MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE

Université 20 août 1955 Skikda

Faculté des sciences

Département d'Informatique

THÉSE DE DOCTORAT

Soumise en conformité avec les exigences du diplôme de doctorat 3^{eme} cycle

Spécialité : Informatique

Option : Réseaux et Systèmes distribués

Intitulée :

Contribution au Clustering de données : Adéquation d'approches et défis de domaines d'application

---

Défendue publiquement par :

M. Ibrahim Zebiri

le 30 Avril 2024, devant le jury composé de :

Pr. Bachir Boucheham Université 20 Août 1955, Skikda	**Président**
Pr. Mohamed Benmohammed Université Abdelhamid Mehri Constantine 2	**Examinateur**
Dr. Mehdi Boulaiche (MCA) Université 20 Août 1955, Skikda	**Examinateur**
Pr. Mohammed Redjimi Université 20 Août 1955, Skikda	**Encadrant**
Dr. Djamel Zeghida (MCA) Université 20 Août 1955, Skikda	**Co-encadrant**

MINISTRY OF HIGHER TEACHING AND SCIENTIFIC RESEARCH

University 20 août 1955 Skikda

Faculty of Sciences

Department of Computer Science

DOCTORAL DISSERTATION

Submitted in partial fulfillment of the requirements of the **doctoral** degree
3^{rd} cycle

Major : Computer Science

Minor : Computer Networks and Distributed Systems

Entitled :

Contribution to Data Clustering : Suitability of Approaches and Challenges in Application Domains

---

Publicly defended by :
M. Ibrahim Zebiri

on April 30, 2024, in front of the examination committee composed of :

Pr. Bachir Boucheham University 20 Août 1955, Skikda	**Main Examiner**
Pr. Mohamed Benmohammed University Abdelhamid Mehri Constantine 2	**Examiner**
Dr. Mehdi Boulaiche (MCA) University 20 Août 1955, Skikda	**Examiner**
Pr. Mohammed Redjimi University 20 Août 1955, Skikda	**Supervisor**
Dr. Djamel Zeghida (MCA) University 20 Août 1955, Skikda	**Co-Supervisor**

I would like to dedicate this thesis to:

my **grandmother**, May ALLAH have mercy on her,

my **loving parents**,

and **Gaza**.

Declaration

I hereby declare that except where specific reference is made to the work of others. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 20,000 words including bibliography, tables and equations. It has more than 108 headers, 32 figures, 24 tables, 597 Math Inline, and 88 Math Display.

Ibrahim Zebiri
May 2024

Acknowledgements

In the Name of ALLAH, the Most Beneficent, the Most Merciful
Alhamdulillah, All the praises be to ALLAH, the Lord of the worlds
May Allah's blessing go to his servant and messenger Muhammad (peace be upon him), his family and his companions.

First and foremost, I must acknowledge my limitless thanks to Allah, the Ever-magnificent, the Ever-Thankful, for His help and bless by giving me the opportunity, courage and enough energy to carry out and complete this PhD dissertation.

Too many people contributed to this work to list them all. However, The author would like to acknowledge in these few lines his parents "My Lord! Bestow on them Your Mercy as they did bring me up when I was child".

The author would also like to acknowledge his supervisors for their guidance and assistance through the completion of this work, his supervisor Pr. M. Redjimi and co-supervisor Dr. D. Zeghida.

The author would like to thank the examiners; Pr. B. Boucheham, Pr. M. Benmohammed and Dr. M. Boulaiche for their willingness to examine this dissertation.

With many thanks to Dr. N.S. Sani for her precious advice and support.

The author is also grateful to his dear friends Omar Ouzzir and Sidahmed Defla for their assistance.

The author would also like to thank all those who he could not mention in name one by one.

Constantine, February, 2024

Ibrahim Zebiri

Abstract

We are swamped in our daily lives by an ever-growing torrent of information, handled by intricate systems with insatiable demands for speed and efficiency. Gleaning knowledge and insights from this deluge requires adaptable techniques that flex with the harsh, ever-shifting realities of data processing. Data clustering, though classified as NP-Hard problem, shines as one such technique. To tackle these computationally prohibitive problems, we turn to metaheuristics, inspired by the marvels in the living world. These approximate solutions allow us to stay ahead of the curve, continually proposing new approaches and optimizing and combining existing ones to achieve high-performance, quality clustering.

This thesis offers a short comprehensive exploration of data clustering, delving into its fundamental concepts, motivations, and diverse applications across scientific fields. It provides an in-depth analysis of the state-of-the-art in data clustering, encompassing its objectives, applications, techniques, and the various measures used to evaluate clustering results. Furthermore, the thesis introduces optimization methods inspired by real-world phenomena like genetic algorithms, rat swarm optimizer, and grey wolf optimizer, and explores their potential for effective data clustering.

This work makes two notable contributions: the Rat Swarm Optimizer for Data Clustering (RSOC) and the Enhanced Grey Wolf Optimizer for Data Clustering (EGWAC). Both aim to address the limitations of some existing clustering techniques and enhance their performance.

RSOC adapts the swarm intelligence metaheuristic RSO to data clustering challenges. It leverages its ability to escape local optima and premature convergence while exploring a broad solution space to find optimal cluster centers. The discussion section showcases RSOC's performance on diverse datasets using various measures, including homogeneity, completeness, v-measure, purity, and error rate. Comparisons with state-of-the-art and recent algorithms demonstrate RSOC's adaptability and superior performance in most cases.

The second contribution (EGWAC) addresses an identifies the issue in the position updation mechanism of the original Grey Wolf Algorithm-based Clustering technique (GWAC). This issue arises from treating the order of cluster centers as significant for finding clusters, when in reality it only affects wolf position updates and can lead to inaccurate solutions. EGWAC optimizes this key element. Experiments on various data clustering benchmarks

comparing EGWAC against GWAC and other well-known algorithms, using measures like precision, recall, g-measure, purity, and entropy, demonstrate its overall capability to identify optimal clusters.

Keywords: Data clustering, hard clustering, metaheuristics, optimization, rat swarm optimizer for data clustering (RSOC), enhanced grey wolf algorithm for data clustering (EGWAC).

ملخص

نحن مغمورون في حياتنا اليومية في سيول متزايدة من المعلومات، يتعامل معها أنظمة معقدة لديها متطلبات كبيرة للسرعة والفعالية. يتطلب استخلاص المعرفة والمعاني العميقة من هذا الطوفان تقنيات تتكيف مع واقع معالجة البيانات القاسي والمتغير باستمرار. يبرز تجميع البيانات، على الرغم من تصنيفه كمشكلة صعبة حسابيًا، باعتباره إحدى هذه التقنيات. ولهذا، فإننا نلجأ إلى أساليب الاستدلال الخفيفة (Metaheuristics) ، المستوحاة من عجائب العالم الحيوي، لحل هذه المشاكل المحظورة حسابيًا. تتيح لنا هذه الحلول التقريبية البقاء في صدارة المنافسة، حيث نقترح باستمرار طرقًا جديدة ونعمل على تحسين وتهجين الطرق الحالية لتحقيق تجميع عالي الأداء والجودة.

تقدم هذه المذكرة عرضا مختصرا لتجميع البيانات، ويتعمق في مفاهيمه الأساسية ودوافعه وتطبيقاته المتنوعة عبر المجالات العلمية. يقدم تحليلًا معمقًا للحالة التقنية لتجميع البيانات، بما في ذلك أهدافه وتطبيقاته وتقنياته والمقاييس المختلفة المستخدمة لتقييم نتائج التجميع. بالإضافة إلى ذلك، يقدم البحث طرقًا للتحسين مستوحاة من ظواهر واقعية مثل الخوارزميات الجينية ومحاكيات أسراب الجرذان ومحاكيات الذئاب الرمادية، ويستكشف إمكاناتها لتجميع البيانات الفعال.

يقدم هذا العمل مساهمتين مهمتين: محاكاة سرب الجرذان لتجميع البيانات (RSOC) ومحاكاة ذئب الرمادي المحسنة لتجميع البيانات (EGWAC) . يهدف كلاهما إلى معالجة قيود تقنيات التجميع الحالية وتحسين أدائها.

RSOC هو تعديل على أسلوب الاستدلال الذكي بالحشود (RSO) تحديات تجميع البيانات. يستفيد من قدرته على الهروب من التطرف المحلي والتقارب المبكر أثناء استكشاف مساحة واسعة للحلول لإيجاد مراكز مجموعات مثالية. يعرض قسم المناقشة أداء RSOC على مجموعات بيانات متنوعة باستخدام مقاييس مختلفة، بما في ذلك التجانس والاكتمال ومقياس v والنقاء ومعدل الخطأ. تُظهر المقارنات مع الخوارزميات التقليدية والحديثة قدرة RSOC على التكيف والأداء العالي في معظم الحالات.

تعالج EGWAC مشكلة تم تحديدها في آلية تحديث الموقع لخوارزمية GWAC الأصلية لتجميع البيانات. تنشأ هذه المشكلة من التعامل مع ترتيب مراكز المجموعة على أنه مهم لإيجاد المجموعات، بينما في الواقع يؤثر فقط على تحديثات موضع الذئب ويمكن أن يؤدي إلى حلول غير دقيقة. EGWAC يحسن هذا العنصر الأساسي. تُظهر التجارب على معايير تجميع البيانات المختلفة التي تقارن EGWAC مع GWAC وخوارزميات أخرى معروفة، باستخدام مقاييس مثل الدقة والاستدعاء ومقياس g والنقاء والأنتروبي، قدرتها الكلية على تحديد المجموعات المثالية.

الكلمات المفتاحية: تجميع البيانات، التجميع المقيد، الاستراتيجيات الفوقية، التحسين، محسن سرب الجرذان لتجميع البيانات (RSOC) ، خوارزمية الذئب الرمادي المحسن لتجميع البيانات (EGWAC) .

Résumé

Nous sommes submergés dans nos vies quotidiennes par un torrent d'informations en constante augmentation, traité par des systèmes complexes aux exigences insatiables de rapidité et d'efficacité. Extraire des connaissances et des insights de ce déluge nécessite des techniques adaptables qui s'adaptent aux réalités dures et en constante évolution du traitement des données. Le clustering de données, bien que classé comme NP-difficile, brille comme l'une de ces techniques. Pour aborder ces problèmes informatiquement prohibitifs, nous nous tournons vers les métaheuristiques, inspirées des merveilles du monde vivant. Ces solutions

approximatives nous permettent de garder une longueur d'avance, en proposant continuellement de nouvelles approches et en optimisant et combinant les approches existantes pour obtenir un clustering de haute performance et de qualité.

Cette thèse propose une breve exploration du clustering de données, en plongeant dans ses concepts fondamentaux, ses motivations et ses diverses applications dans les domaines scientifiques. Elle fournit une analyse approfondie de l'état de l'art du clustering de données, englobant ses objectifs, ses applications, ses techniques et les différentes mesures utilisées pour évaluer les résultats de clustering. En outre, la thèse présente des méthodes d'optimisation inspirées de phénomènes du monde réel comme les algorithmes génétiques, les optimiseurs d'essaims de rats et les optimiseurs de loups gris, et explore leur potentiel pour un clustering de données efficace.

Ce travail apporte deux contributions notables : l'Optimiseur d'Essaim de Rats pour le Clustering de Données (RSOC) et l'Optimiseur de Loup Gris Amélioré pour le Clustering de Données (EGWAC). Tous deux visent à remédier aux limitations des techniques de clustering existantes et à améliorer leurs performances.

RSOC adapte la métaheuristique d'intelligence d'essaim (RSO) aux défis du clustering de données. Il exploite sa capacité à échapper aux optima locaux et à la convergence prématurée tout en explorant un large espace de solutions pour trouver des centres de clusters optimaux. La section discussion présente les performances de RSOC sur divers ensembles de données en utilisant diverses mesures, y compris l'homogénéité, l'exhaustivité, la v-mesure, la pureté et le taux d'erreur. Les comparaisons avec des algorithmes de pointe et récents démontrent l'adaptabilité et les performances supérieures de RSOC dans la plupart des cas.

EGWAC résout un problème identifié dans le mécanisme de mise à jour de position de l'algorithme GWAC d'origine pour le clustering de données. Ce problème provient du fait que l'ordre des centres de clusters est considéré comme significatif pour trouver des clusters, alors qu'en réalité il n'affecte que les mises à jour de position des loups et peut conduire à des solutions inexactes. EGWAC optimise cet élément clé. Des expériences sur divers benchmarks de clustering de données comparant EGWAC à GWAC et à d'autres algorithmes bien connus, en utilisant des mesures telles que la précision, le rappel, le g-mesure, la pureté et l'entropie, démontrent sa capacité globale à identifier des clusters optimaux.

Mots-clés: Clustering de données, clustering dur, metaheuristics, optimisation, Optimiseur de swarm de rats pour le clustering de données (RSOC), Algorithme du loup gris amélioré pour le clustering de données (EGWAC).

Table of contents

List of figures

List of tables

Nomenclature

Greek Symbols

$\delta(.,.)$	proximity function
f_{square}	square wave function
f_{den}	density function
μ	Cluster's center
∇	density function
σ	distance threshold

Subscripts

i	subscript index
j	subscript index
k	subscript index

Other Symbols

C	Clusters
D	dataset
d	number of data features
$d(.,.)$	distance function
K	number of clusters
n	dataset size
O	data object

$P(.,.)$ proximity function

$s(.,.)$ similarity function

$w(.)$ weight function

Acronyms / Abbreviations

ACROA Artificial Chemical Reaction Optimization Algorithm

BATC BAT algorithm-based Clustering

BH Black Hole algorithm

CIF Cauchy's Integral Formula

CRO Chemical Reaction Optimization

DBSCAN Density- Based Spatial Clustering of Applications with Noise

DE Differential Evolution

DENCLUE DENsity-based CLUstEring

EGWAC Enhanced Grey Wolf Optimizer for Data Clustering

Eq Equation

Fig Figure

FPAC Flower Pollination Algorithm-based Clustering

GA Genetic Algorithm

GSA Gravitational Search Algorithm

GWAC Grey Wolf Algorithm-based data Clustering

GWO Grey Wolf Optimizer

GWOTS hybrid Grey Wolf Optimizer and Tabu Search

H-GA Hybrid GA

H-HHO Hybrid Harris Hawks Optimization

H-KHA Hybrid Krill Herd Algorith

H-PSO Hybrid PSO

HGSO Henry Gas Solubility Optimization

HSC Harmony Search-based clustering

HS Harmony Search

KHA Krill Herd Algorithm

Lb Lower bound

MHSC Modified Harmony Search-based clustering

MVO Multi-Verse Optimize

NLP Natural Language Processing

PPF Past Present Future

PSO Particle Swarm Optimization

RSOC RSO-based Clustering

RSO Rat Swarm Optimizer

SPBO Student Psychology Based Optimization

std STandard Deviation

TLBO Teaching-Learning-Based Optimization

TSO Tuna Swarm Optimization

Ub Uower bound

UPGMA Unweighted Pair Group Method using Average

UPGMC Unweighted Pair Group Method using Centroid

WPGMA Weighted Pair Group Method using Average

WPGMC Weighted Pair Group Method using Centroid

Chapter 1

Introduction

1.1 Overview

The categorization of things and perceived objects is an innate and common task in our daily lives. Indeed, since the dawn of time and up to the present, humans have practiced grouping and classification in their everyday lives; at home, at work, in nature, or in the city. Whether it is a physical object or a moral or abstract value, everything around us is seen in an organized way. When we perceive a living being, we automatically distinguish it by the species to which it belongs: dog, cat, mosquito, bird, etc., or by the degree of the danger: dangerous, non-dangerous, semi-dangerous. Even objects are automatically classified in our brains: clothes, drinks, fruit or vegetables. The same goes for abstract things like feelings: sadness, fear, love, etc., or moral values: courage, altruism, kindness, malice and so on. This categorization process is a natural phenomenon to better understand and judge one's surroundings. In the last century, and from the second half onwards, humans began to automate their tasks and use computer tools for problem solving, including data classification and clustering. Data classification and clustering are two different tasks that should be differentiated.

Supervised classification assigns data objects to already known categories by "learning" from examples. It uses labeled examples ("training data") to figure out what goes in each category ("class"). Then, it can sort new, unknown things ("test data") into the right categories. Classification delves into understanding groups of "objects". Its goal is to determine whether these objects can be meaningfully grouped into classes based on their features. This methodology is applicable across countless disciplines.

Unsupervised classification (clustering), similar to classification, but with a twist! While classification groups things into existing classes, clustering discovers the classes themselves with no pre-defined categories exist; the patterns emerge from the data itself (no labeled data). Data clustering uncovers hidden groups and relationships within data, forming groups ("clusters") of similar objects.

Ultimately, the classification task is used to accurately assign new objects to known classes. Conversely, data clustering is used to unveil the hidden structures and uncover unknown relationships.

In this thesis we are interested in data clustering. It is a fundamental task in machine learning and data analysis. Along with the ever-exploding data volumes and the intensification real-world complexity, the need for new and optimized clustering techniques that can convoy the present challenges is indispensable. While existing clustering techniques seem to be handy in various situations, they can get tripped up by large, complex, or high-dimensional datasets. This thesis delves into the realm of data clustering and discusses some of the well-known algorithms along with their drawbacks which gave rise to our contributions in developing and optimizing some clustering techniques.

1.2 Where does it come from?

The central idea of this thesis arose from a conversation about the data clustering problem, its intersection with different scientific fields, the main algorithms that solve this problem and their limitations in terms of quality of results, whereby the problematic takes its place:**"How to overcome the limitations of these techniques?"**. In order to answer this problematic some points need to be clarified;

- **"What is data clustering and what are its main parts?"**
- **"What are the best techniques in solving the data clustering problem?"**
- **"How to optimize these techniques in order to overcome their limitations?"**

Our efforts to address the main research question led to two contributions. Our first contribution is the application of the rat swarm optimizer to data clustering. The second contribution is the optimization of the grey wolf algorithm-based data clustering. The results of the two contributions were quite impressive. The tests were carried out through several benchmarks using different measures to assure the quality of the proposed techniques. The contributions were compared to a bunch of state of the art, recent and well-known algorithms in the data clustering field, where the results demonstrated the power of the two contributions in producing high-quality clustering results among the algorithms compared with.

1.3 Thesis outline

To answer the previously mentioned questions, this dissertation was structured as follows:

- Chapter 1: **Introduction**. This chapter provides an overview of the fundamental concepts and motivations behind the study of data clustering. It sets the stage for understanding the significance of data clustering in various scientific fields and introduces the key themes that will be explored throughout the dissertation.

- Chapter 2: **Data Clustering (State of the Art)**. In this chapter, the current state of data clustering is examined, including its objectives, applications and steps. The chapter also delves into data representation and the various types and scales of data, providing a comprehensive understanding of the foundational principles of data clustering. It also presents some of the well-utilized techniques and the different measures to evaluate the results.

- Chapter 3: **Metaheuristics**. This chapter introduces optimization methods inspired by real-world behaviors and systems. It explores genetic algorithms, rat swarm optimizer, and grey wolf optimizer, providing insights into their applications in the context of data clustering.
  The subsequent chapters will present in-depth discussions and analyses of experimental results culminating in a comprehensive understanding of the contributions and implications of these methods in the data clustering.

- Chapter 4: **Rat Swarm Optimizer for Data Clustering**. This chapter introduces (our first contribution), the Rat Swarm Optimizer for Data Clustering (RSOC) method, describes the adaptation to the problem including details about the RSO-based clustering method, its solution representation, algorithm, main parts, time complexity, and provides a comparison with other algorithms and the discussion of results.

- Chapter 5: **Enhanced Grey Wolf Optimizer for Data Clustering**. This chapter presents an in-depth analysis and comparison of (our second contribution) the Enhanced Grey Wolf Optimizer for Data Clustering (EGWAC) algorithm with the original work; grey wolf algorithm-based data clustering (GWAC) and other well-established clustering algorithms. It focuses on exhibiting the performance of EGWAC across multiple datasets and comparing its effectiveness in producing high-quality clustering results.

- **Conclusion**: The final chapter summarizes the key findings and insights from the preceding chapters, offering a comprehensive conclusion to the researches presented in this dissertation.

Each chapter is structured to respond to predetermined inquiries. Chapter 2 lays the groundwork for understanding data clustering. Chapter 3 exposes optimization methods, paving the way for the contributions aimed at overcoming limitations. Finally, chapters 4 and 5 delve into the development and application of the proposed techniques, answering the "how to optimize" question.

1.4 Where to use data clustering?

Our contributions help in enhancing the accuracy of some existing clustering techniques. They can be utilized in different scientific fields and mainly in artificial intelligence.

- **Customer segmentation:** in finding the optimal customers' categories for product recommendations and personalization.

- **Anomaly detection:** in detecting efficiently unusual patterns which indicates security threats, fraud or system failure.
- **Document clustering:** in organizing documents to facilitate sentiment analysis and topic modeling (uncovering latent topics within document collections).
- **Recommendation:** in grouping similar users in order to suggest relevant content to each group.
- **Gene expression:** in finding similar genes patterns which helps in detecting diseases, or drugs.
- **Human activity:** in identifying various patterns of human activities like resting, walking, or running which is used in healthcare, fitness and smart home applications.
- **Robotics:** in analyzing sensor data from robots and autonomous machines which helps in object detection and decision making.

These applications are just a small slice from the wide range of clustering applications.

1.5 Scientific productions

The scientific papers that were realized during this thesis are:

- Zebiri, I., Zeghida, D., and Redjimi, M. (2022a). Enhanced grey wolf optimizer for data clustering. In International Conference on Artificial Intelligence: Theories and Applications, pages 147–159. Springer.
- Zebiri, I., Zeghida, D., and Redjimi, M. (2022b). Rat swarm optimizer for data clustering. Jordanian Journal of Computers and Information Technology, 8(3).

Chapter 2

Data clustering (State of the art)

2.1 Introduction

In our daily lives, we are compelled to process vast and diverse amounts of information without respite. These processes aid in decision-making and problem-solving and range from simple data representation to advanced analysis. One of these processes is data clustering.

In this chapter, a state of the art of 'data clustering' is proposed. It will present the principle data clustering elements; the definition of clusters and data clustering process, its main objectives, the general and basic steps of data clustering methods, data representation, proximity measures, taxonomy of clustering techniques, and finally evaluation and performance measures.

2.2 Data clustering

Data clustering is considered an unsupervised grouping technique (i.e. there is no data classes assigned by human expert [104], nor any predefined knowledge of the clusters). It can be defined as the process where a set of data objects is subdivided into finite number of groups (clusters) based on characteristics of data. It aims to reveal structure or groups in data [68, 34], where each group contains objects that are in some rate (highly) closer or similar to each other, whereas distant or dissimilar to objects from different groups. Data clustering has a bunch of different definitions, and as it is stated in [83, 75, 54, 12], there is no suitable definition that can fit all applications of data clustering. This issue is the consequence of the different definitions of a single cluster, and the use of vague and not well-defining (and defined) terms such as "distance" and "similarity". Usually a cluster is taken as a primitive and left undefined [12]. Some cluster definitions are presented and discussed in [63, 75, 100, 115]:

- **"A cluster is a set of entities which are alike, and entities from different clusters are not alike"** [63, 75].

- **Clusters are relative dense regions of points separated by relative low-density regions** [63, 75].
- **"A set of similar data entities found by a clustering algorithm"**[89, p. 1].

In literature, data clustering was firstly introduced by Driver and Kroeber [27] in anthropology in **(1932)** , then, in psychology in **(1938)** [132] and **(1939)** [117]. It was celebrated by Cattell [20] in **(1943)**. As a result of computational difficulties, clustering development was delayed until the end of the 1950s where the computerization led to propagating and increasing the number of clustering techniques [89]. By the early 1960s, clustering techniques were emerged in biology by Sokal and Sneath **(1963)** [109] under the name of "numerical taxonomy". Clustering techniques then got researchers' attention from different scientific fields, Ray and Berry [102] in geography **(1966)**, Lorr [79] in psychiatry **(1966)**, Morrison [90] and Green et al. [48] in management **(1967)**, Kaiser [67] in political science **(1967)**, Wingo Jr [123] in urbanization **(1967)**, Jardine and Sibson [66] in mathematics **(1968)** and Fisher [36] **(1969)** in economics [15, 70]. In fact, Clustering activities have been around for over a century, and have been used in a variety of fields [89].

2.2.1 Clustering objectives and applications

Mirkin [89] identified a list of clustering objectives:

- **Structuring:** generally, it is the main objective of data clustering where a group of data is organized on a number of subgroups of similar data points.
- **Description:** is describing clusters based on data characteristics, this goal is not fundamental in finding clusters (e.g. Density-based clustering algorithms don't involve any cluster description to create clusters).
- **Association:** reveals relationships between different aspects of the studied case. An aspect can be represented by one or more features. For instance, analyzing the relationship between having a profile picture (on social media websites) and the popularity of that profile.
- **Generalization:** refers to the process of giving declarations or overviews about the studied case or its features. This goal involves all previous ones (structuring, description and association).
- **Visualization:** refers to the process of representing clusters in a graphical or visual format.

These objectives are not the only ones but the main in the author's opinion, further objectives can be found in [89, 6, 105].

Clustering applications are generally included on the previous clustering objectives. Some popular applications are: data summarizing, anomaly detection, customers segmentation, gene-expression, object and character recognition, video surveillance, information retrieval, Image processing and segmentation and others [33].

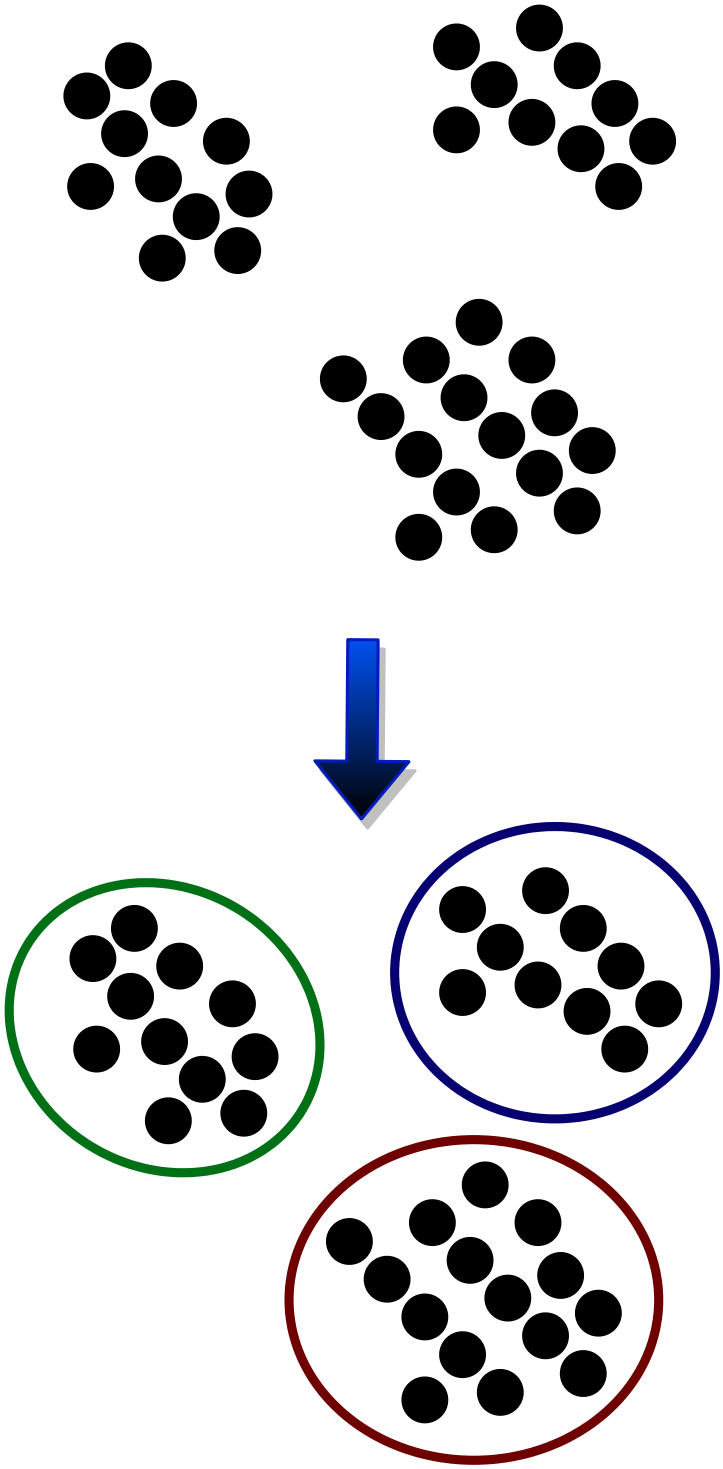


Fig. 2.1 Data clustering

Table 2.1 Data Matrix

Persons	Attributes			
	Height	Weight	Age	Degree
Ammar	178	85	24	17
Assim	178	70	23	18.5
Kais	179	64	23	16
Salah	187	87	25	15.5
Abdou	183	83	25	16

2.2.2 Data clustering steps

In general, a data clustering task includes the following steps [64]:

1. Data representation.
2. Proximity measure selection.
3. Type of clustering selection.
4. Data abstraction (Describes each cluster; mostly by representative points).
5. Output evaluation.

The *data abstraction* and *output evaluation* steps are not always included.

2.3 Data representation

In data clustering, generally, a dataset can be presented by: a data matrix, and extensively, a proximity matrix. The data representation step may incorporate feature selection and/or feature extraction. Feature selection chooses important ones among the total group of them to represent data, while new features are created from the old ones in feature extraction.

2.3.1 Data matrix

Let D a dataset of n objects characterized by d features, an object can be represented by a multidimensional vector as $O = (o_1, o_2, ..., o_d)$ where, o_i represents the i^{th} feature's value. Thus, the whole dataset can be represented by a matrix, where columns and rows represent respectively the features and the objects. For instance, let Ammar, Assim, Kais, Salah and Abdou, five persons represented by height, weight, age, and degree. Each row in the data matrix 2.1 represents an individual (object).

Table 2.2 Distance Matrix (Euclidean distance)

Persons	Ammar	Assim	Kais	Salah	Abdou
Ammar	0				
Assim	15.11	0			
Kais	21.07	6.58	0		
Salah	9.39	19.57	24.44	0	
Abdou	5.57	14.29	19.52	5.68	0

Table 2.3 Distance Matrix (Squared Euclidean distance)

Persons	Ammar	Assim	Kais	Salah	Abdou
Ammar	0				
Assim	228.25	0			
Kais	444	43.25	0		
Salah	88.25	383	597.25	0	
Abdou	31	204.25	381	32.25	0

2.3.2 Proximity matrix

Some clustering techniques work on a proximity (similarity or distance) matrix to create clusters. The main form of this matrix is created by calculating the proximity between each object on the dataset. In general, the half of the matrix is left blank as the proximity function is considered as commutative function ($P(x,y) = P(y,x)$). The diagonal is filled by zeros in case of distance (or by ones in case of similarity) as the proximity between an object and itself is the minimum (or maximum in similarity).

2.3.3 Data types

Data features come in three distinct categories [12]:

1. **Continuous:** these variables can take any value within an uncountable infinite range. Circle's surface is an example.

2. **Discrete:** these variables have finite or countably infinite possible values. For instance, population of a city.

3. **binary:** are variables that can only take two values. Gender is a sample of these variables, where it can just take either male or female.

2.3.4 Data scales

There are two main scales: qualitative (nominal or ordinal) and quantitative (intervall or ratio) [12]:

1. **Nominal scale:** this scale can only provide two functions to distinguish between values, equality and inequality. Nation of birth is an example (Algeria, Tunisia, Morocco ...).

2. **Ordinal scale:** this scale provides in addition to equality and inequality the order. By way of example, difficulty scale (easy, medium and hard).

3. **Interval scale:** besides characteristics of ordinal scale, this scale gives a meaning to the measure of difference between two values, e.g., temperature values using Celsius (35°C is greater (hotter) than 30°C by 5°C).

4. **Ratio scale:** this scale is the most meaningful scale where it has an absolute zero intrinsically immanent to reality, e.g., measuring temperature by Kelvin (the zero here is the absence of any heating source). The Kilogram also has an absolute zero (the absence of any weight). The importance of this scale is that one can say x is twice warmer than y (x/y has a meaningful value). The two well-known temperature measures Celsius and Kelvin are respectively examples of interval and ratio scales. 20°C is not twice heater than 10°C. However, 20K is twice hotter than 10K because Kelvin has an absolute zero that is intrinsically immanent to reality. Unlike Kelvin, the zero in Celsius represent something else (freezing point of water), that's the reason of using the degree sign (°) with Celsius values.

2.4 Proximity measures

Since the basis in constructing clusters is the proximity calculation, clustering algorithms use different methods to calculate proximity between data objects. As data features can be of the same type they can also be of mixed types. Thus, proximity measures can be broken up into four categories: numerical, categorical, binary and mixed measures.

2.4.1 Numerical measures

- **Minkowski distance:** the distance between two points is defined as follows [40]:

$$d_{Mink}(x,y) = \left(\sum_{i=1}^{d} |x_i - y_i|^r \right)^{\frac{1}{r}} \tag{2.1}$$

  Where x and y are two data objects. r can have any natural value greater than 0 ($[1,\infty]$):

  - If $r = 1$, it gives the Manhattan (city block) distance: $d_{Man}(x,y) = \sum_{i=1}^{d} |x_i - y_i|$
  - If $r = 2$, it gives the Euclidean distance: $d_{Euc}(x,y) = \left(\sum_{i=1}^{d} |x_i - y_i|^2 \right)^{\frac{1}{2}}$
  - If $r \to \infty$, it gives the Chebyshev distance: $d_{Max}(x,y) = \max_{1 \le i \le d} |x_i - y_i|$

2.4.2 Categorical measures

- **Simple matching:** the proximity here is calculated as:

$$d_{Simp}(x,y) = \sum_{i=1}^{d} \delta(x_i, y_i) \tag{2.2}$$

For calculating the distance, δ is defined as:

$$\delta(x_i, y_i) = \begin{cases} 0 & \text{if } x_i = y_i \\ 1 & \text{else} \end{cases}$$

For the similarity, it is defined opposingly:

$$\delta(x_i, y_i) = \begin{cases} 1 & \text{if } x_i = y_i \\ 0 & \text{else} \end{cases}$$

2.4.3 Binary measures

Most proximity measures of two binary vectors use these functions [40]:

$$A = x.y = \sum_{i=1}^{d} x_i y_i$$

$$B = \overline{x}.y = \sum_{i=1}^{d} (1 - x_i) y_i$$

$$C = x.\overline{y} = \sum_{i=1}^{d} x_i (1 - y_i)$$

$$D = \overline{x}.\overline{y} = \sum_{i=1}^{d} (1 - x_i)(1 - y_i)$$

- **Jaccard:** similarity function $s(x,y) = \frac{A}{A+B+C}$, dissimilarity $d(x,y) = \frac{B+C}{A+B+C}$
- **Dice:** similarity function $s(x,y) = \frac{A}{2A+B+C}$, dissimilarity $d(x,y) = \frac{B+C}{2A+B+C}$
- **Yule:** similarity function $s(x,y) = \frac{AD-BC}{AD+BC}$, dissimilarity $d(x,y) = \frac{BC}{AD+BC}$

The previous values, A, B, C and D can be formally explained as:

- A: number of features where both x and y have a present value (1).
- B: number of features where x has absent value (0) and y has present value (1).
- C: number of features where x has present value (1) and y has absent value (0).
- D: Number of features with simultaneous absence of values for both x and y.

2.4.4 Mixed types measures

The previous measures can only deal with data features of the same type, and generally data features come with mixed types, in this case the need for mixed type measures is indispensable. Gower [47] designed a mixed-types proximity measure called **The general similarity coefficient** [47, 125, 40]:

$$s_{Gower}(x,y) = \frac{1}{\sum_{i=1}^{d} w(x_i,y_i)} \sum_{i=1}^{d} w(x_i,y_i) s(x_i,y_i) \tag{2.3}$$

Where $w(x_i,y_i)$ takes 1 if the i^{th} attribute of x and y is not missing, else it takes 0. R_i represents the difference between the minimum and maximum values observed for the i^{th} feature (range). The function $s(x_i,y_i)$ is defined as follows:

- For **quantitative** as:

$$s(x_i,y_i) = 1 - \frac{|x_i - y_i|}{R_i}$$

- For **qualitative (binary and nominal)** as:

$$s(x_i,y_i) = \begin{cases} 1 & \text{if } x_i = y_i \\ 0 & \text{else} \end{cases}$$

  $w(x_i,y_i)$ takes 1 if the k^{th} attribute of x and y is not missing. Otherwise, it takes 0.

The general distance coefficient is defined as [40, 125]:

$$d_{Gower}(x,y) = \left(\frac{1}{\sum_{i=1}^{d} w(x_i,y_i)} \sum_{i=1}^{d} w(x_i,y_i) d^2(x_i,y_i) \right)^{\frac{1}{2}} \tag{2.4}$$

Where $d^2(x_i,y_i)$ is calculated:

- For **quantitative** variables as: $d^2(x_i,y_i) = |x_i - y_i|^2$
- For **ordinal** as: $d^2(x_i,y_i) = \left(\frac{|x_i - y_i|}{R_i}\right)^2$
- For **binary and nominal** variables as: $d^2(x_i,y_i) = \begin{cases} 0 & \text{if } x_i = y_i \\ 1 & \text{else} \end{cases}$

2.5 Taxonomy of clustering algorithms

Several properties challenge the notion of a singular classification, such as the results of clustering (hard, fuzzy or overlapping), the mode of finding clusters (joining, splitting, etc), the type of clustering such as: flat or hierarchical, and others [53, 83, 15, 54, 52].

2.5.1 Hard clustering

The property of this type is to find exclusive clusters, an object can be part of one and only one cluster.

2.5.2 Overlapping clustering

In this type, objects can be simultaneously part of several clusters. This type is used to represent cases where an object is a member of more than one group (e.g. a person can be employee and student at the same time).

2.5.3 Fuzzy clustering

Each object in this type has an indicator of membership $M = (m_1, m_2, ..., m_K)$, m_i represents the rate of belonging to i^{th} cluster and K represents the number of clusters. This rate is included between 0 and 1, and the sum of all rates equals 1:

$$\sum_{i=1}^{K} m_i = 1$$

The following sections detailed some well-utilized clustering techniques: partitional, hierarchical, and density-based methods [5, 33].

2.5.4 Partitional methods

These algorithms are considered as the main simplest clustering algorithms [52], whither, a group of objects is organized forming exclusive clusters. Mostly, these algorithms require clusters' number, K, as user input. As primitive, the clusters' number should be between 1 and the dataset's size $1 < K < n$, considering that taking the whole dataset as a cluster or each data object as a single cluster has no sense. Typically, these algorithms depend mostly on the distance function and clusters representatives to find clusters. Due to the reliance on prototypes to represent clusters, these algorithms are also known as prototype-based clustering algorithms [5].
Usually, prototype-based clustering algorithms consist of two main steps:

1. Selection of initial prototypes, and

2. Improvements of prototypes until convergence.

It is better to use a non-deterministic approach in the first step, then, iteratively refine prototypes [61]. There are several partitional algorithms, such as k-means, k-medoids and others.

K-means

K-means is considered as the extremely wide-utilized algorithm, it has a diverse rich history. K-means' independent invention in various domains started by Thorndike [116], followed by Steinhaus [113],

Lloyd [78], Forgy [38], Ball and Hall [16], Jancey [65] and McQueen [85]. Regardless of the fact that it has existed for more than half a century, K-means' ease of use, conceptual clarity, and efficacy remain cornerstones of its enduring popularity [62, 125].

K-means begins by selecting K data objects as initial clusters' centoids, then, data objects are grouped with their nearest centroid, forming the initial clusters by using a distance function (typically, the euclidean distance). The algorithm then updates the clusters' centers by the mean of its members. The reassignment and centroid recalculation process are repeated until the centroids are either still the same, or some other lightweight convergence criterion is met (e.g. until at least 1% of points change their clusters) [5].

Algorithm 1 K-means

Require: dataset (D), number of clusters (k)
Ensure: the clustered dataset
  Initialize centroids by random K data points.
  repeat
    Assign each data point in D to the cluster with the closest centroid.
    Recalculate the mean of clusters as their centroids.
  until Stopping condition is met.

K-medoids

K-medoids is a partitional clustering algorithm that is identical in some sort to K-means. It was derived from the problem of finding K points that serves as cluster prototypes, and then assigning the rest of points to these prototypes in order that the distances between each prototype and its points is minimized. The K-medoids method was differently exposed by Vinod [119], Spath [111], Massart et al. [81], Kaufman and Rousseeuw [68]. Like K-means, K-medoids uses exemplars to represent clusters, which is the mean in case of K-means and the medoid in case of K-medoids. K-medoids pseudo-code is depicted in Algorithm 2.

Algorithm 2 K-medoids

Require: number of clusters (K), dataset (D)
Ensure: the clustered dataset
  Select K points from the dataset as initial prototypes (medoids).
  repeat
    Create K clusters by associating each data object with the nearest medoid.
    Calculate the fitness (sum of the distances between each medoid and the points within its cluster).
    For each non-medoid point in a cluster and its corresponding medoid, swap the values of them and recalculate the fitness.
  until convergence criterion is met.

Same as K-means, K-medoids starts in the first part by initializing the cluster medoids by a random points from the dataset. Then it forms clusters like k-means, it assigns data objects to the closest medoid. Mostly, the utilized distance measure is the Euclidean distance. However, different distance measures can also be used. After that, the algorithm computes the sum of distances between points and their assigned medoids. The calculated value represents the fitness (quality) of the solution. The second part is the repeated process, where the algorithm swaps each cluster's medoid with the cluster's member that minimizes the sum of distances. The second part is repeated until the medoids don't change.

There are some differences between the two algorithms such as [54]:

- The mean is the representative point regarding K-means, while, in K-medoids the representative is one of the cluster's individuals.
- Unlike K-means, K-medoids uses the basic Euclidean distance instead of the squared one.
- To perform well, it is advised that K-means utilizes the Squared Euclidean distance, while the performance of K-medoids does not depend on that.

Many other partitional clustering methods are discussed in [5, 14, 40, 128, 100].

2.5.5 Hierarchical methods

As it is indicated by their name, these methods consist of creating a hierarchy of clusters, the construction of which is done in a gradual manner, the higher up the hierarchy one goes, the less specific the clusters become. Hierarchical clustering results can be depicted through dendrograms. These algorithms fall into two categories: bottom-up (Agglomerative) and top-down (Divisive). Cutting the dendrogram at any level to form clusters is a unique strength of hierarchical clustering compared to partitional methods [5]. This flexibility removes the need to pre-define the number of clusters (K), allowing data-driven cluster discovery. The hierarchical methods are applicable to any attribute type, unlike some partitional methods such as k-means which is applicable to numerical data. They also give a facility to handle different forms of distance or similarity between objects, they use the standard proximity matrix, or other specific matrices, however, they suffer from some problems such as the problem of choosing the stopping criterion, the inability to undo the previous step (merging or splitting), and the missing values [100, 17, 31].

Agglomerative methods

These methods are also known as bottom-up techniques. They consist of creating the cluster hierarchy starting with n objects, each object represents a cluster, successively at each step this method merges the two nearest objects or clusters forming a new cluster until a single group contains all the objects or a stopping condition holds [52].

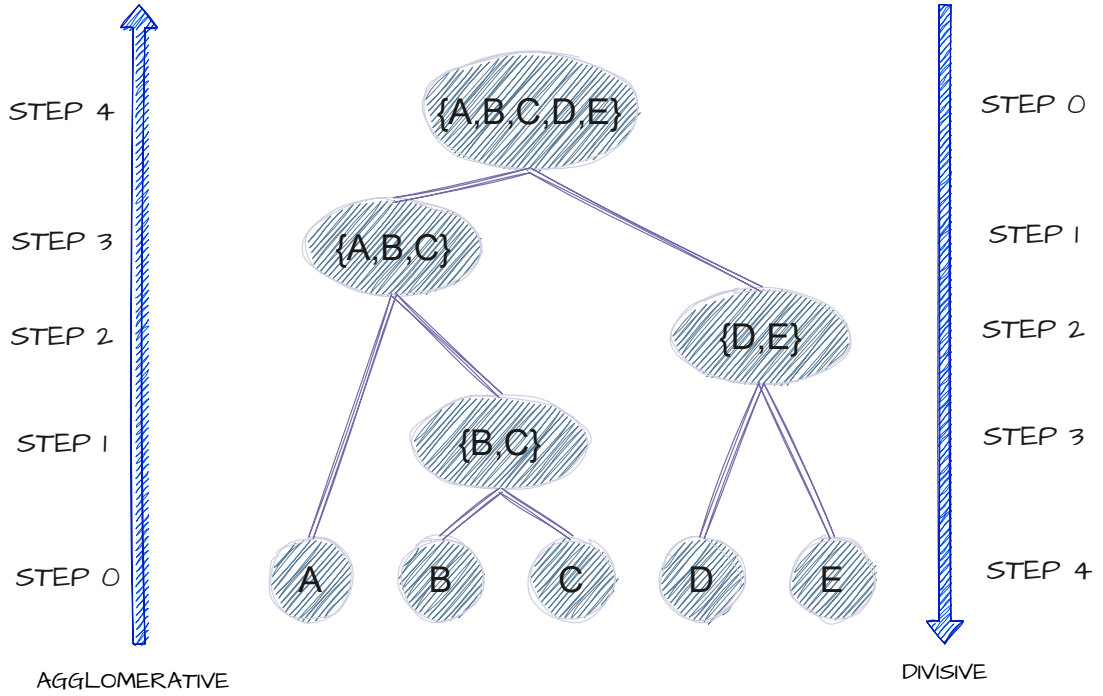


Fig. 2.2 Hierarchical clustering

The bottom-up approach suffers from the major disadvantage; once a partition is made, an object cannot be moved to another cluster.

Algorithm 3 Agglomerative clustering

Require: dataset (D)
Ensure: the clustered dataset
Construct the pairwise distance matrix for the entire dataset.
repeat
Combine the two clusters C_i and C_j with the smallest distance as $C_{i,j} = C_i \cup C_j$.
Update the distance matrix by removing C_i and C_j entries and inserting the distances between $C_{i,j}$ and the rest of clusters.
until Satisfaction of stopping criterion.

At each step in the bottom-up merging process, we need to update the distances between the newly formed cluster and all remaining clusters. The several techniques to measure inter-cluster distance led to create different agglomertive techniques [103]. Some agglomertive techniques are discussed in the following sections.

Single-link

This approach was defined by Florek et al. [37] in **(1951)**, then described by other researchers including Sneath [107]. This method is also called **nearest-neighbour** as a hierarchical process [122]. For any two clusters C_i and C_j, the distance between them is defined as the minimum distance among all pairwise distances between points in C_i and points in C_j [75, 45, 126]:

$$dist_{SL}(C_i, C_j) = \min_{x \in C_i, y \in C_j} d(x, y) \tag{2.5}$$

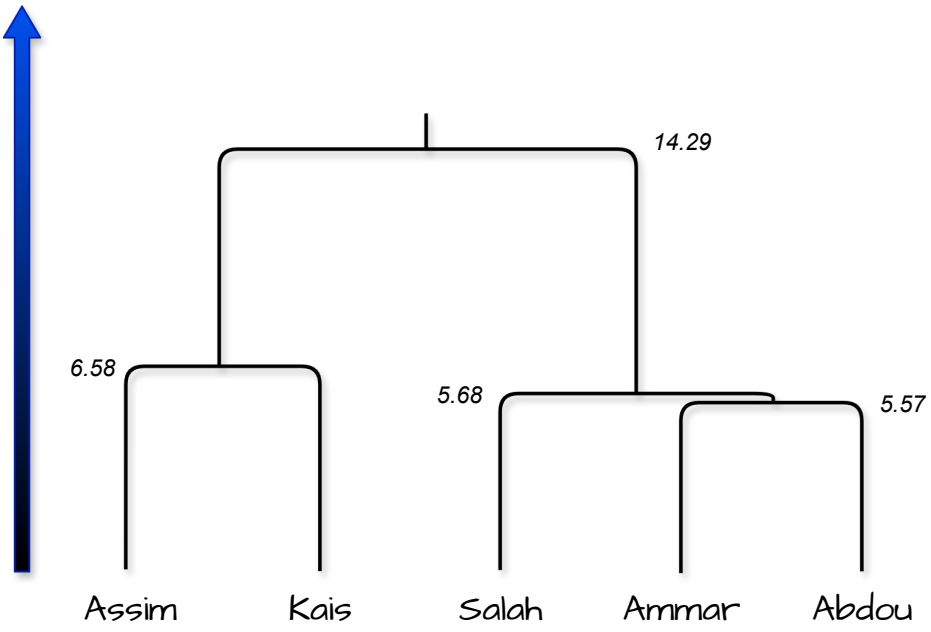


Fig. 2.3 Single link hierarchical clustering

This approach is based on local similarity. Through its local behaviour, it is able to efficiently cluster elongated, non elliptical sets. However, this method struggles with noise and outliers problems, particularly the chaining-effect problem [5]. Two separate clusters will be merged if there is a string of dots between them. The obvious solution to overcome this phenomenon is to remove this noise (chain), then apply the method [74, 66].

Figure 2.3 illustrates the evolving clusters of the persons in table 2.1 using single-link.

Complete-link

Complete-link was proposed by Horn [58] in 1943 , Sørensen [110] in 1948, and King [69] in 1967. In 1965, Macnaughton-Smith [80] proposed its hierarchical version 'furthest-neighbour'. For any two clusters C_i and C_j, the distance between them is defined as the maximum distance among all pairwise distances between points in C_i and points in C_j [45, 96]:

$$dist_{CL}(C_i, C_j) = \max_{x \in C_i, y \in C_j} d(x, y) \tag{2.6}$$

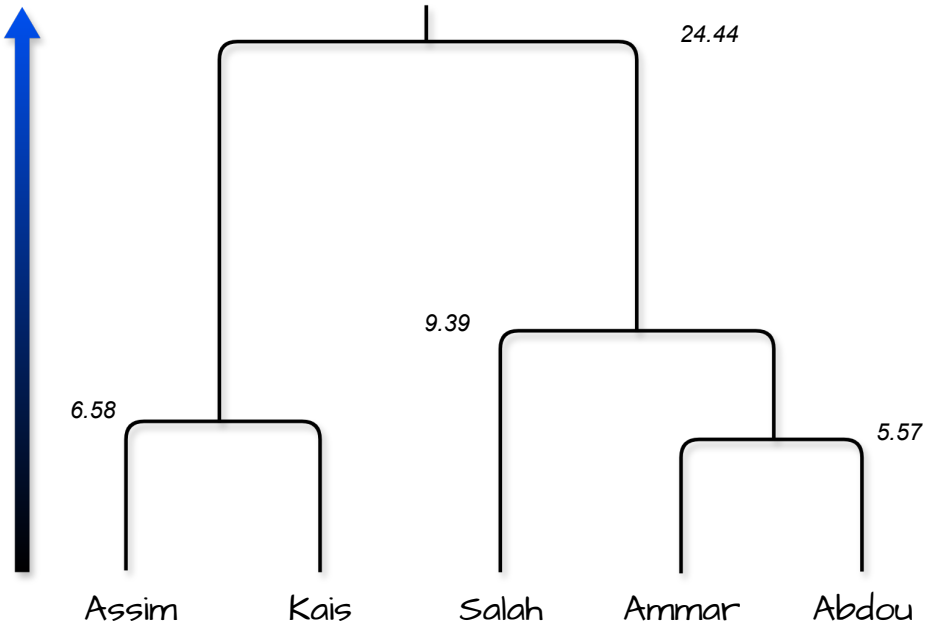


Fig. 2.4 Complete link hierarchical clustering

Unlike single-link, this method has non-local behaviour. Despite its advantage, this method remains susceptible to the disruptive effects of noise and outliers. [5, 15].

Average-link

This approach otherwise called UPGMA (Unweighted Pair Group Method using Average), it was proposed by Sokal and Michener in 1958 [108]. It has reduced the disadvantages associated with the previous approaches, as it calculates all pairwise distances. Consequently, this method becomes increasingly expensive as the number of data points grows [5, 45, 128, 6].

$$dist_{UPGMA}(C_i,C_j) = \frac{1}{n_i.n_j} \sum_{x \in C_i} \sum_{y \in C_j} d(x,y) \tag{2.7}$$

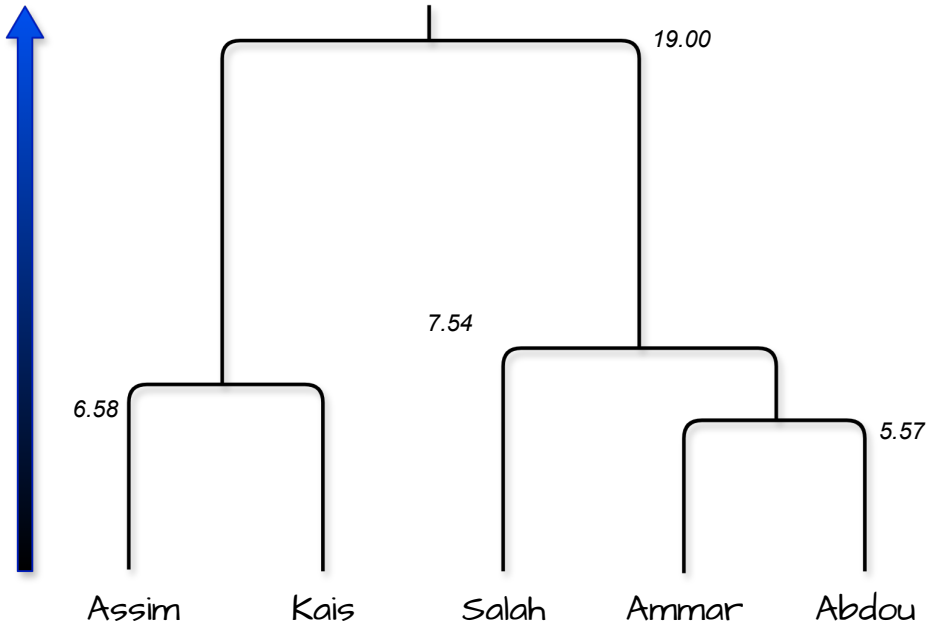


Fig. 2.5 UPGMA (Average link) hierarchical clustering

n_i and n_j are respectively, the cardinalities of clusters C_i and C_j. The distance between a cluster C_l and another cluster $C_{i,j}$ consisting of two smaller clusters C_i and C_j is given by [73]:

$$dist_{UPGMA}(C_l,C_{i,j}) = \frac{1}{2} dist_{UPGMA}(C_l,C_i) + \frac{1}{2} dist_{UPGMA}(C_l,C_j) \tag{2.8}$$

Weighted average-link

McQuitty proposed another approach in 1966 [86] called WPGMA (Weighted Pair Group Method using Average), this approach is a weighted version of the previous one, the distance between a cluster C_l and another composite one $C_{i,j}$ is then given by [40, 128]:

$$dist_{WPGMA}(C_l,C_{i,j}) = \frac{n_i}{n_i+n_j} dist_{WPGMA}(C_l,C_i) + \frac{n_j}{n_i+n_j} dist_{WPGMA}(C_l,C_j) \tag{2.9}$$

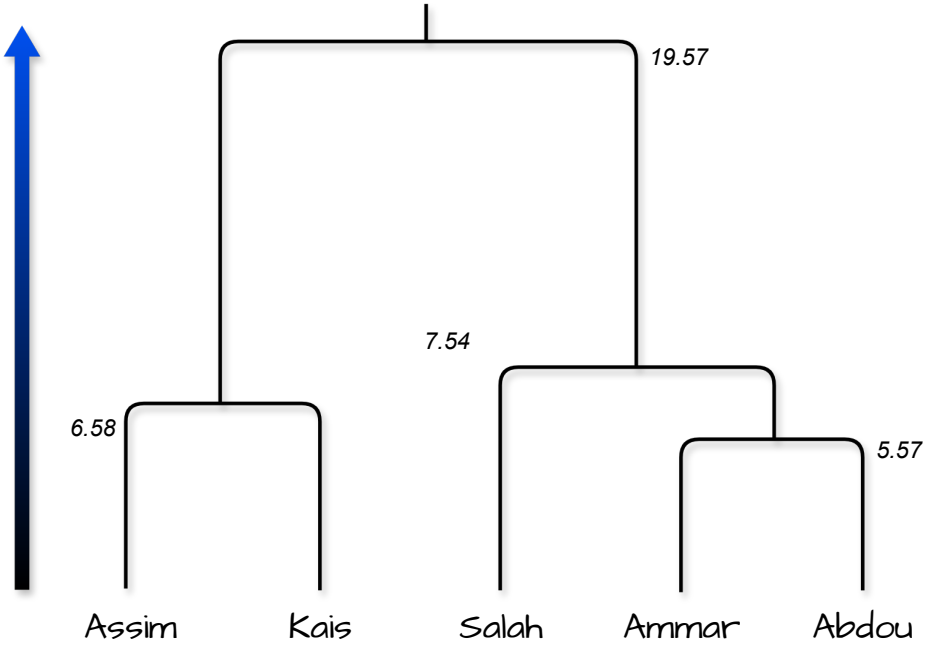


Fig. 2.6 WPGMA hierarchical clustering

Centroid method

In 1958, Sokal and Michener [108] proposed the UPGMC (Unweighted Pair Group Method using Centroid) or the centroid method. This method measures the distance between clusters by averaging the positions of their members (centroids) and then it calculates the distance between these centroids [40, 96, 128]:

$$d_{UC}(C_i, C_j) = d(\mu_i, \mu_j) \tag{2.10}$$

$$\mu_i = \frac{1}{n_i} \sum_{x \in C_i} x$$

μ_i and μ_j are respectively the centroids of the clusters C_i and C_j.

After the junction of two clusters, the new centroid is given by the weighted mean [103]:

$$\mu_{i,j} = \frac{n_i \mu_i + n_j \mu_j}{n_i + n_j}$$

$\mu_{i,j}$ represents the centroid of the new combined clusters. The following formula is used to calculate the distance between the new cluster $C_{i,j}$ and other ones [73]:

Table 2.4 Centroid method (1^{st} merging)

Persons	Attributes			
	Height	Weight	Age	Degree
Assim	178	70	23	18.5
Kais	179	64	23	16
Salah	187	87	25	15.5
{Ammar, Abdou}	180.5	84	24.5	16.5

$$d_{UC}(C_l, C_{i,j}) = \frac{n_i}{n_i + n_j} d_{UC}(C_l, C_i) + \frac{n_j}{n_i + n_j} d_{UC}(C_l, C_j) - \frac{n_i n_j}{(n_i + n_j)^2} d_{UC}(C_i, C_j) \tag{2.11}$$

Unlike UPGMA, which calculates cluster distances by averaging all pairwise distances within a distance matrix, UPGMC relies solely on the pre-calculated centroids of each cluster as its distance measure, see the Table 2.4 [45].

Median method

Gower proposed another centroid-based method called the median method [46] or WPGMC (Weighted Pair Group Method using Centroid), this method was proposed to reduce the disadvantages of the UPGMC method. In scenarios with unequal cluster sizes during merging, the newly calculated centroid is influenced by the larger cluster and might end up residing within its boundaries [45, 40]. In the median method, the new centroid is calculated independently of the cluster size, using the following expression, we can determine the new distances between the combined clusters and the others [73]:

$$d_M(C_l, C_{i,j}) = \frac{1}{2} d_M(C_l, C_i) + \frac{1}{2} d_M(C_l, C_j) - \frac{1}{4} d_M(C_i, C_j) \tag{2.12}$$

Ward's method

This method, introduced in 1963 by Ward Jr [120], is an agglomerative clustering technique that measures the "increase in variance" within a cluster when merging two other clusters, choosing the merge that minimizes this increase. The method employs the sum of squared error (SSE) used in K-means, to determine the clusters to merge. The aim of the Ward's technique is minimizing the increment in SSE resulting from a pair-wise cluster combination. Taking that into consideration, SSE values are determined for all possible combinations of the clusters, whereby the two pairwise closest clusters are combined. The Ward's criterion essentially calculated the squared Euclidean distance between their centroids, weighted by a relative value to their sizes [128]:

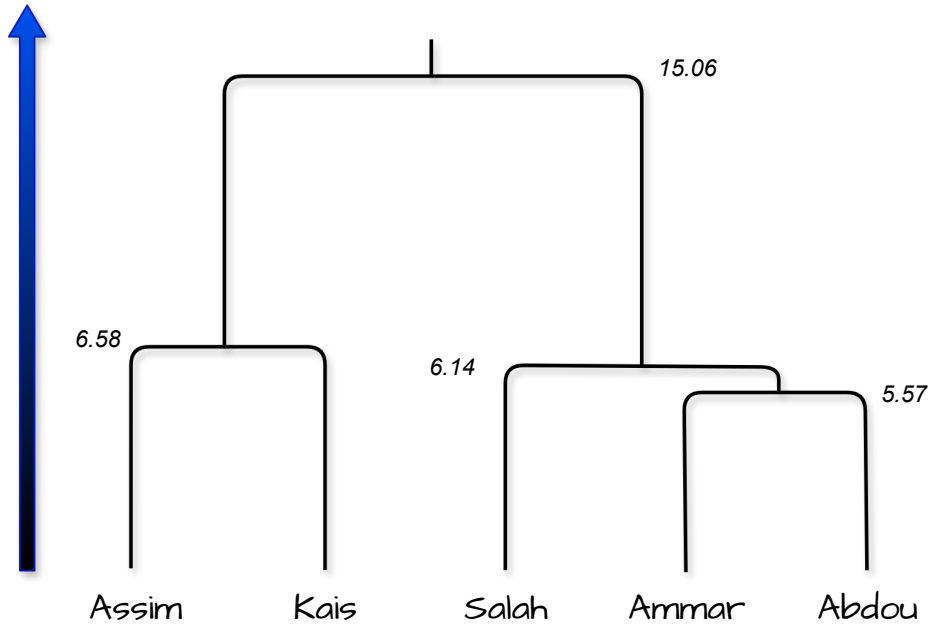


Fig. 2.7 UPGMC hierarchical clustering

$$W(C_i, C_j) = \frac{n_i * n_j}{n_i + n_j} d^2(\mu_i, \mu_j) \tag{2.13}$$

C_i and C_j are the clusters j and i, μ_i, μ_j are their centroids, and n_i and n_j are their cardinalities. d^2 is the squared euclidean distance.

Let C_{ab} a cluster formed by merging C_a and C_b, the SSE values are calculated as follows [103]:

$$SSE_a = \sum_{x \in C_a} d^2(x - \mu_a),$$
$$SSE_b = \sum_{x \in C_b} d^2(x - \mu_b),$$
$$SSE_{ab} = \sum_{x \in C_{ab}} d^2(x - \mu_{ab}).$$

The following formula depicts the calculation of the new centroid μ_{ab}:

$$\mu_{ab} = \frac{n_a \mu_a + n_b \mu_b}{n_a + n_b} \tag{2.14}$$

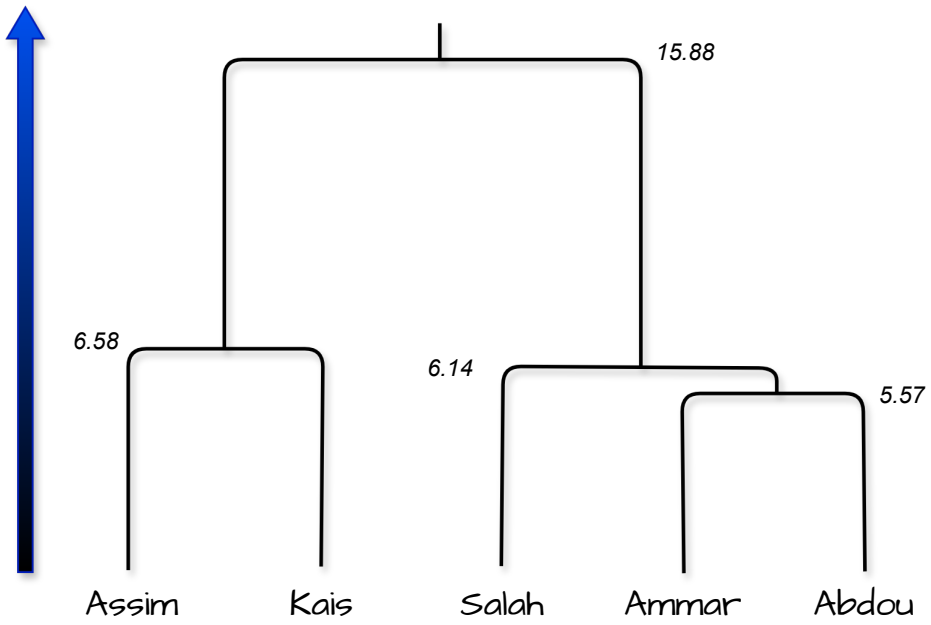


Fig. 2.8 WPGMC hierarchical clustering

Ward's method combine the two cluster whereby the increase in the SSE criterion is the minimum. This increase is calculated as:

$$\Delta_{ab} = SSE_{ab} - (SSE_a + SSE_b) \tag{2.15}$$

The formula in the equation 2.15 can be written as:

$$\Delta_{ab} = n_a d^2(\mu_a, \mu_{ab}) + n_b d^2(\mu_b, \mu_{ab}),$$
$$\Delta_{ab} = \frac{n_a * n_b}{n_a + n_b} d^2(\mu_a, \mu_b). \tag{2.16}$$

The Ward's method shares a close connection with the centroid method. Lets take the equation that calculates the distances between two clusters in the centroid method (Eq. 2.10). If we squared it, the only difference between the two methods will be the coefficient $\left(\frac{n_a * n_b}{n_a + n_b}\right)$, which means that the size of clusters affects the Ward's method. The coefficient can be written like [103]:

$$\frac{n_a * n_b}{n_a + n_b} = \frac{n_a}{1 + (n_a/n_b)}$$

The value of the coefficient increases if one cluster is larger than the other, which results in large value of SSE. Thus, Ward's method is supposed to combine clusters of equal sizes or smaller ones [103].

The distance between a cluster C_l and another one consisting of two other clusters C_a and C_b is given by [124]:

$$D(C_l, C_{ab}) = \frac{n_a + n_l}{n_{ab} + n_l} d^2(\mu_a, \mu_l) + \frac{n_b + n_l}{n_{ab} + n_l} d^2(\mu_b, \mu_l) - \frac{n_l}{n_{ab} + n_l} d^2(\mu_a, \mu_b) \tag{2.17}$$

Unlike the previous methods, this method utilizes the squared Euclidean distance. Lets take the example in Table 2.1.

The first clusters to be combined together are the clusters that result in the minimum augmentation of the SSE index which are the individuals *Ammar* and *Abdou*, by using the equation 2.16 we get $\Delta_{Abdou,Ammar} = \frac{1}{2}(31) = 15.5$. As the initial SSE ($SSE_0$) is set to zero. The SSE after the first clusters' fusion will be:

$$SSE_1 = SSE_0 + \Delta_{Abdou,Ammar} = 15.5$$

The distances between the new combined clusters ($C_{Ab,Am}$) and the others is carried out using the equation 2.17.

$$D(Assim, C_{Ab,Am}) = \frac{2}{3}\left(d^2(Assim, Abdou) + d^2(Assim, Ammar)\right) - \frac{1}{3}d^2(Abdou, Ammar) = 278,$$
$$D(Salah, C_{Ab,Am}) = \frac{2}{3}\left(d^2(Salah, Abdou) + d^2(Salah, Ammar)\right) - \frac{1}{3}d^2(Abdou, Ammar) = 70,$$
$$D(Kais, C_{Ab,Am}) = \frac{2}{3}\left(d^2(Kais, Abdou) + d^2(Kais, Ammar)\right) - \frac{1}{3}d^2(Abdou, Ammar) = 539.67.$$

The $d^2(.,.)$ distances in these equations are taken directly from the distance matrix. The distance matrix will be updated as:

	Assim	Salah	Kais	$C_{Ab,Am}$
Assim	0			
Salah	383	0		
Kais	43.25	597.25	0	
$C_{Ab,Am}$	278	70	539.67	0

Table 2.5 Distance matrix after the 1^{st} merging.

After recalculating the distance matrix, the clusters to be combined together are the individuals Assim and Kais ($C_{As,Ka}$) whereby the SSE increments by $\Delta_{Assim,Kais} = \frac{1}{2}(43.25) = 21.63$. Thus, the SSE value, the distances between clusters and the distance matrix will be:

$$SSE_2 = SSE_1 + \Delta_{Assim,Kais} = 37.13$$

$$D(Salah, C_{As,Ka}) = \frac{2}{3}\left(d^2(Salah, Assim) + d^2(Salah, Kais)\right) - \frac{1}{3}d^2(Assim, Kais) = 639.08,$$
$$D(C_{Ab,Am}, C_{As,Ka}) = \frac{3}{4}\left(d^2(C_{Ab,Am}, Assim) + d^2(C_{Ab,Am}, Kais)\right) - \frac{2}{4}d^2(Assim, Kais) = 591.63.$$

	$C_{As,Ka}$	Salah	$C_{Ab,Am}$
$C_{As,Ka}$	0		
Salah	639.08	0	
$C_{Ab,Am}$	591.63	70	0

Table 2.6 Distance matrix after the 2^{nd} merging.

The clusters to merge in this stage are the individual Salah and $C_{Ab,Am}$. $\Delta_{Salah,C_{Ab,Am}} = \frac{2}{3}(70) = 46.67$. The out-coming augmentation in SSE by the new cluster ($C_{Ab,Am,Sa}$) and the distance between clusters are then calculated.

$$SSE_3 = SSE_2 + \Delta_{Salah,C_{Ab,Am}} = 83.8$$

$$D(C_{As,Ka}, C_{Ab,Am,Sa}) = \frac{3}{5}d^2(C_{As,Ka}, Salah) + \frac{4}{5}d^2(C_{As,Ka}, C_{Ab,Am}) - \frac{2}{5}d^2(Salah, C_{Ab,Am}) = 828.75.$$

	$C_{As,Ka}$	$C_{Ab,Am,Sa}$
$C_{As,Ka}$	0	
$C_{Ab,Am,Sa}$	828.75	0

Table 2.7 Distance matrix after the 3^{rd} merging.

At the last stage we only have one possible move as it lasts only two clusters ($C_{As,Ka}$ and $C_{Ab,Am,Sa}$). The SSE here will be incremented by $\Delta_{C_{Ab,Am,Sa},C_{As,Ka}} = \frac{6}{5}(828.75) = 994.5$.

$$SSE_4 = SSE_3 + \Delta_{C_{Ab,Am,Sa},C_{As,Ka}} = 1078.3$$

The previous example illustrated the functioning of Ward's method in clustering the data.

As it is stated by Anderberg [12], ward's method is not guaranteed to find the absolute best grouping of data points (the one with the lowest SSE), yet, good approximations to the best solution [40].

Lance & Williams formula

The formula of Lance and Williams [74] can summarize the different techniques of calculating the distance between two clusters in the aforementioned methods:

$$D(C_l, C_{ij}) = \alpha_i D(C_l, C_i) + \alpha_j D(C_l, C_j) - \beta D(C_i, C_j) + \gamma |D(C_l, C_i) - D(C_l, C_j)| \tag{2.18}$$

$D(..)$ is the distance extracted from the distance matrix of the corresponding method. α_i, α_j, β and γ are coefficients. Table 2.8, summarizes how to get each method from the Lance & Williams formula (Eq. 2.18).

Method	α_i	α_j	β	γ
Single-link	$\frac{1}{2}$	$\frac{1}{2}$	0	$-\frac{1}{2}$
Complete-link	$\frac{1}{2}$	$\frac{1}{2}$	0	$\frac{1}{2}$
Average-link	$\frac{n_i}{n_i+n_j}$	$\frac{n_j}{n_i+n_j}$	0	0
Weighted average-link	$\frac{1}{2}$	$\frac{1}{2}$	0	0
Centroid	$\frac{n_i}{n_i+n_j}$	$\frac{n_j}{n_i+n_j}$	$-\frac{n_i * n_j}{(n_i+n_j)^2}$	0
Median	$\frac{1}{2}$	$\frac{1}{2}$	-$\frac{1}{4}$	0
Ward	$\frac{n_i+n_l}{n_i+n_j+n_l}$	$\frac{n_j+n_l}{n_i+n_j+n_l}$	$-\frac{n_l}{n_i+n_j+n_l}$	0

Table 2.8 Coefficients' values of Lance & Williams formula.

Additional information about the agglomerative methods can be found in [74, 96, 44].

Divisive methods

The divisive or top-down approach begins with a single group of all the objects, and successively at each iteration, a cluster is divided into two smaller clusters, until each cluster contains a single object or a stopping criterion is met [44]. This process is illustrated in Figure 2.2. Efficiency is the main strength point compared to the bottom-up method, especially when it is not necessary to develop the entire hierarchy.

The major drawbacks of the top-down method are: (i) the complexity of splitting a cluster into two sub-clusters, there are $2^{(n-1)} - 1$ possible splits to divide a cluster that contains n objects into two sub-clusters, where this is considered as an NP-Hard problem [44, 19, 121]. Similarly to the bottom-up approach: (ii) once a partition has been made, an object cannot be moved to another cluster. Another problem arises as early as the second iteration, and that is (iii) choosing which cluster to split up [30, 40, 103].

Algorithm 4 Divisive clustering

Require: dataset (D)
Ensure: the clustered dataset
Begin with a single group containing all the data objects.
repeat
Select the cluster with the largest within-cluster squared error.
Breaks the cluster into two smaller ones using Bisecting k-means to maximize Ward's distance.
until Satisfaction of stopping criterion.

Divisive clustering doesn't necessarily require exhaustive exploration of all potential splits such as the monothetic strategy defined below. Divisive algorithms of polynomial complexity have been proposed, but with no guarantee of optimality [44].

There are two strategies of the divisive method: (i) monothetic (also known as association analyses), and (ii) polythetic. A monothetic cluster is a group in which all objects have the same value of a variable or almost the same, the division of a group into two subgroups is based on a single attribute (variable). Polythetic methods are more flexible, considering all features to form groups. Monothetic approaches are more convinient to binary data.

Pointing out that the specification of monothetic and polythetic methods for just divisive hierarchical methods is not true, because agglomerative methods are polythetic methods [15, 84].

These methods have been less common than bottom-up methods. For an example application of a top-down method, see [15, 44, 30].

Bisecting k-means

This method utilizes the classic k-means algorithm as a building block, but instead of assigning data points to existing clusters, it focuses on iteratively splitting clusters into smaller subgroups [5].

Algorithm 5 Bisecting k-means

Require: dataset (D)
Ensure: the clustered dataset
Begin with a single group containing all the data objects.
repeat
Pick a cluster to break down.
repeat
Select two data points from the cluster as centroids of the new sub-clusters.
until Until convergence.
until Satisfaction of stopping criterion.

2.5.6 Density-based methods

These methods consider clusters as areas with a high concentration of data points, separated by areas with relatively fewer data points, clusters expand towards any direction the density leads as long as it does not exceed a threshold. From this functioning, clusters have arbitrary forms, and have protection against noise.

Generally, density is the count of nearby objects within a designated area or volume (neighborhood). Density-based methods have good scalability and require no parameters to specify the number of clusters. These exceptional properties come with certain disadvantages. Mainly, a cluster can contain two sub-clusters of very different densities, both exceeding the threshold. The second drawback comes from the parameters, a small value difference of a parameter can produce radical differences in clusters. The third is the lack of interpretability, where usually clusters found by a density-based algorithm can't be interpreted by features since the individuals within the same clusters may show extremely different features' values [100, 17].

Density-based algorithms can be broken up into two major approaches [100, 17]: (i) density-based connectivity, and (ii) density functions.

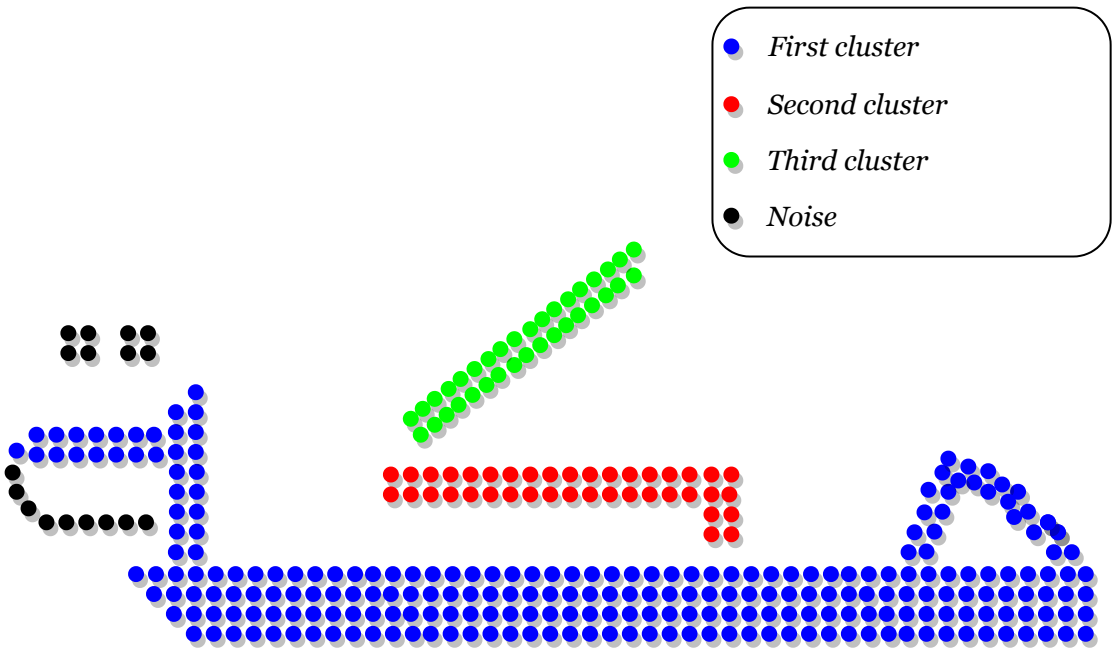


Fig. 2.9 Density clustering

density-based connectivity

In these techniques, there are two crucial concepts, density and connectivity, which are measured according to the local distribution of the nearest neighbours. Connectivity is an equivalence relationship. DBSCAN and OPTICS are two examples of these methods [17].

DBSCAN

To discover clusters of arbitrary forms, Ester et al. [32] introduced the DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm. Understanding the functionality of the algorithm requires prior definition of several key concepts. Two crucial parameters, EPS (epsilon) and $MinPts$, govern its operation. EPS defines the neighborhood of a data point, encompassing all points within a specified distance (EPS) of the reference point.

$$v_{EPS}(p) = \{q \in X | d(p,q) <= EPS\}$$

The other parameter ($MinPts$) is the minimum number of points, which is used to define core objects. A core object is a datapoint that possesses more datapoints than $MinPts$ in its neighborhood. Whenever an object q possesses less datapoints than $MinPts$ on its neighborhood $|v_{EPS}(q)| \leq MinPts$, and this object is on the neighborhood of a core object p according to EPS, $d(p,q) <= EPS$, it will be considered as boundary point. However, if it falls outside a certain distance ("neighborhood") from the core objects it will be called noise. Boundary points are objects located at the edge of clusters. Noise objects are not covered by any cluster [100].

Algorithm 6 DBSCAN algorithm

Require: dataset (D)
Ensure: the clustered dataset
  repeat
    Select an arbitrary not visited object q.
    if $|v_{EPS}(q)| \geq MinPts$ **then**
      collect all density-reachable objects from q to form new cluster.
      if q is classified in another cluster **then**
        merge the two clusters.
      end if
    end if
  until all objects are visited.

  Return the clusters.

An object p is directly reachable by the density of another one q, if q is a core and p falls within its EPS-neighborhood. p is density-reachable by q if there exists a sequence of directly reachable points connecting them, i.e. $p_1, p_2, ..., p_m$ where $p_1 = q$ and $p_m = p$ and for every two consecutive

points p_i and p_{i+1}, p_i is directly reachable by p_{i+1}. An object is connected by density to another one, if they are both reachable by a core point. Cluster membership hinges on reachability from the same core point.

Density functions

As the name suggests, these methods use a function to calculate density. The total density is determined by adding up the density functions of all objects. Clusters are defined by density attractors, which represent the function's local maximums. DENCLUE [55] is an example of these methods [100].

DENCLUE

DENCLUE (DENsity-based CLUstEring) was proposed by Hinneburg and keim [55]. The algorithm uses as density function the sum of influence functions. The latter could be the square wave function, which is defined as follows [5]:

$$f_{square}(p,q) = \begin{cases} 0 & \text{if } d(p,q) > \sigma \\ 1 & \text{else} \end{cases} \tag{2.19}$$

The mathematical representation for the density of a point p is given by:

$$f_{den}(p) = \sum_{q \in D} f_{square}(p,q) \tag{2.20}$$

Such that: D is the dataset. A cluster is defined by density attractors. A density attractor p^* is a local optimum of the density function f_{den}. If its density function value exceeds a threshold ε, then this point and the points attracted to it by the density, form a cluster. A point p is deemed attracted to an attractor p^* if there is p^k that validates the following equation [55, 128]:

$$d(p^*, p^k) \leq \varepsilon \tag{2.21}$$

where:

$$p^0 = p, and$$

$$p^i = p^{i-1} + \delta \frac{\nabla f(p^{i-1})}{\|\nabla f(p^{i-1})\|}$$

$$\nabla f(p) = \sum_{q \in D} (q-p) f_{square}(p,q)$$

ε is a parameter to be defined, and δ is the control of convergence speed. The calculation of p^i ends when $f(p^{i-1}) < f(p^i)$ [55].
If the density value of the p^* is less than ε, then the points attracted to it by the density are considered

as noise. If two attractors of different clusters are connected through a chain of points and the density value of each point of the chain is equal to or greater than ε, then the two clusters will be merged [55].

By using the previous influence function, if we define $\sigma = EPS$ and $\varepsilon = MinPts$, DENCLUE forms arbitrary clusters equivalent to those produced by DBSCAN [55, 5].

2.5.7 Other clustering methods:

There are many other clustering methods that are different or intersect with the afore-discussed ones such as: grid-based, graph-based, probabilistic and fuzzy clustering methods [40, 14, 52].

2.6 Clustering quality measures

In literature, plenty clustering algorithms were proposed for different applications. Since the clustering is an unsupervised process it is important to use special measures that can assess the quality of results to reveal the performance of the algorithms. There are two types of measures: evaluation and performance measures. The first type measures are mainly used to assess the quality of a clustering result. Performance measures on the other hand are utilized for comparing algorithms' efficiency using some benchmarks such as the execution time [14]. In this thesis, we are concerned about the first type. The following section presents a brief overview about these measures.

2.6.1 Clustering validity indices

The quality of clustering's results can be evaluated by validity indices. There are three types of validity indices [50, 51]:

- **Internal validity indices:** in this type, the clustering evaluation is based on the term of quantities that interlace the vectors of the dataset themselves (e.g., the proximity matrix).
- **External validity indices:** in this type, the evaluation of the results is based on a pre-specified structure imposed on the dataset which may be the classes created by an expert.
- **Relative validity indices:** in this type, the evaluation is done by comparing the clusters' structures obtained by the same algorithm applied to the same dataset but with different parameter values, or by different clustering algorithms.

Internal measures

- **Davies and Bouldin index:** This index is a combination of the homogeneity and separation measures [24].

– Homogeneity:

$$H_i = \frac{1}{n_i} \sum_{x \in C_i} d(x, \mu_i)$$

Where, C_i, μ_i, and n_i are respectively the cluster number i, its center and size. $d(x, \mu_i)$ represents the distance between the data point x and μ_i.

– Separation:

$$S_{i,j} = d(\mu_i, \mu_j)$$

Davies and Bouldin index is defined as [24]:

$$DB = \frac{1}{K} \sum_{i=1}^{K} DB_i \tag{2.22}$$

where:

$$DB_i = \max_{1 \leq j \leq K, i \neq j} \frac{H_i + H_j}{S_{i,j}}$$

where K is the number of clusters. This index is an internal measure as it does not include any knowledge besides the dataset features. Low DB index values indicate better results.

- **Dunn index:** This index is designed to find compact and well-separated cluster [29, 51].

$$Dunn = \min_{i=1,\ldots,K} \left\{ \min_{j=i+1,..,K} \left(\frac{d(C_i, C_j)}{\max_{l=1,..,K} diam(C_l)} \right) \right\} \tag{2.23}$$

Where:

– $d(C_i, C_j)$ is the distance between the clusters i and j which is calculated as:

$$d(C_i, C_j) = \min_{x \in C_i, y \in C_j} d(x, y)$$

– $diam(C_l)$ is the diameter of the cluster l calculated as:

$$diam(C_l) = \max_{x,y \in C_l} d(x, y)$$

This index is an internal measure as it does not include any knowledge besides the data features. High Dunn index values reflect the better results.

- **Sum of squared error:** This index is based on the concept of minimizing the intra-clusters distances (distances between the center and the individuals of the same cluster), which tends towards finding more compact clusters. This index is the objective function utilized by k-means algorithm [78].

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} d^2(x, \mu_i) \tag{2.24}$$

This index is an internal measure. Low SSE values indicate the good results.

External measures

- **Homogeneity:** the homogeneity index is used to indicate the rate where the individuals within the same cluster are from the same classe.

$$Hom = 1 - \frac{H(CL|C)}{H(CL)} \tag{2.25}$$

Where:

$$H(CL|C) = -\sum_{i=1}^{K}\sum_{j=1}^{Q}\frac{n_{i,j}}{n}.\log\left(\frac{n_{i,j}}{n_j}\right)$$

$$H(CL) = -\sum_{j=1}^{Q}\frac{n_j}{n}.\log\left(\frac{n_i}{n}\right)$$

n_i, n_j, $n_{i,j}$, and n are, respectively, the number of individuals in the class i, the cluster j, the number of common individuals between the class i and the cluster j, and the size of the dataset (total number of data objects). K and Q are respectively the total number of clusters and classes. High values of the Homogeneity index is an indication for the good quality of results.

- **Completeness:** the completeness index indicates the rate where all clusters of the same class are clustered in the same cluster.

$$Com = 1 - \frac{H(C|CL)}{H(C)} \tag{2.26}$$

Where:

$$H(C|CL) = -\sum_{j=1}^{Q}\sum_{i=1}^{K}\frac{n_{i,j}}{n}.\log\left(\frac{n_{i,j}}{n_j}\right)$$

$$H(C) = -\sum_{i=1}^{K}\frac{n_i}{n}.\log\left(\frac{n_i}{n}\right)$$

High values of the Completeness index is an indication for the good quality of results.

- **V-measure:** this index is a combination of the two previous measures. It is calculated as follows:

$$V - measure = 2.\frac{Hom.Com}{Hom + Com} \tag{2.27}$$

High values of the V-measure index is an indication for good quality of results.

- **Purity:** this index represents the rate of the data points that are clustered correctly.

$$Purity = \frac{1}{n}\sum_{i=1}^{K}\max_{j=1,\ldots,Q}(n_{i,j}) \tag{2.28}$$

High values of the purity index indicate the good quality of results.

- **Entropy:** this index is calculated as follows:

$$Entropy = \sum_{i=1}^{K} \frac{n_i}{n} E(C_i) \tag{2.29}$$

where:

$$E(C_i) = -\frac{1}{\log(Q)} . \sum_{j=1}^{Q} \frac{n_{i,j}}{n_i} \log\left(\frac{n_{i,j}}{n_i}\right)$$

Low values of this index is a sign for the good results.

- **Precision:** this index is calculated as follows:

$$Precision = \frac{TP}{TP+FP} \tag{2.30}$$

where:

  - **True Positives (TP):** number of pairs that are clustered together and are from the same class.
  - **False Positives (FP):** number of pairs from different classes that are clustered in the same cluster.
  - **False Negatives (FN):** number of pairs from the same class that are clustered in different clusters.

High values of this index indicate the good quality of results.

- **Recall:** this index is calculated as follows:

$$Recall = \frac{TP}{TP+FN} \tag{2.31}$$

High values of the recall index indicate the good quality of results.

- **G-measure:** this index is calculated as follows:

$$G-measure = \sqrt{Precision * Recall} \tag{2.32}$$

High values of this index indicate the good quality of results.

- **Error rate:** this index is calculated as follows:

$$ErrorRate = \frac{Number-of-misplaced-points}{n}.100 \tag{2.33}$$

Low values of this index indicate the good quality of results.

There are plenty other measures. Further information about evaluation and performance measures are provided in [14, 50, 51, 8].

2.7 Conclusion

Chapter 2 provided a comprehensive state of the art of data clustering, shedding light on its objectives, applications, its general main steps and the measures to assess the results. The chapter has delved into the fundamental principles of data clustering.

The historical evolution of data clustering, from its early roots in some specific fields to its widespread adoption in several scientific fields, underscores the enduring relevance and impact of this technique. The chapter highlighted some diverse definitions of data clustering and its emergence as a tool for different principle objectives including structuring, describing, associating, generalizing, and visualizing data in a wide array of applications.

Furthermore, the chapter provided insights into the taxonomy of clustering techniques, including partitional methods, hierarchical methods, and density-based methods, offering a comprehensive understanding of the diverse approaches to data clustering. The chapter was closed by exhibiting different measures to assess the clustering results including the utilized ones in this thesis.

The techniques presented in this chapter have fatal drawbacks in addition to the ones introduced in each technique, which are the premature convergence and the stagnation in local optima while searching the solution space.

In the next chapter, we will present some of the well-known techniques in solving optimization problems including data clustering. These methods are called metaheuristics. They are capable of finding good solution to the combinatorial problems in acceptable amount of time and avoid the drawbacks of the previous techniques.

Chapter 3

Metaheuristics

3.1 Introduction

Metaheuristics are optimization methods inspired by the behaviors, different working systems and theories in the real world. Particle swarm optimization (PSO), tuna swarm optimization (TSO), rat swarm optimizer (RSO) are some examples of metaheuristics drawing inspiration from the social behavior of animals, fish, and birds. Others are inspired by human activities and its characteristics, for instance: teaching-learning-based optimization (TLBO), student psychology based optimization algorithm (SPBO), and past present future (PPF) algorithm. Others are inspired by the spacial systems, events, and physical theories such as: gravitational search algorithm (GSA), black hole algorithm (BH), and multi-verse optimizer (MVO). There are also chemical-based mechanisms metaheuristics. For instance: chemical reaction optimization (CRO), henry gas solubility optimization (HGSO), and artificial chemical reaction optimization algorithm (ACROA). Some other metaheuristics are inspired by evolutionary systems, the most well-utilized one in this category is genetic algorithm (GA).

Metaheuristics can be classified using several criteria, for instance: the utilized optimization method **(deterministic, or stochastic)**, the navigation scope strategy **(local, or global)**, the solution search strategy **(trajectory-based, or population-based)**, the memory strategy **(memory-less, or memory-based)**, the environment type **(static, or dynamic)**, the parallelism **(mono-processing, or multi-processing)**, the objective function manipulation **(static, or dynamic)** and other criteria [98].

One popular classification, is based on the source of inspiration. There is no universal classification based on the source of inspiration. This is due to two main reasons: (i) the ratio of detailing and specification, and (ii) the evolving number of recent-developed metaheuristics from different emerging areas that can't be fit in one static classification. Some researchers consider four categories, such as: *evolutionary, human, physics, and swarm intelligence-based* metaheuristics [11], others distinguish between physical and chemical-based ones [92], other ones include bio-inspired (not swarm intelligence) instead of human-based class [59].

Unlike heuristics, which are dedicated to solve only one specific problem, metaheuristics are used in any optimization problem. The only task to do for each one is to adjust the metaheuristic to fit each problem if needed. Another advantage about metaheuristics is their ability to escape local optima and search in a wide area of the solution space.

Some examples of metaheuristics and their applications to data clustering are presented in the following sections.

3.2 Genetic Algorithms

3.2.1 Inspiration

The genetic algorithms (GA) is an evolutionary-based algorithm that was first introduced by Goldberg [42] at the end of 1980s, while, some of its concepts where published by Holland [57] in 1970s. GA fells on evolutionary-based algorithms as it is inspired by the genetic biological system [106]. In GA, durability to the best is the primary simulated process. Better individuals have a better chance of surviving and dispersing their genes to subsequent generations. Inspired by chromosomes and genes, this algorithm considers chromosomes (individuals) as solutions, and genes as their variables.

The GA is a population-based technique that simultaneously explores diverse regions of the solution space. It uses stochastic parameters and operators to avoid getting stuck at the local optima. Its main parts are introduced in the following sections.

3.2.2 Generation of population

In this algorithm, the chromosomes represent the solutions. Thus, it starts by assigning random points to the chromosomes. This step can be performed using the following equation:

$$G_i = (Ub_i - Lb_i) * r - Lb_i \tag{3.1}$$

G_i is a gene (variable) in a chromosome (solution), Ub_i, and Lb_i are respectively the max and min bounds of the i^{th} variable. r represents a random number between 0 and 1.

3.2.3 Selection

In GA, the chromosome's chance for surviving is correlated to its fitness score. In each iteration, a number of individuals are selected to be a part of some operations. This mechanism can be simulated with a roulette wheel, where the better the individual is the larger its portion on the pie chart.

Table 3.1 Detailed description of individuals (maximization fitness)

Chromosomes	Fitness value	Percentage in the pie chart
Yellow	9	10%
Blue	13.5	15%
Red	27	30%
Green	40.5	45%
Sum	90	100%

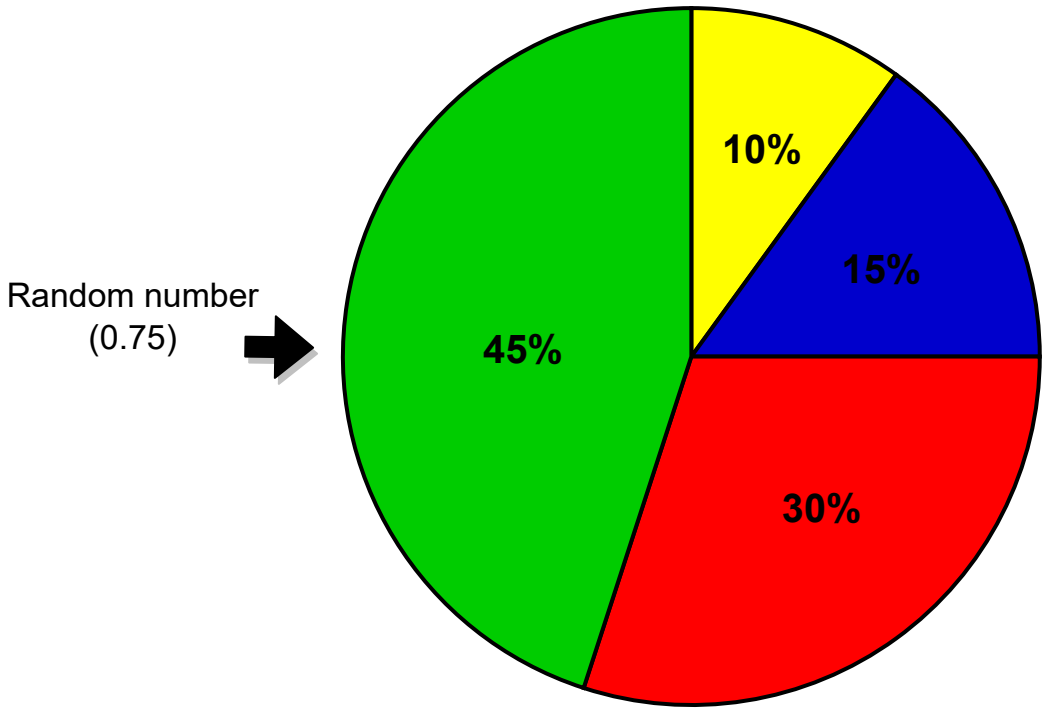


Fig. 3.1 Roulette wheel

Figure 3.1 and table 3.1 explain more the roulette wheel, where the objective type in this example is maximization (Greater fitness value equals better result). Some other examples of selection mechanism include: Boltzmann selection [43], tournament selection [87], steady state selection [114], and rank selection [71].

3.2.4 Crossover

The selected individuals, pass by the first operation which is crossover (combination). In this step the selected (parent) chromosomes interchange their genes to generate child chromosomes. There are

several methods to do so, one method consists of splitting the parent chromosomes into two or more pieces and combine them to generate child individuals. Figure 3.2 illustrates this process.

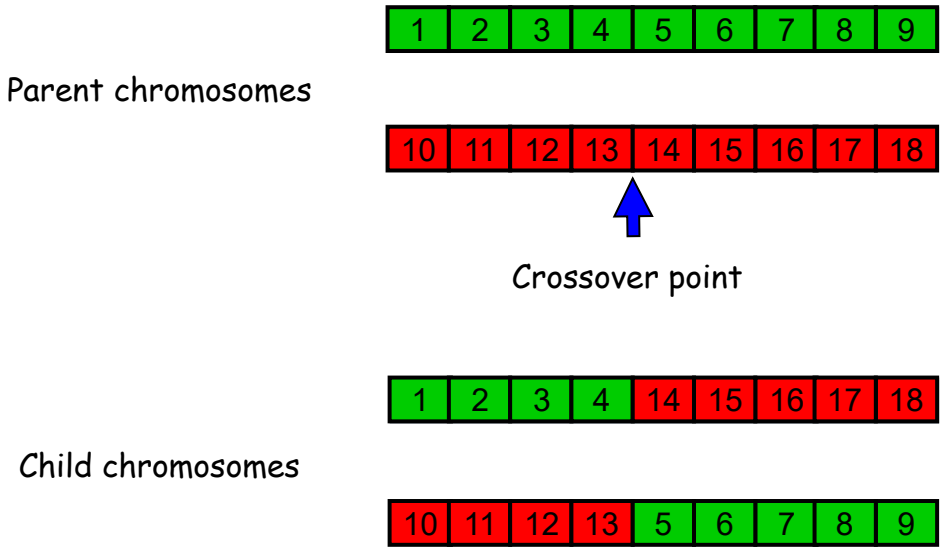


Fig. 3.2 Crossover

The crossover step includes the parameter P_c that represents the child's propagation probability. For each child chromosome, if a stochastic calculated value is less than P_c, then, the current child continue to spread for the next iteration, if not, the parent propagates. Some other examples of selection mechanism include: cycle crossover [97], and three parents crossover [118].

3.2.5 Mutation

After the crossover step, the resulting chromosomes passes by another operation which is mutation. A random number between 0 and 1 is generated for every chromosomes' gene. If the random generated number is less than the probability parameter P_m then, an aleatory value in $[L_b, U_b]$ is attributed to the gene.

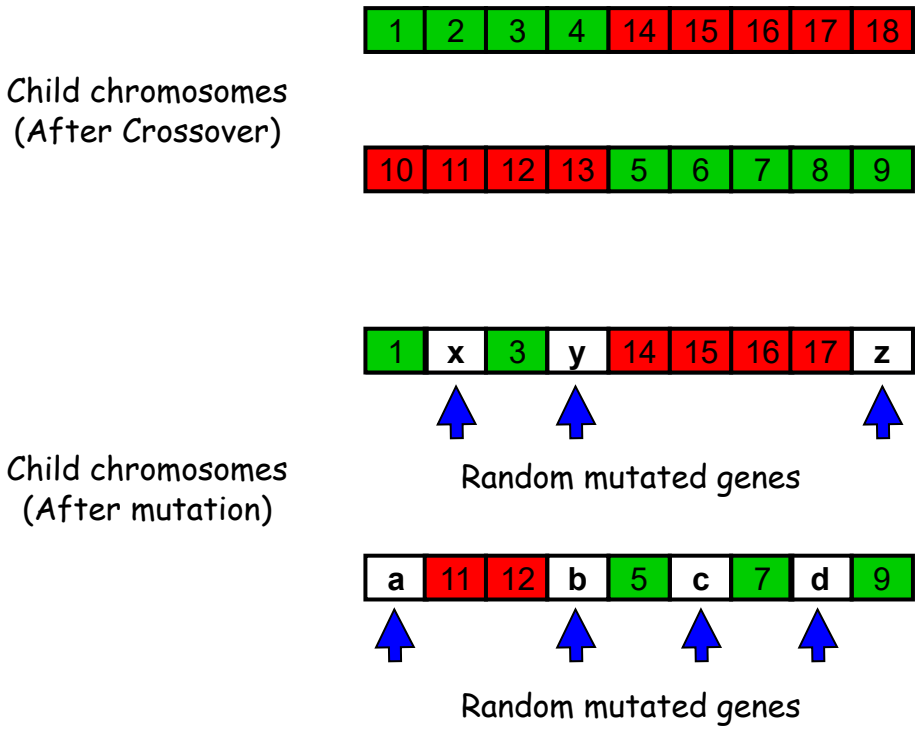


Fig. 3.3 Mutation

Figure 3.3 illustrates this process. Other mutation techniques include; uniform [112], Gaussian [56], and non-uniform [95].

3.2.6 Elitism

Elitism is a mechanism that prevents the good solution loosing due to the aforementioned operations. Due to its huge influence on the GA's performance, researchers have included this method. The fittest chromosomes from the present generation are preserved and passed on unchanged to the following generation. The list of best individuals is updated by ranking their fitness values.

The Genetic Algorithm (GA) initiates by initializing the population with random solutions. Then it calculates the fitness value of each chromosomes. Some individuals are selected using a selection operator to reproduce their off-spring through crossover. The exploration mechanism in GA is exhibited in the mutation operator, whereby, GA ensure exploration new solutions. The processes from calculating the fitness until mutation, are repeated until stopping criteria is met. The repeated process ensure the gradual optimization of solutions.

Algorithm 7 GA

Require: Population size n, max number of iterations Max_Iter.

Ensure: the best solution S^*.

Initialize the population of chromosomes. ▷ Population Initialization

Set the selection, crossover, and mutation probability parameters.

while $(t < Max\_Iter)$ **do**

Calculate the fitness value of each chromosome.

Select some chromosomes as parents.

Apply crossover to generate new solutions.

Apply mutation to some chromosomes' genes.

Add the best solutions to the next population.

$t \leftarrow t + 1$.

end while

Return: The best solution.

3.3 Genetic algorithm for data clustering

GA has been widely applied to data clustering [22, 60, 77, 82, 94, 99, 14]. They exhibit remarkable versatility in addressing various clustering problems, including [14]:

- Centroid-based clustering: GA excel at finding optimal centroid locations, representing cluster centers within the data space.
- Medoid-based clustering: Identification of medoids, real data points that act as cluster representatives, is done by GA very well too. However, in here there should be a function to ensure that the genes are one of data points.

The fitness function in GA depends on each different application. GA demonstrate superior data handling capabilities, they can effectively cluster data containing both numerical and categorical attributes, and they often produce better results than k-means [14].

3.4 Rat swarm optimizer

3.4.1 Inspiration

Rat swarm optimizer (RSO) is a recent metaheuristic proposed by Dhiman et al. [25]. It imitates the behaviour of rats in wild chase and combat. Rats have a natural social intelligence. They assist one another and participate in different jobs. They form bands and they are well recognized for their

aggressive chase and fight behaviors.

3.4.2 Mathematical model and optimization algorithm

RSO is a population based algorithm. The set of solution here is represented by rats population. Each rat symbolizes a distinct one. As a first step, RSO generate the population randomly. After this step, the algorithm assesses the rats (solutions) with the help of an objective function, and the solution with the best result is saved in $\overrightarrow{P}_r$ which represents the best rat. The algorithm then, iteratively, updates the rats position through the two behaviours: chasing and fighting the prey and calculates their fitness. The algorithm then, updates the parameters and adjust any solution that have moved beyond the search space. The algorithm updates $\overrightarrow{P}_r$ if there is a better solution. RSO terminates once the iterative process is executed a certain number of times and returns the best solution $\overrightarrow{P}_r$.

Algorithm 8 RSO [25]

Require: Population size n, max number of iterations Max_Iter.
Ensure: the best solution P_r.
Initialize $\overrightarrow{R}$, $\overrightarrow{A}$ and $\overrightarrow{C}$ and set $t \leftarrow 0$. ▷ Parameter Initialization
Initialize the group of rats $P_i(i = 1, ..., n)$. ▷ Population Initialization
calculate the fitness value of each rat.
The best solution is assigned to $\overrightarrow{P_r}$.
while $(t < Max\_Iter)$ **do**
  for each rat **do**
    Update the position of the current rat by Eqs.(3.2, 3.6).
  end for
  Update $\overrightarrow{R}$, $\overrightarrow{A}$ and $\overrightarrow{C}$ by Eqs.(3.3, 3.4 and 3.5).
  Adjust the rats out of the search space.
  Calculate the fitness value of each rat.
  if there is a better solution than $\overrightarrow{P_r}$ **then**
    The position of $\overrightarrow{P_r}$ is updated to the position of the best solution.
  end if
  $t \leftarrow t + 1$.
end while
Return: $\overrightarrow{P_r}$.

3.4.3 Chasing the Prey

Since prey capture is a collective endeavor, RSO identifies the individual ($\overrightarrow{P}_r$) with the most accurate knowledge of the prey's location as the leader. The remaining rats subsequently adjust their positions based on $\overrightarrow{P}_r$'s[25]:

$$\overrightarrow{P} = \overrightarrow{A}.\overrightarrow{P_i}(t) + \overrightarrow{C}.(\overrightarrow{P_r}(t) - \overrightarrow{P_i}(t)) \tag{3.2}$$

$\overrightarrow{P_i}(t)$ refers to the current location of rat number i at iteration (t). $\overrightarrow{C}$ is a random number in $[0,2]$, $\overrightarrow{A}$ is a coefficient that represents besides $\overrightarrow{C}$, the exploitation and exploration parameters. They are calculated as:

$$\overrightarrow{A} = \overrightarrow{R} - t.\left(\frac{\overrightarrow{R}}{maxIteration}\right) \tag{3.3}$$

$$\overrightarrow{C} = rand(0,2) \tag{3.4}$$

$\overrightarrow{R}$ represents a random number drawn from the uniform distribution between 1 and 5. $maxIter$ denotes the maximum number of allowable repetitions for the iterated section.

$$\overrightarrow{R} = rand(1,5) \tag{3.5}$$

3.4.4 Fighting the Prey

This behavior represent the updation of position, the next position of a random rat is calculated as:

$$\overrightarrow{P_i}(t+1) = \left|\overrightarrow{P_r}(t) - \overrightarrow{P}\right| \tag{3.6}$$

In the RSO algorithm, the parameters $\overrightarrow{A}$ and $\overrightarrow{C}$ regulate the trade-off between exploring the search space and exploiting promising regions. A moderate values of $\overrightarrow{A}$ and $\overrightarrow{C}$ (close to 1) will intensify the search around the best solution, while more distant values will lead to diversify the search process.

3.5 Grey Wolf Optimizer

3.5.1 Inspiration

Grey Wolf Optimizer (GWO) is a SI metaheuristic that was developed by Mirjalili et al. [88] in 2014. The main distinctiveness of GWO than other SI algorithms is that it imitates the social hierarchy of grey wolves' hunting behavior. They like living in packs. On average, there are 12 to 15 people in the group. They follow a strict dominating social structure.

A Group of grey wolves consists of four categories. Alpha (α) is the wolf that leads the group. He has the power to decide where to rest, when to hunt, and so on. The orders of the leader are obeyed by the pack. The leader has the ability to effectively manage and guide the pack. Beneath the α in the pack hierarchy reside the Beta β wolves. These individuals act as deputies, supporting the Alpha in decision-making and various pack activities. Their experience and leadership qualities make

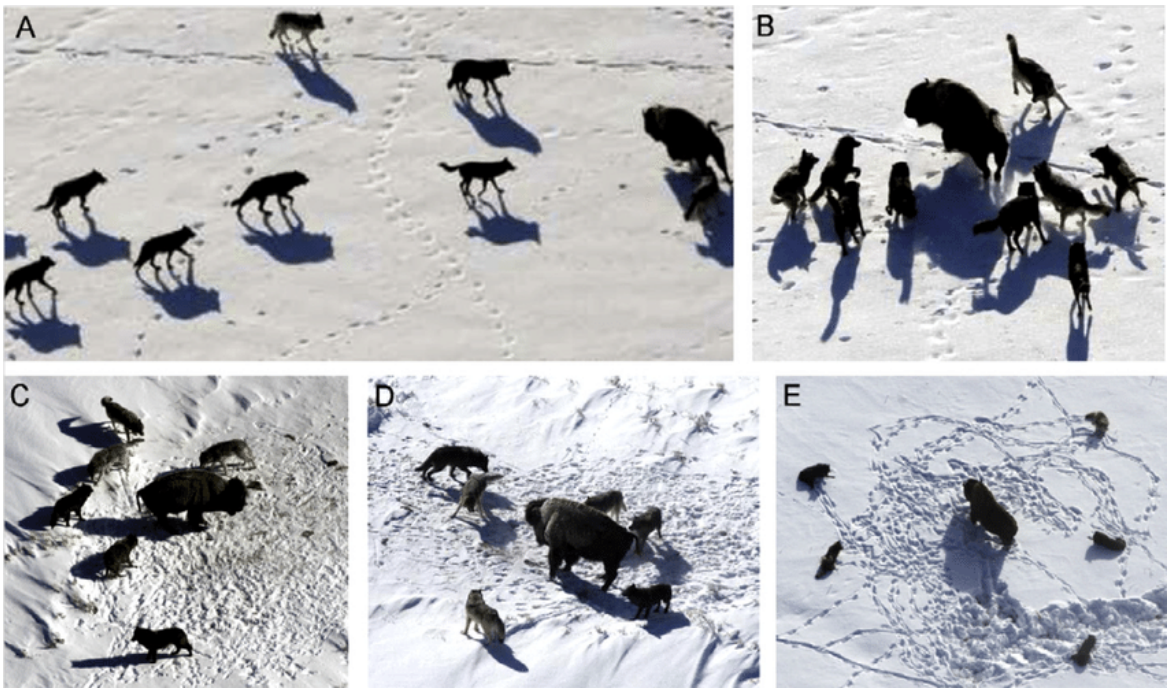


Fig. 3.4 Grey wolves hunting steps [88].

them prime candidates to succeed the α upon its demise or decline due to age. The lowest wolves in the hierarchy are the Omegas (ω). These individuals are required to obey the commands of all other wolves with higher ranks, including the α, β, and δ. As a result of their lower social standing, Omegas are often the last to eat.

3.5.2 Mathematical modelling

According to Muro et al. [91, 88], all agents (wolves) are homogeneous, and there is no prior hierarchy. The three best solutions are, respectively, Alpha, Beta and Delta. The remaining are assumed to be Omegas. The three best wolves are the ones that direct the hunting process.

3.5.3 Prey's encircling

In nature, wolves surround their prey, this behaviour is modeled as follows [88]:

$$\overrightarrow{D} = |\overrightarrow{C}.\overrightarrow{X}_p(t) - \overrightarrow{X}(t)| \tag{3.7}$$

$$\overrightarrow{X}(t+1) = \overrightarrow{X}_p(t) - \overrightarrow{A}.\overrightarrow{D} \tag{3.8}$$

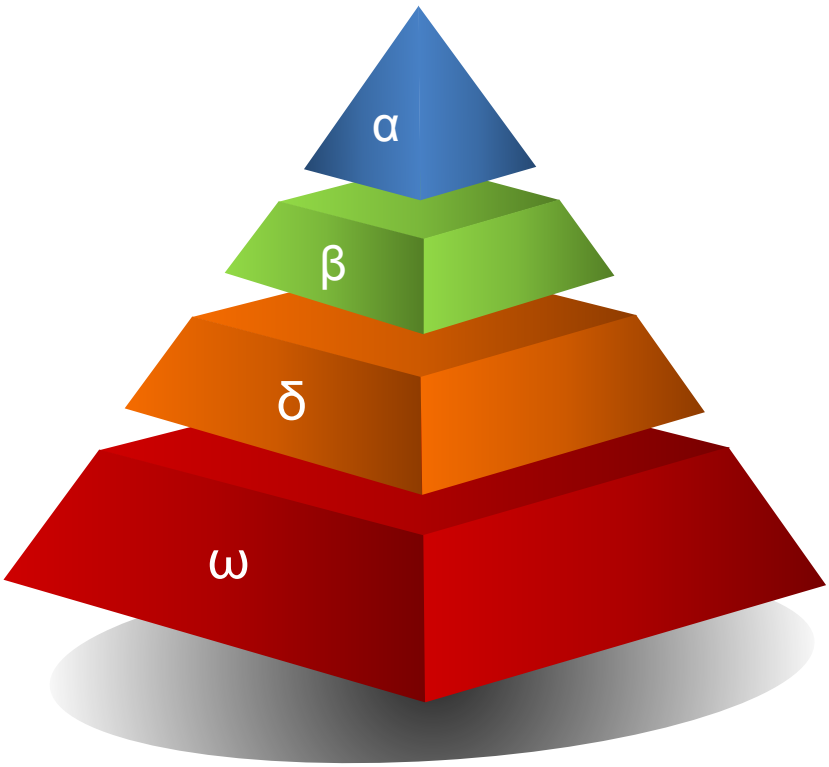


Fig. 3.5 Grey wolves' social hierarchy.

Where $\overrightarrow{D}, t, \overrightarrow{X}_p(t)$, and $\overrightarrow{X}(t)$ represent respectively, the distance between the wolf and the prey, the ongoing iteration, the prey's position, and current the wolf's position. $\overrightarrow{A}$ and $\overrightarrow{C}$ are parameters given by:

$$\overrightarrow{A} = 2\overrightarrow{a}.\overrightarrow{r}_1 - \overrightarrow{a} \tag{3.9}$$

$$\overrightarrow{C} = 2\overrightarrow{r}_2 \tag{3.10}$$

The vector $\overrightarrow{a}$ undergo a linear decrease from 2 to 0 during the iterations. $\overrightarrow{r}_1$, and $\overrightarrow{r}_2$ are vectors in the interval [0,1]. The value of $\overrightarrow{a}$ is calculated as:

$$\overrightarrow{a} = 2 - t\left(\frac{2}{T}\right) \tag{3.11}$$

T here represents the max limit of iterations.

A wolf can move to different positions around the prey with the help of the two equations (3.7) and (3.8) by adapting the vectors $\overrightarrow{A}$ and $\overrightarrow{C}$. The following figure exposes some possible moves.

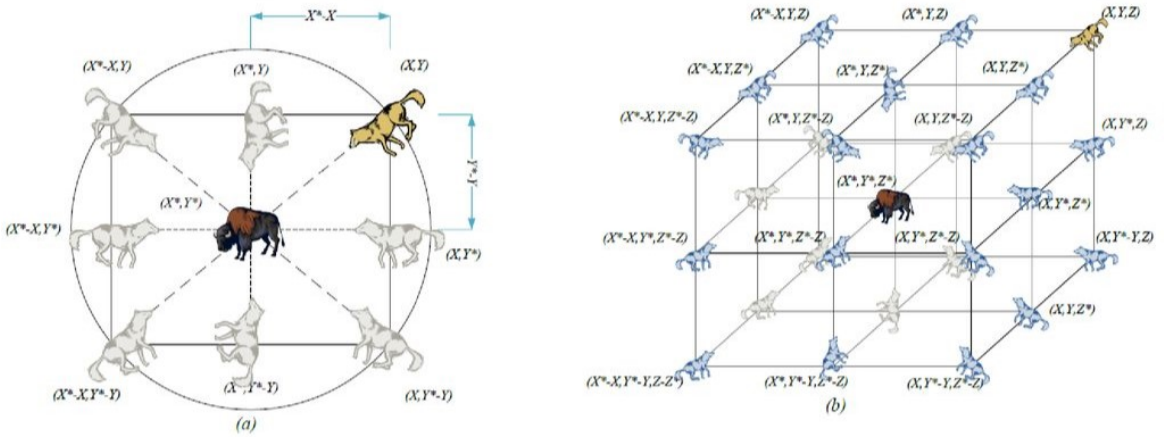


Fig. 3.6 Examples of grey wolves possible moves [88].

3.5.4 Hunting

Hunting is generally guided by Alpha. Occasionally, Betas and Deltas may also take part in the hunt. Nevertheless, within the context of abstract search spaces the optimal solution's position remains unknown. To model this mechanism, it is assumed that the three best agents α, β and δ are more aware of the prey's position. Consequently, the three best solutions identified by these wolves guide the movement of all other agents for further exploration and optimization [88].

$$\overrightarrow{D}_\alpha = |\overrightarrow{C}_1.\overrightarrow{X}_\alpha - \overrightarrow{X}(t)|, \overrightarrow{D}_\beta = |\overrightarrow{C}_2.\overrightarrow{X}_\beta - \overrightarrow{X}(t)|, \overrightarrow{D}_\delta = |\overrightarrow{C}_3.\overrightarrow{X}_\delta - \overrightarrow{X}(t)|. \tag{3.12}$$

$$\begin{aligned} \overrightarrow{X}_1 &= \overrightarrow{X}_\alpha - \overrightarrow{A}_1.\overrightarrow{D}_\alpha, \\ \overrightarrow{X}_2 &= \overrightarrow{X}_\beta - \overrightarrow{A}_2.\overrightarrow{D}_\beta, \\ \overrightarrow{X}_3 &= \overrightarrow{X}_\delta - \overrightarrow{A}_3.\overrightarrow{D}_\delta. \end{aligned} \tag{3.13}$$

$$\overrightarrow{X}(t+1) = \frac{\overrightarrow{X}_1 + \overrightarrow{X}_2 + \overrightarrow{X}_3}{3} \tag{3.14}$$

3.5.5 Attacking the prey

The behaviour of attacking the prey is manifested in the algorithm when the value of $\overrightarrow{A}$ lies between 1 and -1, which obliges wolves to search between the prey and their location. This behavior represents

the notion of exploitation. Other values of $\overrightarrow{A}$ can imply the notion of exploration. The parameter $\overrightarrow{C}$ can also lead to exploration while it takes random values [88].

3.6 Grey wolf algorithm-based data clustering

The main idea in GWAC [72] is to find the best cluster centers. GWO is used to find the optimal cluster centers which are fed to k-means to cluster the dataset.

3.6.1 Agent representation

In GWAC, each wolf (solution) is represented by the cluster centers, which are points on the feature space. For a clustering problem with two clusters, a wolf should consist of two centers, for another one with three clusters, a wolf should consist of three centers. Thus a wolf can be represented as follows [72]:

$$w = (\mu_1, \mu_2, ..., \mu_3)$$

As aforementioned, a cluster center is a point in the feature space. By replacing each center by its features, a wolf can be represented as:

$$w = ((\mu_{1.1}, \mu_{1.2}, ..., f_{1.d}), (\mu_{2.1}, \mu_{2.2}, ..., \mu_{2.d}), ..., (\mu_{K.1}, \mu_{K.2}, ..., \mu_{K.d}))$$

where $\mu_{i.j}$ is the feature number j of the i^{th} cluster, d and K are, respectively, the number of features and clusters.

3.6.2 Population initialization

The initialization of the population is made by assigning K random points from the given dataset to each wolf [72].

3.6.3 Fitness function

In here, the fitness values are calculated throughout two steps. first, the clusters are produced according to the cluster centers of each wolf, where each point is attributed to the cluster with the nearest center to the point. The cluster centers of the current wolf are then updated to the mean of the cluster members. After this process, the fitness value of the results is calculated using SSE [72]. For instance, let $\mu_1 = (a,b,c)$ a center of a cluster C_1, and let $p_l = (1,2,3)$ and $p_m = (4,5,6)$ the only points closer to μ_1, i.e., the two points will be on its cluster. μ_1 will then be updated to the mean of its cluster

members (p_l and p_m) which gives the following result:

$$\mu = (2.5, 3.5, 4.5)$$

3.6.4 Position updation

The position updation of the wolves is done with the help of the positions of the three best solutions (alpha, beta, and delta) as stated in equations (3.13,3.14) [72]. Let $\overrightarrow{X}_1 = ((a,b,c),(d,e,f))$, $\overrightarrow{X}_2 = ((aa,bb,cc),(dd,ee,ff))$ and $\overrightarrow{X}_3 = ((u,v,w),(x,y,z))$. The next position of the current wolf is calculated as follows:

$$\overrightarrow{X}(t+1) = ((\frac{a+aa+u}{3}, \frac{b+bb+v}{3}, \frac{c+cc+w}{3}), (\frac{d+dd+x}{3}, \frac{e+ee+y}{3}, \frac{f+ff+z}{3}))$$

Algorithm (9) represent the GWAC.

Algorithm 9 GWAC[72]

Require: Control Coefficient a, population size n, max number of iterations Max_Iter.
Ensure: the best solution $\overrightarrow{X}_\alpha$.
Generate initial population of n grey wolves.
Set iteration counter t = 0.
Compute the fitness of each grey wolf.
Identify the first three best grey wolves based on fitness.
while t<Max_Iter **do**
  for each grey wolf **do**
    Compute control coefficients $\overrightarrow{A}$, and $\overrightarrow{C}$ by Eqs. (3.9 and 3.10).
    Update the position of the current wolf using Eq. (3.14).
  end for
  Update the coefficient a.
  Compute the fitness of all grey wolves.
  Update the value of the three best position (alpha, beta and delta).
  t=t+1.
end while
Return: the best solution $\overrightarrow{X}_\alpha$.

The algorithm consists of two parts, the first one starts by the initialization of parameters and population where it assigns to each wolf K random points from the dataset. Then, it computes the fitness of each wolf as explained in section (3.6.3) and updates the values of $\overrightarrow{X}_\alpha$, $\overrightarrow{X}_\beta$ and $\overrightarrow{X}_\delta$. The second part starts by checking the boundaries of each wolf. For each solution, the parameters $\overrightarrow{A}$, $\overrightarrow{C}$ and the current wolf's position are updated using equations (3.9, 3.10, 3.14), the updation of position is done as it is already explained. The value of a is then updated, and after the fitness calculation, the position of $\overrightarrow{X}_\alpha$, $\overrightarrow{X}_\beta$ and $\overrightarrow{X}_\delta$ is updated. This part of the algorithm is repeated a number of times

defined by the user. At the end, GWAC returns the best cluster centers ($\overrightarrow{X}_\alpha$ position) and the clustered dataset by $\overrightarrow{X}_\alpha$ [72].

3.7 Conclusion

In conclusion, Chapter 3 provided a comprehensive overview of metaheuristics, offering insights into optimization methods inspired by real-world behaviors and systems. The chapter has introduced various metaheuristic algorithms, including genetic algorithms, rat swarm optimizer, and grey wolf optimizer, each drawing inspiration from different natural or social behaviors.

The discussion on genetic algorithms has highlighted their foundation in genetic biological systems, emphasizing the concept of survival of the fittest and the simulation of chromosomes and genes as solutions with a brief view of GA for data clustering. Furthermore, the exploration of rat swarm optimizer has shed light on its emulation of the chasing and fighting behaviors of rats in nature, underscoring the social intelligence and cooperative nature of rats as a source of inspiration for optimization algorithms. Additionally, the examination of grey wolf optimizer has provided insights into its unique approach, where it imitates the social hierarchy and hunting behavior of grey wolves. The chapter also exhibits the application of GWO for the data clustering with a comprehensive explanation of its main parts.

By presenting these metaheuristic algorithms, the chapter has contributed to a deeper understanding of some diverse sources of inspiration and their applications in data clustering. The insights provided in this chapter will inform subsequent discussions and help to a comprehensive understanding of our contributions which are some practical applications and enhancements of metaheuristic algorithms in the context of data clustering.

The next chapter will present our first contribution, which is the adaptation of the rat swarm optimizer to solve the data clustering problem. The chapter will expose the adapted algorithm and explain its main points.

Chapter 4

Rat Swarm Optimizer for Data Clustering (First contribution)

4.1 Introduction

This chapter introduces our first contribution: the Rat Swarm Optimizer for Data Clustering (RSOC) method. It begins by describing the adaptation of the rat swarm optimizer algorithm for the data clustering problem and explains the quality of results of this method compared to other algorithms. The chapter also provides details about the RSO-based clustering method, including its solution representation, algorithm, its main parts and time complexity. Additionally, it presents the experimental results and discussion of RSOC compared to other algorithms, which is divided into two parts, which are the comparison with, respectively, MVO and H-HHO and other algorithms. The findings highlight its performance across various benchmark datasets. The rat swarm optimizer was chosen in our work [131] to address the data clustering because of its flexibility in handling different problems as stated in [25], few parameters to setup and its simplicity to be adapted to different problems.

4.2 RSO-based clustering method

RSO-based clustering method (RSOC) is an adaptation of the RSO algorithm in order to handle the data clustering problem. The main idea of RSOC is to find the best cluster centers.

4.2.1 Solution representation

As in RSO (Sec.3.4.1), each rat represents a solution. The solution here is the cluster centers. As each center is a point on the feature space, a random solution P_s can be represented as:

$$P_s = ((\mu_{1,1}, \mu_{i,1,2}, ..., \mu_{1,d}), (\mu_{2,1}, \mu_{2,2}, ..., \mu_{2,d}), ..., (\mu_{K,1}, \mu_{K,2}, ..., \mu_{K,d}))$$

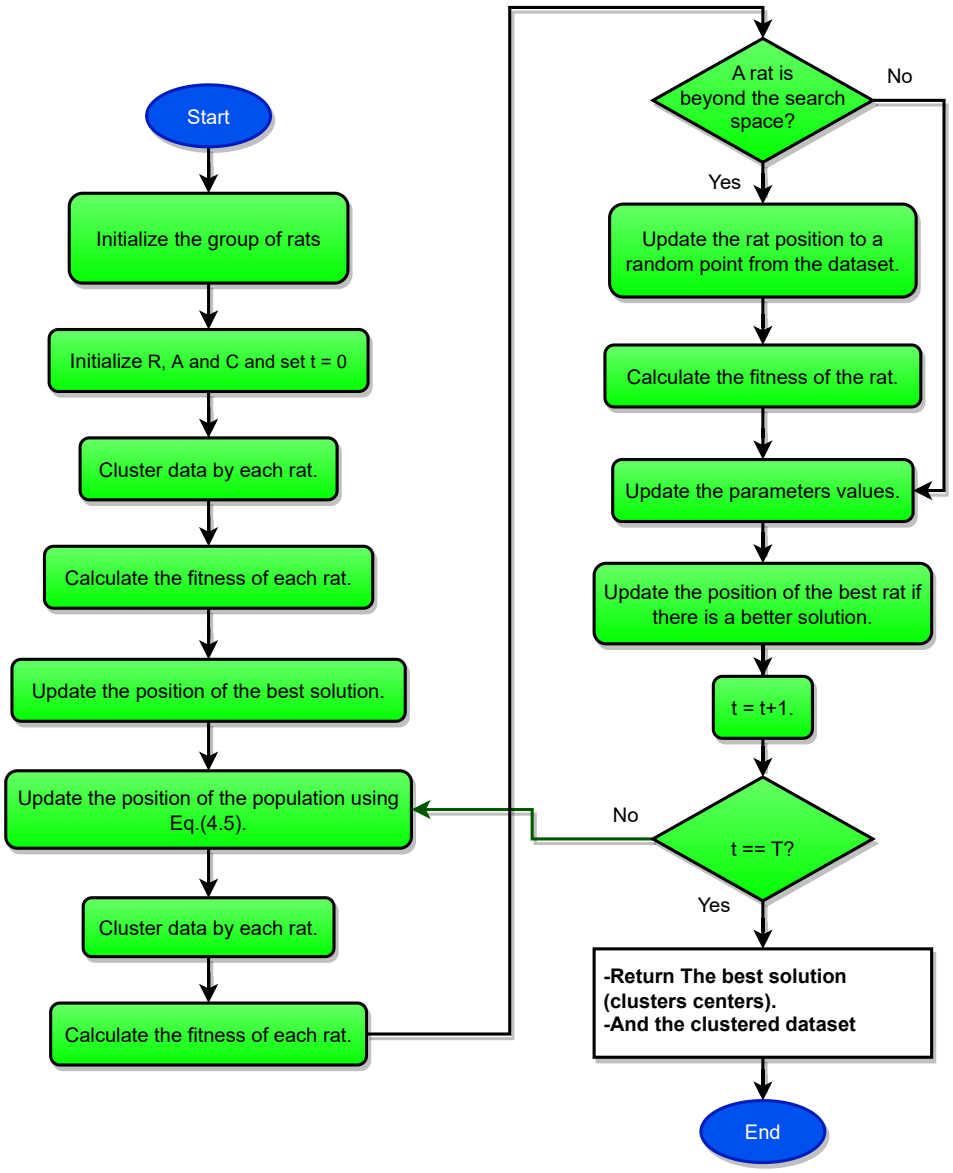


Fig. 4.1 RSOC flowchart

K and d are, respectively, the number of clusters, and features. $\mu_{i,j}$ is the feature number j of the i^{th} cluster.

4.2.2 RSOC algorithm

The first part of RSOC algorithm begins by initialize parameters A, C and R. Rats (cluster centers) are then initialized by K random points from the dataset. After this, the algorithm uses the rats to

cluster the dataset by assigning each data point to the closest center. Since using a rat (cluster centers) results a clustering outcome, by applying the previous process with all rats, we get as much different clusterings of the dataset as the number of rats. After this, the algorithm assesses the results using the objective function and sets the value of the best rat $\overrightarrow{P_r}$ to the one with the best fitness value. The utilized fitness function here is the SSE Eq.2.24.

The second part, starts by updating the rats position and parameters using Eqs.(3.3-3.6). Then, it updates the positions of rates that go beyond the search space with K random points from the dataset. Finally, the dataset is clustered by rats and the results are assessed by the objective function. If there is a better solution than $\overrightarrow{P_r}$, then, its position is updated to the best solution. The second part of the algorithm is the repeated process, it will be repeated a number of times defined by the user. At the end, the algorithm returns the best solution and the clustered dataset. The utilized objective function here is the SSE. The pseudo-code (10) and figure 4.1 depict the discussed RSOC.

Algorithm 10 RSOC [131]

Require: Population size n, dataset, number of clusters, max number of iterations Max_Iter.
Ensure: the best solution P_r and the clustered dataset.

```
Initialize the group of rats P_i(i = 1,...,n).            ▷ Population Initialization
Initialize R⃗, A⃗ and C⃗ using Eqs. (3.3- 3.5) and set t = 0.   ▷ Parameter Initialization
Cluster data using rats.
Calculate the fitness value of rats using Eq.2.24.
Assign the best solution to P⃗_r.
while (t < Max_Iter) do
    for each rat do
        Update the position of the current rat by Eqs.(3.2, 3.6).
        Cluster the dataset using the current rat.
        if the current solution is beyond the search space then
            Update its position randomly.
        end if
        Update R⃗, A⃗ and C⃗ using Eqs.(3.3-3.5).
        Calculate the fitness value of the rat using Eq.2.24.
    end for
    if there is a better solution than P⃗_r then
        The position of P⃗_r is updated to the position of the best solution.
    end if
    t ← t + 1.
end while
Return: P⃗_r and the clusters.
```

4.2.3 Position updation

According to (Eq.3.6), the whole population of rats will update their position in proportion to the best rat position $\overrightarrow{P_r}$, each one will either get closer or not. Getting closer to the best rat represents the intensification mechanism (i.e. exploitation) since the algorithm will verify the neighborhood of the best rat (the rats that got closer to $\overrightarrow{P_r}$). The other case; when rats update their position far from the best rat, represents the diversification mechanism (exploration) where, the algorithm here will verify other regions.

These two mechanisms, which are handled by the different values of the parameters $\overrightarrow{A}$ and $\overrightarrow{C}$, help the algorithm in finding a balance between searching the near region to the best solution and other far regions from it.

4.2.4 Time complexity

The time complexity calculation of the RSOC algorithm will be divided into three main parts to make it clear and easy to understand. The first part constitutes the instructions before the while loop, and part two is the loop **while**. The third part represents the instruction after the loop.

The following items explain the time complexity of each instruction in the algorithm.

1. **Population initialization:** in this process, the main instruction is the assigning of K random data points to each rat. As it was explained in the solution representation section (Sec.4.2.1) each data point is represented by a d features, which results in $K * d$ operations to assign the K data points to a rat (initialization of one rat). Thus, the time complexity to initialize a population of n rats is $O(n * K * d)$.

2. **Parameter initialization:** this process is a simple one, without explicit neither implicit loops or recursive processes, which results in constant time complexity $O(1)$.

3. **Cluster data using rats:** the process of clustering a dataset of s data points is done for each rat which means n times. This process consists of three sub-processes, (i) calculation of the distance between each data point and the K centers, (ii) verification of the shortest distance between the K calculated ones for each data point (the calculated distances between the data point and the K centers), and (iii) the assgnement of labels to each data point. These sub-processes are explained as follows:

    - Calculating the distance between each data point and the K centers has $O(s * K * d)$ time complexity. As each data point has d features, calculating the distance between K centers and a dataset will result in $O(K * d)$. For the whole dataset, the result will be $O(s * K * d)$.

- Verifying the shortest distance among the K calculated ones for each data point (the previous calculation) has also $O(s*K)$ as comparing K resulted distances (simple values) is $O(K)$, and with this process repeated for all data points it will give $O(s*K)$.
- The cluster label assignment for one data point has $O(1)$ time complexity as it is a simple instruction. Thus, for the whole dataset it will be $O(s)$.

The time complexity of the third process (Cluster data using rats) will be the greatest among the previous detailed ones, which is $O(s*K*d)$ for the clustering process using one rat. As we have n rats this phase will result in $O(n*s*K*d)$.

4. **Fitness calculation:** consists of calculating the distance between the cluster centers and their corresponding members which will result in $O(s*d)$. As this process is repeated by each rat it will result in $O(n*s*d)$ time complexity.
5. **Updating $\overrightarrow{P}_r$:** this operation consists of assigning a solution ($K*d$ values) to $\overrightarrow{P}_r$ which has $O(K*d)$.
6. **Updating rats position:** it has $O(n*K*d)$ for the whole population of rats as the updation of a solution has $O(K*d)$.
7. **Rturning back solutions that go beyond the search space:** at the worst case all rats will be out of the search space which will be as the population initialization ($O(n*K*d)$).
8. **Updating parameters:** this process has n times the complexity of the parameters initialization as they will be recalculated for each rat which gives $O(n)$.
9. **Increment the value of t by one:** has $O(1)$.
10. **Returning back the best solution and the clusters:** this process has an $O(s*d)$ as it consists of two sequential processes which are returning the best solution ($O(K*d)$) and the clusters ($O(s*d)$). The grater one between the two is ($O(s*d)$) because the number of clusters K should be less than the number of data points s.

To sum up, the time complexity of the RSOC algorithm can be explained in three points, first part before the while loop processes, second is the while loop and third processes after the loop:

1. **Processes before the while loop:** here there are five sequential processes, this means we will take the bigger time complexity among them, which is $O(n*s*K*d)$.
2. **The while loop:** inside the while loop there are the best solution updation, the updation of the loop index (t) and five sequential processes inside a for loop where their time complexity was calculated previously, the bigger time complexity is $O(n*s*K*d)$. And as these processes are repeated inside the while loop Max_Iter times, the total time complexity of the while loop is $O(Max\_Iter*n*s*K*d)$.

3. **After the while loop:** returning back the best solution and the clusters has $O(s*d)$ time complexity.

Accordingly, the total time complexity of this algorithm is $O(Max\_Iter*n*s*K*d)$.

4.3 Experimental results and discussion

In this section, we will present the experimental results of RSOC compared to other algorithms along with the discussion of the results. The RSOC was compared to two sets of algorithms, in the first one, RSOC was compared to Multi-verse Optimizer (MVO) and other algorithms using four measures. In the second, RSOC was compared to hybrid Harris Hawks Optimization (H-HHO) and twelve other algorithms using an error-rate measure.

The utilized benchmark datasets for the comparisons can be found in the UCI Machine Learning Repository [28]. Table 4.1 details the datasets characteristics.

Table 4.1 Utilized datasets.

Datasets	Number of instances	Number of features	Number of classes
Iris	150	4	3
Ecoli	336	7	8
Glass	214	9	6
Heart	270	13	2
Cancer	683	10	2
Seeds	210	7	3
Wine	178	13	3
CMC	1473	9	3

4.3.1 Benchmark datasets description

The "class" attribute is removed from datasets before the clustering process. The following is a description of the utilized datasets.

- The **Iris** dataset is widely recognized and frequently used in the fields of machine learning and statistics, including data clustering. It contains 150 instances representing three Iris species: Iris setosa, Iris virginica, and Iris versicolor. Each instance is defined by four measurements in centimeters, encompassing the length and width of both sepals and petals. Originally introduced by British statistician Ronald Fisher in 1936 [35]. The features type of the Iris dataset is "real".

- There are 336 samples in the **Ecoli** dataset, each of which represents a single strain of Escherichia coli. The dataset contains information on the biochemical properties of these bacteria. This dataset was gathered in 1992 by Kenta Nakai and Minoru Kanehisa. Each sample is described by seven characteristics with the exclusion of the sequence name and class features. The dataset contains eight classes. The features type is "real" [93].

- The **Glass identification** is a dataset that is used in different domains such as machine learning, the dataset contains 214 samples with 11 features including: Id number (of the glass), refractive index, sodium content, magnesium content, aluminum content, silicon content, potassium content, calcium content, barium content, iron content, type. The data are divided into six different classes and the features type is "real". The dataset was utilized with excluding Id number feature (which is just a number) and the class feature (type) [41].

- The **statlog (Heart)** dataset is a widely used benchmark in machine learning for classification and clustering. It comprises 270 samples patients and the samples are characterized by 13 features (without the class feature), namely: age, sex, chest pain type, resting blood pressure, serum cholestoral in mg/dl, fasting blood sugar > 120 mg/dl, resting electrocardiographic results, maximum heart rate achieved, exercise induced angina, oldpeak = ST depression induced by exercise relative to rest, the slope of the peak exercise ST segment, number of major vessels (0-3) colored by flourosopy, and thal: 3=normal; 6=fixed defect; 7=reversable defect. The features types are categorical and real. The dataset was collected by the German Institute for Medical Documentation and Information (DIMDI) and initialy released in 1998 [26]. It is classified into two classes; presence or absence of heart disease.

- **Breast Cancer Wisconsin** is a benchmark dataset used in machine learning for classification and clustering. This dataset has different versions, the original one contains 699 samples (and 683 with the exclusion of the 16 samples with missing values), each one represents an individual patient undergoing breast cancer diagnosis. The samples are characterized by 10 features (excluding the class label). The features are: clump thickness, uniformity of cell size, uniformity of cell shape, Marginal Adhesion, Single, Epithelial Cell Size, Bare Nuclei, Bland Chromatin, Normal Nucleoli, and Mitoses. The dataset was created by William Wolberg and Olvi Mangasarian in 1992 [127]. The samples are classified into two classes, which are: benign, with 458 samples, and malignant, with 241 samples.

- The **Seeds** dataset is widely used benchmark in machine learning for data clustering. It comprises 210 samples with 7 features representing wheat kernels. These features are: The surface area of the kernel, Perimeter, Compactness, Length, Width, Asymmetry, Kernel depth. There are three classes of the wheat kernels, and the features type is "real" [21].

- The **Wine** Dataset provides information about different wines' chemical composition and sensory characteristics. It includes 178 samples with 13 integer and real features, namely:

alcohol, malic acid, ash, alcalinity of ash, magnesium, total phenols, flavanoids, nonflavanoid phenols, proanthocyanins, color intensity, hue, OD280/OD315 of diluted wines, and proline. The Wine dataset consists of three classes (class1: 59, class2: 71, and class3: 48 samples) [4].

- The **Contraceptive Method Choice** (CMC) is a sub-dataset extracted from the 1987 National Indonesia Contraceptive Prevalence survey. It consists of 1473 samples which represent married women asked about their contraceptive method choice. Ten attributes collected about each women including the class label (contraceptive method used): wife's age, wife's education, husband's education, number of children ever born, wife's religion, wife's now working?, husband's occupation, standard-of-living index, media exposure, and the contraceptive method used (class feature) [76].

The RSOC results for both comparisons were carried out in a laptop with Intel(R) Core(TM) i7-7500U CPU @ 2.70GHz 2.90 GHz processor, DDR4 4.0 GB RAM using Matlab R2021a under windows 10 home 64-bit operating system.

4.3.2 Comparing to MVO

In this part, RSOC is compared to four algorithms: MVO, GA, PSO, and differential evolution (DE) which are tested on the aforedescribed five datasets; Iris, Ecoli, Glass, Heart and Seeds. The comparison results are assessed through four measures, notably: homogeneity, completeness, v-measure and purity. High values of these measures are an index of the effectiveness of the clustering algorithm in accurately identifying clusters within different types of datasets.

The results of the algorithms compared with are directly taken from [10]. Table 4.2 presents the algorithms parameters. The max number of iterations and the population size were set for all algorithms respectively to 200 and 50 while the results were collected through 10 independent runs. The machine used for this assessment for MVO and the other algorithms is stated in [10].

Table 4.2 The parameters of algorithms.

Algorithm	Parameter	Value
PSO	Inertia factor	0.1
	c_1	2
	c_2	2
DE	C Crossover	0.5
	F Scaling	$[0.2, 0.8]$
GA	Crossover	0.8
	Mutation	0.02
MVO	WEP_{max}	1
	WEP_{min}	0.2
RSOC	R	[1,5]
	C	[0,2]

Tables 4.3- 4.7, provide clustering results for different datasets using various algorithms. These tables present the values of different evaluation measures such as homogeneity, completeness, v-measure, and purity for each algorithm and dataset. The summarized results in the tables reveal the ability rate of each one of the algorithms to accurately cluster the data. Based on the results, RSOC consistently performs well across multiple datasets, achieving high values for homogeneity, completeness, v-measure, and purity, indicating its effectiveness in clustering the data accurately. Other algorithms such as DE, PSO, GA, and MVO show varying levels of performance across different datasets. However, they generally have lower values compared to RSOC, indicating that RSOC outperforms them in terms of clustering quality. The best results in tables are in bold.

Table 4.3 presents the clustering results of the RSOC approach compared to other optimization algorithms specifically for the Iris dataset. The results show that RSOC outperformed all other algorithms in terms of all measures but completeness where it showed the second best result after PSO regarding the Iris dataset. Table 4.4 presents the clustering results of algorithms for the Ecoli dataset. As in Iris, RSOC showed the best results through all measures but completeness where it ranked the second after PSO. In Table 4.5, the results suggest that RSOC was effective in producing homogeneous and pure clusters for the Glass dataset where it gave the best results for all measures except for completeness which was given by MVO. Table 4.6 presents the clustering results of RSOC and the other algorithms for the Heart dataset. In this dataset RSOC did not perform well whereas MVO showed the best results. The results of clustering regarding Seeds dataset are presented in Table 4.7. In this case, RSOC demonstrated competitive performance compared to other algorithms.

It achieved higher values for homogeneity, completeness, v-measure, and purity compared to MVO, GA, PSO and DE.

Overall, the tables provide a comprehensive comparison of different clustering algorithms and their performance on various datasets, with RSOC emerging as the top-performing algorithm in most cases.

Table 4.3 Clustering results of Iris dataset.

	Homogeneity	Completeness	V-Measure	Purity
DE	0.72778	0.75507	0.74096	0.86733
	(0.04379)	(0.04469)	(0.04293)	(0.03777)
PSO	0.65750	**0.82877**	0.72629	0.77133
	(0.07052)	**(0.09641)**	(0.02481)	(0.09270)
GA	0.60002	0.69056	0.64046	0.75333
	(0.09578)	(0.09905)	(0.09045)	(0.08433)
MVO	0.73642	0.74749	0.74191	0.88667
	(0.00000)	(0.00000)	(0.00000)	(0.00000)
RSOC	**0.74847**	0.76149	**0.75492**	**0.89200**
	(0.00635)	(0.00738)	**(0.00686)**	**(0.00281)**

Table 4.4 Clustering results of Ecoli dataset.

	Homogeneity	Completeness	V-Measure	Purity
DE	0.43868	0.56485	0.49188	0.69235
	(0.10838)	(0.12341)	(0.11438)	(0.07287)
PSO	0.22629	**0.74693**	0.31740	0.57187
	(0.14762)	**(0.17063)**	(0.20040)	(0.09071)
GA	0.44054	0.52583	0.47512	0.67890
	(0.08253)	(0.08403)	(0.06779)	(0.06717)
MVO	0.50214	0.71637	0.58060	0.72508
	(0.13705)	(0.04119)	(0.10298)	(0.07459)
RSOC	**0.69627**	0.54324	**0.61021**	**0.81815**
	(0.01868)	(0.02423)	**(0.02162)**	**(0.01559)**

Table 4.5 Clustering results of Glass dataset.

	Homogeneity	Completeness	V-Measure	Purity
DE	0.18996	0.46231	0.26717	0.45047
	(0.05362)	(0.08873)	(0.06913)	(0.03201)
PSO	0.17044	0.46871	0.24495	0.44206
	(0.07987)	(0.11835)	(0.10986)	(0.04295)
GA	0.24416	0.40213	0.30203	0.48972
	(0.04901)	(0.08900)	(0.05786)	(0.03763)
MVO	0.24341	**0.50376**	0.32666	0.47804
	(0.03544)	**(0.07557)**	(0.04368)	(0.02136)
RSOC	**0.36172**	0.43519	**0.39355**	**0.56028**
	(0.02865)	(0.07616)	**(0.04263)**	**(0.02611)**

Table 4.6 Clustering results of Heart dataset.

	Homogeneity	Completeness	V-Measure	Purity
DE	0.13902	0.13881	0.13890	0.68815
	(0.11303)	(0.11186)	(0.11245)	(0.10174)
PSO	0.17086	0.20987	0.18129	0.70148
	(0.11114)	(0.08492)	(0.11056)	(0.10612)
GA	0.14584	0.15514	0.15021	0.70519
	(0.09743)	(0.10498)	(0.10090)	(0.08145)
MVO	**0.25875**	**0.25761**	**0.25816**	**0.78222**
	(0.06571)	**(0.06283)**	**(0.06432)**	**(0.05627)**
RSOC	0.01881	0.01944	0.01912	0.59074
	(0.00086)	(0.00092)	(0.00089)	(0.00195)

Table 4.7 Clustering results of Seeds dataset.

	Homogeneity	Completeness	V-Measure	Purity
DE	0.55015	0.64305	0.58691	0.77048
	(0.10567)	(0.03752)	(0.06628)	(0.09162)
PSO	0.54263	0.68222	0.59593	0.76095
	(0.11405)	(0.05097)	(0.06504)	(0.11586)
GA	0.54015	0.61663	0.57184	0.76762
	(0.06536)	(0.05254)	(0.03513)	(0.08056)
MVO	0.61098	0.67855	0.63709	0.82810
	(0.09793)	(0.03824)	(0.05412)	(0.10025)
RSOC	**0.69394**	**0.69689**	**0.69541**	**0.89524**
	(0.00793)	**(0.00877)**	**(0.00835)**	**(0.00224)**

4.3.3 Comparing to H-HHO

In this part, RSOC is compared to the hybrid Krill Herd Algorithm (H-KHA) [3], hybrid Harris Hawks Optimization (H-HHO) [1] and eleven other algorithms including: K-means (KM), K-means++ (KM++) [13], Spectral, Agglomerative [23], DBSCAN [32], GA [82], PSO [101], Harmony Search (HS) [7], Krill Herd Algorithm (KHA) [2], Hybrid GA (H-GA), and Hybrid PSO (H-PSO). The algorithms were tested on five widely utilized benchmark datasets in data clustering, namely: Iris, Wine, Cancer, CMC, and Glass. The results were assessed using the error rate index. Low values of these measures are an index of the effectiveness of the clustering algorithm in accurately identifying clusters within different types of datasets.

The results of the algorithms compared with are directly taken from [3, 1]. Table 4.8 presents the algorithms parameters mentioned in [3]. The parameters for RSOC are the same as in the previous comparison. The max number of iterations was set for all algorithms including RSOC to 1000. RSOC results were collected through 15 independent runs. The parameters mentioned in the papers compared with are just of H-KHA.

Table 4.8 The parameters of algorithms.

Algorithm	Parameter	Value
H-KHA	Population	20
	V_f	0.2
	D^{max}	0.002
	N^{min}	0.01
	N^{max}	0.05
	HMCR	0.90
	PARmin	0.45
	PARmax	0.90
	bwmin	0.10
	bwmax	1.00
RSOC	R	[1,5]
	C	[0,2]

Tables 4.9 and 4.10, as well as Figure 4.2, provide a comprehensive comparison of the performance of the aforementioned algorithms including RSOC using the error rate index. The findings provide insights into the performance of the Rat Swarm Optimizer for Clustering (RSOC) algorithm compared to other algorithms.

Table 4.9 present the error rate of the fourteen algorithms applied on Iris, Wine, Cancer, CMC, and Glass. Three criterion are stated in the table, the mean, best and worst error rate results of each algorithm. The error rates are expressed as percentages, representing the proportion of misclustered instances in the datasets. Table 4.10 presents the ranks of different clustering algorithms based on their performance on the same five datasets. The ranks are determined based on the error rates, with lower error rates leading to higher ranks. The sum column provides the total rank for each algorithm across all datasets, allowing for an overall comparison of their performance. Figure 4.2 visually compares the error rates of different algorithms on the five datasets. Each algorithm is represented by a different color or marker.

Based on the information presented in Tables 4.9 and 4.10, as well as Figure 4.2, it is evident that RSOC consistently outperforms other clustering algorithms in terms of error rates across the datasets. It consistently shows the lowest error rates in all datasets except for Cancer, which indicates its effectiveness in clustering the data accurately. It is also observed that in CMC and Glass datasets, there is a gap between RSOC results and the second best algorithm in these datasets (respectively H-KHA and HS). These findings which show the effectiveness of RSOC are due to its ability to search in different regions of the solution space and the smooth balance (which is guaranteed by the parameters $\overrightarrow{A}$ and $\overrightarrow{C}$ discussed in Sec 3.4 and 4.2.3) between intensification and diversification

mechanisms which allow the algorithm to verify the neighborhood of the local optima along with other regions. The ranks in Table 4.10 summarize the algorithms ranks in each dataset and the overall ranking which further reinforce the superior performance of RSOC, as it is ranked first among all algorithms, indicating its overall effectiveness in clustering datasets. Figure 4.2 visually confirms the superior performance of RSOC, as its error rates are consistently lower compared to other algorithms in four out of five datasets.

In summary, the Tables 4.9 and 4.10 and Figure 4.2 provide strong evidence to support the effectiveness of RSOC as a clustering algorithm, demonstrating its ability to produce accurate and reliable clustering results across multiple datasets. However it still has a weakness with the Cancer datasets.

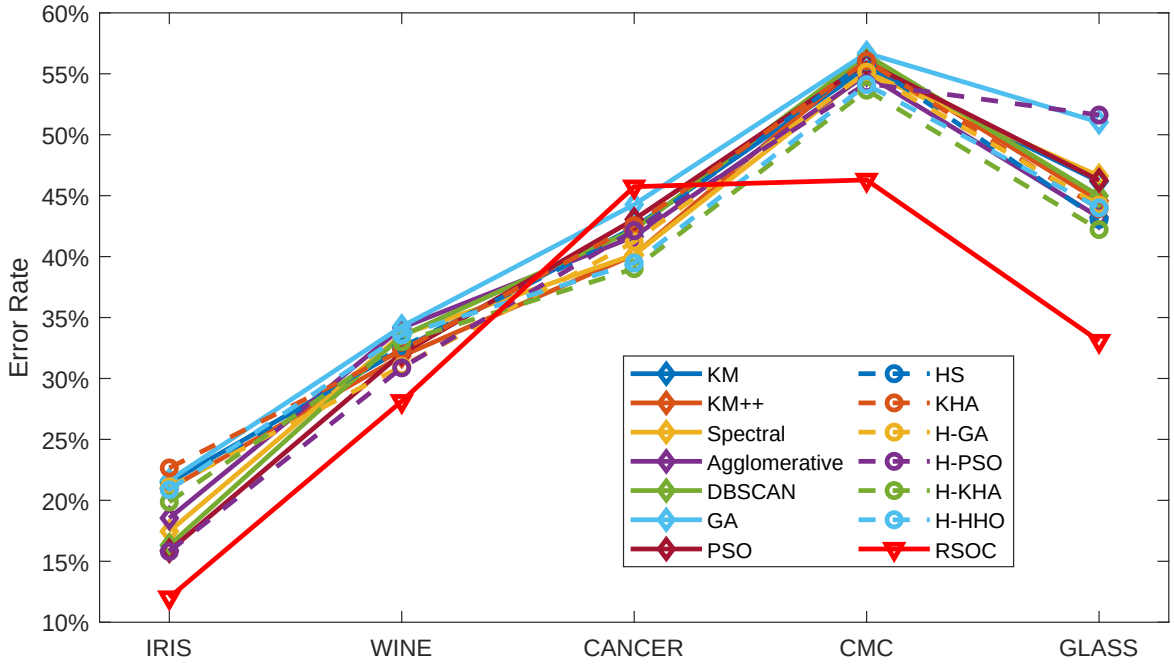


Fig. 4.2 Visual comparison of error-rate results.

Table 4.9 Error Rate results.

	Criterion	Iris	Wine	Cancer	CMC	Glass	Rank
K-means	MEAN	21.467	32.388	42.388	55.470	46.154	
	BEST	10.660	29.775	39.865	54.660	42.262	12
	WORST	56.667	43.820	45.970	56.667	46.215	
KM++	MEAN	20.983	31.841	40.145	56.258	44.566	
	BEST	10.101	30.546	39.500	52.003	45.123	07
	WORST	54.274	43.534	44.965	57.001	45.250	
Spectral	MEAN	17.458	33.585	40.154	55.120	46.614	
	BEST	10.547	29.189	38.111	53.541	38.541	09
	WORST	55.541	43.137	44.685	54.044	51.991	
Agglomerative	MEAN	18.544	34.154	41.645	54.944	43.222	
	BEST	9.874	30.665	39.148	52.391	32.001	06
	WORST	48.397	42.688	46.699	57.487	52.140	
DBSCAN	MEAN	16.311	33.487	42.199	56.544	44.984	
	BEST	9.987	30.140	39.654	54.280	33.717	11
	WORST	43.111	42.009	44.021	56.654	51.123	
GA	MEAN	21.652	34.270	44.270	56.697	51.028	
	BEST	10.666	29.310	39.510	54.656	42.991	14
	WORST	43.333	47.753	47.753	57.296	56.075	
PSO	MEAN	15.867	32.051	43.051	55.899	46.262	
	BEST	10.667	29.775	40.775	54.101	43.925	10
	WORST	43.447	44.449	45.455	56.486	52.804	
HS	MEAN	21.054	32.568	42.054	56.001	43.054	
	BEST	10.509	29.865	40.111	55.430	41.162	08
	WORST	44.286	44.467	45.640	57.906	46.255	
KHA	MEAN	22.658	32.303	42.543	56.056	43.925	
	BEST	9.430	29.213	39.256	53.936	38.318	12
	WORST	42.548	47.191	47.191	56.999	50.476	
H-GA	MEAN	21.100	30.989	41.214	55.142	44.219	
	BEST	9.765	29.654	40.254	53.124	35.249	05
	WORST	44.667	44.001	46.214	56.214	51.985	
H-PSO	MEAN	15.800	30.871	42.125	54.204	51.617	
	BEST	9.666	29.775	39.775	53.201	41.589	04
	WORST	44.333	43.888	46.758	55.333	56.075	
H-KHA	MEAN	19.866	33.000	**39.012**	53.656	42.219	
	BEST	9.000	29.650	38.670	52.213	32.242	02
	WORST	43.333	42.134	44.154	54.333	51.420	
H-HHO	MEAN	20.866	33.564	39.470	54.109	44.002	
	BEST	9.332	29.653	39.119	53.165	34.242	03
	WORST	43.333	43.584	45.365	55.693	51.445	
RSOC	MEAN	**12.027**	**28.134**	45.737	**46.292**	**33.070**	
	BEST	12.027	28.134	45.737	46.208	31.587	**01**
	WORST	12.027	28.134	45.737	46.392	33.970	

Table 4.10 Ranks of algorithms.

	Iris	Wine	Cancer	CMC	Glass	Sum	average rank
k-means	12	7	10	8	10	47	9.4
km++	9	4	3	12	8	36	7.2
Spectral	5	12	4	6	12	39	7.8
Agglomerative	6	13	6	5	4	34	6.8
DBSCAN	4	10	9	13	9	45	9
GA	13	14	13	14	13	67	13.4
PSO	3	5	12	9	11	40	8
HS	10	8	7	10	2	37	7.4
KHA	14	6	11	11	5	47	9.4
H-GA	11	3	5	7	6	32	6.4
H-PSO	2	2	8	4	14	30	6
H-KHA	7	9	1	2	3	22	4.4
H-HHO	8	11	2	3	5	29	5.8
RSOC	1	1	14	1	1	18	3.6

4.4 Conclusion

In conclusion, Chapter 4 introduced our method, the rat swarm optimizer for data clustering (RSOC), its main body, its algorithm, and its time complexity, along with a comprehensive comparison with well-established algorithms-based data clustering. The experimental results unequivocally demonstrated the superior ability of RSOC to yield overall better solutions compared to the algorithms it was pitted against. Overall, the findings suggest that RSOC is a promising algorithm for data clustering. However, it is important to note that RSOC may not be suitable for all types of datasets, which calls for the no free lunch theorem, as evidenced by its unexpected performance on the Heart and Cancer datasets. Further research and improvements are needed to enhance the performance of RSOC such as including in a phase of MVO or H-KHA to overcome this weakness. It is also suggested to explore its applicability to other optimization and engineering problems.

The next chapter will present our second contribution, which is the enhanced grey wolf algorithm for data clustering, along with the original version, grey wolf algorithm-based data clustering, includ-

ing a comparison between the two versions and other algorithms, along with the experimental results and discussion.

Chapter 5

Enhanced Grey Wolf Optimizer for Data Clustering (Second contribution)

5.1 Introduction

The chapter presents an in-depth analysis and comparison of the Enhanced Grey Wolf Optimizer for Data Clustering EGWAC algorithm (our second contribution) with other well-established clustering algorithms. The focus of this chapter is to exhibit the performance of EGWAC across multiple datasets and compare its effectiveness in producing high-quality clustering results.

The chapter begins by providing a detailed overview of our method, the enhanced grey wolf algorithm for data clustering, highlighting its key features, enhancements, and the underlying principles that differentiate it from the original version. This introduction sets the stage for a comprehensive exploration of EGWAC's performance in comparison to other algorithms.

Furthermore, the chapter outlines the benchmark datasets used for the evaluation, including their characteristics and relevance to real-world clustering applications. This ensures a thorough and meaningful comparison of EGWAC's performance across diverse types of data.

Overall, this chapter serves as a comprehensive evaluation of EGWAC's performance, offering valuable insights into its strengths and capabilities as a state-of-the-art clustering algorithm.

5.2 Enhanced Grey wolf algorithm-based data clustering

The enhanced grey wolf algorithm-based data clustering (EGWAC) is an optimized version of GWAC, where some points in the latter are optimized. It has the same main body as GWAC, wolves population represents the set of solutions. Each wolf is represented by the cluster centers.

5.2.1 Population initialization

In EGWAC, the population of solutions is initialized by the same technique in the GWAC, each wolf is initialized by K random points from the dataset.

5.2.2 Main body of EGWAC

The first part of the algorithm starts by initialize the parameters and the population position, cluster the dataset by each wolf, calculate the fitness value (SSE), and then, update the position of the three best solutions $\overrightarrow{X}_{\alpha}$, $\overrightarrow{X}_{\beta}$ and $\overrightarrow{X}_{\delta}$.

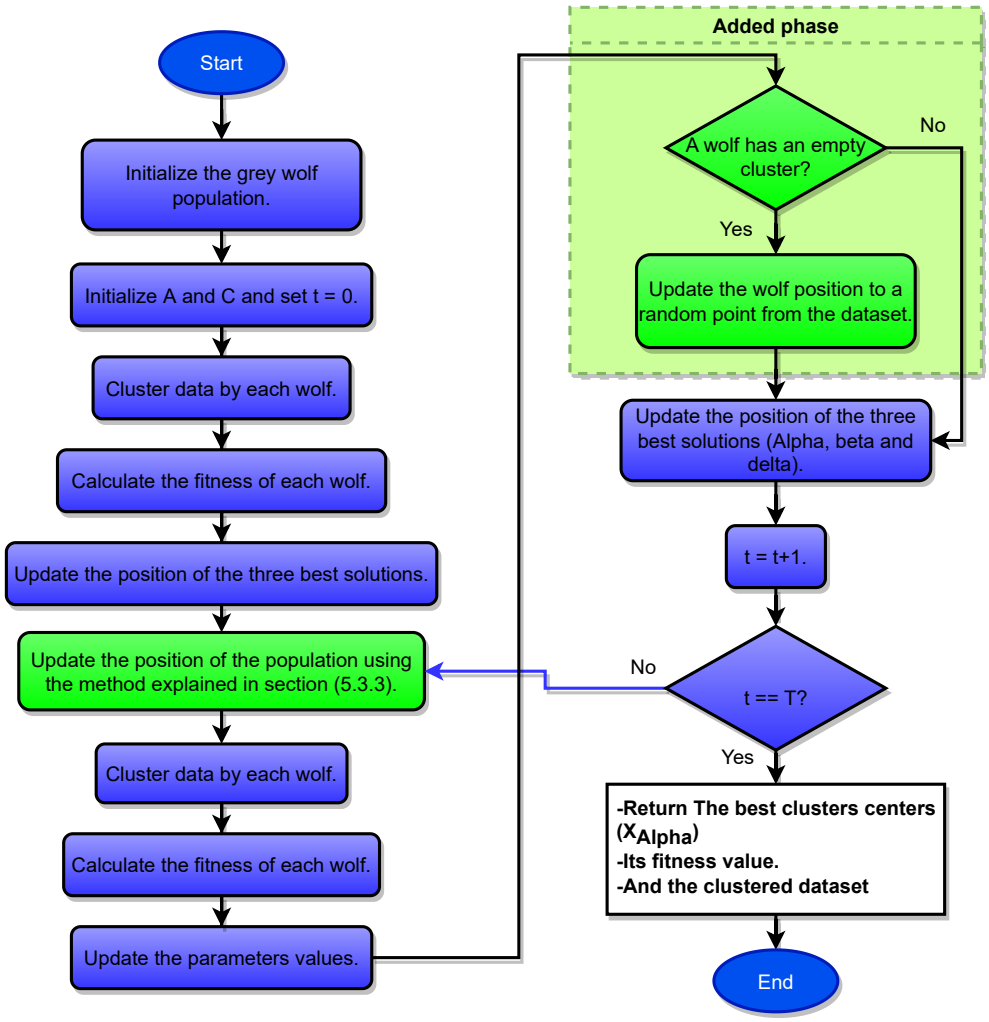


Fig. 5.1 EGWAC flowchart

The second part starts by update wolves position, cluster the dataset, and calculate the fitness value of each one. The position of a wolf is updated randomly if it produces any empty cluster. The

parameters $\overrightarrow{A}$, $\overrightarrow{C}$ and $\overrightarrow{X}_\alpha$, $\overrightarrow{X}_\beta$ and $\overrightarrow{X}_\delta$ positions are updated. This process is repeated a defined number of times.

At the end of the algorithm, it returns the best cluster centers $\overrightarrow{X}_\alpha$, and its clustered dataset. Algorithm (11) depicts the EGWAC.

Algorithm 11 EGWAC [130]

Require: number of clusters k, population size, max number of iterations T and the dataset.
Ensure: the best solution $\overrightarrow{X}_\alpha$ and the clustered dataset.
Initialize the grey wolf population $\overrightarrow{X}_i (i = 1, ..., n)$.
Initialize $\overrightarrow{A}$ and $\overrightarrow{C}$ and set $t = 0$.
Cluster data by each wolf.
Calculate the fitness of each wolf (using Eq.2.24) and update the position of $\overrightarrow{X}_\alpha, \overrightarrow{X}_\beta$ and $\overrightarrow{X}_\delta$.
while $(t < T)$ **do**
  for each wolf **do**
    Update the position of the current wolf by the method explained in section (5.2.3).
    Cluster data by the current wolf.
    Update $\overrightarrow{A}$ and $\overrightarrow{C}$ by Eqs.(3.9, 3.10).
  end for
  Update $\overrightarrow{d}$ by Eq.(3.11).
  Calculate the fitness of all solutions using Eq.2.24.
  if a wolf has an empty cluster **then**
    Update the wolf position to a random point from the dataset.
  end if
  Update $\overrightarrow{X}_\alpha, \overrightarrow{X}_\beta, \overrightarrow{X}_\delta$.
  $t = t + 1$.
end while
Return: $\overrightarrow{X}_\alpha$ and the clustered the dataset.

5.2.3 Position updation

As it was explained in section (3.6.4), GWAC uses the obvious technique to calculate the mean of the three positions $\overrightarrow{X}_1$, $\overrightarrow{X}_2$ and $\overrightarrow{X}_3$. It summates the first component of the first center of the three positions, and divides the result by three to get the first component of the center number one of the new position, and so on for the other components. The same calculation is applied for the rest of centers to get the new position. This technique can lead to an arbitrary position, whereas, the intended result is the mean of the three positions. This mislead appears because the position of cluster centers is meaningless, if there are two wolves $W_1 = (\mu_1, \mu_2, \mu_3)$ and $W_2(\mu_3, \mu_1, \mu_2)$ with the same centers but in different order, the clustering results using the first one will be the same as using the second, actually any combination of the same centers will do the same as it is illustrated in figure (5.2).

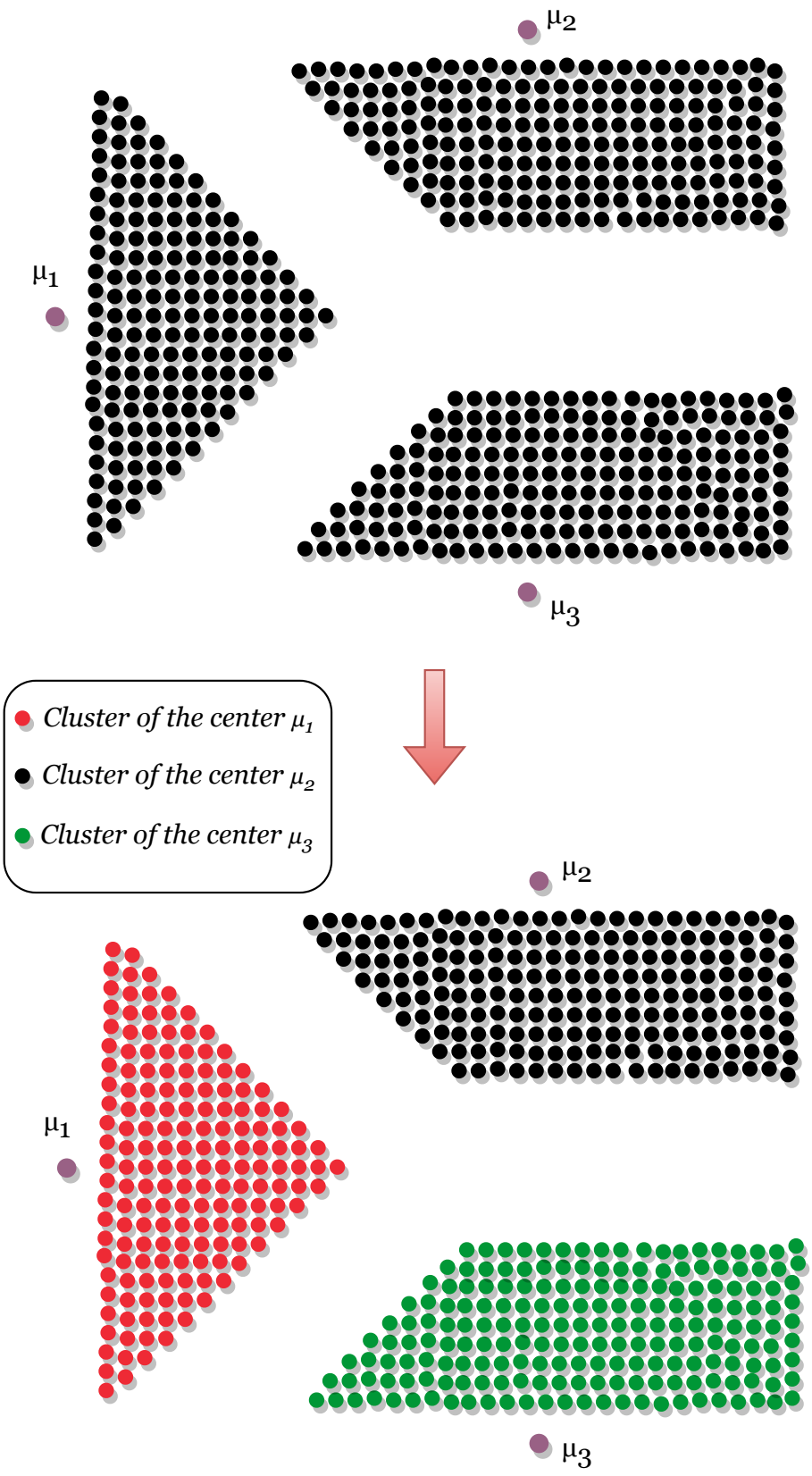


Fig. 5.2 Clustering results using W_1 and W_2.

This is because in the clustering process each data point will be assigned to the cluster with the nearest center, in other words the algorithm calculates the distance between the data point and the centers to find the closest one. Thus, if a data point p is closer to the center μ_3, it will be assigned to the cluster with the center μ_3 for both W_1 and W_2, which is the third center in W_1 and the first center regarding W_2.
From the aforementioned reason, the two wolves are identical which means the mean of them should be themselves. However, the outcomes of calculating it with the GWAC technique will be quite different, it will give: $W_m = (\frac{\mu_1+\mu_3}{2}, \frac{\mu_2+\mu_1}{2}, \frac{\mu_3+\mu_2}{2})$ istead of $(\frac{\mu_1+\mu_1}{2}, \frac{\mu_2+\mu_2}{2}, \frac{\mu_3+\mu_3}{2})$
From this point, the GWAC calculation of the next position, which is supposed to be a point at the center of the three positions $\overrightarrow{X}_1$, $\overrightarrow{X}_2$ and $\overrightarrow{X}_3$ (the mean), could lead to other positions far away from the intended one. To explain it more, let $\overrightarrow{X}_1 = ((1,2),(40,50),(10,11))$, $\overrightarrow{X}_2 = ((40,50),(10,11),(1,2))$ and $\overrightarrow{X}_3 = ((10,11),(1,2),(40,50))$. The next position will be as:

$$\overrightarrow{X}(t+1) = \left(\frac{(1+40+10,2+50+11)}{3}, \frac{(40+10+1,50+11+2)}{3}, \frac{(10+1+40,11+2+50)}{3}\right)$$

$$\overrightarrow{X}(t+1) = ((17,21),(17,21),(17,21))$$

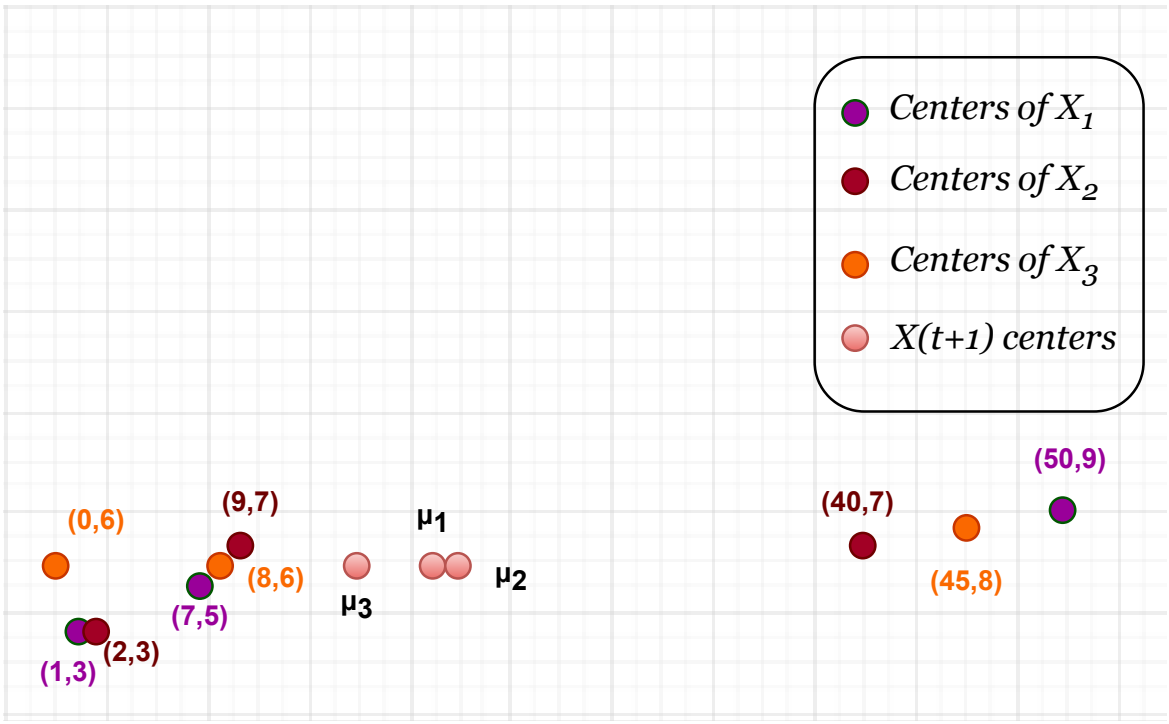
However, a mean of the three points should consist of any combination the three centers $(1,2)$, $(40,50)$ and $(10,11)$. The same thing will happen if $\overrightarrow{X}_1$, $\overrightarrow{X}_2$ and $\overrightarrow{X}_3$ are different. Let $\overrightarrow{X}_1 = ((1,3),(50,9),(7,5))$, $\overrightarrow{X}2 = ((9,7),(2,3),(40,7))$ and $\overrightarrow{X}3 = ((45,8),(8,6),(0,6))$. The next position using GWAC will be $\overrightarrow{X}(t+1) = ((18.33,6),(20,6),(15.66,6))$, which is clearly far away from the three positions.

The main idea proposed in EGWAC is to calculate the distance between centers of the three positions in order to know the centers that present the same or the similar cluster from each position. Taking the last example, it is clear that the closest centers from $\overrightarrow{X}_2$ and $\overrightarrow{X}_3$ to the first center from $\overrightarrow{X}_1$ are respectively the second $(2,3)$ and the last $(0,6)$. $\overrightarrow{X}_2$'s last $(40,7)$ and $\overrightarrow{X}_3$'s first $(45,8)$ one for $\overrightarrow{X}_1$'s second center and the $\overrightarrow{X}_2$'s first $(9,7)$ and $\overrightarrow{X}_3$'s second $(8,6)$ ones regarding $\overrightarrow{X}_1$'s third center. The next position using EGWAC will be calculated as:

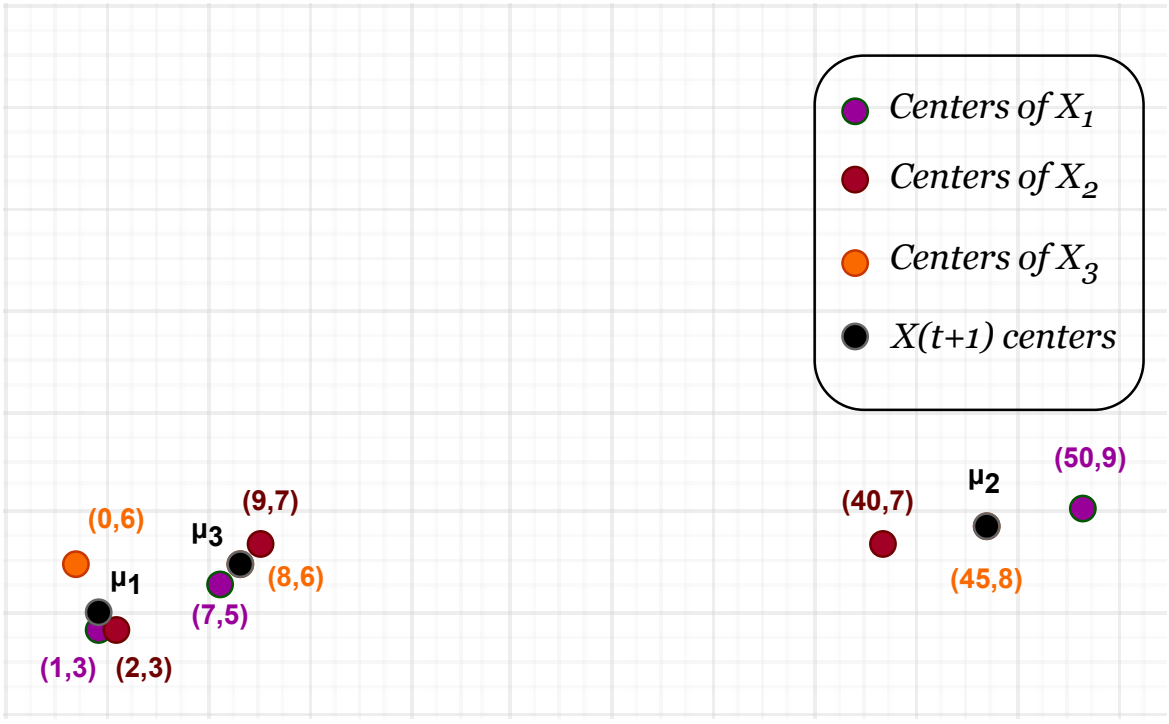
$$\overrightarrow{X}(t+1) = \left(\frac{(1+2+0,3+3+6)}{3}, \frac{(50+40+45,9+7+8)}{3}, \frac{(7+9+8,5+7+6)}{3}\right)$$

$$\overrightarrow{X}(t+1) = ((1,4),(45,8),(8,6))$$

which is clearly more closer to the three points. Fig. 5.3 illustrates the next position calculated by GWAC and EGWAC.



(a) GWAC next position.



(b) EGWAC next position.

Fig. 5.3 Next position using GWAC and EGWAC

To illustrate the effect of this issue to the clustering results, let the dataset in Fig. 5.4.

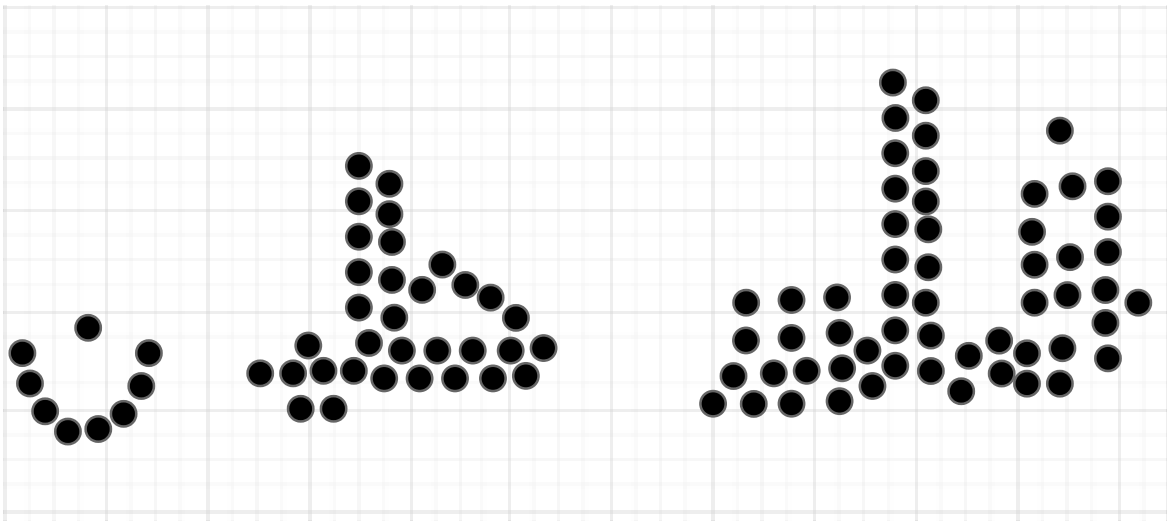


Fig. 5.4 Next position using GWAC and EGWAC

The clustering using EGWAC method will give similar results to the three positions X_1, X_2 and X_3 as it is illustrated in Fig. 5.6. However, the clustering results using GWAC method messed up the results and did not follow the lead of the three positions which is the principle mechanism in GWO metaheuristic. Furthermore, GWAC method results in splitting a clearly compact cluster which is represented by the green color in EGWAC clustering into three parts, and merging two separated clusters the red cluster with some points of the green one. GWAC also merged two other separated clusters that are represented in the white cluster and some points of the green one as it is illustrated in Fig. 5.5.

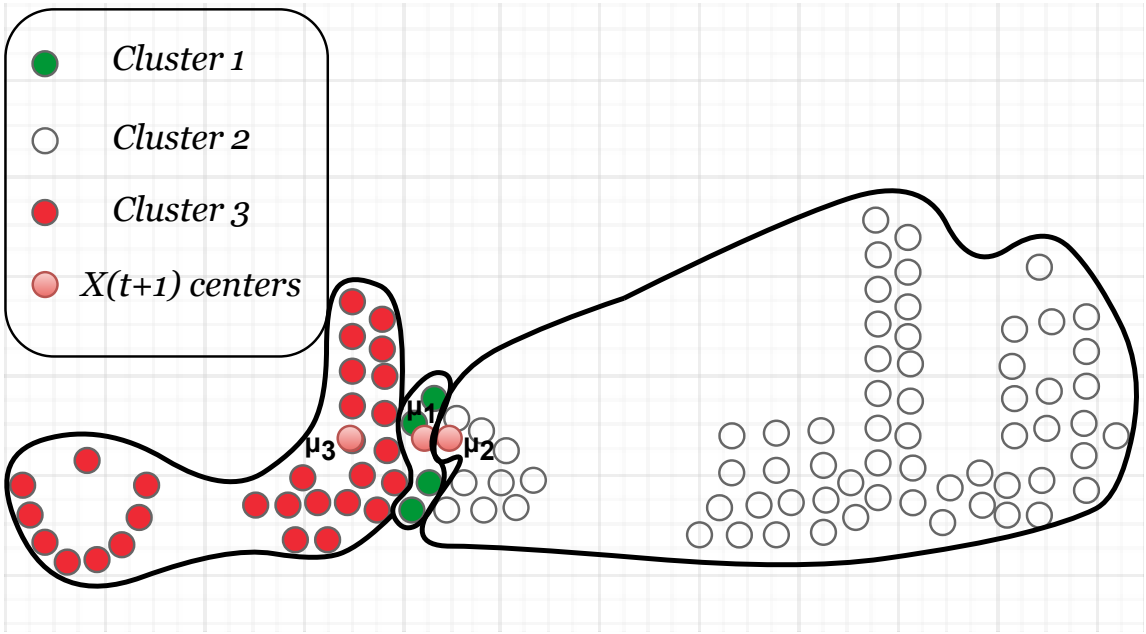
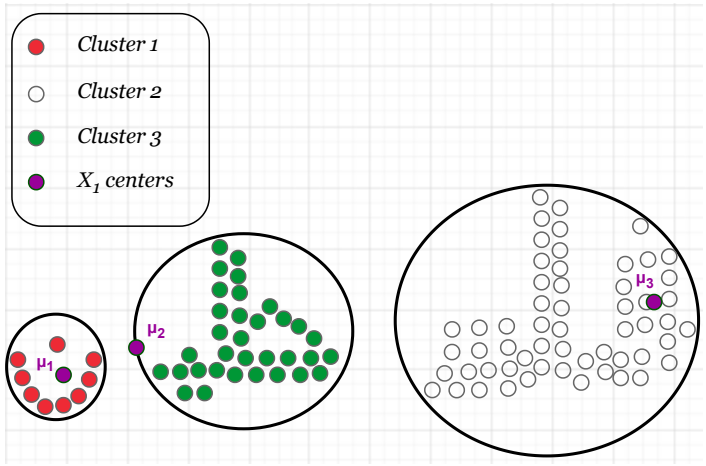
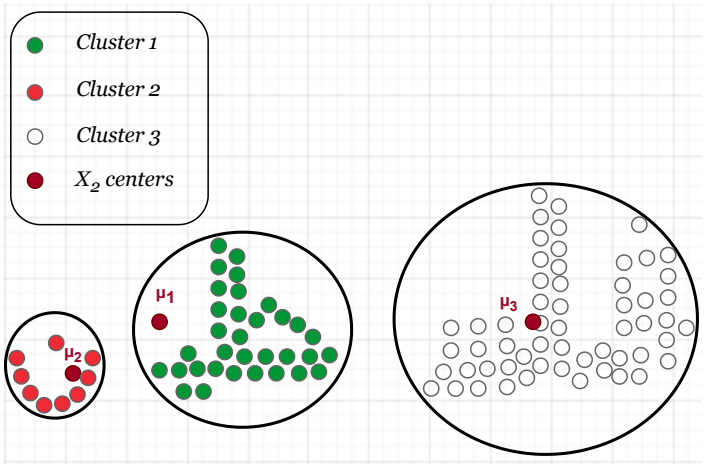


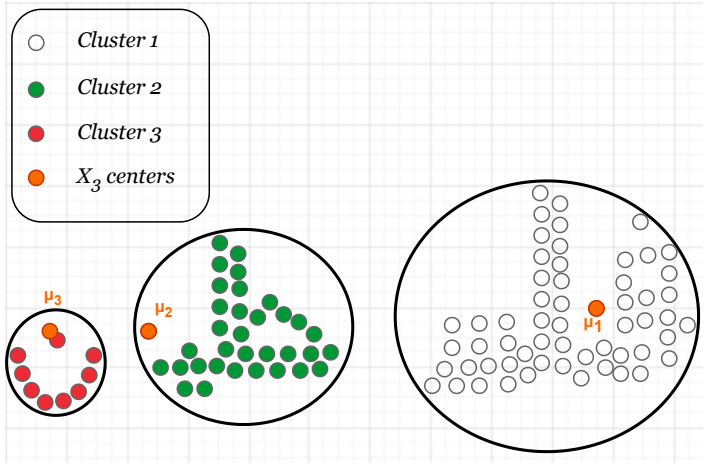
Fig. 5.5 Clustering using GWAC's new position



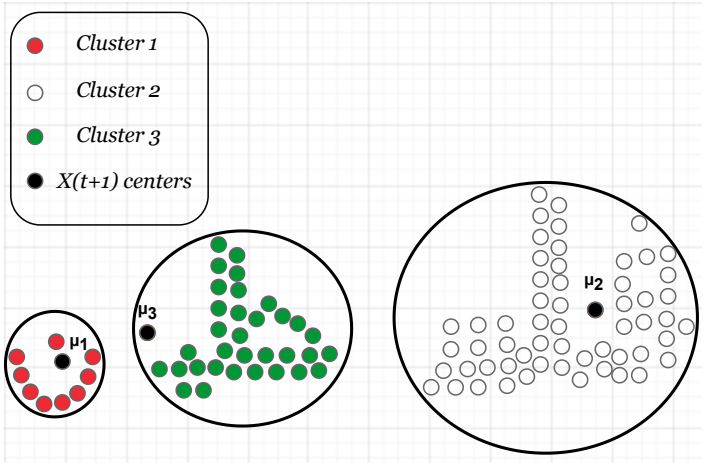
(a) Clustering using X_1 position



(b) Clustering using X_2 position



(c) Clustering using X_3 position



(d) Clustering using EGWAC's new position

Fig. 5.6 Clustering using the three positions and EGWAC

5.2.4 Handling empty clusters

Clustering algorithms often come with empty clusters, this may happen when there is a center that is far away from the data points (i.e. other centers are closer to the data points than this center). In this case, as the data points will be assigned to the closest center, the cluster of farthest one will be empty. Same thing, if there are identical centers in the same wolf, then one of the clusters will be empty. The two centers are identical means they will have the same distance between them and any other data point. Thus, when comparing the two values (distances between a data point and the two centers) the data point will go either in the cluster with the first center or the one with the second center, regardless of which cluster the data points will be in, they will be in one cluster, which means the other one will be empty. To resolve this problem, EGWAC includes a phase where any wolf resulting empty clusters its position will be updated to a K random points from the dataset.

5.2.5 Time complexity

The time complexity calculation of the EGWAC will either be divided into three main parts to make it clear and easy to be understood. The first part constitutes the instructions before the while loop, and part two is the loop **while**. The third part represent the instruction after the loop.

The following items explain the time complexity of each instruction in the algorithm.

1. **Population initialization:** as it was explained with RSOC (Section 4.2.4, the main instruction is the assigning of K random data points to each wolf. Each data point is represented by a d features, which results in $K * d$ operations to assign the K data points to a wolf (initialization of a wolf). Thus, the time complexity to initialize a population of n individual is $O(n * K * d)$.

2. **Parameter initialization:** this process is a simple one, without explicit neither implicit loops or recursive process, which results in constant time complexity $O(1)$.

3. **Cluster data by each wolf:** the process of clustering a dataset of s data points consists of three operations, (i) calculation of the distance between each data point and the K centers, (ii) verification of the shortest distance among the K calculated ones for each data point (the calculated distances between the data point and the K centers), and (iii) the assignement of labels to each data point. these operations are detailed as follows:

    - Calculating the distance between each data point and the K centers has $O(s * K * d)$ time complexity. As each data point has d features, calculating the distance between K centers and a dataset will result in $O(K * d)$. For the whole dataset, the result will be $O(s * K * d)$.

    - Verifying the shortest distance among the K calculated ones for each data point (the previous calculation) has also $O(s * K)$ as comparing K resulted distances (simple values) is $O(K)$, and with this process repeated for all data points it will give $O(s * K)$.

- The cluster label assignment for one data point has $O(1)$ time complexity as it is a simple instruction. Thus, for the whole dataset it will be $O(s)$.

The time complexity of the third process (Cluster data by each wolf) will be the greatest among the previous detailed ones, which is $O(s*K*d)$ for the clustering process using a wolf. As we have n wolves this phase will result in $O(n*s*K*d)$.

4. **Fitness calculation:** consists of calculating the distance between the cluster centers and their corresponding members which will result in $O(s*d)$. As this process is repeated by each wolf it will result in $O(n*s*d)$ time complexity.

5. **Updating the three best solutions $\overrightarrow{X}_\alpha$, $\overrightarrow{X}_\beta$, and $\overrightarrow{X}_\delta$:** this operation consists of assigning Three solutions ($3*K*d$ values) to $\overrightarrow{X}_\alpha$, $\overrightarrow{X}_\beta$, and $\overrightarrow{X}_\delta$ which has $O(K*d)$ time complexity.

6. **Updating wolves position:** it has $O(n*K*d)$ for the whole population of wolves as the updation of a solution has $O(K*d)$. To make this clear the updation of a wolf position is explained as follows:

   - **The calculation of the three vectors $\overrightarrow{D}_\alpha$, $\overrightarrow{D}_\beta$, and $\overrightarrow{D}_\delta$:** this process result in three operations to the $K*d$ features, first, the multiplication between $\overrightarrow{C}_i$ and $\overrightarrow{X}_j$, then, the subtraction of the $\overrightarrow{X}(t)$ from the previous calculated value, and finally, the assigning of the result to $\overrightarrow{D}_i$. This, results in $3*3*K*d$ whcich gives $O(K*D)$.
   - **The calculation of $\overrightarrow{X}_1$, $\overrightarrow{X}_2$, and $\overrightarrow{X}_3$:** has the same operations as the previous point which means the same time complexity $O(K*d)$.
   - **Searching the closest centers to calculate the $\overrightarrow{X}(t+1)$:** this process consists of calculating the distance between each center from $\overrightarrow{X}_\alpha$ and centers from $\overrightarrow{X}_\beta$ and $\overrightarrow{X}_\delta$ and compare the results to know the closest ones. It has $O(K*d)$.
   - **The calculation of the mean ($\overrightarrow{X}(t+1)$:** has $3*K*d+K*d$ operations, i.e., $O(K*d)$.

   As the four processes are in sequential order, the time complexity to update a wolf position will be the greater value among them ($O(K*d)$). For the whole population of n wolves, it will be $O(n*K*d)$.

7. **Updating parameters:** this process has n times the complexity of the parameters initialization as they will be recalculated for each wolf which gives $O(n)$.

8. **Rturning back solutions resulting empty clusters:** at the worst case all wolves will outcome an empty cluster. In this case it will be like the population initialization ($O(n*K*d)$).

9. **Increment the value of t by one:** has $O(1)$.

10. **Returning back the best solution and the clusters:** this process has an $O(s*d)$ as it consists of two sequential processes which are returning the best solution ($O(K*d)$) and the clusters ($O(s*d)$), and it is obvious that the grater one between the two is ($O(s*d)$) as the number of clusters K should be less than the number of data points s.

To sum up, the time complexity of the EGWAC can be simplified in three points, first part before the while loop processes, second is the while loop and third processes after the loop:

1. **Processes before the while loop:** instructions before the while loop are sequential processes, this means we will take the bigger time complexity among them to represent this part, which is $O(n*s*K*d)$.

2. **The while loop:** inside the while loop we have the instructions of updating wolves position, clusteing the dataset, updating parameters, calculating the wolves fitness, updating solutions with empty clusters, and updating the position fo the three best solutions where their time complexity was calculated previously. As they are in a sequential order, we take the bigger time complexity, which is $O(n*s*K*d)$. And as these processes are repeated inside the while loop T times, the total time complexity of the while loop is $O(T*n*s*K*d)$.

3. **After the while loop:** returning back the best solution and the clusters has $O(s*d)$ time complexity.

Accordingly, the total time complexity of the EGWAC is $O(T*n*s*K*d)$.

5.3 Experimental results and discussion

We will present in this section the experimental results of EGWAC compared to other algorithms along with the interpretation and discussion of the results. This section is divided into two parts, the first one exposes and discusses the comparison of the EGAWC to the GWAC and other algorithms. The second part present the comparison of the EGWAC to the hybrid grey wolf optimizer and tabu search (GWOTS). This part is attached by the interpretation and discussion of the comparison findings.

The utilized benchmark datasets for the comparison can be found in the UCI Machine Learning Repository [28]. Table 5.1 details the datasets characteristics.

Table 5.1 Utilized datasets.

Datasets	Number of instances	Number of features	Number of classes
Iris	150	4	3
Blood	748	4	2
Glass	214	9	6
Seeds	210	7	3
Wine	178	13	3
Haberman	306	3	2
Heart	270	13	2
Liver (Bupa)	345	6	2
Planning relax	182	12	2
CMC	1473	9	3

5.3.1 Benchmark datasets description

The "class" attribute is removed from datasets before the clustering process. Some datasets are already described in the previous section 4.3.1. The following is a description of the rest datasets.

- The **Blood Transfusion Service Center** benchmark is a dataset that pertains to blood donations and transfusions, it encompasses 748 instances (donors). The samples were selected randomly from a donor database gathered in Hsin-Chu City in Taiwan. Each sample is characterized by five features including the class label, namely, R (Recency - months since last donation), F (Frequency - total number of donation), M (Monetary - total blood donated in c.c.), T (Time - months since first donation), and the last feature represent the blood donation in March 2007 (1 for donating; 0 for not donating), this feature represent the classes [129].

- The **Haberman's Survival** is a dataset on the surviving of patients who underwent surgery for breast cancer. This study was performed at the University of Chicago's Billings Hospital between 1958 and 1970. Each sample is characterized by four features including: age at surgery, year of surgery, number of positive axillary nodes, and survival status (1 = the patient survived for at least 5 years after surgery, 2 = the patient died within 5 years of surgery) which represents the class label [49].

- The **Liver (Bupa)** is a benchmark dataset for machine learning tasks such as data clustering. This benchmark was created by the BUPA Medical Research Ltd. It consists of 345 samples, each of which is characterized by six features: mean corpuscular volume (mcv), alkphos alkaline phosphotase, alamine aminotransferase (sgpt), aspartate aminotransferase (sgot),

gamma-glutamyl transpeptidase (gammagt), number of half-pint equivalents of alcoholic beverages drunk per day (drinks), and a seventh variable, which is considered as the class label [39].

- The **Planning relax** benchmark is a collection of ECG signals designed to classify two mental states: planning and relaxation. Rajen Bhatt created this dataset which contains 182 samples (ECG signals) of five seconds long, each of which is characterized by 13 features (including the class label) such as: EEG record, Mu Rhythm, Event, elated Desynchronization (ERD), Electrodes, EOG (Electrooculogram), and Experiment [18].

The EGWAC results for both comparisons were carried out in a laptop with Intel(R) Core(TM) i7-7500U CPU @ 2.70GHz 2.90 GHz processor, DDR4 4.0 GB RAM using Matlab R2021a under windows 10 home 64-bit operating system.

5.3.2 Comparing with GWAC

In this part, the EGWAC is compared to its original version (GWAC) and other algorithms in order to reveal the effect of the improvements in EGWAC. The algorithms compared here are: K-means (KM) , GA-based clustering (GAC), harmony search-based clustering (HSC), modified HSC (MHSC), PSO-based clustering (PSOC), flower pollination algorithm-based clustering (FPAC), bat algorithm-based clustering (BATC), GWAC, and EGWAC. The algorithms here are tested on six real datasets notably: Iris, Wine, Glass, Haberman, Bupa, and CMC benchmarks. The results were assessed using three measures including: precision, recall, and g-measure.

The results of the algorithms compared with are directly taken from [72]. All population-based algorithms have a population size of 30 and a maximum number of iterations of 500.

Table 5.2 The parameters of algorithms.

Algorithm	Parameter	Value
GAC	Crossover	0.8
	Mutation	0.01
HSC	Harmony memory consideration rate (HMCR)	0.9
	Pitch adjustment rate (PAR)	0.5
	Bandwidth	0.01
MHSC	HMCR	[0.5, 0.95]
	PAR	[0.01, 0.99]
	Bandwidth	[0.001, 0.1]
PSOC	Inertia weight	0.72
	Acceleration constants	1.49
	Maximum velocity	255
FPAC	Switch probability	0.8
BATC	Loudness	[1, 2]
	Pulse emission rate	[0, 1]
	α and γ	0.9

The results were collected through more than 20 independent runs.

Table 5.3 provides a comprehensive comparison of the aforementioned clustering algorithms. The comparison is based on three key performance measures such as precision, recall, and G-measure across multiple datasets. High values of these measures represent an index of the effectiveness of the clustering algorithm in accurately identifying clusters within different types of datasets.

The table presents the mean values, standard deviations (std), and ranks with the three measures for each algorithm and dataset. This allows for a detailed analysis of how each algorithm performs across different datasets and provides a view into their consistency and variability. By comparing the mean performance, std, and ranking of each algorithm on different datasets. The best values are highlighted in bold.

Figures 5.7 to 5.12, represent the boxplot of the EGWAC to better visualize the distribution of clustering performance achieved by it on different datasets. Each boxplot illustrates the variability and central tendency of the algorithm's performance across multiple runs on a specific dataset. The boxplots provide a visual representation of the algorithm's stability, reliability, and consistency in performance across different runs on each dataset.

It is observed from the table and figures that EGWAC showed the best results across all datasets in all measures outperforming the algorihtms compared with except for Bupa dataset, where the

EGWAC unexpectedly gave the second best result in precision after FPAC. The enhanced version (EGWAC) outperformed the original one (GWAC) in the whole comparison. Regarding Iris, Wine, Glass, EGWAC results were a way better than the others. In the Glass dataset we can observe the significant gap between EGWAC and the other algorithms. The narrow spread and elevated median values in the boxplots signify consistent and resilient performance of the EGWAC. These findings indicate that EGWAC holds promise as an algorithm for data clustering, demonstrating robust and dependable performance across a range of evaluation measures and datasets which is due to the enhancement applied to the GWAC where mainly the position updation was optimized.

To summarize, the results presented in Table 5.3 and Figures 5.13 to 5.15 highlight the effectiveness of EGWAC as a clustering algorithm and its potential to deliver consistent and reliable clustering results across diverse datasets.

Table 5.3 Clustering results in different dataset

	Measure	KM	GAC	HSC	MHSC	PSOC	FPAC	BATC	GWAC	EGWAC
Iris	Precision	0.3018	0.3534	0.2428	0.4586	0.4299	0.4266	0.4396	0.5688	**0.8046**
	Std	(0.2584)	(0.3193)	(0.2782)	(0.3178)	(0.3482)	(0.1936)	(0.4159)	(0.2323)	**(0.0020)**
	Rank	7	8	9	3	5	6	4	2	**1**
	Recall	0.3020	0.3733	0.2393	0.4427	0.4460	0.4385	0.4360	0.5813	**0.8357**
	Std	(0.2569)	(0.3011)	(0.2669)	(0.3126)	(0.3316)	(0.1870)	(0.4090)	(0.2179)	**(0.0035)**
	Rank	8	7	9	4	3	5	6	2	**1**
	G-Measure	0.1144	0.2046	0.1102	0.4082	0.4364	0.4319	0.4359	0.5682	**0.8200**
	Std	(0.2828)	(0.3735)	(0.2870)	(0.3454)	(0.3385)	(0.1893)	(0.4107)	(0.2208)	**(0.0028)**
	Rank	8	7	9	6	3	5	4	2	**1**
Wine	Precision	0.2718	0.2925	0.3569	0.4343	0.3851	0.3629	0.3872	0.4994	**0.5840**
	Std	(0.1365)	(0.2191)	(0.2554)	(0.2108)	(0.2002)	(0.1286)	(0.2381)	(0.1808)	**(0.0006)**
	Rank	9	8	7	3	5	6	4	2	**1**
	Recall	0.2742	0.3105	0.3579	0.4087	0.3757	0.3900	0.3773	0.4778	**0.5833**
	Std	(0.1409)	(0.2323)	(0.2312)	(0.1897)	(0.1805)	(0.1045)	(0.2233)	(0.1694)	**(0.0005)**
	Rank	9	8	7	3	6	4	5	2	**1**
	G-Measure	0.0777	0.1082	0.2110	0.2519	0.3795	0.4074	0.3807	0.4878	**0.5837**
	Std	(0.2457)	(0.2445)	(0.3153)	(0.2940)	(0.1913)	(0.1290)	(0.2296)	(0.1679)	**(0.0005)**
	Rank	9	8	7	6	5	3	4	2	**1**
Glass	Precision	0.1573	0.1086	0.1976	0.2411	0.1441	0.1577	0.2043	0.2921	**0.4152**
	Std	(0.1573)	(0.0466)	(0.1035)	(0.1266)	(0.0930)	(0.0755)	(0.1321)	(0.0937)	**(0.0090)**
	Rank	7	9	5	3	8	6	4	2	**1**
	Recall	0.1363	0.1158	0.1889	0.2510	0.1629	0.1538	0.2016	0.2621	**0.6167**
	Std	(0.1083)	(0.0661)	(0.0741)	(0.1176)	(0.1005)	(0.0645)	(0.1252)	(0.0818)	**(0.0420)**
	Rank	8	9	5	3	6	7	4	2	**1**
	G-Measure	0.1413	0.1624	0.1319	0.1927	0.1408	0.1409	0.1919	0.2511	**0.5056**
	Std	(0.0736)	(0.0959)	(0.0987)	(0.1044)	(0.0902)	(0.0596)	(0.1201)	(0.0779)	**(0.0126)**
	Rank	6	5	9	3	8	7	4	2	**1**

Table 5.3 Continued...

	Measure	KM	GAC	HSC	MHSC	PSOC	FPAC	BATC	GWAC	EGWAC
Haberman	Precision	0.4972	0.4995	0.4955	0.4996	0.4981	0.4983	0.4941	0.5042	**0.6483**
	Std	(0.1146)	(0.0181)	(0.0226)	(0.0157)	(0.0144)	(0.0158)	(0.0106)	(0.0121)	**(0.0000)**
	Rank	7	4	8	3	6	5	9	2	**1**
	Recall	0.4964	0.4994	0.4953	0.4994	0.4977	0.4978	0.4925	0.5054	**0.5006**
	Std	(0.0179)	(0.0229)	(0.0279)	(0.0189)	(0.0184)	(0.0206)	(0.0134)	(0.0154)	**(0.0000)**
	Rank	7	3	8	3	6	5	9	2	**1**
	G-Measure	0.4556	0.4647	0.4596	0.4602	0.4756	0.4764	0.4703	0.4768	**0.5697**
	Std	(0.0161)	(0.0226)	(0.0340)	(0.0209)	(0.0105)	(0.0241)	(0.0090)	(0.0148)	**(0.0000)**
	Rank	9	6	8	7	4	3	5	2	**1**
Bupa	Precision	0.5000	0.4787	0.5078	0.5119	0.5138	**0.5411**	0.5243	0.4944	0.5097
	Std	(0.0132)	(0.0463)	(0.0490)	(0.0465)	(0.0421)	**(0.0672)**	(0.0643)	(0.0256)	(0.0000)
	Rank	7	9	6	4	3	**1**	2	8	5
	Recall	0.5000	0.4892	0.5015	0.5054	0.5078	0.5253	0.5242	0.4978	**0.8054**
	Std	(0.0082)	(0.0271)	(0.0266)	(0.0195)	(0.0221)	(0.0412)	(0.0570)	(0.0102)	**(0.0000)**
	Rank	7	9	6	5	4	2	3	8	**1**
	G-Measure	0.4302	0.3839	0.4108	0.4324	0.4684	0.5090	0.5032	0.4100	**0.6407**
	Std	(0.0119)	(0.0514)	(0.0362)	(0.0491)	(0.0124)	(0.0018)	(0.0601)	(0.0191)	**(0.0000)**
	Rank	6	9	7	5	4	2	3	8	**1**
CMC	Precision	0.3432	0.3378	0.3559	0.3615	0.3502	0.3090	0.3166	0.3620	**0.4477**
	Std	(0.0417)	(0.0599)	(0.0136)	(0.0185)	(0.0589)	(0.0806)	(0.0863)	(0.0417)	**(0.0010)**
	Rank	6	7	4	3	5	9	8	2	**1**
	Recall	0.3320	0.3372	0.3520	0.3569	0.3472	0.3124	0.3186	0.3614	**0.3640**
	Std	(0.0444)	(0.0570)	(0.0144)	(0.0184)	(0.0514)	(0.0748)	(0.0827)	(0.0347)	**(0.0023)**
	Rank	7	6	4	3	5	9	8	2	**1**
	G-Measure	0.3139	0.3196	0.3352	0.3459	0.3440	0.3061	0.3134	0.3538	**0.4037**
	Std	(0.0569)	(0.0588)	(0.0218)	(0.0223)	(0.0535)	(0.0759)	(0.0842)	(0.0345)	**(0.0011)**
	Rank	7	6	5	3	4	9	8	2	**1**
Overall rank	(Sum)	8 (134)	7 (128)	6 (123)	3 (70)	4 (90)	5 (94)	5 (94)	2 (54)	**1 (18)**

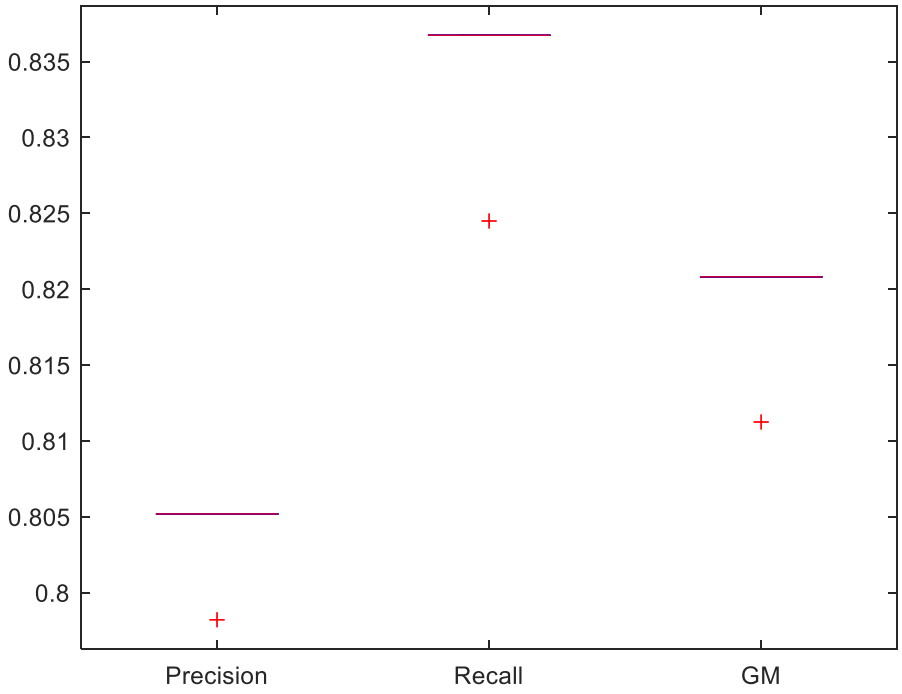


Fig. 5.7 Iris boxplot of EGWAC

5.3.3 Comparing with GWOTS

In this part, EGWAC is compared to a hybrid version of GWO with tabu search (TS) called GWOTS and other seven well-utilized algorithms including: GWO, TS, DE, PSO, GA, and simulated annealing (SA). The algorithms were tested on 9 real datasets that were used to test clustering algorithms performance [72, 10, 9]. The results were measured through two measures: purity, and entropy. High values in purity measure and low ones in entropy indicate the effectiveness of the algorithm.

The results of the algorithms compared with are taken from [9]. The maximum number of iteration were set for TS and SA to 10000 as they are trajectory-based algorithms. The maximum number of iteration were set to 200 for the population-based algorithms (rest), and their population size to 50. The results were gathered over 30 independent runs.

Tables 5.4 and 5.5 provide a detailed comparison of the previously mentioned clustering algorithms across two performance measures and nine different datasets. These tables offer insights into the effectiveness and reliability of each algorithm in clustering diverse types of data.

Table 5.4 presents the purity clustering results for different algorithms across multiple datasets, including IRIS, Wine, Glass, Haberman, Bupa, and CMC. The table includes the mean purity value, standard deviation (STD), median, best, worst values, and the ranking of each algorithm based on its performance. The best results are highlighted in bold. The EGWAC algorithm consistently outperformed the other algorithms, achieving the best purity results in five out of nine datasets (IRIS,

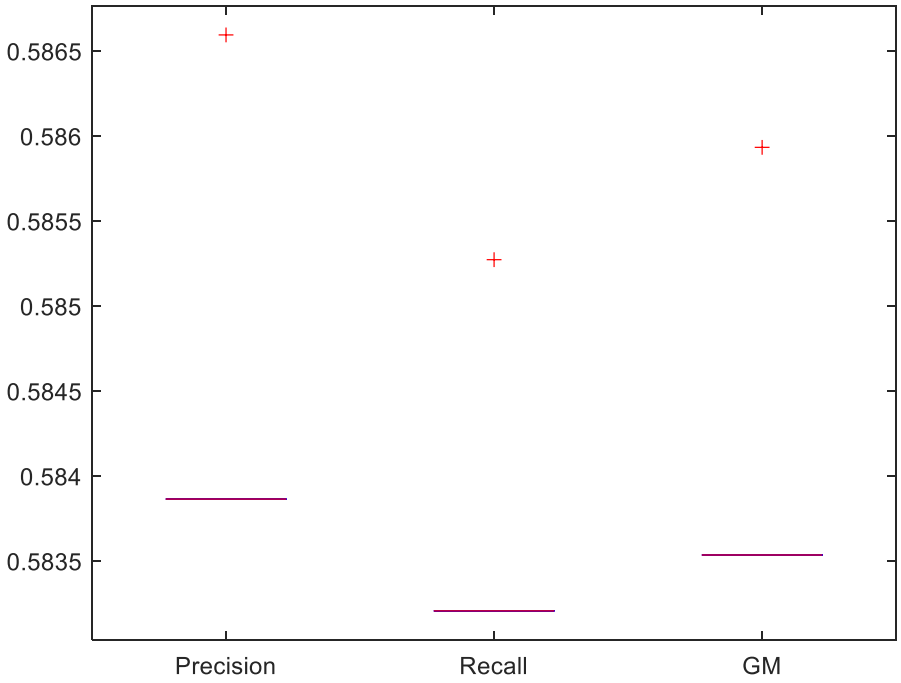


Fig. 5.8 Wine boxplot of EGWAC

Blood, Glass, Seeds, and Haberman), followed by GWOTS which gave the best results in Wine and Heart datasets. The EGWAC stuck in a local optima in Heart dataset, where it is observed that the mean median, best worst values are the same (0.59259) along with the consistency in STD. EGWAC had the lowest standard deviation, indicating a high level of consistency in its performance across different datasets. Overall, EGWAC was ranked first in the majority of the datasets, demonstrating its effectiveness in producing high-quality clustering results.

Table 5.5 exhibits the entropy clustering results for the algorithms across the different datasets. Similar to Table 4, it includes the mean entropy value, standard deviation (STD), median, best, worst values, and the ranking of each algorithm based on its performance. The best results are highlighted in bold. In this table, the EGWAC algorithm also demonstrated strong performance, achieving the best entropy results in several datasets, including IRIS, Blood, Glass, Seeds and Haberman. Like in purity, EGWAC doesn't perform well and get trapped in local optima as it is presented in the table where the mean, median, best, and worst values are the same (0.97169) with the Nil value of the STD.

In summary, previous tables demonstrate that the Enhanced Grey Wolf Optimizer for Data Clustering (EGWAC) algorithm in overall outperformed the other algorithms in terms of purity and entropy across a range of datasets. Its strong performance and low variability make it a promising algorithm for data clustering tasks. However it doesn't perform well in Heart dataset regarding the two measures, purity and entropy where it gets trapped in local optima.

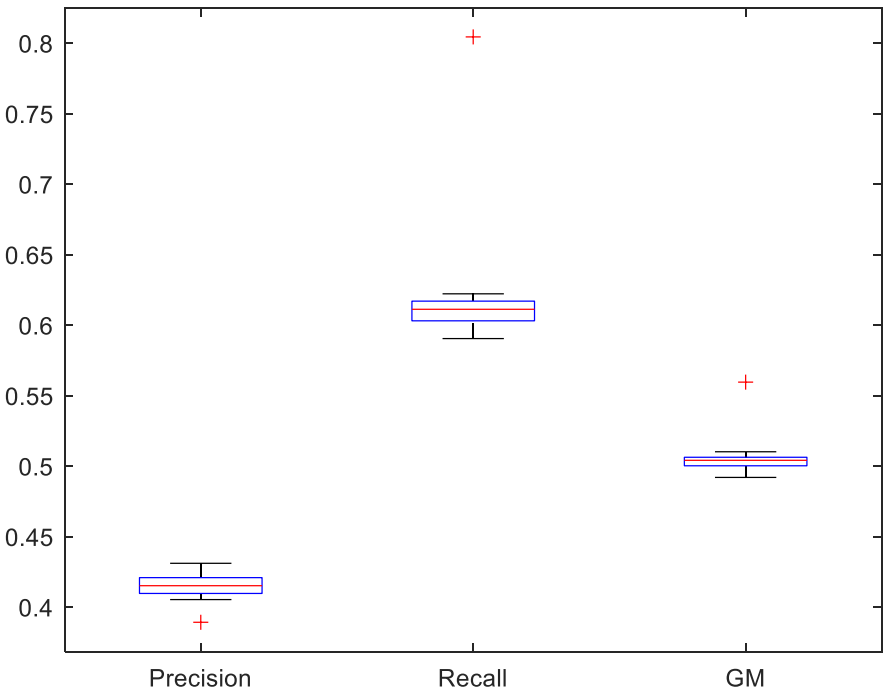


Fig. 5.9 Glass boxplot of EGWAC

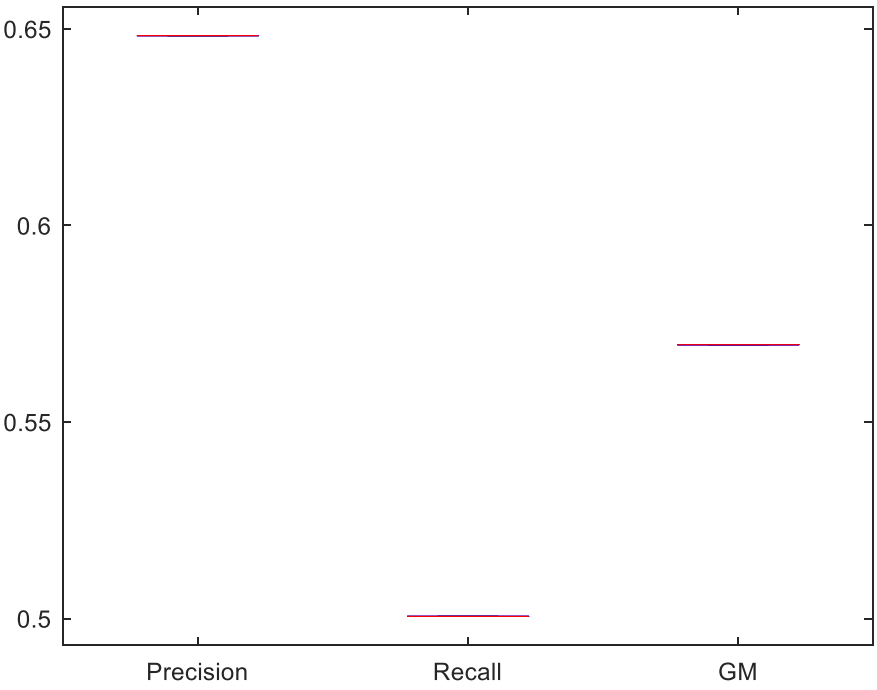


Fig. 5.10 Haberman boxplot of EGWAC

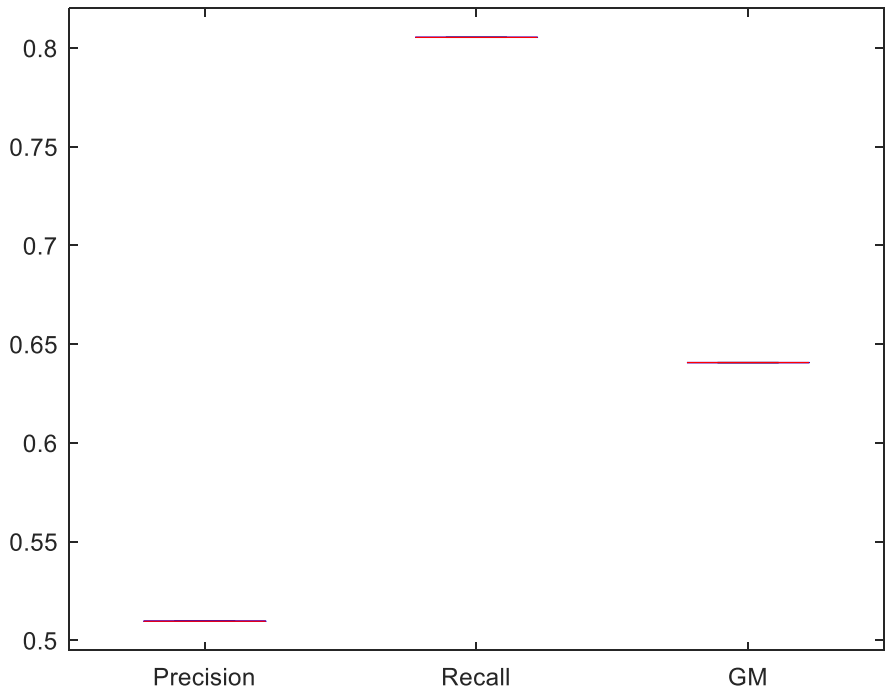


Fig. 5.11 Bupa boxplot of EGWAC

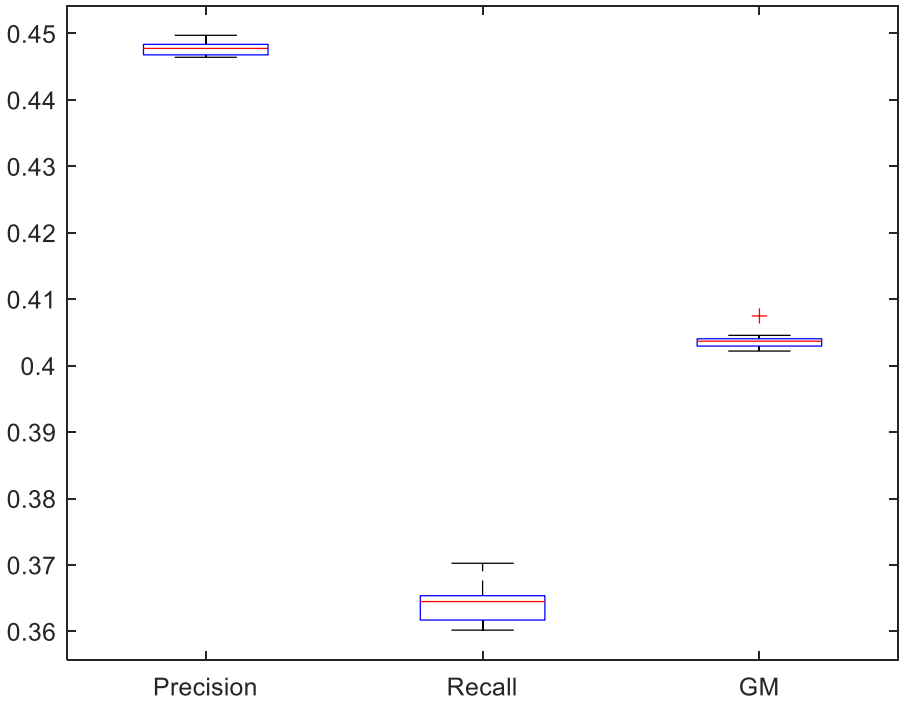


Fig. 5.12 CMC boxplot of EGWAC

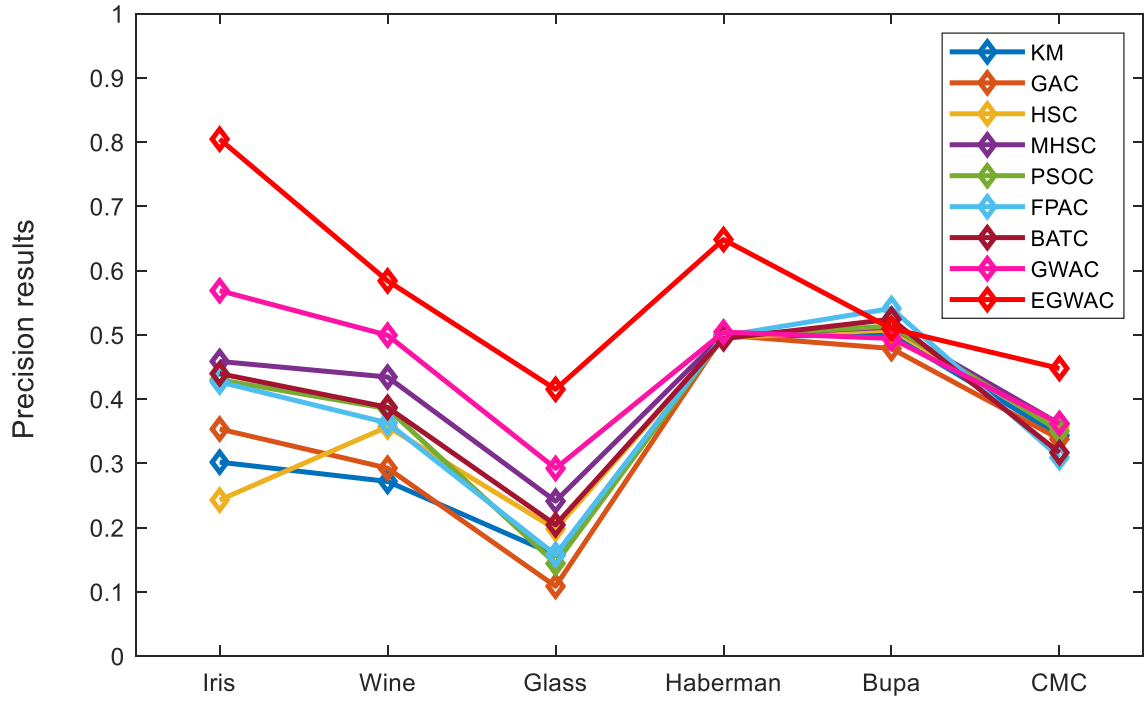


Fig. 5.13 Visual comparison of Precision.

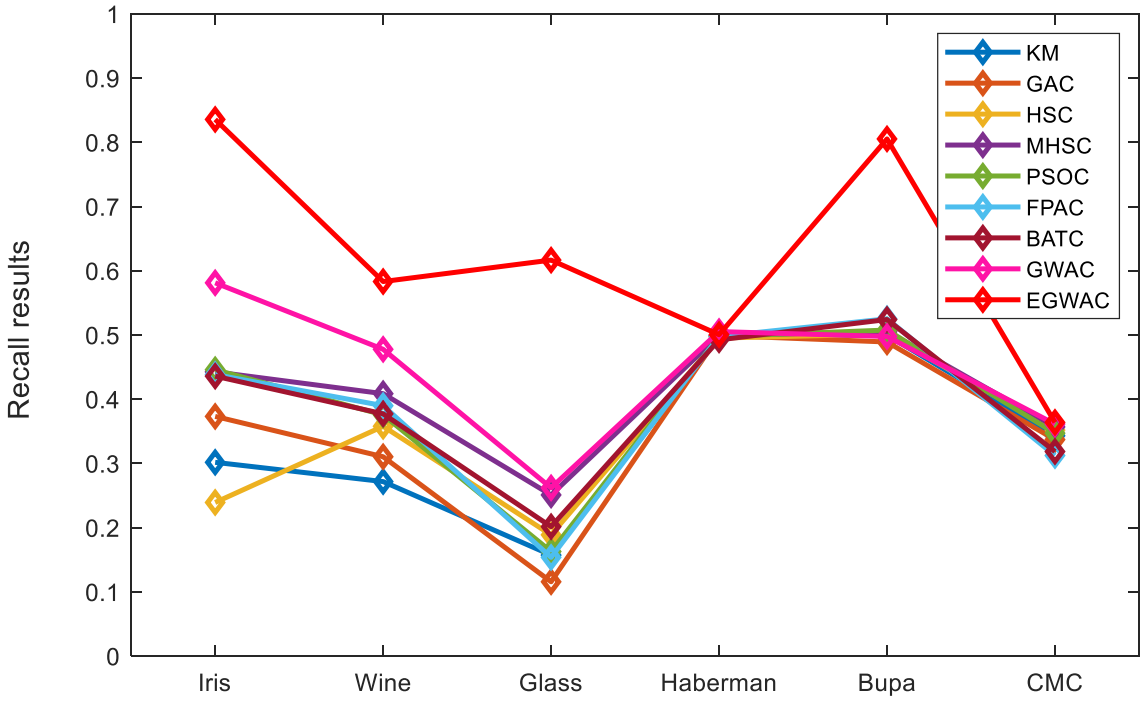


Fig. 5.14 Visual comparison of Recall.

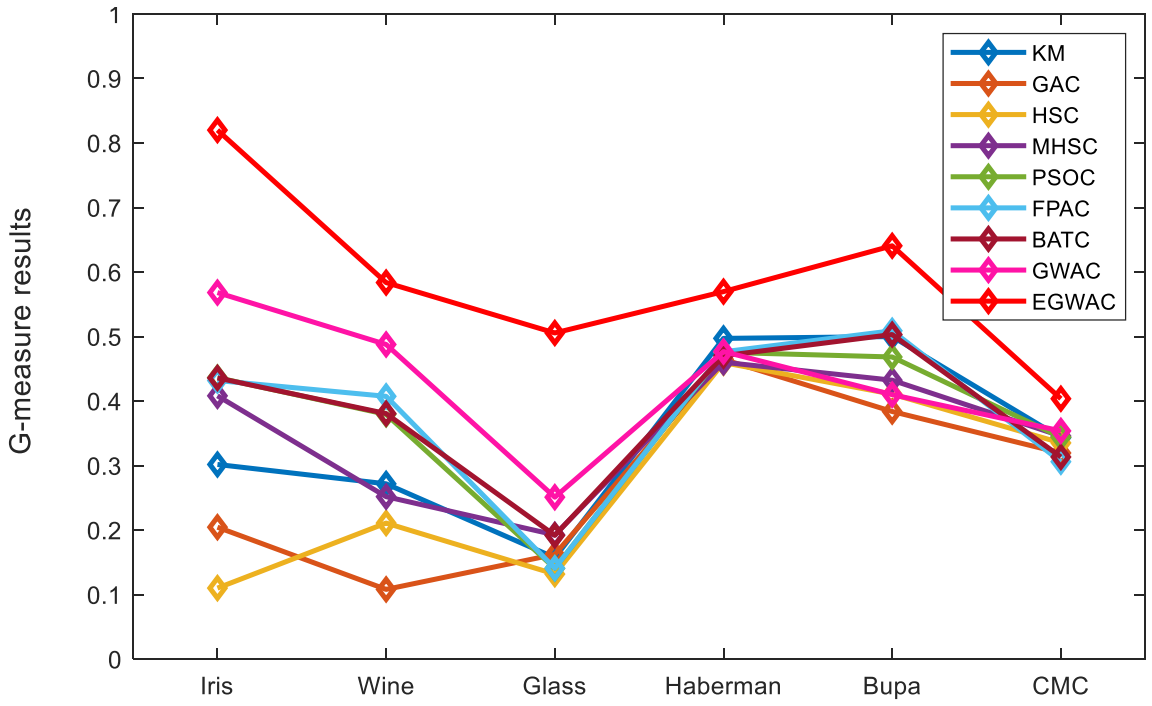


Fig. 5.15 Visual comparison of G-measure.

Table 5.4 Purity clustering results

Algorithm	S. measure	Iris	Blood	Glass	Seeds	Wine	Haberman	Heart	Liver	P. relax
EGWAC	MEAN	**0.89267**	**0.83289**	**0.57928**	**0.89302**	0.70243	**0.75490**	0.59259	0.57971	0.71429
	STD	0.00203	0.00000	0.01045	0.00325	0.00103	0.00000	0.00000	0.00000	0.00000
	MEDIAN	0.89333	0.83289	0.58411	0.89524	0.70225	0.75490	0.59259	0.57971	0.71429
	BEST	0.89333	0.83289	0.59346	0.89524	0.70787	0.75490	0.59259	0.57971	0.71429
	WORST	0.88667	0.83289	0.55608	0.88571	0.70225	0.75490	0.59259	0.57971	0.71429
	RANK	1	1	1	1	3	1	8	2	3
GWOTS	MEAN	0.88667	0.76203	0.54439	0.88857	**0.95506**	0.73529	**0.77704**	0.57971	0.71429
	STD	0.00000	0.00000	0.00832	0.00246	0.00917	0.00000	0.06483	0.00000	0.00000
	MEDIAN	0.88667	0.76203	0.54673	0.89048	0.95506	0.73529	0.79630	0.57971	0.71429
	BEST	0.88667	0.76203	0.55140	0.89048	0.96629	0.73529	0.80000	0.57971	0.71429
	WORST	0.88667	0.76203	0.52804	0.88571	0.93820	0.73529	0.59259	0.57971	0.71429
	RANK	2	4	2	2	1	2	1	2	3
GWO	MEAN	0.82200	0.76203	0.49579	0.87048	0.59888	0.73529	0.76148	0.57971	0.71429
	STD	0.10419	0.00000	0.01713	0.07785	0.07268	0.00000	0.04905	0.00000	0.00000
	MEDIAN	0.88667	0.76203	0.49533	0.90000	0.62360	0.73529	0.78519	0.57971	0.71429
	BEST	0.88667	0.76203	0.52336	0.91429	0.64607	0.73529	0.80000	0.57971	0.71429
	WORST	0.66667	0.76203	0.46262	0.65714	0.39888	0.73529	0.65185	0.57971	0.71429
	RANK	4	4	4	3	4	2	2	2	3
TS	MEAN	0.82067	0.76203	0.50654	0.86524	0.92079	0.73529	0.72852	0.57971	0.71429
	STD	0.10627	0.00000	0.02673	0.07316	0.10259	0.00000	0.09732	0.00000	0.00000
	MEDIAN	0.88667	0.76203	0.50701	0.88810	0.94944	0.73529	0.79630	0.57971	0.71429
	BEST	0.88667	0.76203	0.55140	0.89048	0.96629	0.73529	0.80000	0.57971	0.71429
	WORST	0.66667	0.76203	0.44860	0.65714	0.62921	0.73529	0.59259	0.57971	0.71429
	RANK	5	4	3	4	2	2	3	2	3

Table 5.4 Continued

Algorithm	S. measure	Iris	Blood	Glass	Seeds	Wine	Haberman	Heart	Liver	P. relax
DE	MEAN	0.86733	0.76203	0.45047	0.77048	0.58202	0.73529	0.68815	0.57971	0.71429
	STD	0.03777	0.00000	0.03201	0.09162	0.11971	0.00000	0.10174	0.00000	0.00000
	MEDIAN	0.87333	0.76203	0.45093	0.80476	0.60955	0.73529	0.71111	0.57971	0.71429
	BEST	0.93333	0.76203	0.48131	0.85238	0.73034	0.73529	0.80370	0.57971	0.71429
	WORST	0.82000	0.76203	0.36916	0.62857	0.39888	0.73529	0.55556	0.57971	0.71429
	RANK	3	4	6	5	6	2	7	2	3
PSO	MEAN	0.77133	0.76270	0.44206	0.76095	0.41966	0.73529	0.70148	**0.58000**	**0.71484**
	STD	0.09270	0.00144	0.04295	0.11586	0.06378	0.00000	0.10612	0.00092	0.00174
	MEDIAN	0.77333	0.76203	0.45561	0.75000	0.39888	0.73529	0.74259	0.57971	0.71429
	BEST	0.88667	0.76604	0.48131	0.88571	0.60112	0.73529	0.81481	0.58261	0.71978
	WORST	0.66667	0.76203	0.35514	0.62381	0.39888	0.73529	0.55556	0.57971	0.71429
	RANK	6	2	8	7	8	2	6	1	1
GA	MEAN	0.75333	0.76243	0.48972	0.76762	0.59888	0.73529	0.70519	0.57971	0.71429
	STD	0.08433	0.00127	0.03763	0.08056	0.07718	0.00000	0.08145	0.00000	0.00000
	MEDIAN	0.74000	0.76203	0.47430	0.77143	0.61236	0.73529	0.73148	0.57971	0.71429
	BEST	0.92667	0.76604	0.56542	0.86667	0.69663	0.73529	0.80000	0.57971	0.71429
	WORST	0.66667	0.76203	0.44393	0.64762	0.44944	0.73529	0.58148	0.57971	0.71429
	RANK	7	3	5	6	4	2	5	2	3
SA	MEAN	0.75067	0.76203	0.44252	0.71476	0.52584	0.73529	0.70630	0.57971	**0.71484**
	STD	0.09544	0.00000	0.04215	0.08445	0.10206	0.00000	0.09650	0.00000	0.00174
	MEDIAN	0.71667	0.76203	0.45327	0.70000	0.58427	0.73529	0.70926	0.57971	0.71429
	BEST	0.90000	0.76203	0.48131	0.87619	0.62921	0.73529	0.82222	0.57971	0.71978
	WORST	0.66667	0.76203	0.36449	0.61429	0.39888	0.73529	0.56667	0.57971	0.71429
	RANK	8	4	7	8	7	2	4	2	1

Table 5.5 Entropy clustering results

Algorithm	S. measure	Iris	Blood	Glass	Seeds	Wine	Haberman	Heart	Liver	P. relax
EGWAC	MEAN	**0.25002**	**0.73452**	**0.52311**	**0.30231**	0.56456	**0.82199**	0.97169	0.98096	0.86307
	STD	0.00460	0.00000	0.01312	0.00645	0.00052	0.00000	0.00000	0.00000	0.00012
	MEDIAN	0.24852	0.73452	0.52182	0.30654	0.56465	0.82199	0.97169	0.98096	0.86311
	BEST	0.24852	0.73452	0.49509	0.29247	0.56180	0.82199	0.97169	0.98096	0.86269
	WORST	0.26358	0.73452	0.54613	0.31200	0.56465	0.82199	0.97169	0.98096	0.86312
	RANK	1	1	1	1	3	1	8	5	8
GWOTS	MEAN	0.26358	0.79142	0.56357	0.33019	**0.14427**	0.83309	**0.74351**	0.98154	0.86039
	STD	0.00000	0.00000	0.01751	0.00463	0.02278	0.00001	0.07796	0.00006	0.00125
	MEDIAN	0.26358	0.79142	0.56040	0.32660	0.14471	0.83309	0.72126	0.98158	0.85997
	BEST	0.26358	0.79142	0.53241	0.32660	0.11713	0.83309	0.71299	0.98146	0.85861
	WORST	0.26358	0.79142	0.59110	0.33557	0.18701	0.83311	0.96516	0.98158	0.86181
	RANK	2	6	2	2	1	5	1	8	4
GWO	MEAN	0.32035	0.79146	0.62082	0.34362	0.61111	0.83302	0.76078	0.98122	0.86010
	STD	0.09203	0.00008	0.03400	0.08287	0.13461	0.00049	0.05061	0.00040	0.00343
	MEDIAN	0.26358	0.79142	0.63095	0.31305	0.57427	0.83309	0.73919	0.98133	0.86141
	BEST	0.26358	0.79142	0.56312	0.27930	0.51743	0.83176	0.71299	0.98051	0.85467
	WORST	0.46659	0.79157	0.66959	0.55760	0.97907	0.83375	0.86730	0.98158	0.86312
	RANK	5	8	4	3	4	4	2	6	2
TS	MEAN	0.31529	0.79142	0.59599	0.35329	0.18998	0.83309	0.80528	0.98150	0.86016
	STD	0.08419	0.00000	0.03803	0.07193	0.13784	0.00000	0.11911	0.00012	0.00119
	MEDIAN	0.26358	0.79142	0.58555	0.33108	0.15222	0.83309	0.72126	0.98158	0.85969
	BEST	0.26358	0.79142	0.54014	0.32660	0.11713	0.83309	0.71299	0.98130	0.85861
	WORST	0.46659	0.79142	0.66323	0.55760	0.58062	0.83309	0.96516	0.98158	0.86181
	RANK	4	6	3	4	2	5	3	7	3

Table 5.5 Continued

Algorithm	S. measure	Iris	Blood	Glass	Seeds	Wine	Haberman	Heart	Liver	P. relax
DE	MEAN	0.27222	0.79084	0.68205	0.44985	0.69294	0.83313	0.85330	0.97984	0.86128
	STD	0.04379	0.00149	0.04515	0.10567	0.18513	0.00055	0.11203	0.00208	0.00205
	MEDIAN	0.27664	0.79135	0.67478	0.41566	0.59598	0.83312	0.85073	0.98063	0.86161
	BEST	0.21191	0.78663	0.63147	0.33300	0.51678	0.83219	0.70634	0.97471	0.85610
	WORST	0.34021	0.79159	0.79821	0.63013	0.97426	0.83375	0.98294	0.98151	0.86312
	RANK	3	5	6	5	5	7	7	4	5
PSO	MEAN	0.34250	0.78967	0.69849	0.45737	0.94772	0.83282	0.82174	0.97952	0.86210
	STD	0.07052	0.00267	0.06725	0.11405	0.12182	0.00056	0.11015	0.00172	0.00244
	MEDIAN	0.34402	0.79084	0.67235	0.48304	0.98855	0.83309	0.79934	0.98005	0.86312
	BEST	0.24642	0.78475	0.64625	0.32839	0.60128	0.83176	0.69125	0.97619	0.85559
	WORST	0.42062	0.79164	0.83655	0.59974	0.98855	0.83309	0.98793	0.98158	0.86312
	RANK	6	3	7	6	8	3	4	3	7
GA	MEAN	0.39998	0.78546	0.63641	0.45985	0.72193	0.83214	0.84654	**0.97880**	0.86157
	STD	0.09578	0.00962	0.04127	0.06536	0.10238	0.00250	0.09656	0.00550	0.00171
	MEDIAN	0.40526	0.78878	0.63960	0.46180	0.73591	0.83308	0.83261	0.98111	0.86205
	BEST	0.19146	0.75963	0.56619	0.36536	0.57990	0.82597	0.70599	0.96391	0.85862
	WORST	0.51821	0.79151	0.69909	0.57000	0.92031	0.83376	0.97198	0.98158	0.86311
	RANK	8	2	5	7	6	2	6	1	6
SA	MEAN	0.36360	0.79045	0.70300	0.52086	0.76438	0.83320	0.83038	0.97883	**0.85972**
	STD	0.07384	0.00125	0.06329	0.09839	0.18297	0.00063	0.11629	0.00345	0.00486
	MEDIAN	0.40394	0.79101	0.68946	0.54009	0.67757	0.83336	0.84784	0.98052	0.86301
	BEST	0.24595	0.78828	0.61750	0.33188	0.54954	0.83170	0.67510	0.97240	0.84956
	WORST	0.42062	0.79164	0.81684	0.68006	0.98855	0.83376	0.98701	0.98159	0.86312
	RANK	7	4	8	8	7	8	5	2	1

5.4 Conclusion

In conclusion, this chapter presented our proposed method (Enhanced Grey Wolf Optimizer for Data Clustering algorithm). EGWAC has shown remarkable performance across multiple datasets when compared to other well-utilized clustering algorithms. EGWAC outperformed the original Grey Wolf Algorithm for Clustering (GWAC) and also showed superiority over a hybrid version of Grey Wolf Optimizer with Tabu Search (GWOTS).

The consistent and resilient performance of EGWAC, as evidenced by its low standard deviation and high median values, highlights its reliability and stability in producing high-quality clustering results. The algorithm's effectiveness was particularly notable in datasets such as IRIS, Blood, Glass, Seeds, and Haberman, where it consistently ranked first in purity and entropy measures.

Overall, the results presented in this chapter underscore the potential of EGWAC as a robust and dependable algorithm for data clustering tasks. Its enhanced parts such as position updation mechanism have demonstrated significant improvements over existing algorithms, making EGWAC a promising choice for a wide range of clustering applications. In spite of that, EGWAC still have some limitations like stacking in local optima as it is observed in Heart dataset.

Conclusion and Future works

This thesis has made significant contributions to the field of data clustering, offering a comprehensive overview of the field and proposing new and optimized algorithms to deal with data clustering effectively. In this research, it has been shown that data clustering is of fundamental importance under modern computer science. The research study began from a discussion about the data clustering problem, domains of utilization and some of the current well-known solving methods, which consequenced in understanding and uncovering some limitations in the existing data clustering techniques. It thus gained an insight into trying to overcome these limitations by proposing adapted algorithms and optimizing some existing ones.

The thesis provides a detailed understanding of data clustering, its objectives, applications, and various techniques used in the field. Along with that, it delves into the measures used to assess the quality of clustering results, including validity indices and performance measures, providing insights into the evaluation of clustering techniques. The thesis introduces optimization methods inspired by real-world behaviors and systems, such as genetic algorithms, rat swarm optimizer, and grey wolf optimizer, and explores their applications in the context of data clustering. These sections are the fruit of answering the questions about the data clustering problem and techniques to solve it. The thesis, then presents our contributions, the Rat Swarm Optimizer for Data Clustering (RSOC) and the Enhanced Grey Wolf Optimizer for Data Clustering (EGWAC), which are the results of our efforts in finding how to address the limitations of the clustering techniques and improve them.

In Chapter 4, the thesis introduced the Rat Swarm Optimizer for Data Clustering (RSOC) and compared its performance with well-established algorithms. The experimental results demonstrated the superior ability of RSOC to yield better solutions compared to the algorithms it was pitted against. In Chapter 5, the research presented the Enhanced Grey Wolf Optimizer for Data Clustering (EGWAC) and compared its performance with the original Grey Wolf Optimizer and other algorithms. The results showed that EGWAC outperformed the original algorithm and other established algorithms across multiple datasets, demonstrating its potential as an effective clustering algorithm. The findings highlighted the effectiveness and consistency of EGWAC in producing high-quality clustering results.

Overall, this research work significantly contributes to the field of data clustering by providing a comprehensive understanding of the subject, introducing new algorithms, and offering valuable insights for future research and improvements in the field. However, it is important to acknowledge

its limitations as there remained some cases where our two contributions have weaknesses in, as it was stated in the previous chapters for Cancer and Heart datasets.

Further research and improvements are needed to enhance the performance of data clustering algorithms in diverse application domains. The findings presented in this thesis serve as a valuable resource for researchers, and students in the field of computer science, paving the way for more effective and efficient solutions in the future. As future works, it is decided to propose a hybridization of the rat swarm optimizer with tabu search or harmony search to avoid sticking in local optima. It is also decided to apply the main idea in EGWAC to other clustering algorithms and other domains to that can be performed in.

Ultimately, this thesis forms a framework for understanding what data clustering means, its applications and implementation barriers. It is hoped that the views and findings expressed in this work will provide impetus toward better and refined approaches to data clustering in the future.

References

[1] Abualigah, L., Abd Elaziz, M., Shehab, M., Ahmad Alomari, O., Alshinwan, M., Alabool, H., and Al-Arabiat, D. A. (2021). Hybrid harris hawks optimization with differential evolution for data clustering. In *Metaheuristics in Machine Learning: Theory and Applications*, pages 267–299. Springer.

[2] Abualigah, L. M., Khader, A. T., AlBetar, M. A., and Hanandeh, E. S. (2017a). A new hybridization strategy for krill herd algorithm and harmony search algorithm applied to improve the data clustering. In *First EAI international conference on computer science and engineering*, pages 54–63.

[3] Abualigah, L. M., Khader, A. T., Hanandeh, E. S., and Gandomi, A. H. (2017b). A novel hybridization strategy for krill herd algorithm applied to clustering techniques. *Applied Soft Computing*, 60:423–435.

[4] Aeberhard, S. and Forina, M. (1991). Wine. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5PC7J.

[5] Aggarwal, C. C. and Reddy, C. K. (2014). *Data clustering: Algorithms and applications*. Chapman and Hall/CRC.

[6] Aldenderfer, M. S. and Blashfield, R. K. (1984). Cluster analysis. newberry park.

[7] Alia, O. M., Al-Betar, M. A., Mandava, R., and Khader, A. T. (2011). Data clustering using harmony search algorithm. In *Swarm, Evolutionary, and Memetic Computing: Second International Conference, SEMCCO 2011, Visakhapatnam, Andhra Pradesh, India, December 19-21, 2011, Proceedings, Part II 2*, pages 79–88. Springer.

[8] Aljarah, I., Habib, M., Nujoom, R., Faris, H., and Mirjalili, S. (2021). A comprehensive review of evaluation and fitness measures for evolutionary data clustering. *Evolutionary Data Clustering: Algorithms and Applications*, pages 23–71.

[9] Aljarah, I., Mafarja, M., Heidari, A. A., Faris, H., and Mirjalili, S. (2020a). Clustering analysis using a novel locality-informed grey wolf-inspired clustering approach. *Knowledge and Information Systems*, 62:507–539.

[10] Aljarah, I., Mafarja, M., Heidari, A. A., Faris, H., and Mirjalili, S. (2020b). Multi-verse optimizer: theory, literature review, and application in data clustering. *Nature-inspired optimizers*, pages 123–141.

[11] Almufti, S. M., Shaban, A. A., Ali, Z. A., Ali, R. I., and Fuente, J. A. D. (2023). Overview of metaheuristic algorithms. *Polaris Global Journal of Scholarly Research and Trends*, 2(2):10–32.

[12] Anderberg, M. (1973). *Cluster Analysis for Applications*. Probability and Mathematical Statistics : a series of monographs and textbooks. Academic Press.

[13] Arthur, D. and Vassilvitskii, S. (2007). K-means++ the advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035.

[14] Bagirov, A. M., Karmitsa, N., and Taheri, S. (2020). Partitional clustering via nonsmooth optimization. *Cham, Switzerland: Springer Nature*.

[15] Bailey, K. (1975). (1974)" cluster analysis," pp. 59-128 in dr heise (ed.).

[16] Ball, G. H. and Hall, D. J. (1965). Isodata, a novel method of data analysis and pattern classification. Technical report, Stanford research inst Menlo Park CA.

[17] Berkhin, P. (2006). A survey of clustering data mining techniques. *Grouping multidimensional data: Recent advances in clustering*, pages 25–71.

[18] Bhatt, R. (2012). Planning Relax. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5T023.

[19] Brucker, P. (1978). On the complexity of clustering problems. In *Optimization and Operations Research: Proceedings of a Workshop Held at the University of Bonn, October 2–8, 1977*, pages 45–54. Springer.

[20] Cattell, R. B. (1943). The description of personality: Basic traits resolved into clusters. *The journal of abnormal and social psychology*, 38(4):476–506.

[21] Charytanowicz, M., Niewczas, J., Kulczycki, P., Kowalski, P., and Lukasik, S. (2012). seeds. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5H30K.

[22] Cowgill, M. C., Harvey, R. J., and Watson, L. T. (1999). A genetic algorithm approach to cluster analysis. *Computers & Mathematics with Applications*, 37(7):99–108.

[23] Davidson, I. and Ravi, S. (2005). Agglomerative hierarchical clustering with constraints: Theoretical and empirical results. In *European Conference on Principles of Data Mining and Knowledge Discovery*, pages 59–70. Springer.

[24] Davies, D. L. and Bouldin, D. W. (1979). A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, (2):224–227.

[25] Dhiman, G., Garg, M., Nagar, A., Kumar, V., and Dehghani, M. (2021). A novel algorithm for global optimization: rat swarm optimizer. *Journal of Ambient Intelligence and Humanized Computing*, 12:8457–8482.

[26] (DIMD). Statlog (Heart). UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C57303.

[27] Driver, H. E. and Kroeber, A. L. (1932). Quantitative expression of cultural relationships. *University of California Publications in American Archaeology and Ethnology*, 31(2):211–256.

[28] Dua, D. and Karra Taniskidou, E. (2017). Uci machine learning repository.

[29] Dunn, J. C. (1974). Well-separated clusters and optimal fuzzy partitions. *Journal of cybernetics*, 4(1):95–104.

[30] Edwards, A. W. F. and Cavalli-Sforza, L. L. (1965). A method for cluster analysis. *Biometrics*, 21(2):362–375.

[31] El Bouchefry, K. and de Souza, R. S. (2020). Learning in big data: Introduction to machine learning. In *Knowledge discovery in big data from astronomy and earth observation*, pages 225–249. Elsevier.

[32] Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pages 226–231.

[33] Ezugwu, A. E., Ikotun, A. M., Oyelade, O. O., Abualigah, L., Agushaka, J. O., Eke, C. I., and Akinyelu, A. A. (2022). A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects. *Engineering Applications of Artificial Intelligence*, 110:104743.

[34] Fielding, A. H. (2006). *Cluster and classification techniques for the biosciences*. Cambridge University Press.

[35] Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188.

[36] Fisher, W. D. (1969). Clustering and aggregation in economics.

[37] Florek, K., Łukaszewicz, J., Perkal, J., Steinhaus, H., and Zubrzycki, S. (1951). Sur la liaison et la division des points d'un ensemble fini. In *Colloquium mathematicum*, volume 2, pages 282–285.

[38] Forgy, E. W. (1965). Cluster analysis of multivariate data: efficiency versus interpretability of classifications." biometrics 21.

[39] Forsyth, R. S. (1990). Liver Disorders. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C54G67.

[40] Gan, G., Ma, C., and Wu, J. (2007). *Data clustering: Theory, algorithms, and applications*. SIAM, 1 edition.

[41] German, B. (1987). Glass Identification. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5WW2P.

[42] Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc.

[43] Goldberg, D. E. (1990). A note on boltzmann tournament selection for genetic algorithms and population-oriented simulated annealing. *Complex Systems*, 4:445–460.

[44] Gordon, A. D. (1999). *Classification*. CRC Press.

[45] Gore, P. A. (2000). 11 - cluster analysis. In Tinsley, H. E. and Brown, S. D., editors, *Handbook of Applied Multivariate Statistics and Mathematical Modeling*, pages 297–321. Academic Press, San Diego.

[46] Gower, J. C. (1967). A comparison of some methods of cluster analysis. *Biometrics*, pages 623–637.

[47] Gower, J. C. (1971). A general coefficient of similarity and some of its properties. *Biometrics*, pages 857–871.

[48] Green, P. E., Frank, R. E., and Robinson, P. J. (1967). Cluster analysis in test market selection. *Management science*, 13(8):B–387.

[49] Haberman, S. (1999). Haberman's Survival. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5XK51.

[50] Halkidi, M., Batistakis, Y., and Vazirgiannis, M. (2002a). Cluster validity methods: part i. *ACM Sigmod Record*, 31(2):40–45.

[51] Halkidi, M., Batistakis, Y., and Vazirgiannis, M. (2002b). Clustering validity checking methods: Part ii. *ACM Sigmod Record*, 31(3):19–27.

[52] Han, J., Kamber, M., and Pei, J. (2012). Data mining concepts and techniques third edition. *University of Illinois at Urbana-Champaign Micheline Kamber Jian Pei Simon Fraser University*.

[53] Hartigan, J. A. (1975). *Clustering algorithms*. John Wiley & Sons, Inc.

[54] Hennig, C., Meila, M., Murtagh, F., and Rocci, R. (2015). *Handbook of cluster analysis*. CRC press.

[55] Hinneburg, A. and Keim, D. A. (1998). An efficient approach to clustering in large multimedia databases with noise. In *Proceedings of the Fourth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 58–65.

[56] Hinterding, R. (1995). Gaussian mutation and self-adaption for numeric genetic algorithms. In *Proceedings of 1995 IEEE International Conference on Evolutionary Computation*, volume 1, page 384. IEEE.

[57] Holland, J. H. (1975). Adaptation in natural and artificial systems/john h. holland. *Ann Arbor: University of Michigan Press*.

[58] Horn, D. (1944). A study of personality syndromes. *Journal of Personality*, 12(4):257–274.

[59] Houssein, E. H., El-din Helmy, B., Oliva, D., Elngar, A. A., and Shaban, H. (2021). *Multi-level Thresholding Image Segmentation Based on Nature-Inspired Optimization Algorithms: A Comprehensive Review*, pages 239–265. Springer International Publishing.

[60] Hruschka, E. R., Campello, R. J. G. B., Freitas, A. A., and Ponce Leon F. de Carvalho, A. C. (2009). A survey of evolutionary algorithms for clustering. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 39(2):133–155.

[61] Hämäläinen, J. (2018). *Improvements and Applications of the Elements of Prototype-Based Clustering*. PhD thesis.

[62] Jain, A. K. (2010). Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666.

[63] Jain, A. K. and Dubes, R. C. (1988). *Algorithms for clustering data*. Prentice-Hall, Inc.

[64] Jain, A. K., Murty, M. N., and Flynn, P. J. (1999). Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323.

[65] Jancey, R. (1966). Multidimensional group analysis. *Australian Journal of Botany*, 14(1):127–130.

[66] Jardine, N. and Sibson, R. (1968). The construction of hierarchic and non-hierarchic classifications. *The computer journal*, 11(2):177–184.

[67] Kaiser, H. F. (1966). An objective method for establishing legislative districts. *Midwest Journal of Political Science*, 10(2):200–213.

[68] Kaufman, L. and Rousseeuw, P. J. (1990). Finding groups in data. an introduction to cluster analysis. *Wiley Series in Probability and Mathematical Statistics. Applied Probability and Statistics*.

[69] King, B. (1967). Step-wise clustering procedures. *Journal of the American Statistical Association*, 62(317):86–101.

[70] Krippendorff, K. (1980). Clustering.

[71] Kumar, R. (2012). Blending roulette wheel selection & rank selection in genetic algorithms. *International Journal of Machine Learning and Computing*, 2(4):365.

[72] Kumar, V., Chhabra, J. K., and Kumar, D. (2017). Grey wolf algorithm-based clustering technique. *Journal of Intelligent Systems*, 26(1):153–168.

[73] Lance, G. and Williams, W. (1966). A generalized sorting strategy for computer classifications. *Nature*, 212(5058):218–218.

[74] Lance, G. N. and Williams, W. T. (1967). A general theory of classificatory sorting strategies: 1. hierarchical systems. *The computer journal*, 9(4):373–380.

[75] Landau, S., Leese, M., Stahl, D., and Everitt, B. (2011). *Cluster Analysis*. Wiley Series in Probability and Statistics. Wiley.

[76] Lim, T.-S. (1997). Contraceptive Method Choice. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C59W2D.

[77] Liu, Y., Wu, X., and Shen, Y. (2011). Automatic clustering using genetic algorithms. *Applied Mathematics and Computation*, 4(218):1267–1279.

[78] Lloyd, S. (1982). Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137.

[79] Lorr, M. (1966). *Explorations in typing psychotics*. Pergamon.

[80] Macnaughton-Smith, P. (1965). Some statistical and other numerical techniques for classifying individuals.

[81] Massart, D., Plastria, F., and Kaufman, L. (1983). Non-hierarchical clustering with masloc. *Pattern Recognition*, 16(5):507–516.

[82] Maulik, U. and Bandyopadhyay, S. (2000). Genetic algorithm-based clustering technique. *Pattern recognition*, 33(9):1455–1465.

[83] McGarigal, K., Cushman, S., and Stafford, S. (2000a). *Multivariate Statistics for Wildlife Ecology Research*.

[84] McGarigal, K., Stafford, S., Cushman, S., McGarigal, K., Stafford, S., and Cushman, S. (2000b). Discriminant analysis. *Multivariate statistics for wildlife and ecology research*, pages 129–187.

[85] McQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proc. Fifth Berkeley Symposium on Mathematical Statistics and Probability, 1967*, pages 281–297.

[86] McQuitty, L. L. (1966). Similarity analysis by reciprocal pairs for discrete and continuous data. *Educational and Psychological measurement*, 26(4):825–831.

[87] Miller, B. L. and Goldberg, D. E. (1995). Genetic algorithms, tournament selection, and the effects of noise. *Complex Systems*, 9(3):193–212.

[88] Mirjalili, S., Mirjalili, S. M., and Lewis, A. (2014). Grey wolf optimizer. *Advances in engineering software*, 69:46–61.

[89] Mirkin, B. (2005). *Clustering for data mining: a data recovery approach*. Chapman and Hall/CRC.

[90] Morrison, D. G. (1967). Measurement problems in cluster analysis. *Management science*, 13(12):B–775.

[91] Muro, C., Escobedo, R., Spector, L., and Coppinger, R. (2011). Wolf-pack (canis lupus) hunting strategies emerge from simple rules in computational simulations. *Behavioural processes*, 88(3):192–197.

[92] Naik, A. and Satapathy, S. C. (2021). Past present future: a new human-based algorithm for stochastic optimization. *Soft Computing*, 25:12915–12976.

[93] Nakai, K. (1996). Ecoli. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5388M.

[94] Naldi, M. C., de Carvalho, A. C., Campell, R. J. G. B., and Hruschka, e. R. (2008). Genetic clustering for data mining. *Soft computing for knowledge discovery and data mining*, pages 113–132.

[95] Neubauer, A. (1997). A theoretical analysis of the non-uniform mutation operator for the modified genetic algorithm. In *Proceedings of 1997 IEEE International Conference on Evolutionary Computation (ICEC'97)*, pages 93–96. IEEE.

[96] Odell, P. L. and Duran, B. S. (1974). *Cluster Analysis: A Survey*. Springer.

[97] Oliver, I., Smith, D., and Holland, J. R. (1987). A study of permutation crossover operators on the traveling salesman problem. In *Proceedings of the Second International Conference on Genetic Algorithms on Genetic algorithms and their application*, pages 224–230.

[98] Peres, F. and Castelli, M. (2021). Combinatorial optimization problems and metaheuristics: Review, challenges, design, and development. *Applied Sciences*, 11(14):6449.

[99] Rahman, M. A. and Islam, M. Z. (2014). A hybrid clustering technique combining a novel genetic algorithm with k-means. *Knowledge-Based Systems*, 71:345–365.

[100] Rai, P. and Singh, S. (2010). A survey of clustering techniques. *International Journal of Computer Applications*, 7(12):1–5.

[101] Rana, S., Jasola, S., and Kumar, R. (2011). A review on particle swarm optimization algorithms and their applications to data clustering. *Artificial Intelligence Review*, 35:211–222.

[102] Ray, D. M. and Berry, B. J. (1966). Multivariate socioeconomic regionalization: a pilot study in central canada. *Papers on regional statistical studies*, pages 75–130.

[103] Rencher, A. C. (2002). Methods of multivariate analysis, 2nd edn., a john wiley & sons. *New York*.

[104] Schütze, H., Manning, C. D., and Raghavan, P. (2008). *Introduction to information retrieval*, volume 39. Cambridge University Press Cambridge.

[105] Siddiqi, U. F. and Sait, S. M. (2022). Heuristic for the data clustering problem. US Patent 11,442,978.

[106] Singh, A. and Kumar, A. (2021). Applications of nature-inspired meta-heuristic algorithms: A survey. *International Journal of Advanced Intelligence Paradigms*, 20(3-4):388–417.

[107] Sneath, P. H. (1957). The application of computers to taxonomy. *Microbiology*, 17(1):201–226.

[108] Sokal, R. R. and Michener, C. D. (1975). A statistical method for evaluating systematic relationships. *Multivariate statistical methods, among-groups covariation*, page 269.

[109] Sokal, R. R. and Sneath, P. H. A. (1963). *Principles of numerical taxonomy*. Freeman.

[110] Sorensen, T. A. (1948). A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on danish commons. *Biol. Skar.*, 5:1–34.

[111] Spath, H. (1985). *The cluster dissection and analysis theory fortran programs examples*. Prentice-Hall, Inc.

[112] Srinivas, M. and Patnaik, L. M. (1994). Adaptive probabilities of crossover and mutation in genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, 24(4):656–667.

[113] Steinhaus, H. (1957). Sur la division des corps matériels en parties. *Bulletin de l'Académie Polonaise des Sciences*, 4:801–804.

[114] Syswerda, G. (1991). A study of reproduction in generational and steady-state genetic algorithms. In *Foundations of genetic algorithms*, volume 1, pages 94–101. Elsevier.

[115] Tan, P.-N., Steinbach, M., and Kumar, V. (2005). *Introduction to Data Mining*. Addison-Wisley.

[116] Thorndike, R. (1953). Who belongs in the family? *Psychometrika*, 18(4):267–276.

[117] Tryon, R. C. (1939). Cluster analysis: correlation profile and orthometric analysis for the isolation of unities in mind and personality. *Ann Arbor: Edward Brothers*.

[118] Tsutsui, S., Yamamura, M., and Higuchi, T. (1999). Multi-parent recombination with simplex crossover in real coded genetic algorithms. In *Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation-Volume 1*, pages 657–664.

[119] Vinod, H. D. (1969). Integer programming and the theory of grouping. *Journal of the American Statistical association*, 64(326):506–519.

[120] Ward Jr, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244.

[121] Welch, W. J. (1982). Algorithmic complexity: three np-hard problems in computational statistics. *Journal of Statistical Computation and Simulation*, 15(1):17–25.

[122] Williams, W. and Lambert, J. t. (1966). Multivariate methods in plant ecology: V. similarity analyses and information-analysis. *The Journal of Ecology*, pages 427–445.

[123] Wingo Jr, L. (1967). Recent patterns of urbanization among latin american countries. *Urban Affairs Quarterly*, 2(3):81–109.

[124] Wishart, D. (1969). 256. note: An algorithm for hierarchical classifications. *Biometrics*, pages 165–170.

[125] Wishart, D. (2003). K-means clustering with outlier detection, mixed variables and missing values. In *Exploratory Data Analysis in Empirical Research: Proceedings of the 25 th Annual Conference of the Gesellschaft für Klassifikation eV, University of Munich, March 14–16, 2001*, pages 216–226. Springer.

[126] Wishert, D. (1969). Mode analysis: a generalization of nearest neighbour which reduces chaining effects (with discussion). *Numerical taxonomy*, pages 282–311.

[127] Wolberg, W. (1992). Breast Cancer Wisconsin (Original). UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5HP4Z.

[128] Wunsch, D. and Xu, R. (2008). *Clustering*. John Wiley & Sons.

[129] Yeh, I.-C. (2008). Blood Transfusion Service Center. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5GS39.

[130] Zebiri, I., Zeghida, D., and Redjimi, M. (2022a). Enhanced grey wolf optimizer for data clustering. In *International Conference on Artificial Intelligence: Theories and Applications*, pages 147–159. Springer.

[131] Zebiri, I., Zeghida, D., and Redjimi, M. (2022b). Rat swarm optimizer for data clustering. *Jordanian Journal of Computers and Information Technology*, 8(3).

[132] Zubin, J. (1938). A technique for measuring like-mindedness. *The Journal of Abnormal and Social Psychology*, 33(4):508.