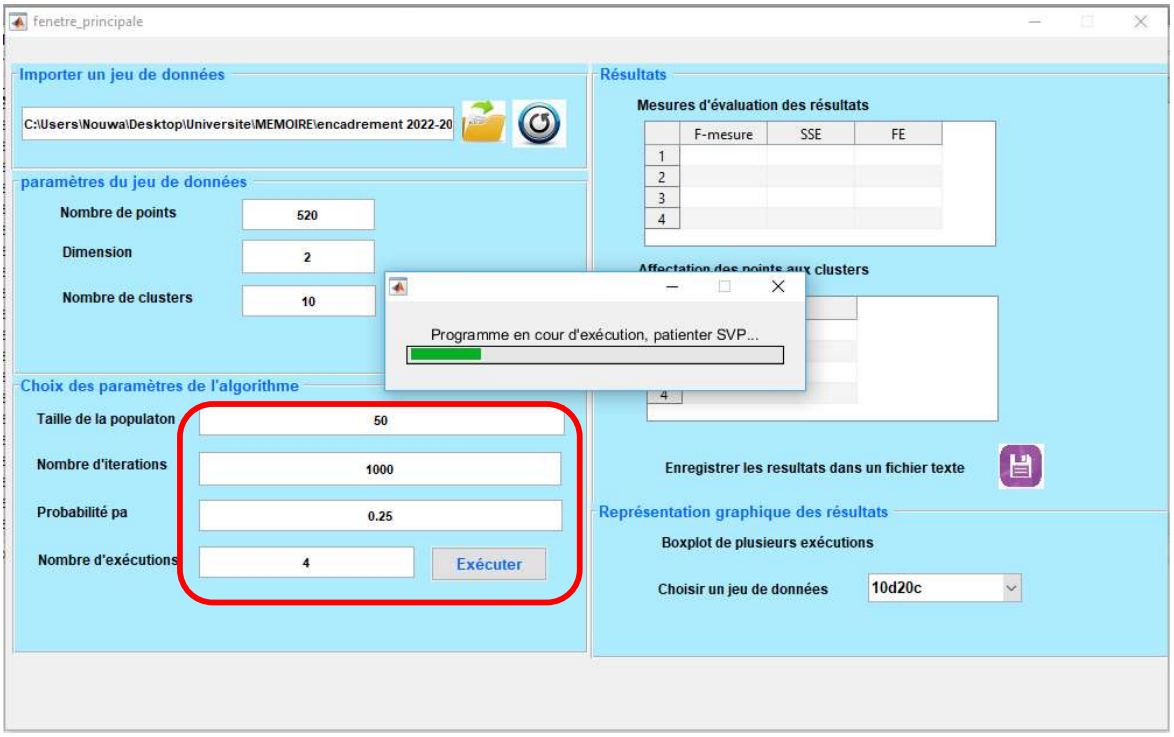


Figure 4.5 : La saisie des paramètres et barre de progression après l'exécution

4) Section 4 «Résultats» : Une foie que l'exécution termine, les résultats s'affiche dans les deux tableaux : Mesures d'évaluation des résultats et affectation des points aux clusters (voire Figure 4.6).

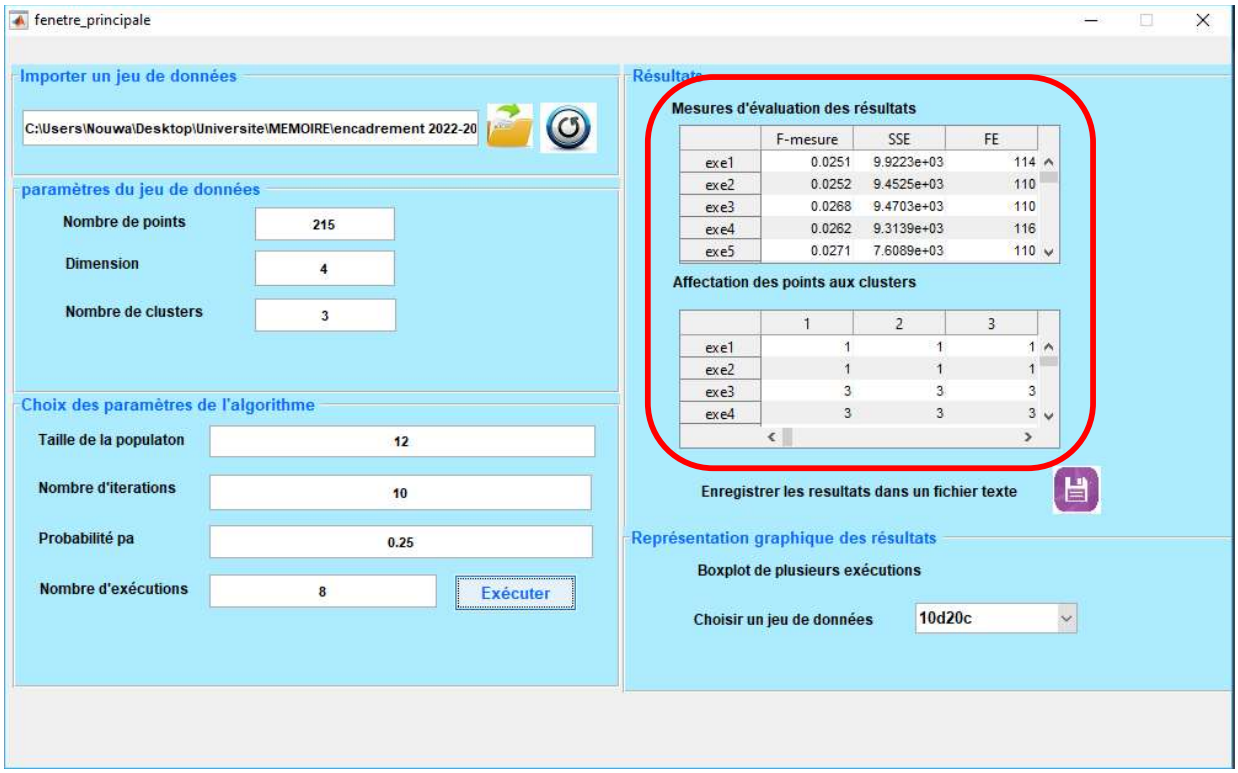


Figure 4.6 : Les résultats d'exécutions

L'utilisateur peut aussi cliquer sur le bouton  pour sauvegarder les résultats dans un fichier texte (voire Figure 4.7).

Figure 4.7 : Enregistrement des résultats

5) Section 5 «La représentation graphique» : les jeux de données sont divisé en 2 parties,

Partie 1 : Les jeux de données pour lesquelles la classification correcte est connue (IRIS, WISCONSINE, DERMATOLOGY, 10D20C, 10D10C, 2D4C, 2D10C) où les résultats s'affichent sous forme de boxplots de F-Mesure, SSE et indice FE (voir Figure 4.8).

Partie 2 : Les jeux de données pour lesquelles la classification correcte n'est pas connue (COVIDDATA). Lorsque l'utilisateur sélectionne un jeu de données, les fenêtres de visualisation du clustering des points et une carte graphique pour les pays s'affichent. L'utilisateur peut cliquer sur un point pour plus d'information (voir Figure 4.9 et Figure 4.10).

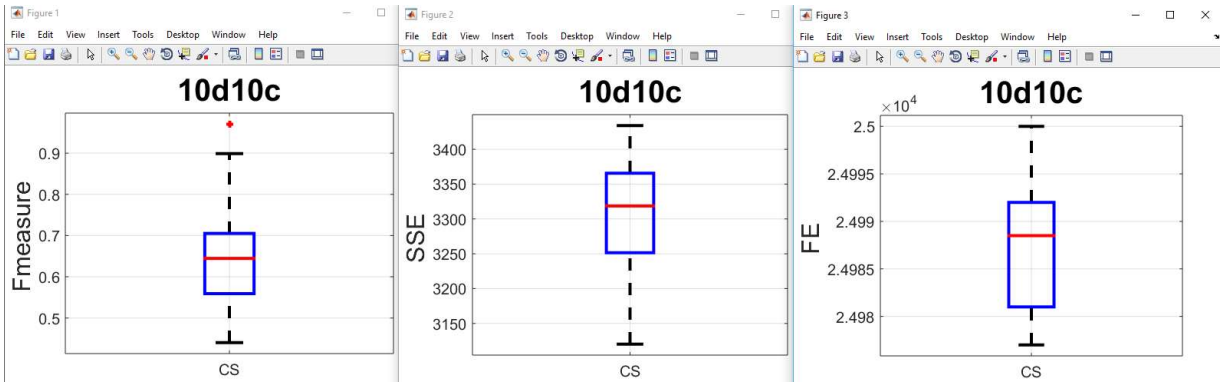


Figure 4.8 : Représentation graphique par boxplots

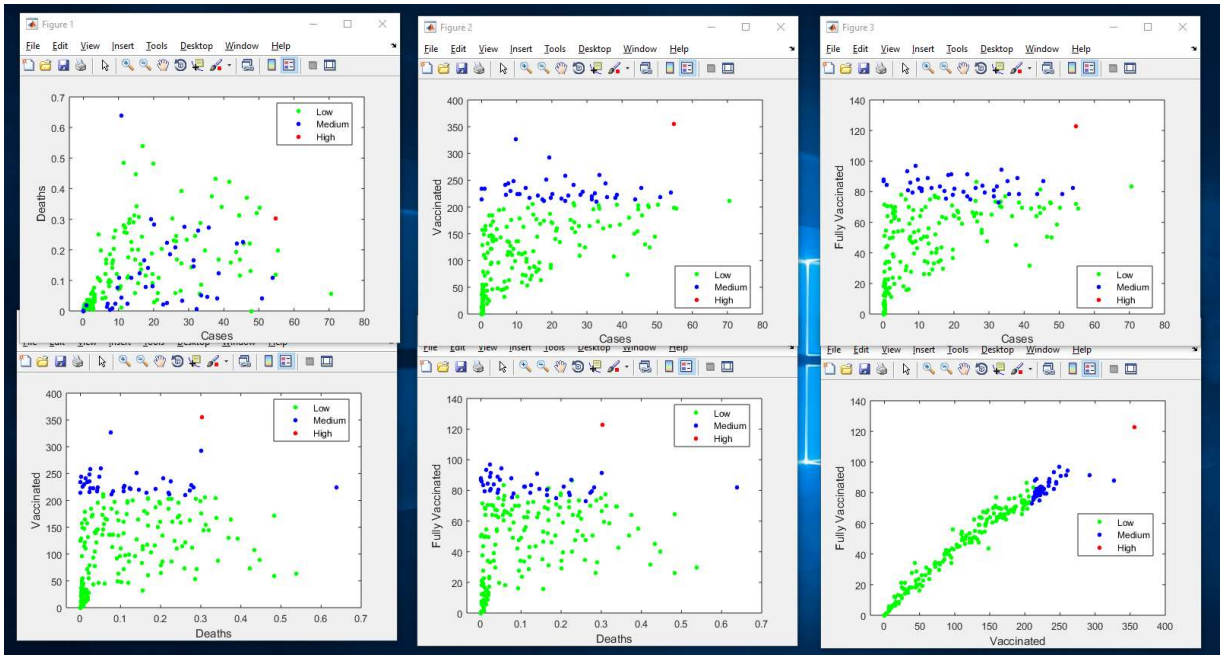


Figure 4.9 : Représentation graphique des clusters de 4 dimensions

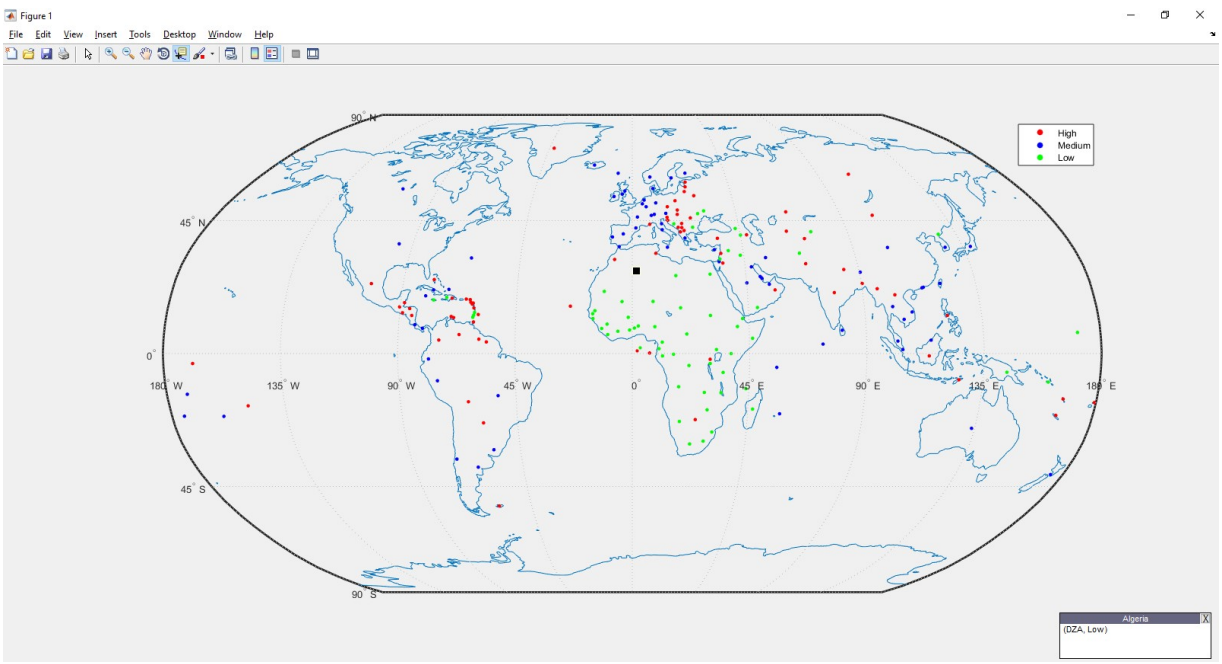


Figure 4.10 : Représentation graphique des clusters sur une carte géographique

4.3. Conclusion

Dans ce chapitre, nous avons présenté l'implémentation de notre application, nous avons montré le déroulement de notre application, les outils pour le développement, ainsi que la présentation détaillée des interfaces. Dans le prochain chapitre, nous passons à l'expérimentation en montrant les tests, les résultats et leur analyse.

Chapitre 5

Expérimentation : tests, résultats et analyse

5.1 Introduction

Dans ce chapitre, nous allons présenter les résultats des tests effectués ainsi que leur visualisation graphique en utilisant les Boxplot, et la représentation des résultats de clustering sur la carte géographique. A la fin, nous passons à l'analyse et la discussion de ces résultats.

5.2 Jeux de données

Des tests ont été effectués sur des données synthétiques et des données réelles.

A. Jeux de données synthétiques

Les jeux de données synthétiques sont générés avec un générateur de cluster gaussien décrit dans [36]. Le tableau 5.1 montre le détail de ces jeux de données comportant les variables suivantes :

- *m* : nombre total de points de données dans le jeu de données.
- *ni* : le nombre de points appartenant au groupe *i.*
- *Dim* : nombre de dimensions.
- *K* : le nombre de clusters (groupes).

Jeu de données synthétiques	*K*	*Dim*	*m*	*Ni*
2d4c	4	2	1123	369,471,53,230
2d10c	10	2	520	67,15,19,53,83,64,65,68,68,18
10d10c	10	10	436	18,83,57,26,67,50,12,72,39,12
10d20c	20	10	433	20,30,12,9,24,31,10,23,15,41 32,9,13,26,10,36,13,40,10,29

Tableau 5.1 : Résumé des jeux de données synthétiques

B. Jeux de données réels

Les jeux de données réels sont apportés d'un répertoire de bases de données d'apprentissage appelé l'UCI [37]. Le détail de ces jeux de données est présenté dans le tableau 5.2.

Jeu de données réels	*K*	*Dim*	*m*	*Ni*
Iris	3	4	150	50,50,50
Dermatology	6	33	366	112,61,72,49,52,20
Wisconsin	2	9	699	458,241

Tableau 5.2 : Résumé des jeux de données réels

C. Jeu de données de COVID19

Les données utilisées ici sont extraites du site spécifique créé par l'Université John Hopkins sur la COVID-19 [39]. Ce jeu de données contient des données épidémiologiques sur la COVID-19

pour 206 pays prises de la période du 4 avril 2020 au 4 avril 2022. Ce jeu de données inclut 8 attributs :

iso_code : code du pays.

continent : le continent du pays.

location : le pays

total_cases : le nombre total de cas de covid19

total_deaths : le nombre total

total_vaccinations : le nombre de vaccins administrés.

People fully vaccinated : le nombre de personnes totalement vaccinés.

Population : la taille de la population d'un pays.

Seuls les dimensions (total_cases , total_deaths , total_vaccinations ,People fully vaccinated) ont été inclut dans la tâche de clustering et une sorte de normalisation a été effectuée sur le jeu de données covid19, elle consiste à diviser les valeurs des chacunes des dimensions citées précédemment par la taille de la population et multiplié par 100 afin d'obtenir un pourcentage à la place d'un nombre.

Jeu de données	*K*	*Dim*	*m*
Covid Data	3	4	206
Covid Data Cases	3	1	206
Covid Data Deaths	3	1	206
Covid Data Vaccinations	3	1	206
Covid Data Fully Vaccinated	3	1	206

Tableau 5.3 : Résumé des jeux de données de Covid19

Dans le tableau 5.3, les quatre derniers jeux de données sont extraites due premier jeu de donnée Covid Data en choisissant à chaque foie un seul attribut.

5.3.L'évaluation des résultats

5.3.1. Mesures externes et internes

Pour faire l'évaluation on a utilisé la mesure externe F-mesure [38]. La fonction fait la comparaison d'un jeu de donnée de la qualité du clustering au classes données et les classes correctes.

Chaque classe R_i contient N_i points de données, chaque cluster C_j (généré par l'algorithme) est considéré comme l'ensemble de N_j points de données. N_{ij} donne le nombre de

points de la classe R_i dans le cluster C_j et N donne le nombre total des points du jeu de données. Pour chaque classe R_i et un cluster C_j , la précision et le rappel sont alors défini comme [6] :

$$Prec(R_i, C_j) = \frac{N_{ij}}{N_j} \; et \; Rep(R_i, C_j) = \frac{N_{ij}}{N_i} \qquad (5.1)$$

Et la valeur de F-mesure correspondante est :

$$Fmes(R_i, C_j) = \frac{(b^2+1)\cdot Prec(R_i,C_j)\cdot Rep(R_i,C_j)}{b^2\cdot Prec(R_i,C_j)+Rep(R_i,C_j)} \qquad (5.2)$$

Où des coefficients égaux de Prec(R_i ,C_j) et Rep(R_i ,C_j) sont obtenu si b=1. La valeur globale de F-mesure F pour toute la partition est calculée comme

$$F(C) = \sum_{i=1}^{k'} \max_{C_i \in C}(Fmes(R_i, C_j)) \qquad (5.3)$$

La deuxième évaluation est en utilisant la fonction objective interne SSE. Qui fait le calcul de la somme de la distance entre des points de données et leur centroïdes de cluster correspondant

$$SSE = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \qquad (5.4)$$

Où :

y_i : la valeur observée

$\hat{y}_i$: la valeur estimée par la ligne de régression

Finalement on fait le calcul de nombre d'évaluations de la fonction objective FE pour avoir une idée de sa vitesse et sa qualité de convergence.

5.3.2. Représentations graphique par Boxplot

L'outil Boxplot est intégré dans MATLAB qui nous permet de simplifier la visualisation des résultats pour chaque jeu de données d'une manière claire et simple (voir Figure 5.1)

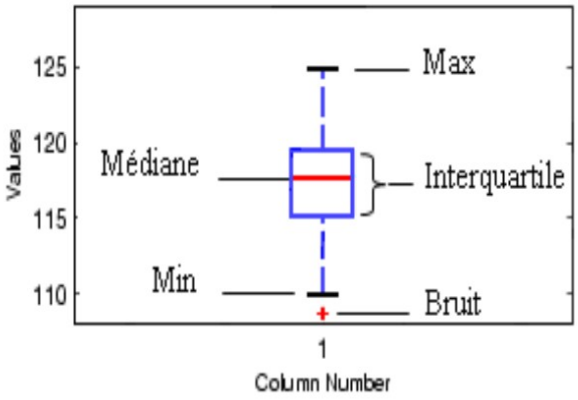


Figure 5.1 : Informations données par un Boxplot

5.4 Résultats des tests

A. Tests sur les jeux de données réels et synthétiques

Pour les sept jeux de donnée présentés ci-dessus, nous avons choisi les paramètres suivants pour l'algorithme CS :

- *Nombre de population* : 25
- *Nombre d'itérations* : 1000
- *Probabilité Pa :* 0.25
- *Nombre d'exécutions :* 30

Tous les résultats sont montrés dans les tableaux ci-dessous (Tableau 5.4, 5.5, 5.6) qui montrent les valeurs de médiane, l'interquartile, le maximum et le minimum de chacune des mesures F-mesure, SSE et FE, suivi par leurs visualisations en utilisant les Boxplots (Figure 5.2 ,5. 3, 5. 4)

Jeu de données		***Cuckoo Search***
2d4c	Médiane	0.7171
	Interquartile	0.1257
	Maximum	0.9446
	Minimum	0.5362
2d10c	Médiane	0.6612
	Interquartile	0.0717
	Maximum	0.9056
	Minimum	0.5491
10d10c	Médiane	0.6445
	Interquartile	0.1460
	Maximum	0.9706
	Minimum	0.4402

10d20c	Médiane	0.3187
	Interquartile	0.0343
	Maximum	0.4161
	Minimum	0.2657
Iris	Médiane	0.7741
	Interquartile	0.1584
	Maximum	0.9465
	Minimum	0.4606
Dermatology	Médiane	0.4859
	Interquartile	0.1061
	Maximum	0.6278
	Minimum	0.3796
Wisconsin	Médiane	0.7937
	Interquartile	0.1751
	Maximum	0.9648
	Minimum	0.5815

Tableau 5.4 : Médiane, interquartile, max et min de la F-mesure obtenus

Jeu de données		***Cuckoo Search***
2d4c	Médiane	1887.8465
	Interquartile	0.0085
	Maximum	1887.8680
	Minimum	1887.8428
2d10c	Médiane	530.7568
	Interquartile	4.5649
	Maximum	541.3295
	Minimum	523.1658
10d10c	Médiane	3318.5833
	Interquartile	114.2121
	Maximum	3434.2265
	Minimum	3120.0060
10d20c	Médiane	4114.1865
	Interquartile	66.5226
	Maximum	4252.7341
	Minimum	4030.4276
Iris	Médiane	96.6557
	Interquartile	0.0005
	Maximum	96.6578
	Minimum	96.6555

Dermatology	Médiane	1509.7061
	Interquartile	33.7341
	Maximum	1543.5824
	Minimum	1338.0480
Wisconsin	Médiane	2964.3888
	Interquartile	0.0018
	Maximum	2964.3918
	Minimum	2964.3870

Tableau 5.5 : médiane, interquartile, max et min de la fonction objective SSE obtenus

Jeu de données		***Cuckoo Search***
2d4c	Médiane	24988
	Interquartile	12
	Maximum	25000
	Minimum	24976
2d10c	Médiane	24988
	Interquartile	12.0000
	Maximum	25000
	Minimum	24976
10d10c	Médiane	24988
	Interquartile	11
	Maximum	25000
	Minimum	24977
10d20c	Médiane	24991
	Interquartile	9
	Maximum	24998
	Minimum	24977
Iris	Médiane	24989
	Interquartile	14
	Maximum	25000
	Minimum	24976
Dermatology	Médiane	24986
	Interquartile	7
	Maximum	24999
	Minimum	24976
Wisconsin	Médiane	24985
	Interquartile	14
	Maximum	25000
	Minimum	24976

Tableau 5.6 : Médiane, interquartile, max et min de FE obtenus

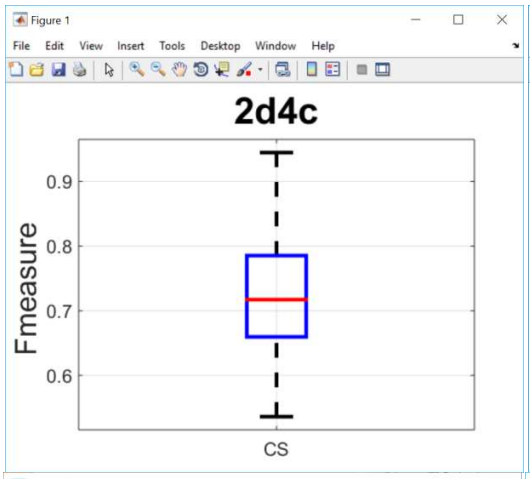


Figure 1
File Edit View Insert Tools Desktop Window Help
2d4c
Fmeasure
0.9
0.8
0.7
0.6
CS

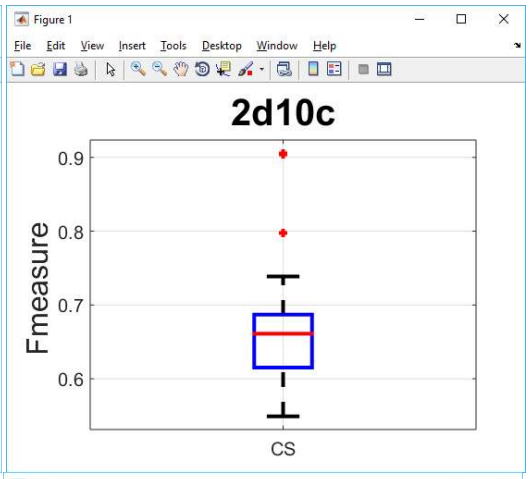


Figure 1
File Edit View Insert Tools Desktop Window Help
2d10c
Fmeasure
0.9
0.8
0.7
0.6
CS

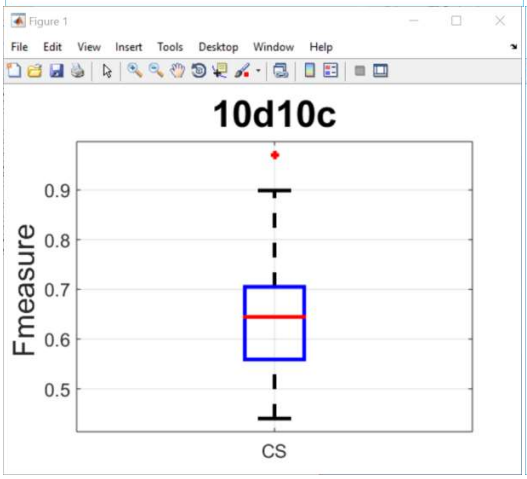


Figure 1
File Edit View Insert Tools Desktop Window Help
10d10c
Fmeasure
0.9
0.8
0.7
0.6
0.5
CS

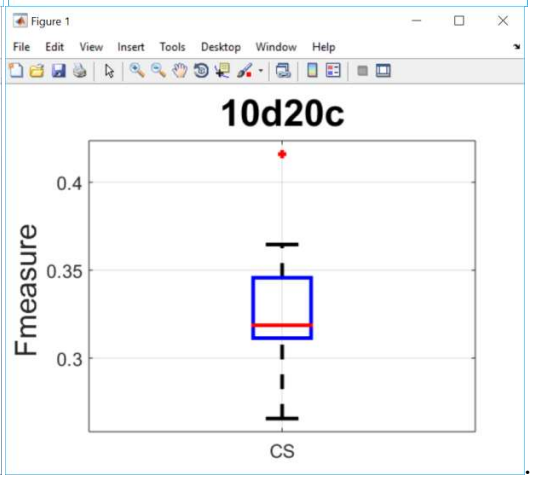


Figure 1
File Edit View Insert Tools Desktop Window Help
10d20c
Fmeasure
0.4
0.35
0.3
CS

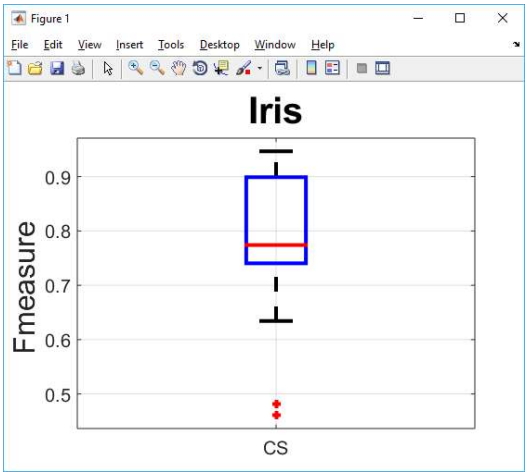


Figure 1
File Edit View Insert Tools Desktop Window Help
Iris
Fmeasure
0.9
0.8
0.7
0.6
0.5
CS

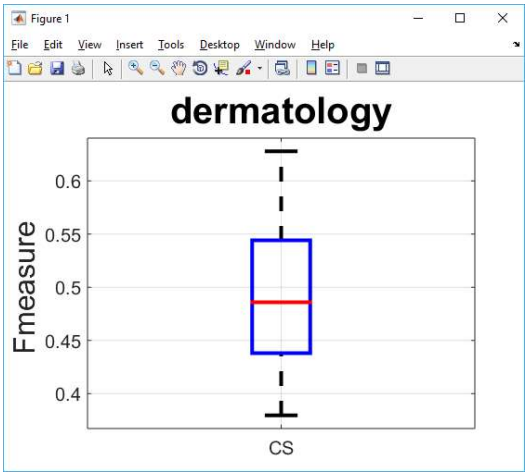


Figure 1
File Edit View Insert Tools Desktop Window Help
dermatology
Fmeasure
0.6
0.55
0.5
0.45
0.4
CS

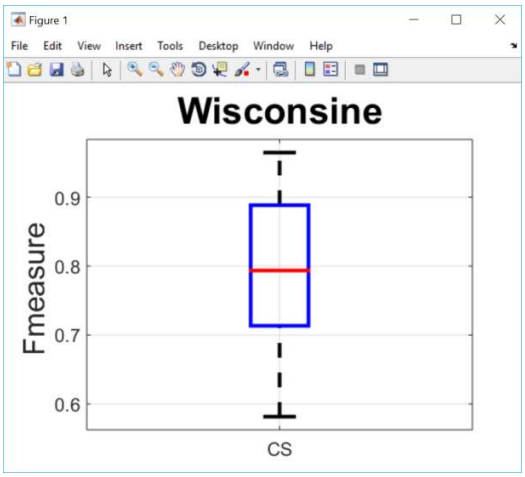
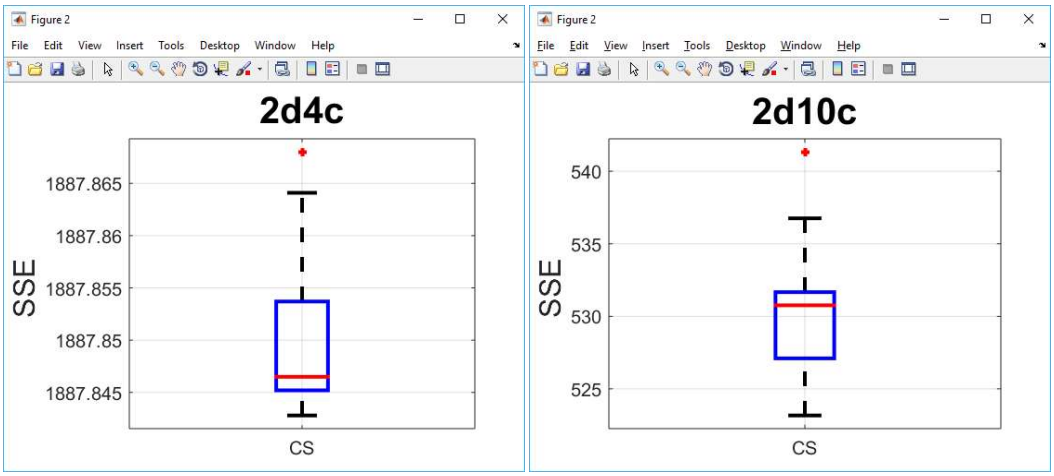
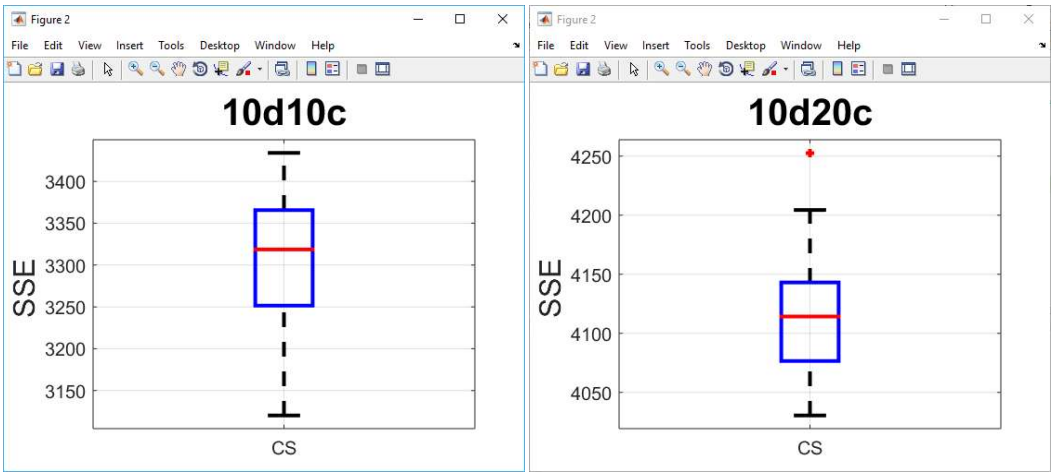


Figure 5.2 : Boxplots des résultats évalués avec F-Mesure





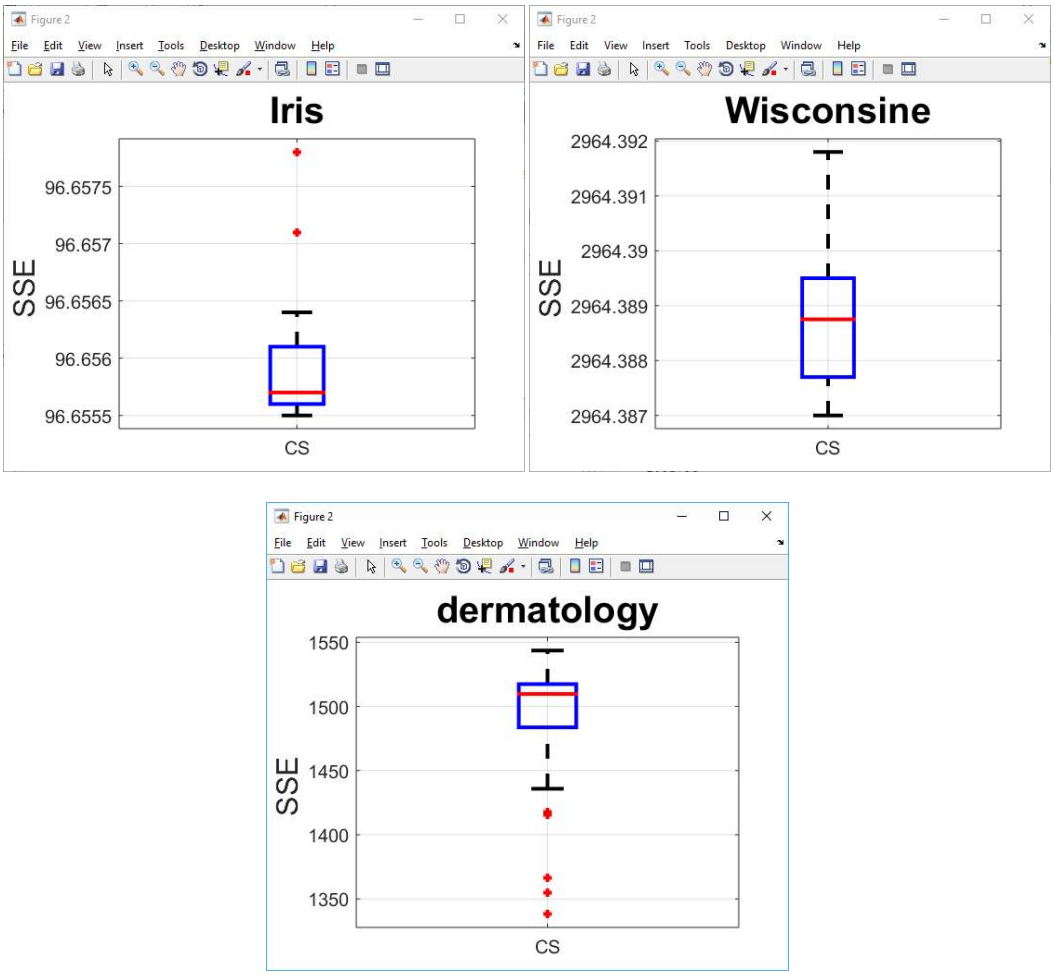
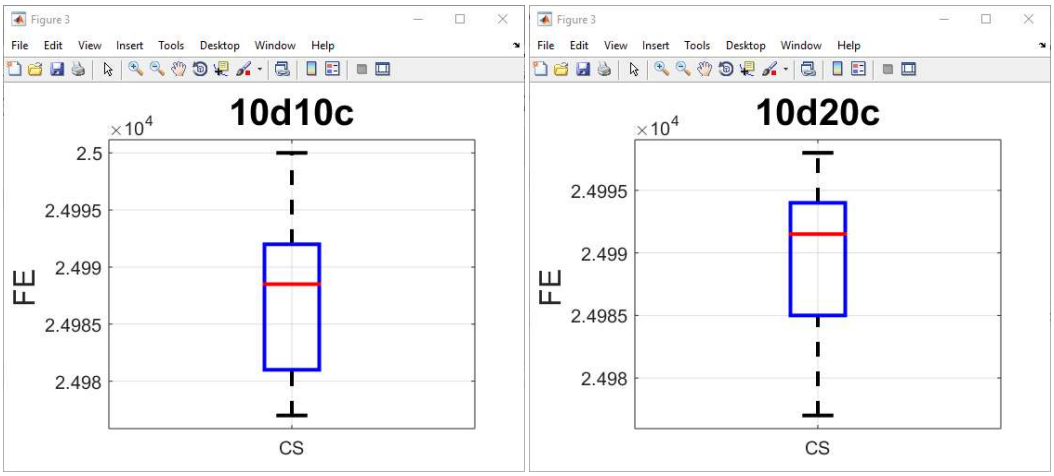
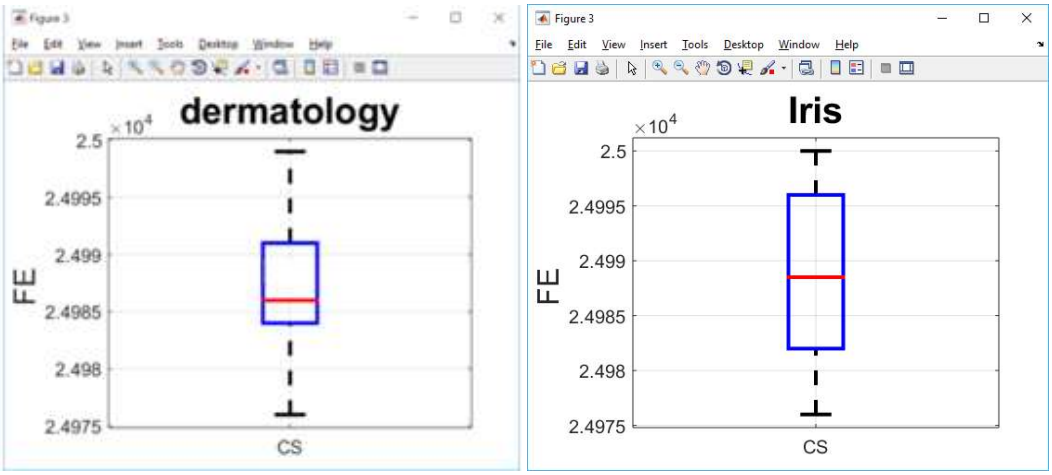


Figure 5.3 : Boxplots des résultats évalués avec SSE





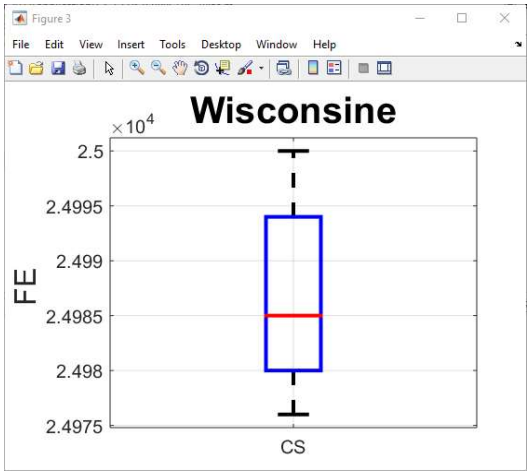


Figure 5.4 : Boxplots des résultats évalués avec FE

B. Tests sur le jeu de données de COVID19

Nous avons appliqué l'algorithme CS pour le clustering du jeu de données COVID19 en fixant les paramètres suivants :

- *Nombre de population* : 25
- *Nombre d'itérations* : 1000
- *Probabilité Pa :* 0.25
- *Nombre d'exécutions :* 1

Les résultats numériques sont montrés dans les tableaux ci-dessous (Tableau 5.7, 5.8) qui montrent les valeurs de chacune des mesures SSE et FE.

Jeu de données	*Cuckoo Search*
Covid Data	5491.6302
Covid 3 dim	2690.8265
Covid Data Cases	826.9793
Covid Data Deaths	6.5998
Covid Data Vaccinated	4324.1162
Covid Data Fully Vaccinated	1566.6940

Tableau 5.7 : Valeurs de la fonction objectif SSE obtenus

Jeu de données	*Cuckoo Search*
Covid Data	24991
Covid 3 dim	25000
Covid Data Cases	24987
Covid Data Deaths	24977
Covid Data Vaccinated	24983
Covid Data Fully vaccinated	25000

Tableau 5.8 : Valeurs de FE obtenus

La représentation graphique du clustering du jeu de données covid19 est représentée dans la figure 5.5. et puisque le jeu de données est constitué de 4 dimensions, nous avons présenté la distribution de résultat de clustering selon les 6 paires de dimensions.

La figure 5.11 représente le résultat de clustering des pays sur la carte géographique.

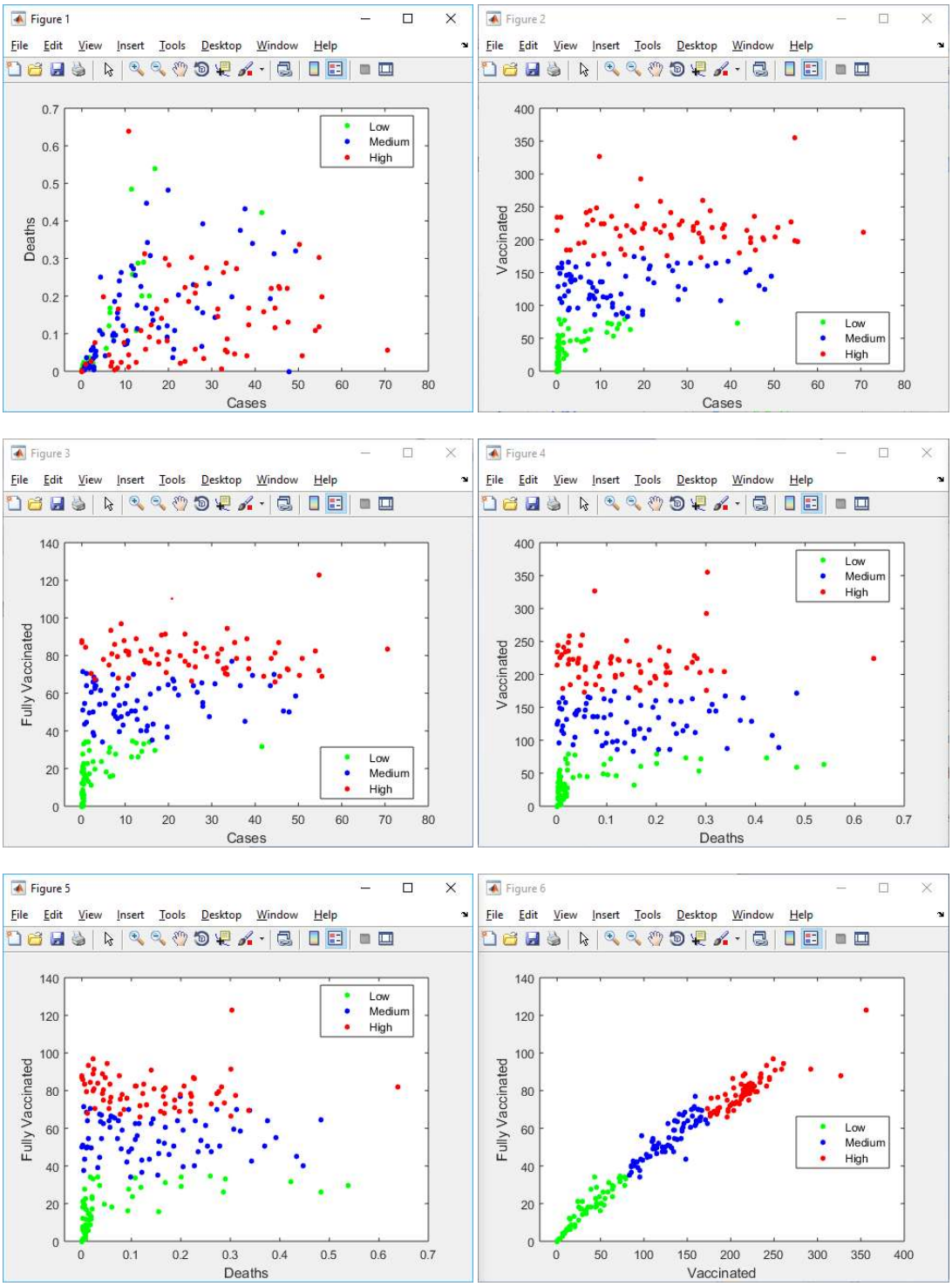


Figure 5.5 : Représentation graphique de clustering du jeu de données de COVID19

La figure 5.6 et 5.12 représente le clustering du jeu de données covid19 en prenant en compte 3 dimensions (*Cases, Deaths, Fully vaccinated*).

Les figures 5.7, 5.8, 5.9, 5.10 représentent le clustering d'une seule dimension (cases, deaths, Vaccinated, Fully Vaccinated respectivement) .

Les figures 5.11, 5.12, 5.13, 5.16 donne la représentation sur une carte géographique du clustering des pays basé sur une seule dimension (cases, deaths, Vaccinated, Fully Vaccinated respectivement).

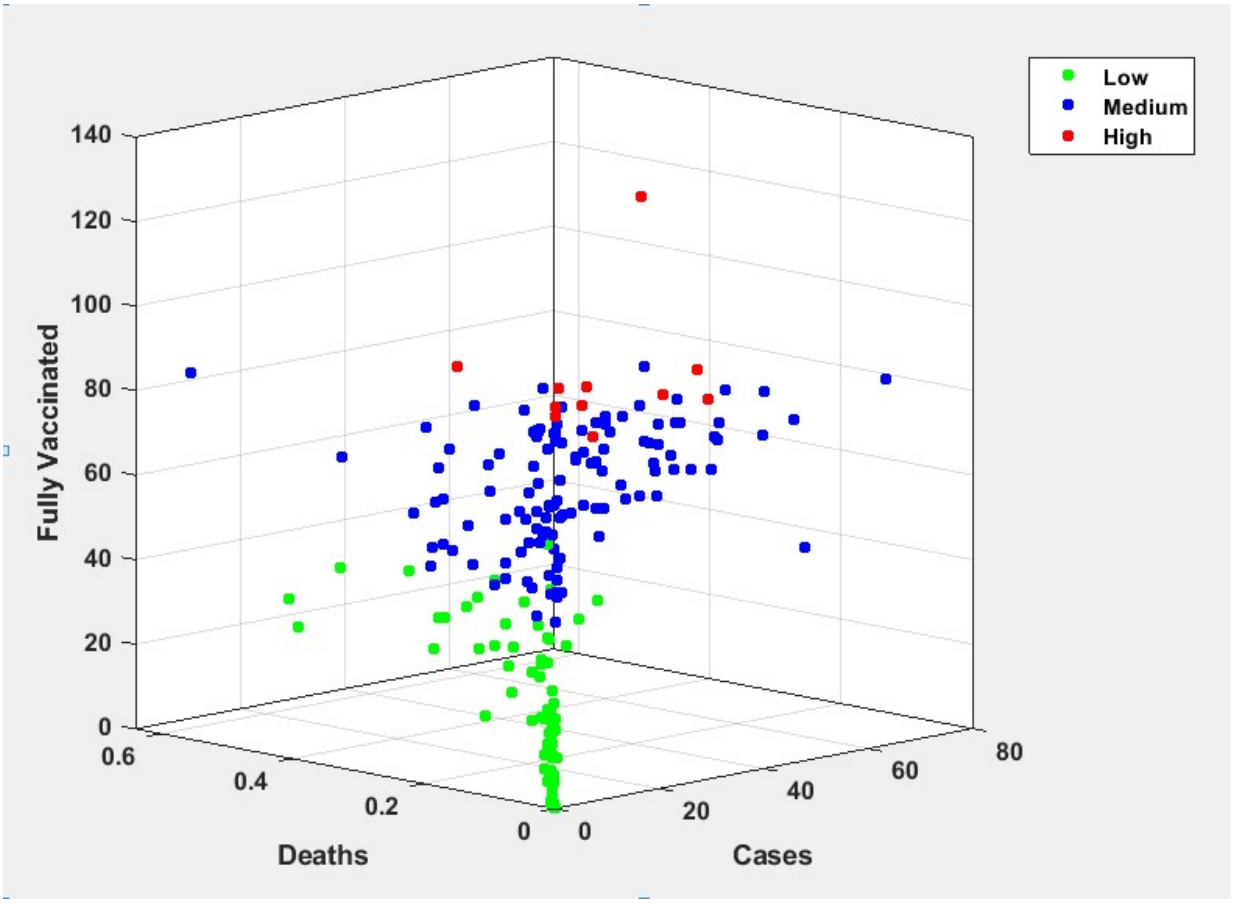


Figure 5.6 : Représentation graphique en 3D de clustering du jeu de données COVID19

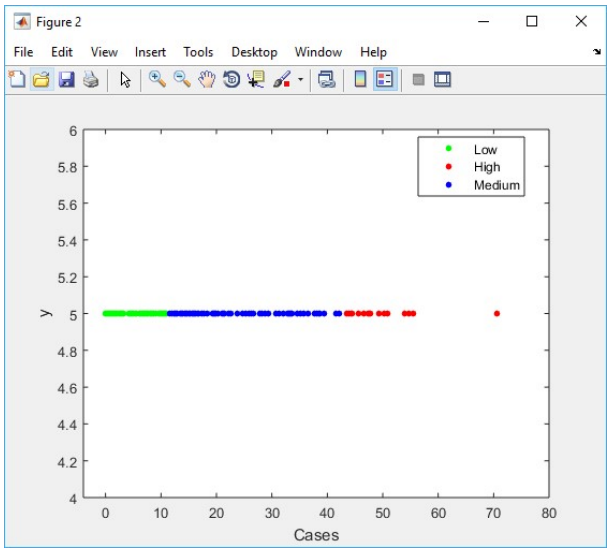


Figure 5.7 : Représentation de clustering des cas (attribut : cases)

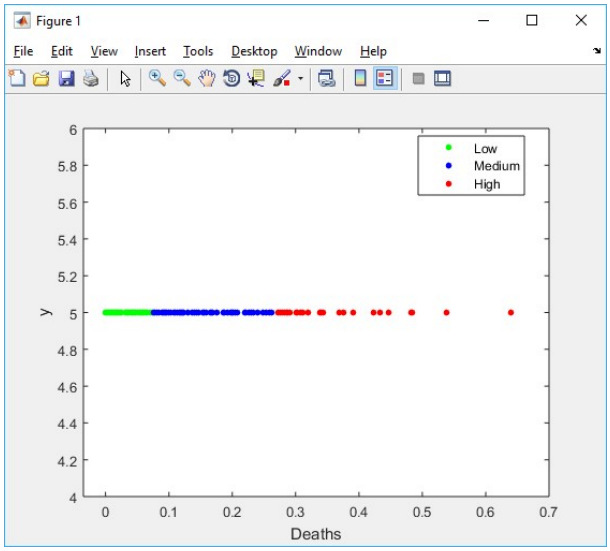


Figure 5.8 : Représentation de clustering des morts (attribut : Deaths)

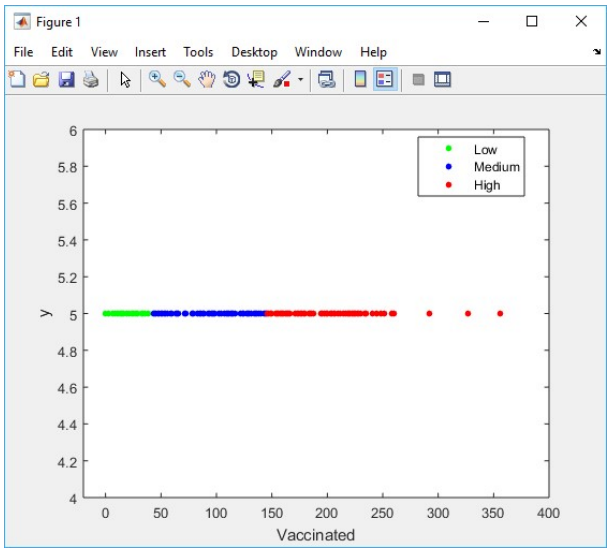


Figure 5.9 : Représentation de clustering des personnes vaccinées (attribut : Vaccinated)

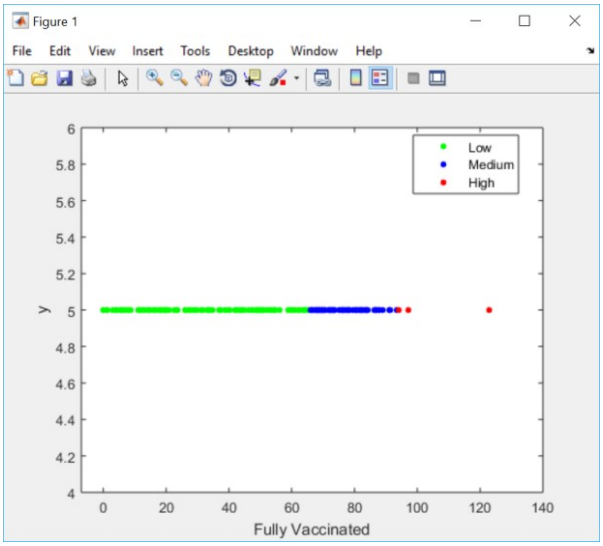


Figure 5.10 : Représentation de clustering des personnes complètement vaccinées (attribut : Fully Vaccinated)

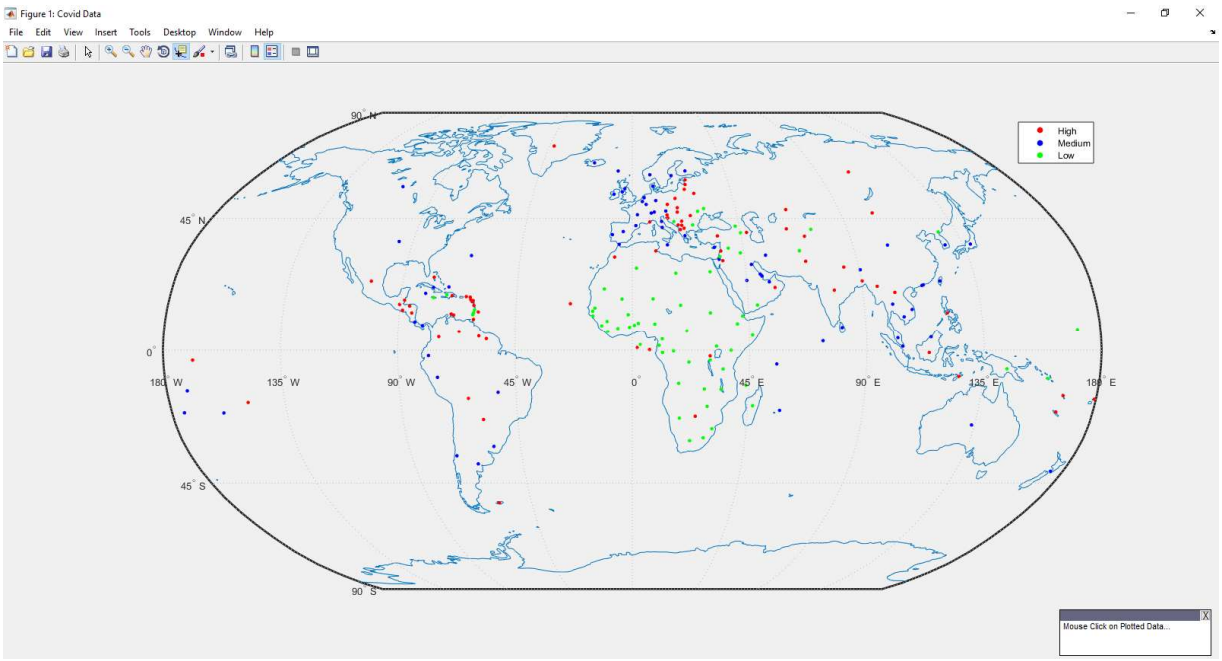


Figure 5.11 : Représentation graphique du clustering des pays sur une carte géographique

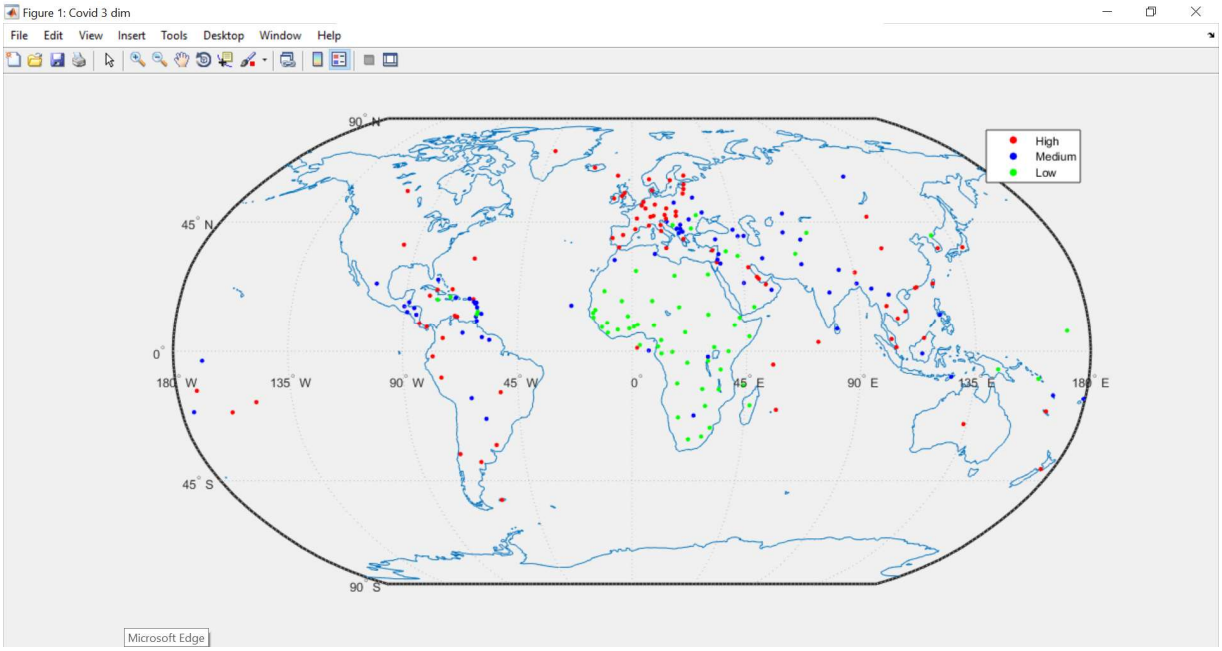


Figure 5.12 : Représentation graphique du clustering des pays basé sur 3 attributs (Cases, Deaths, Fully Vaccinated) sur une carte géographique.

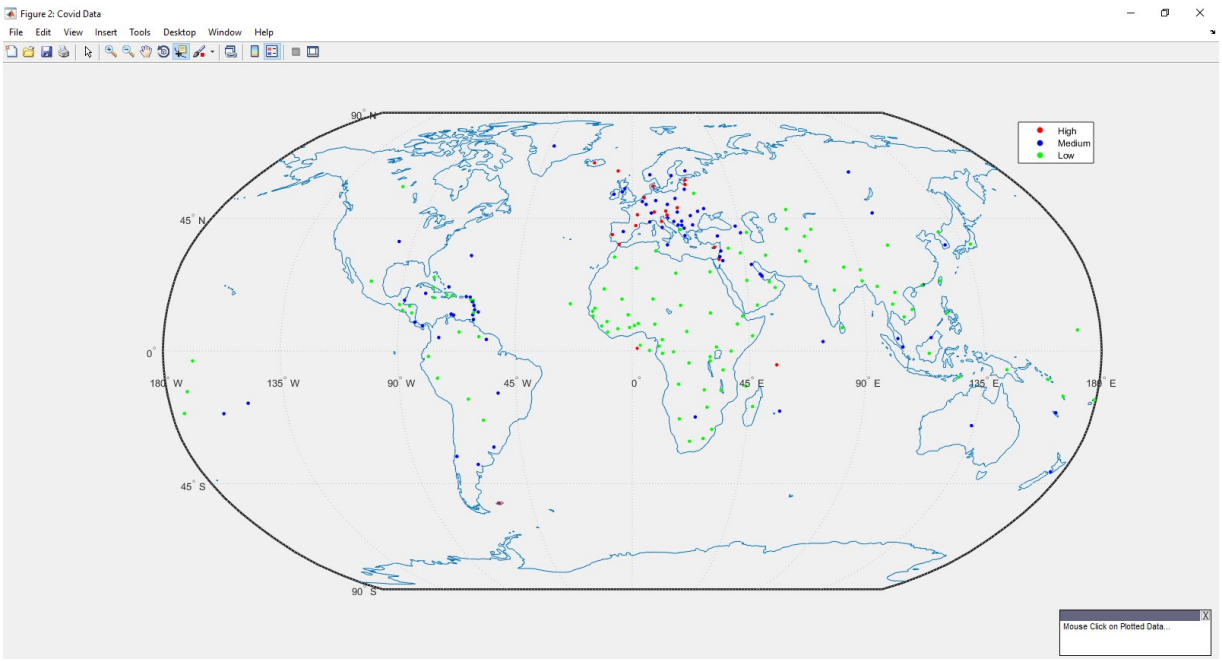


Figure 5.13 : Représentation sur la carte géographique de clustering des pays selon les cas (Cases).

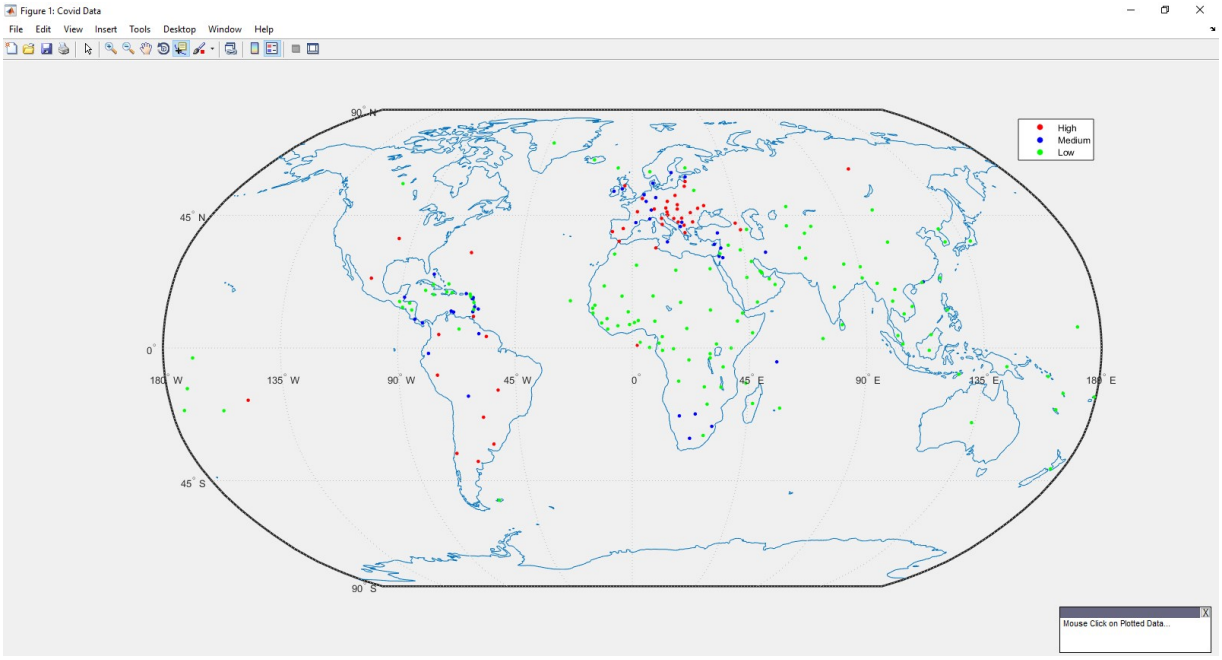


Figure 5.14 : Représentation sur la carte géographique de clustering des pays selon les morts (*Deaths*).

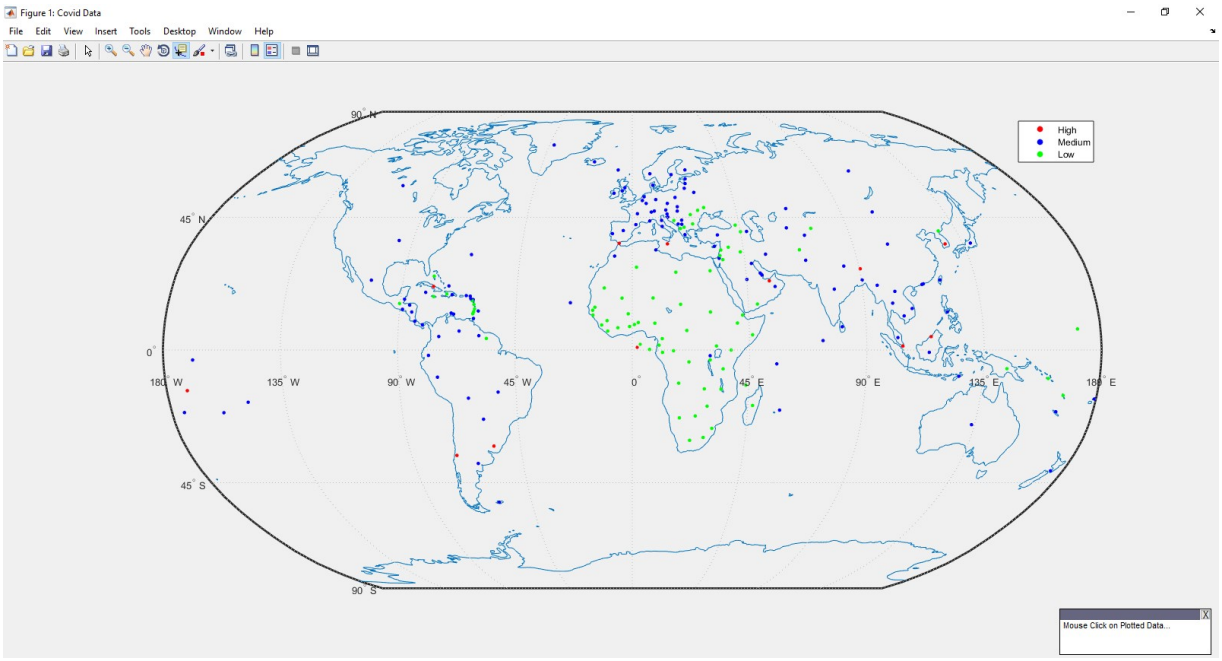


Figure 5.15 : Représentation sur la carte géographique de clustering des pays selon les vaccinées (*Vaccinated*).

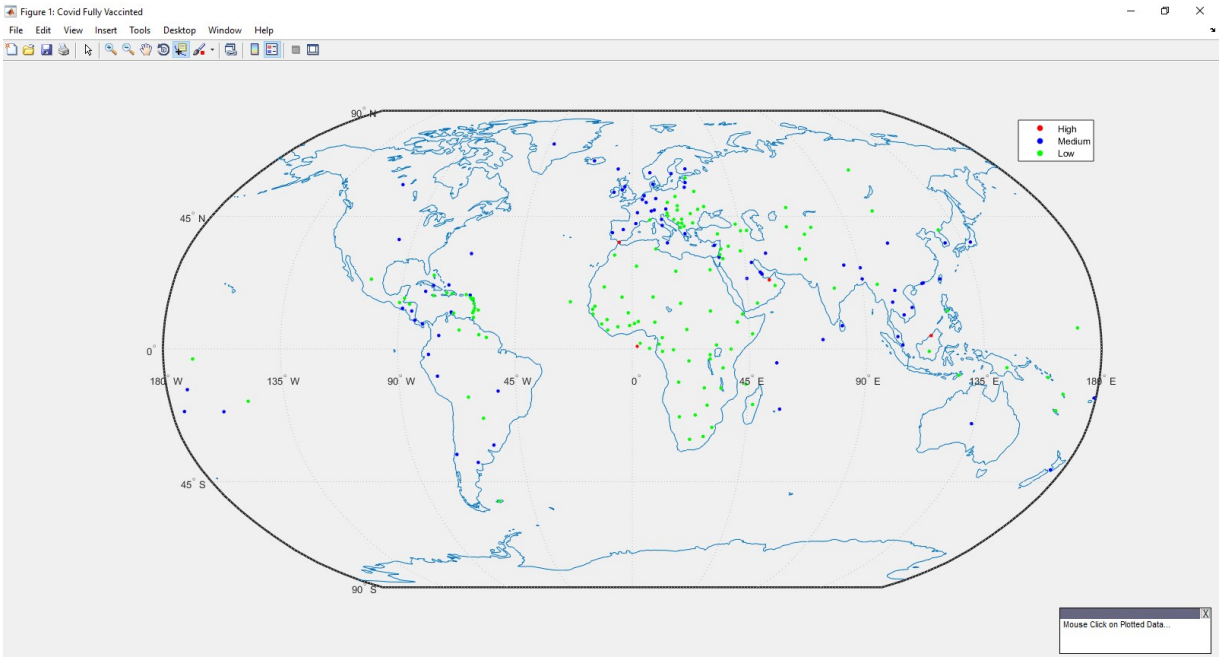


Figure 5.16 : Représentation sur la carte géographique de clustering des pays selon les personnes totalement vaccinées (*Fully Vaccinated*).

5.5 Analyse des résultats

Dans la figure 5.5, la représentation graphique des clusters basée sur les deux dimensions (Vaccinated et Fully vaccinated) montre qu'il y a une forte corrélation linéaire entre ces deux attributs ça veut dire qu'on peut se contenter d'un seul attribut parmi les deux. D'un autre coté, cela montre aussi qu'en général et quelque soit le taux de vaccination élevé ou faible, les personnes reçoivent le vaccin complet.

La représentation graphique des clusters basée sur les deux dimensions (cases et Fully vaccinated) montre qu'il y a un groupe de pays ayant un taux très faible de cas de contaminations et un taux faible de vaccinations et il y a un groupe de pays ayant un taux très faible de cas de contaminations et un taux élevé de vaccinations comme il y a aussi des pays ayant un taux élevé de cas et un taux élevé de vaccinations. Cela peut nécessiter des données supplémentaires sur la période d'administration de vaccin.

La représentation graphique des clusters basée sur les deux dimensions (Deaths et Fully vaccinated) montre qu'il y a un groupe de pays ayant un taux faible de morts et un taux faible de vaccinations et il y a un groupe de pays ayant un taux moyen de morts et un taux élevé de vaccinations.

La figure 5.7 montre que la vaccination contribue à réduire le taux des morts plus que la réduction du taux de cas de contamination.

Sur la carte géographique représentée dans la figure 5.11, pour le jeu de données complet on remarque que l'algorithme regroupe presque tous le continent d'Afrique avec une sévérité basse (*Low*) plus que d'autre pays Asiatique, pour l'Europe le coté Ouest est moins sévère que l'Est ainsi que la majorité de l'Asie.

Sur la carte géographique représentée dans la figure 5.13, pour la carte des Cas presque la majorité de l'Afrique et l'Asie enregistrent des taux bas et un mélange de taux bas et moyen sont enregistré en Amérique, par contre l'Europe c'est du taux bas et haut de cas.

Même remarque pour le jeu de données de Morts,(figure 5.14) l'Afrique et l'Asie enregistrent des taux bas par contre l'Amérique enregistre des taux bas et moyen et l'Europe enregistrent des taux moyen et élevé de morts.

Pour la vaccination sur la carte géographique de la figure 5.15, L'Afrique enregistre des taux faible de vaccinations. Le reste des pays enregistrent des taux moyens et il ya peu de pays qui ont enregistré un taux élevé de vaccination.

Pour les jeux de données synthétiques et réelles, l'algorithme CS détecte les clusters avec un taux plus de 70% pour 2d4c, Iris et Wisconsin. Et un taux plus de 64 % pour 2d10c et 10d10c. et un taux faible pour Dermatology et 10d20c.

5.6 Conclusion

Dans ce chapitre, nous avons présenté les expérimentations effectués sur plusieurs jeux de données synthétiques et réels .Nous avons concentré sur les expérimentations sur des données de COVID19. L'évaluation de résultats est faite avec plusieurs mesures. A la fin, nous avons présenté une analyse et une discussion de nos résultats.

Conclusion Générale

Conclusion générale

Au cours de ce travail de master, nous nous somme intéressés au problème de clustering de données, plus précisément nous avons présenté une application de la méthode de la recherche de coucou au clustering des pays en utilisant des données épidémiologiques de la COVID19.

Dans ce mémoire, nous avons commencé par présenter les notions et les concepts de base du domaine de clustering de données, tels que les définitions de clustering et de cluster, les techniques principales et techniques de validation. Ensuite nous avons passé à étudier les concepts, les principes et le comportement de la méthode de recherche de coucou ainsi que la description détaillée de son algorithme.

Ensuite, nous avons passé à l'adaptation de la méthode de la recherche de coucou au clustering des données de covid 19 en menant une série d'expérimentations, d'évaluation et d'analyse basée sur des mesures internes et externes et des représentations graphiques dans des espaces de 2 et 3 dimensions ainsi que la représentation des résultats sur la carte géographique.

A la fin, ce travail ouvre la voie pour différents enrichissements qui peuvent être apportés à notre application tels que le passage au clustering spectral et au clustering spatial pour traiter ce genre de données.

Bibliographie

[1] : Laurent Candillier, rapport technique : "La classification non supervisée", Septembre 2004.

[2] : V. Kumar, rapport technique: "An Introduction to Cluster Analysis for Data Mining", C.S. Dept. Univ. Minnesota, 2000.

[3] : P. Berkhin, Rapport techniques: "Survey of Clustering Data Mining Techniques", Accrue software, San Jose, California, 2002.

[4] : A. K. Jain et R.C. Dubes: "Algorithms for Clustering Data", Prentice Hall series de reference avancée, 1988.

[5] : Laetitia Jourdan, thèse de doctorat : "Métaheuristiques pour l'extraction de connaissances application à la génomique", Novembre 2003.

[6] : Ramdane Chafika, mémoire de magistère :"Le clustering des données : une nouvelle approche évolutionnaire quantique ", Université Mentouri de Constantine, Juin 2006.

[7] : D.P.Mercer, linacre college : " Clustering large Datasets ", Octobre 2003

[8] : Krishnapuram R. et Keller, J.M. " A possibilistic approach to clustering", IEEE Trans.Fuzzy Syst. 1,pp : 98–110, 1993.

[9] : Yang, M.S. et Wu, K.L. 'Unsupervised possibilistic clustering', Pattern Recognition,Vol. 39, No. 1, pp.5–21., 2006.

[10] : Ramdane Chafika, thése de doctorat :" Apports du Calcul Quantique et des Concepts Possibilistes à la Classification non supervisée Evolutionnaire ", Université Mentouri de Constantine, Novembre 2013.

[11] : Wu X. , Wu,B., Sun,J. et, Fu,H " Unsupervised Possibilistic Fuzzy Clustering", Journal of Information & Computational Science 7:5 pp. 1075–1080, 2010.

[12] : J.Bilmes, A.Vahdat, W.Hsu et E.J.Im. "Empirical observations of probabilistic heuristics for the clustering problem". Technical Report TR-97-018, International Computer Science Institute, University of California, Berkeley, CA, 1997.

[13] : Z. Michalewicz, Genetic Algorithms + Data Structures" Evolution Programs, Springer, New York, 1992.

[14] : Eberhart RC, Kennedy J (1995) A new optimizer using particle swarm theory. In: Proceedings of the 6th international symposium on micro machine and human science, pp 39–43, Nagoya, Japan, Mar 13–16, 1995

[15] : Kennedy J, Eberhart RC (1995) Particle swarm optimization? In: Proceedings of the IEEE international conference on neural networks, pp 1942–1948, Perth, Australia

[16] : Clerc M, Kennedy J (2002) The particle swarm-explosion, stability and convergence in a multi dimensional complex space. IEEE Trans Evolut Comput 6(2):58–73

[17] : M. Halkidi, Y. Batistakis, M. Vazirgiannis, “On Clustering Validation Techniques”, Intelligent Information Systems Journal, Kluwer Pulishers, vol.17, n°2-3, pp. 107- 45, 2001.

[18] : M. Halkidi, Y. Batistakis, M. Vazirgiannis. "Cluster Validity Methods: art II", SIGMOD Record, 2002

[19] : J Handl, Ant-based methods for tasks of clustering and topographic mapping: extensions, analysis and comparison with alternative techniques. Masters Thesis, Université d’Erlangen-Nurnberg, Erlangen, Germany, 2003.

[20] : M. Halkidi ,M.Vazirgiannis. "Clustering Validity Assessment: Finding the optimal partitioning of a data set ",in the Proceedings of ICDM Conference ,California, USA, 001.

[21] : Guillaume Cleuziou, thèse de doctorat : "Une méthode de classification nonsupervisée pour l’apprentissage de règles et la recherche d’information", Décembre 2004.

[22] : Julia Handl, mémoire de magistère:" Ant-based methods for tasks of clustering and topographic mapping, improvements, evaluation and comparison with alternative methods", 2003.

[23] : Guillaume Cleuziou, thèse de doctorat : "Une méthode de classification nonsupervisée pour l’apprentissage de règles et la recherche d’information", Décembre 2004.

[24] : Alexandre Blansché, thèse de doctorat :" La classification non supervisée avec pondération d'attributs par des méthodes évolutionnaires ", Septembre 2006.

[25] : J.Handl et J.Knowles. "Exploiting the trade-off -- the benefits of multiple objectives in data clustering". In the Proceedings of the Third International Conference on Evolutionary Multi-Criterion Optimization, pp 547-560, LNCS 3410, 2005.

[26] : K. Ando, K. Orimo, K. Ueyoshi, H. Yonekawa, S. Sato, H. Nakahara, M. Ikebe, T. Asai, S. Takamaeda-Yamazaki, T. Kuroda, and M. Motomura, "BRein Memory: A 13-layer 4.2 K neuron/0.8 M synapse binary/ternary reconfigurable in-memory deep neural network accelerator in 65 nm cmos," In 2017 IEEE Symposium on VLSI Circuits (VLSI-Circuits), pp. C24-C25, Kyoto, Japan, 2017.

[27] : K. Janocha and W.M. Czarnecki, "On loss functions for deep neural networks in classification," ArXiv e-prints, February 2017.

[28] : Bak, P: ″How nature works″. Oxford universitypress Oxford,1997.

[29] : Aziz OUAARAB :Thèse de Doctorat ″Résolution de Problèmes d'Optimisation Combinatoire par des Métaheuristiques Inspirées de la Nature : Recherche du Coucou via les Vols de Lévy″ ,2015.

[30] : Payne. R. B., Sorenson M. D., and Klitz K : The Cuckoos, Oxford UniversityPress, 2005.

[31] : R. Rajabioun. "CuckooOptimizationAlgorithm. Applied Soft Computing". Vol. 11, N° 8,pp. 5508-5518, 2011.

[32] : Reynolds, A. M. et Frye, M. A. : ″ Free-flight odortracking in drosophilais consistent with an optimal intermittent scale-free search″. PloS one, 2(4):e354, 2007.

[33] : Yang X. S. et Deb, S : " Engineering optimisation by cuckoosearch".

International Journal of MathematicalModelling and Numerical Optimisation, 1(4):330{343,

2010.

[34] : Xin-She Yang :″CuckooSearch and FireflyAlgorithm″: Theory and Applications; Studies in ComputationalIntelligence ,Vol. 516 Springer International Publishing; 2014.

[35] : Ishak B.S, Kamel.N et Bendjeghada.O : Article " A New Algorithm for Data Clustering Based on Cuckoo Search Optimization " Advances in Intelligent Systems and Computing 238, Springer International Publishing Switzerland 2014.

[36] : Handl, J. and Knowles, J. "Improvements to the scalability of multiobjective clustering ",IEEE Congress on Evolutionary Computation, Vol. 3, pp.2372–2379. (2005a)

[37] : Blake, C.L. and Merz, C.J. (1998) "UCI repository of machine learning databases", disponible en http://www.ics.uci.edu/~mlearn/Machine-Learning.html.

[38] : Stein, B., Eissen, S.M.Z. and Wißbrock, F. ″ On cluster validity and the information need of users ″ , 3rd IASTED Int. Conference on Artificial Intelligence and Applications, pp.216–221, 2003.

[39] : Clustering analysis of countries using the COVID-19 cases dataset, Available online 29 May 2020, https://www.arcgis.com/apps/opsdashboard/index.html#/bda7594740fd40299423467b48e9ecf6

[40] : Yang X. S. et Deb, S : " Engineering optimisation by cuckoo search". International Journal of Mathematical Modelling and Numerical Optimisation, 1(4):330{343, 2010.