symmetry

MDPI

Article

Fractional Derivative to Symmetrically Extend the Memory of Fuzzy C-Means

Safaa Safouan [1,†], Karim El Moutaouakil [1,*,†] and Alina-Mihaela Patriciu [2,*,†]

[1] Laboratory of Engineering Sciences, Multidisciplinary Faculty of Taza, Sidi Mohamed Ben Abdellah University, Taza 35000, Morocco; safaa.safouan@usmba.ac.ma
[2] Department of Mathematics and Computer Sciences, Faculty of Sciences and Environment, Dunărea de Jos University of Galaţi, 800201 Galaţi, Romania
* Correspondence: karim.elmoutaouakil@usmba.ac.ma (K.E.M.); alina.patriciu@ugal.ro (A.-M.P.)
† These authors contributed equally to this work.

Abstract: The fuzzy C-means (FCM) clustering algorithm is a widely used unsupervised learning method known for its ability to identify natural groupings within datasets. While effective in many cases, FCM faces challenges such as sensitivity to initial cluster assignments, slow convergence, and difficulty in handling non-linear and overlapping clusters. Aimed at these limitations, this paper introduces a novel fractional fuzzy C-means (Frac-FCM) algorithm, which incorporates fractional derivatives into the FCM framework. By capturing non-local dependencies and long memory effects, fractional derivatives offer a more flexible and precise representation of data relationships, making the method more suitable for complex datasets. Additionally, a genetic algorithm (GA) is employed to optimize a new least-squares objective function that emphasizes the geometric properties of clusters, particularly focusing on the Fukuyama–Sugeno and Xie–Beni indices, thereby enhancing the balance between cluster compactness and separation. Furthermore, the Frac-FCM algorithm is evaluated on several benchmark datasets, including Iris, Seed, and Statlog, and compared against traditional methods like K-means, SOM, GMM, and FCM. The results indicate that Frac-FCM consistently outperforms these methods in terms of the Silhouette and Dunn indices. For instance, Frac-FCM achieves higher Silhouette scores of most cases, indicating more distinct and well-separated clusters. Dunn's index further shows that Frac-FCM generates clusters that are better separated, surpassing the performance of traditional methods. These findings highlight the robustness and superior clustering performance of Frac-FCM. The Friedman test was employed to enhance and validate the effectiveness of Frac-FCM.

Keywords: fuzzy C-means; fractional derivative; genetic algorithm

Citation: Safouan, S.; El Moutaouakil, K.; Patriciu, A.-M. Fractional Derivative to Symmetrically Extend the Memory of Fuzzy C-Means. *Symmetry* **2024**, *16*, 1353. https://doi.org/10.3390/sym16101353

Academic Editor: Hsien-Chung Wu

Received: 12 September 2024
Revised: 5 October 2024
Accepted: 9 October 2024
Published: 12 October 2024

1. Introduction

Clustering is an unsupervised learning technique commonly applied in dynamic data analysis across various domains. Its main objective is to uncover inherent groups within a dataset, where each group or cluster represents data points sharing similar characteristics. These newly formed clusters are valuable, with their evaluation typically being focused on internal metrics. Clustering has broad applications, such as anomaly detection [1], image segmentation [2], market segmentation [3], and recommendation systems [4].

Clustering techniques can be broadly categorized into two types: hard and fuzzy. Hard clustering methods assign each data point exclusively to one cluster, whereas fuzzy clustering allows data points to belong to multiple clusters with different degrees of membership. Since Zadeh introduced fuzzy sets theory [5], various fuzzy clustering techniques have been presented. The fuzzy C-means (FCM) method is one of the most used approaches. Unlike traditional clustering [6], the FCM method is particularly useful when dealing with uncertainty, as it permits overlapping cluster membership, thus providing a more flexible and realistic representation of data relationships. However, FCM has notable

drawbacks: (a) its performance is heavily influenced by the initial cluster center positions; (b) it lacks a guarantee of reaching the global optimum, particularly with larger datasets; and (c) it often exhibits slow convergence when distinguishing between clusters [7].

Aiming to address the above-described defects, several extensions have been proposed in recent years. This study focuses on integrating fractional derivatives (FDs) into the FCM framework. Unlike integer order derivatives, fractional derivatives offer a more flexible and refined way to describe dynamic systems by capturing non-local dependencies and long memory effects [8]. These features are particularly useful for clustering complex datasets, where relationships between data points may span multiple clusters over time, providing a more accurate representation of real-world dynamical systems. This approach addresses some of FCM's key drawbacks, such as slow convergence and a sensitivity to initialization, while also enhancing its ability to manage non-linear and overlapping cluster structures.

A significant challenge in clustering is determining the optimal data partitioning, particularly in partition-based algorithms like k-means and FCM. This process requires selecting the correct number of clusters, a critical factor in ensuring meaningful and reliable results. Various clustering validity indices (CVIs) have been proposed to evaluate the quality of clustering outcomes and help determine the optimal number of clusters. CVIs can be classified into three categories: external, relative, and internal [9,10]. External indices, such as purity and entropy, assess clustering by comparing results to external benchmarks when available [11]. Relative indices compare clustering outcomes generated by the same algorithm under different parameter settings [12]. Internal indices, which are the most commonly used, focus on the dataset's geometric structure and evaluate clusters based on factors like compactness, separation, and connectivity [13]. Among these, popular measures like the Dunn index, Silhouette score, Davies–Bouldin index, Xie–Beni (XB) index, and Fukuyama–Sugeno (FS) index are frequently used to assess clustering quality by balancing intra-cluster compactness and inter-cluster separation. Fuzzy clustering techniques, however, face challenges in dealing with complex, overlapping, or irregular structures. The traditional centroid-based calculations may not effectively differentiate the underlying geometry of clusters. This issue can be addressed more effectively through an overlap-separation measure, which utilizes fuzzy membership degrees to provide a more refined evaluation of cluster quality. Building on this concept, we propose minimizing a new least-squares objective function that incorporates these measures.

In this paper, we present a novel clustering technique that incorporates fractional derivatives into the fuzzy C-means framework, referred to as fractional fuzzy C-means (Frac-FCM). This approach aims to improve the optimization of data partitioning by minimizing the least-squares objective function, focusing on the role of membership degrees. By utilizing fractional differentiation of order α, Frac-FCM integrates infinite memory effects, enhancing the regularity of the model and providing a more accurate representation of real dynamical systems. The core objective of this work is to advance a new fuzzy clustering model through the incorporation of a fractional derivative of order α and fuzzy logic. We achieve this by redefining the update rules for cluster centers and membership degrees, employing the particular Grünwald–Letnikov fractional derivative definition in the range $\alpha \in (0, 1)$. This methodological innovation captures non-local interactions that conventional derivatives are unable to represent effectively. The calculation from centroid-based measures often falls short in differentiating the geometric structures of clusters, as they primarily focus on compactness and separation. This limitation is best addressed by the overlap-separation measure, which employs an aggregate operation of fuzzy membership degrees, providing a more nuanced assessment of cluster quality. Additionally, a genetic algorithm (GA) has been proposed to optimize fuzzy partitions, enhancing criteria based on geometric cluster structures with particular emphasis on the Fukuyama–Sugeno and Xie–Beni indices. For another key consideration, to identify appropriate fuzzy clusters with-

out prior knowledge of the fractional derivative order, fuzziness, or the number of clusters, we propose an optimization model designed to enhance the performance of Frac-FCM.

The rest of the document is organized as follows: Section 2 details related work. Section 3 presents the main theoretical elements required to develop our method. Section 4 describes our model. Section 5 introduces the optimization method. Section 6 provides experimental results. Section 7 outlines some conclusions and prospects.

2. Related Works and Contributions

Clustering is a fundamental technique in unsupervised learning, widely applied across various domains. Among the many clustering algorithms developed, the fuzzy C-means algorithm stands out due to its ability to handle uncertainty in datasets. Developed as a fuzzy extension of the K-means algorithm by Dunn and refined over the years by Bezdek, FCM minimizes a fuzzy version of the least squares error criterion, allowing each data point to belong to multiple clusters with varying degrees of membership [14]. This flexibility enables FCM to perform better than K-means, especially in overlapping clusters [15]. However, FCM also inherits some of the drawbacks of K-means, such as sensitivity to noise and the tendency to converge to local minima, which can lead to sub-optimal clustering results [16].

To address these limitations, several variants of FCM have been proposed over the years. Notable among these is possibilistic C-means (PCM) [17], which introduces a degree of membership independent of the cluster centers [18]. Numerous attempts have been made to improve results by combining FCM with the possibilistic approach [19–21], recognizing the need to apply membership and typicality values simultaneously [22]. Additionally, the generalized fuzzy C-means (GFCM) method extends the basic FCM model to accommodate a wider range of data distributions [23]. A new kernel-based fuzzy clustering method was proposed to address arbitrary cluster shapes and extract expressive features [24]. This approach reduces challenges of high-dimensional data clustering. multiple kernel fuzzy clustering (MKFC) expanded the fuzzy C-means algorithm with multiple kernel-learning settings [25], offering better resistance to useless features and kernels [26]. These adaptations have significantly improved the robustness and applicability of FCM in various contexts.

Despite these advancements, the challenge of avoiding local minima remains a critical issue in FCM and other clustering algorithms. To overcome this, researchers have turned to evolutionary algorithms (EAs). EAs, including genetic algorithms (GAs), particle swarm optimization (PSO), and ant colony optimization (ACO), are particularly advantageous in FCM clustering due to their ability to escape local minima and find global optima. The adaptability and exploratory nature of EAs make them a powerful complement to FCM, enhancing its performance in complex and multidimensional datasets.

One of the most prominent EAs applied to clustering is particle swarm optimization, proposed by Kennedy and Eberhart [27]. Its versatility and simplicity have made it a valuable tool in numerous applications [28]. Silva et al. combined FCM with an advanced version of PSO, which dynamically adjusts PSO parameters during execution to achieve a better balance between exploration and exploitation [29]. Similarly, ant colony optimization has been effectively combined with FCM to address challenges like sensitivity to initialization and local minima [30]. Kernelized FCM was integrated with hybrid ACO to improve clustering in ECG data classification [31]. A dynamic fuzzy clustering approach combines ACO with a genetic algorithm, improving convergence and precision, particularly in image segmentation tasks [32]. The artificial bee colony algorithm, introduced by Karaboga [33], has been applied to a wide range of optimization problems, including data clustering, where it has shown promise in handling high-dimensional and noisy datasets. FCM was embedded into the scout bee phase of the ABC algorithm, which significantly improves the quality of clustering results [34]. A novel hybrid method, IABCFCM, merges an advanced ABC with FCM, aiding the clustering process in avoiding local minima and delivering superior outcomes on established datasets [35]. Furthermore, the SFCM-MeanABC tech-

nique leverages spatial information in fuzzy clustering, particularly for medical image segmentation [36].

Another well-studied evolutionary approach is the genetic algorithm, which has been extensively applied to FCM clustering to address its inherent limitations. One optimization method involved the parameters of a subtractive clustering algorithm through an iterative search, combined with GA, to determine the optimal number of clusters and the appropriate weighting exponent for FCM. It results in improved clustering accuracy [37]. However, while subtractive clustering provides a good estimate of the number of clusters, it may struggle with complex or overlapping cluster shapes, and its performance is highly dependent on parameters like the radius for excluding points near cluster centers. A multi-objective genetic algorithm approach optimizes fuzzy partitions by simultaneously targeting the overlap-separation measure and the fuzzy JM index. This method represents cluster centers as real-coded values and varies the number of clusters, enabling the GA to identify optimal solutions for different cluster shapes [38]. Yet, optimizing multiple objective functions at once results in trade-offs between goals, which adds complexity that may make it difficult to interpret. Furthermore, the kernel-based fuzzy C-means (KFCM) algorithm was optimized through a hybrid approach that combines improved genetic algorithms with kernel techniques, resulting in the GAKFCM algorithm. Despite the fact that GAs are used to improve the initial clustering centers, poor initialization strategies within the GA can still affect the clustering performance, potentially leading to sub-optimal solutions [39].

The primary contribution of this paper is the introduction of an innovative method for addressing the fuzzy clustering problem by utilizing fractional derivatives. These derivatives differ from conventional ones by taking advantage of non-local features and the "infinite memory" effect, offering a more precise representation of real system dynamics. Additionally, we propose a genetic algorithm that optimizes fuzzy partitions by concurrently improving criteria based on geometric cluster structures, particularly using Fukuyama–Sugeno and Xi–Beni indices. This method is designed to yield superior results by: (i) improving search space exploration through the GA, (ii) adapting multiple variables with the aid of fractional derivatives to better reflect the system's behavior, and (iii) determining optimal clustering solutions that can accommodate diverse cluster shapes, including well-separated, hyper-spherical, and overlapping clusters. Initially, we developed the fractional fuzzy C-means (Frac-FCM) algorithm, which incorporates a fractional derivative into the FCM minimization process, specifically using the Grünwald–Letnikov definition with an order of $\alpha \in (0,1)$. This captures small variations in the data structure and models complicated relationships within the data to better differentiate between clusters that overlap and address non-linear boundaries. Following this, a genetic algorithm is employed to optimize the order α along with other key parameters, guided by clustering quality measures in a newly formulated objective function. Its capacity to investigate several solutions simultaneously enhances global search efficiency, minimizes the risk of local optima, and speeds up convergence. This optimization substantially enhances the effectiveness of the proposed Frac-FCM method. To assess the performance of our genetic algorithm optimization, we tested it on various academic datasets. We focused on identifying optimal values and evaluating performance through several metrics, including the XB and FS indices. These metrics offered a thorough assessment of the algorithm's effectiveness. Additionally, we evaluated the Frac-FCM method with other well-known clustering techniques such as K-means [40], self-organizing maps (SOMs) [41], FCM [6], and Gaussian mixture models (GMMs) [42]. The comparison utilized the Silhouette and Dunn indices to analyze the performance of the Frac-FCM algorithm against these established methods. This detailed evaluation aimed to provide a clear understanding of how the Frac-FCM method performs relative to other popular clustering approaches. Friedman tests were performed on the cluster evaluation findings to determine the statistical significance of Frac-FCM's quality improvement. The Friedman

test results show that Frac-FCM produces much better quality clusters than the other existing approaches.

3. Preliminaries

In this section, we give the main tools that we combine in next sections to build our original model, namely fractional fuzzy C-means (Frac-FCM). In this regard, we give the classical fuzzy C-means, the fractional calculus, and the kernel version of the used genetic algorithm.

3.1. Fuzzy C-Means

The fuzzy mean square clustering algorithm, known as fuzzy C-means, allows one sample of data to belong to each cluster with different degrees; this method is frequently used in pattern recognition [43]. The objective function of the FCM method consists of minimizing the following error:

$$J_m(u,c) = \sum_{i=1}^{N}\sum_{j=1}^{N_{clus}} u_{ij}^m \|\mathbf{x}_i - c_j\|^2. \tag{1}$$

where the real value m determines the fuzziness of the generated clusters ($m > 1$), N is the data size, N_{clus} is the number of clusters, u_{ij} is the degree of membership of sample $\mathbf{x}_i$ to the cluster j, c_j is the d-dimensional center of cluster j, and $\|\cdot\|$ represents any norm denoting the similarity between each measured datum and the centers.

An iterative optimization process is employed to achieve fuzzy partitioning by minimizing the objective function presented above. This objective function quantifies the distance between each data point and its corresponding cluster center, with the distance being weighted by the data point's membership in the cluster [44]. The updates of membership u_{ij} and the cluster centers c_j are given by:

$$u_{ij} = \frac{1}{\sum_{k=1}^{N_{clus}} \left(\frac{\|\mathbf{x}_i - c_j\|}{\|\mathbf{x}_i - c_k\|}\right)^{\frac{2}{m-1}}}; \quad c_j = \frac{\sum_{i=1}^{N} u_{ij}^m \mathbf{x}_i}{\sum_{i=1}^{N} u_{ij}^m} \tag{2}$$

This iteration will stop when $\max_{ij}\{|u_{ij}^{(k+1)} - u_{ij}^{(k)}|\} < \varepsilon$, where ε is a termination criterion ranging between 0 and 1, whereas instances of k are the iteration steps. This procedure converges to a local minimum or a saddle point of J_m [45].

3.2. Fractional Calculus (FC)

Fractional calculus (FC), an extension of classical mathematics, deals with the determination of derivatives where the order is fractional. Initially introduced by mathematicians like Euler, Laplace, and Fourier, it gained significance for its ability to accommodate a wide range of values. FC has found applications across various fields, including economics [46], physics [47], and chemistry [48]. It is particularly useful in analyzing nonlocal continuum models with power-law non-locality as it operates with both fractional derivatives and integrals. Known for its ability to describe anomalous diffusion, long memory processes, and similar phenomena [49], fractional calculus enhances traditional models by incorporating memory effects into functions, offering a natural extension beyond ordinary derivatives.

Since its inception, extending the concept of derivatives to non-integer orders α has led to numerous approaches. As a result, several alternative definitions of fractional derivatives have emerged in the literature. Among the most commonly used are the Grünwald–Letnikov, Riemann–Liouville, and Caputo definitions in fractional calculus [50]. Among the three definitions, the more generally applied derivative and the most straightforward is clearly the method based on the Grünwald–Letnikov fractional derivative [51], which is derived from the limit of the integer order difference, involving taking a weighted

sum of function values at equally spaced points, with the weights being determined by the order of the derivative and the spacing between the points. It significantly impacts the numerical calculation. Utilizing the idea of short memory is very helpful in the investigation of non-local events in physics and engineering. It also has applications in the disciplines of signal processing, economics, and biology and becomes part of the FC application.

The Grünwald–Letnikov fractional derivatives extend the concept of the standard differentiation of functions $f(x)$ with non-integer order n in both forward and backward forms, where the fractional derivative at a point x is defined as a limit of a sum involving the function values at nearby points. Given a step size h and a centered at the point x, the forward and backward finite differences of $f(x)$ of order n are, respectively, expressed as:

$$\Delta_h^n f(x) = \sum_{k=0}^{n} (-1)^k \binom{n}{k} f(x + (n-k)h),$$
$$\nabla_h^n f(x) = \sum_{k=0}^{n} (-1)^k \binom{n}{k} f(x - kh).$$

These formulas involve a sum over an infinite number of terms, but in practice, a finite number of terms are used to approximate the derivative. The binomial coefficient $\binom{n}{k}$ shows the number of different options there are to select k items from a set of n items. The choice of n defines the derivative's order, and non-integer values of n allow for fractional differentiation.

Notice that:

$$\Delta_h^n f(x) = (-1)^n \nabla_{-h}^n f(x)$$

The infinite series yields the following results in determining the difference of a fractional order $\alpha > 0$:

$$\nabla_h^\alpha f(x) = \sum_{k=0}^{\infty} (-1)^k \binom{\alpha}{k} f(x - kh), \tag{3}$$

where the definition of the binomial coefficients is:

$$\binom{\alpha}{k} = \frac{\Gamma(\alpha+1)}{\Gamma(k+1)\Gamma(\alpha-k+1)}$$

and $\Gamma(x)$ is the Euler gamma function

$$\Gamma(x) = \int_0^\infty t^{x-1} \mathrm{e}^{-t}\, \mathrm{d}t,$$

an essential function in fractional calculus, as it expands the factorial to real inputs. It is feasible to confirm that $\Gamma(x+1) = x\Gamma(x)$; therefore, considering $\Gamma(1) = 1$, it follows that:

$$\Gamma(n+1) = n! \quad \text{for any } n \in \mathbb{N}.$$

For $h > 0$, the differential in (3) is referred to as the left-sided fractional difference, while for $h < 0$, it is known as the right-sided fractional difference. Given any bounded function $f(x)$ and $\alpha > 0$, the series in (3) is guaranteed to converge absolutely and uniformly.

- Grünwald–Letnikov Fractional Derivative

To define the Grünwald–Letnikov fractional derivatives, the forward and backward finite differences mentioned earlier are extended by replacing n with a fractional order $\alpha > 0$. Consequently, h^n becomes h^α, and the finite-order difference ∇_h^n is substituted by the fractional-order difference ∇_h^α.

The left- and right-sided Grünwald–Letnikov derivatives of order $\alpha > 0$ are defined as follows:

$$ {}^{GL}D^{\alpha}_{x\pm}f(x) = \lim_{h\to 0+} \frac{\nabla^{\alpha}_{\pm h} f(x)}{|h|^{\alpha}}. \tag{4}$$

The Grünwald–Letnikov derivatives for integer orders $\alpha = n \in \mathbb{N}$ is presented as follows:

$$ {}^{GL}D^{n}_{x\pm}f(x) = (-1)^{n} D^{n}_{x} $$

- Grünwald–Letnikov Fractional Integral

Notably, the Equation (3) can be applied for $\alpha < 0$ [52], and Equation (4) establishes the Grünwald–Letnikov fractional integral under the condition:

$$ |f(x)| < c(1+|x|)^{-u}, \quad u > |\alpha|. $$

Essentially, the Grünwald–Letnikov fractional integral provides a unified definition of both fractional derivatives and integrals, which enables a unified approach to the integral elasticity and the fractional gradient.

In conclusion, we may provide the definition of Grünwald–Letnikov, where h is the time increment by:

$$ {}^{GL}D^{\alpha}f(x) = \lim_{h\to 0}\left[\frac{1}{h^{\alpha}}\sum_{k=0}^{+\infty}\frac{(-1)^{k}\Gamma(\alpha+1)f(x-kh)}{\Gamma(k+1)\Gamma(\alpha-k+1)}\right] \tag{5}$$

This expression exposes a crucial fact that the fractional derivatives require an infinite number of terms that carry an implicit memory of all previous occurrences and possess inherent memory capacity.

The set of functions that take into account the Grünwald–Letnikov definition of the fractional derivative is very constrained as it only includes functions that are $(m+1)$ continuously differentiable. Nevertheless, the majority of real-world problems in fields like continuous physical, chemical, and additional systems involve very smooth functions, which fall within this restricted class. Despite this, the Grünwald–Letnikov fractional derivative, which is defined as a limit of a backward difference of fractional order, is not convenient to manipulate.

To obtain an elegant and straightforward statement expression, it is preferable if the integral is present in it:

$$ {}^{GL}D^{\alpha}f(x) = \begin{cases} \frac{1}{\Gamma(m-\alpha)}\frac{d^{n}}{dx^{m}}\int_{x}^{\infty} t^{m-\alpha-1}f(t)\,dt & m-1 \le \alpha < m \\ \frac{\mathrm{d}^{m}f(t)}{\mathrm{d}t^{m}} & \alpha = m \end{cases} $$

This equation, often known as the Riemann–Liouville definition, is one of the most well-known definitions, which is equivalent to the Grünwald–Letnikov definition (3) assuming we take into account a class of functions $f(t)$ that have $(m+1)$ continuous derivatives for $t > 0$ [53].

Expression (5) is often approximated by the following discrete time implementations equation:

$$ {}^{GL}D^{\alpha}f(x) = \frac{1}{N^{-\alpha}}\sum_{k=0}^{m}\frac{(-1)^{k}\Gamma(\alpha+1)f(x-kh)}{\Gamma(k+1)\Gamma(\alpha-k+1)} \tag{6}$$

where N is the sampling period and m is the truncation order.

Depending on the Grünwald–Letnikov definition [54], the discrete approximation of the fractional derivative has been designed and adopted to solve the fractional advection-dispersion issues [55]. Likewise, under this formula, the finite difference method was developed and is commonly utilized to solve a variety of fractional differential equa-

tions [56–58]. By slightly altering this definition, the periodicity of the fractional derivative of a non-constant periodic function can be maintained [59]. Due to its built-in memory, mathematics has used this powerful tool to solve some problems that could not be solved in the traditional sense. Additionally, the Grünwald–Letnikov fractional derivative was used in every application that preprocessed with a real fractional derivative. The Grünwald method achieves a good balance between precision and computing time, whereas the integrator method is faster and has medium precision.

3.3. Genetic Algorithm

Genetic algorithms are among the most prominent population-based optimization techniques, renowned for their effectiveness in solving complex combinatorial optimization problems [60]. By drawing inspiration from natural selection and genetic principles, GAs offer robust solutions that often outperform traditional optimization methods. These algorithms evolve populations through iterative processes, applying genetic operators such as selection, mutation, recombination, and crossover. The adaptive nature of GAs allows them to modify search strategies dynamically, using crossover and mutation probabilities to converge on optimal solutions. Their ability to encode, evaluate, and generate multiple potential solutions rapidly enhances their global search capabilities, reducing the risk of getting trapped in local optima.

The GA procedure operates on a population of potential solutions, evolving them over successive generations to identify the optimal solution. Figure 1 illustrates the cycle of a genetic algorithm, showcasing its key phases and processes. In genetic algorithms, a "chromosome" represents a candidate solution encoded as an array of parameter values. For a problem with N dimensions, the chromosome is an N-element array ($[p_1, p_2, \ldots, p_N]$, where p_i represents the value of the i-th parameter). The core components and operations of a genetic algorithm include:

- Population initialization: The genetic algorithm starts by randomly initializing a population of potential solutions, called chromosomes. This initial population is denoted as $P_0 = \{x_1, x_2, \ldots, x_{p_s}\}$, where p_s represents the population size. Each chromosome is then evaluated by a fitness function to assess its effectiveness in solving the problem.
- Fitness function: Each solution x_i is assessed with a fitness function $f(x_i)$, which measures how well the solution performs or meets the desired criteria. This function is essential for directing the evolutionary process towards finding optimal solutions:

$$f(x_i) = \text{ObjectiveFunction}(x_i)$$

  The selection process favors individuals with higher fitness values for reproduction.
- Selection: This involves choosing individuals from the current population for reproduction based on their fitness and a defined probability distribution (selection function). This process creates a mating pool for the next generation. There are various methods of selection: for example:
  Roulette wheel selection: the likelihood of selecting an individual x_i is proportional to its fitness $f(x_i)$:

$$P(x_i) = \frac{f(x_i)}{\sum_j f(x_j)}$$

  Tournament selection: a random subset of individuals is selected, and the fittest individual within this subset is chosen for reproduction.
- Crossover (recombination): This involves merging genetic material from two parent individuals to create offspring. This process is controlled by a crossover probability p_c. For example, in a single-point crossover:

$$\text{Offspring}_1 = \text{Parent}_1[1:k] + \text{Parent}_2[k+1:N]$$

$$\text{Offspring}_2 = \text{Parent}_2[1:k] + \text{Parent}_1[k+1:N]$$

where k denotes the crossover point and N represents the chromosome length. This technique blends genetic traits from both parents, aiming to combine beneficial features and produce high-quality offspring.

- Mutation: This introduces random changes to the offspring to preserve genetic diversity and prevent premature convergence. This process is controlled by a mutation probability p_m. In the case of binary-encoded solutions, mutation typically involves flipping a bit (changing a 0 to a 1, or vice versa).

$$x_i' = \begin{cases} \text{flip}(x_i[j]), & \text{if random} < p_m \\ x_i[j], & \text{otherwise} \end{cases}$$

where $x_i[j]$ denotes the j-th gene of the individual x_i.
After crossover and mutation, the fitness of the offspring is evaluated using the fitness function. The population is then updated by replacing some or all individuals with the new offspring, ensuring that the population evolves towards better solutions. This iterative process continues until specific convergence criteria are met.

- Convergence criteria: These define when the genetic algorithm is considered to have reached an optimal solution. Typically, convergence is achieved when either the maximum number of generations is reached (generation $t \geq M_{gnr}$) or when a predefined fitness threshold is met ($\max(f(x_i)) \geq f_{\text{threshold}}$).

Fine-tuning parameters like population size p_s, mutation probability p_m, and crossover probability p_c is essential for optimizing GAs for specific problem domains. Convergence analysis further aids in understanding the effectiveness of GAs by assessing metrics like convergence speed and quality. GAs are versatile and find applications across diverse fields, including engineering design [61], scheduling [62], financial modeling [63], and bioinformatics [64], making them valuable tools for continuous and discrete optimization challenges.

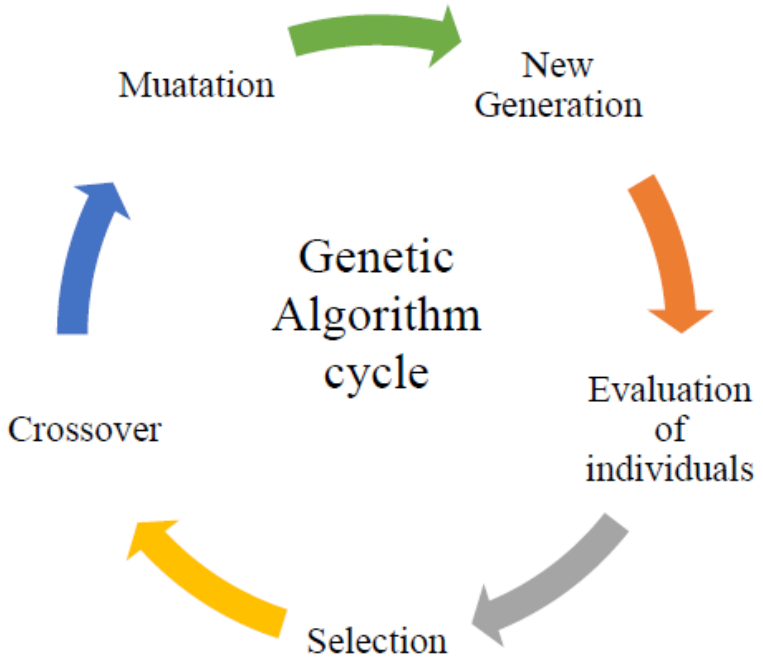

Figure 1. Genetic algorithm cycle.

4. Frac-FCM: The Proposed Fractional Fuzzy C-Means Model

This section introduces the proposed fractional fuzzy C-means (Frac-FCM) method, a novel clustering model that integrates fractional derivatives into the fuzzy C-means minimization step. By leveraging fractional calculus, the Frac-FCM method can capture long-term dependencies and provide smoother variations, enhancing clustering performance, particularly in scenarios involving overlapping clusters and memory effects. The approach is inspired by the established concept of using a fractional derivative of order $\alpha \in (0, 1)$, specifically the Grünwald–Letnikov fractional derivative, to replace the first-order derivative. This substitution improves the algorithm's computational efficiency and benefits from memory effects. Due to their non-local nature and the "infinite memory" effect, fractional-order systems have demonstrated superior accuracy in modeling the behavior of real-world dynamic systems compared with traditional approaches. Additionally, a genetic algorithm is employed to optimize the fractional derivative order α along with other key parameters, further advancing the effectiveness of the proposed Frac-FCM model.

In the fuzzy C-means algorithm, ensuring a robust assignment of membership for the sample indexed by i to a single cluster involves incorporating a set of linear constraints that guide the membership values. Specifically, for all $i = 1, \ldots, N$, we define the following constraint:

$$H_i(u) = \sum_{j=1}^{N_{clus}} u_{ij} - 1 = 0; \tag{7}$$

After constraining the non-linear optimization procedure, we arrive at the following:

$$(P) : \begin{cases} \min J_m(u, c) = \sum_{i=1}^{N} \sum_{j=1}^{N_{clus}} u_{ij}^m \|\mathbf{x}_i - c_j\|^2 \\ \quad \text{Subject to: } \sum_{j=1}^{N_{clus}} u_{ij} = 1, \quad \forall i = 1, \ldots, N \\ \qquad\qquad u_{ij} \geq 0, \quad \forall i = 1, \ldots, N, \quad \forall j = 1, \ldots, N_{clus} \end{cases}$$

We reformulated the update process by adding a novel fractional derivative. First, we applied gradient descent and projected gradient descent [65] to update the center variable and the membership, respectively, expressing the right-hand side with two sequential terms. Then, we incorporated the Grünwald–Letnikov definition into both the center and membership update formulas.

4.1. Centers Tuning Rule

As previously mentioned, the update procedure incorporates the Grünwald–Letnikov derivative. To perform the center update for each training example x_i, we begin by computing the gradient descent concerning the center c_j^{k-1} (at the $(k-1)$-th iteration). The resulting equation for the center vector is as follows:

$$c_j^k = c_j^{k-1} + 2\beta \sum_{i=1}^{N} (u_{ij}^{k-1})^m (\mathbf{x_i} - c_j^{k-1}).$$

This expression can be rewritten as:

$$c_j^k - c_j^{k-1} = 2\beta \sum_{i=1}^{N} (u_{ij}^{k-1})^m (\mathbf{x_i} - c_j^{k-1}).$$

The left side $c_j^k - c_j^{k-1}$ is the discrete version of the derivation of order $\alpha = 1$ in (6) if we assume that $N = 1$. This results in the next expression:

$${}^{GL}D^{\alpha}[c_j^k] = 2\beta \sum_{i=1}^{N} (u_{ij}^{k-1})^m (\mathbf{x_i} - c_j^{k-1}), \tag{8}$$

where the derivative's order can be extended to a real number $0 \leq \alpha \leq 1$, resulting in a smoother variation and longer memory effects. Therefore, Equation (8) can be solved by considering the first three terms of the derivative as:

$$c_j^k - \alpha c_j^{k-1} + \frac{1}{2}\alpha(\alpha-1)c_j^{k-2} - \frac{1}{6}\alpha(\alpha-1)(\alpha-2)c_j^{k-3} = 2\beta\sum_{i=1}^{N}(u_{ij}^{k-1})^m(\mathbf{x_i} - c_j^{k-1})$$

Accordingly, the center update of the Frac-FCM at the (k)-th iteration is given by:

$$c_j^k = \alpha c_j^{k-1} - \frac{1}{2}\alpha(1-\alpha)c_j^{k-2} + \frac{1}{6}\alpha(1-\alpha)(2-\alpha)c_j^{k-3} + 2\beta\sum_{i=1}^{N}(u_{ij}^{k-1})^m(\mathbf{x_i} - c_j^{k-1}) \quad (9)$$

4.2. Memberships Tuning Rule

Repeatedly, to update the membership matrix for each training example, the Grünwald–Letnikov definition is implemented in the process. We start by using the projected gradient descent to deal with the linear constraints $H_i(u)$, and the projected descent algorithm is described below:

- Define the feasible set: the feasible set is determined by the constraint; in our case, the feasible set is defined as the set of u_{ij} values that satisfy the constraint H_i;

$$\mathrm{F} = \{u_{ij} : \ u_{ij} \geq 0, \text{ and } \sum_{j=1}^{N_{clus}} u_{ij} = 1; \ \forall i = 1, \ldots, N, \ \forall j = 1, \ldots, N_{clus}\}.$$

- Define the projection operator (P) by taking the current values of u_{ij} and project them onto the feasible set (F). Projecting u_{ij} to satisfy the constraint directly may not be straightforward; one way to accomplish this is to use a projection method that enforces the constraint indirectly as

$$P_F(u_{ij}) = \max\left(0, \frac{u_{ij}}{\sum_{j=1}^{N_{clus}} u_{ij}}\right). \quad (10)$$

This projection formula ensures that the updated membership values u_{ij} for each data point i meet both the sum-to-one and non-negativity constraints, thereby maintaining the validity of the membership values throughout the optimization process.

First, project the existing solution back onto the constraint set after applying a gradient descent on it. This can be stated as:

$$u_{ij}^k = u_{ij}^{k-1} - \lambda.m(u_{ij}^{k-1})^{m-1}\|\mathbf{x}_i - c_j^{k-1}\|^2.$$

This expression can be rewritten as:

$$u_{ij}^k - u_{ij}^{k-1} = -\lambda.m(u_{ij}^{k-1})^{m-1}\|\mathbf{x}_i - c_j^{k-1}\|^2. \quad (11)$$

Likewise, by assuming $N = 1$, the left side $u_{ij}^k - u_{ij}^{k-1}$ is the discrete version of the derivation of order $\alpha = 1$ in (11). It provides the expression below:

$${}^{GL}D^{\alpha}[u_{ij}^k] = -\lambda.m(u_{ij}^{k-1})^{m-1}\|\mathbf{x}_i - c_j^{k-1}\|^2.$$

Considering the first $m = 3$ terms of differential derivative, yielding

$$u_{ij}^k - \alpha u_{ij}^{k-1} + \frac{1}{2}\alpha(\alpha-1)u_{ij}^{k-2} - \frac{1}{6}\alpha(\alpha-1)(\alpha-2)u_{ij}^{k-3} = -\lambda.m(u_{ij}^{k-1})^{m-1}\|\mathbf{x}_i - c_j^{k-1}\|^2,$$

where α is a constant in the interval $[0,1]$, u_{ij}^{k} represents the membership at iteration k, u_{ij}^{k-1} is that at iteration $k-1$, and so on. From the last equation, the membership update is given below:

$$u_{ij}^{k} = \alpha u_{ij}^{k-1} - \frac{1}{2}\alpha(1-\alpha)u_{ij}^{k-2} + \frac{1}{6}\alpha(1-\alpha)(2-\alpha)u_{ij}^{k-3} - \lambda.m(u_{ij}^{k-1})^{m-1}\|\mathbf{x}_i - c_j^{k-1}\|^2. \quad (12)$$

Apply the projection operator P_F to the obtained u_{ij}^{k} as $u_{ij}^{k} = P_F(u_{ij}^{k})$.

5. Optimized Fractional Fuzzy C-Means Method

The suggested fractional fuzzy C-means (Frac-FCM) clustering algorithm extends the FCM algorithm by allowing data points to be assigned to multiple clusters with varying degrees of membership. The update rules for cluster centers c_j and membership values u_{ij} rely on five different variables: derivative order α, gradient descent rates β and λ, fuzziness m, and number of clusters N_{clus}. Controlling these factors is necessary to increase Frac-FCM's performance. This method is beneficial for handling overlapping clusters, including uncertainty in cluster allocations, and for avoiding local optima.

The optimized method aims to determine the optimal variables $\omega = (\alpha, \beta, \lambda, m, N_{clus})$ to advance the Frac-FCM model by formulating an objective function that incorporates clustering validity indices. The objective function utilized in the optimization process is based on minimizing the cost to identify suitable variables $\omega^* = (\alpha^*, \beta^*, \lambda^*, m^*, N_{clus}^*)$. This function relies on clustering quality measures optimized through the genetic algorithm. To enhance performance, we evaluate metrics such as the Xie–Beni (XB) index and the Fukuyama–Sugeno (FS) index. These indices assess the compactness and separation of clusters: the XB index emphasizes the compactness within clusters and the separation between them, while the FS index offers a geometric perspective by considering the overall centroid. By combining these measures, we leverage their strengths to ensure a more robust evaluation of clustering validity.

5.1. GA-Frac-FCM: Mathematical Model

In this section, we give the mathematical basis of the proposed method. It is an optimization model for which we indicate the variables, the constraints, and the objective function.

- Variables

  The variables of the optimized method are as follows:

· α: The derivative order.
· β, λ: The gradient descent rates.
· m: The fuzziness.
· N_{clus}: The number of clusters.

- Objective function

The objective function of the optimized method builds upon the combination of the FS index with the XB index. The function is defined as follows:

$$E(\alpha, \beta, \lambda, m, N_{clus}) = V_{FS} + V_{XB}.$$

Therefore:

$$E(\alpha, \beta, \lambda, m, N_{clus}) = \left(1 + \frac{1}{N \cdot \theta_{p,q}}\right) \sum_{i=1}^{N}\sum_{j=1}^{N_{clus}} u_{ij}^{m}\|x_i - c_j\|^2 - \sum_{i=1}^{N}\sum_{j=1}^{N_{clus}} u_{ij}^{m}\|c_j - \hat{c}\|^2 \quad (13)$$

where: $\theta_{p,q} = \min_{1\le p\neq q\le N_{clus}} \|c_p - c_q\|^2$ and $\hat{c} = \frac{1}{N_{clus}}\sum_{j=1}^{N_{clus}} c_j$.

- Constraints

The constraints involve determining the lower and upper bounds for each variable. The derivative order of the Grünwald–Letnikov definition is between 0 and 1. Thus, we have the following constraint:

$$0 \leq \alpha \leq 1. \tag{14}$$

The gradient descent rates are positive real values, generally in the range of $[0.01, 1]$. Therefore:

$$0 < \beta \leq 1, \text{ and } 0 < \lambda \leq 1. \tag{15}$$

The real value m determines the fuzziness of the generated clusters ($m > 1$). The lower bound ensures that the clustering remains soft clustering. To strike a balance between the interpretability of clusters and the level of fuzziness, we choose the upper bound as $m_{ub} = 2 + \frac{N}{N_{clus}}$ to increase it with higher dimensionality and decrease it with more clusters, preserving a balance between data complexity and distinctness of clusters. This leads to the following constraint:

$$1 < m \leq 2 + \frac{N}{N_{clus}}. \tag{16}$$

The number of clusters in a dataset must be greater than one and less than or equal to N (the data size). This research imposes the following constraint:

$$1 \leq N_{clus} \leq N. \tag{17}$$

- Recap

Finally, the mixed optimization problem that gives the optimal decision is as follows:

$$(P'): \begin{cases} \min E(\alpha, \beta, \lambda, m, N_{clus}) = \left(1 + \frac{1}{N \cdot \theta_{i,j}}\right) \sum_{i=1}^{N} \sum_{j=1}^{N_{clus}} u_{ij}^{m} \|x_i - c_j\|^2 - \sum_{i=1}^{N} \sum_{j=1}^{N_{clus}} u_{ij}^{m} \|c_j - \hat{c}\|^2 \\ \quad \text{Subject to:} \\ \qquad 0 \leq \alpha \leq 1 \\ \qquad 0 < \beta \leq 1 \\ \qquad 0 < \lambda \leq 1 \\ \qquad 1 < m \leq 2 + \dfrac{N}{N_{clus}} \\ \qquad 1 \leq N_{clus} \leq N \\ \qquad \alpha, \beta, \lambda, m \in \mathbb{R}^+ \text{ and } N_{clus} \in \mathbb{N} \end{cases}$$

The traditional FCM clustering approach, which relies on the Lagrange coefficient method for optimizing cluster centers and data assignments, often struggles with sub-optimal solutions due to poor initialization and slow convergence, especially in large problem spaces. To overcome these limitations, evolutionary algorithms, such as the genetic algorithm (GA), offer a more robust solution by intelligently and randomly exploring the problem space [66]. In our study, the genetic algorithm is employed to optimize the mathematical model (P'), utilizing specific operators to ensure the discovery of the optimal solution. By integrating the genetic algorithm with clustering quality measures, we not only enhance the evaluation criteria but also introduce a novel method for determining the optimal parameters for our proposed model. The next algorithm section outlines the implementation of this process to achieve the optimal solution in the proposed approach. The diagram in Figure 2 illustrates this new technique.

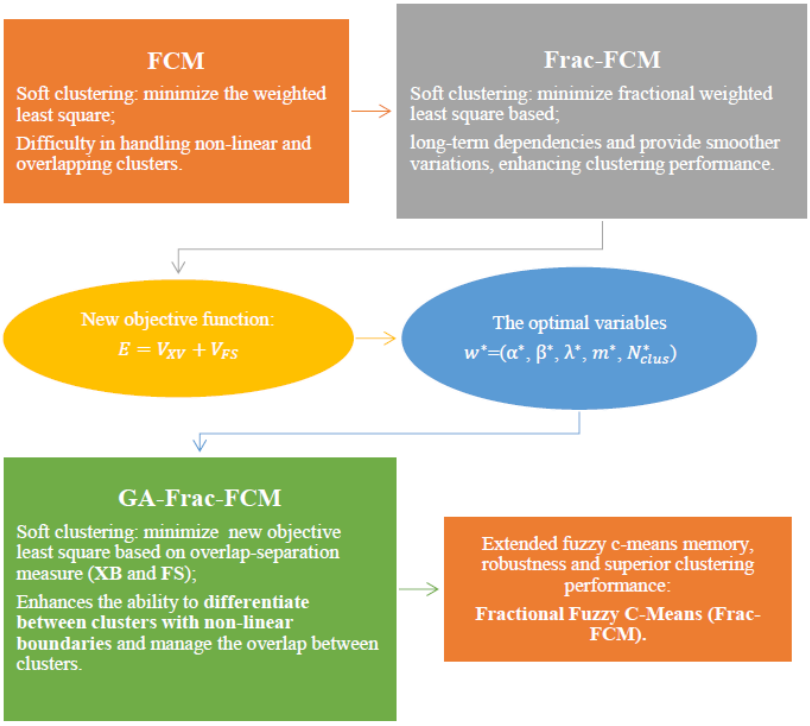

Figure 2. Methodology utilized in this paper to develop the Frac-FCM model.

5.2. GA-Frac-FCM: Algorithms

Algorithm 1 illustrates the genetic procedure proposed in this research to identify the most effective variables for developing the innovative Frac-FCM clustering model. This model integrates an objective function based on the FS and XB indices. To initiate the optimization process, Algorithm 1 generates a random population of parameters (α, β, λ, m, and N_{clus}) that meet the model's criteria (P'). The genetic algorithm systematically refines these populations through iterative processes to determine the optimal parameters and the corresponding cluster cost function for the Frac-FCM method. This involves evaluating fitness, selecting promising candidates, performing genetic operations, and updating the population accordingly. It is important to note that Algorithms 2 and 3 are integral components utilized within Algorithm 1.

Algorithm 2 represents the fitness function, which computes the fitness value corresponding to the given parameter values by evaluating the objective function. This function, utilizing the FS and XB metrics, plays a critical role in the optimization process, providing essential feedback to identify the optimal solutions.

Finally, Algorithm 3 describes the novel Frac-FCM approach; it employs a sequential update process. At each phase, this process computes new parameter values based on observations from the previous three iterations. During the initial iterations ($k = 0$, 1, and 2), the parameters are initialized using the standard fuzzy C-means method. In our approach, the first three iterations mark the initialization phase, due to utilizing the Grünwald–Letnikov fractional derivative. This methodological shift enables a more refined initialization process by leveraging the information gathered from the clustering step, leading to a more effective parameter initialization.

Algorithm 1: Pseudo Code for GA-Frac-FCM Optimization

Input: Input data A, maximum number of generations M_{gnr}, population size p_s.
Output: Optimal parameters $\omega^* = (\alpha^*, \beta^*, \lambda^*, m^*, N^*_{clus})$ for each dataset;
Optimal cluster cost function for dataset $E^* = V^*_{FS} + V^*_{XB}$.
Begin
Initialize:
Define suitable bounds for parameters: L_b, U_b;
Number of variables = 5;
$iter = 0$; set the initial population P_0 randomly;
for $iter = 1$ *to* M_{gnr} **do**
Fitness (P_{iter});
Selection ();
Crossover ();
Mutation ();
Update (P_{iter});
end
ω^* = the optimal ω that minimize $P_{M_{gnr}}$;
$E^* = Frac - FCM(\omega^*)$.

Algorithm 2: Fitness Function

Input: Derivative order α, gradient descent rates β, λ, fuzziness m, and number of cluster N_{clus}.
Output: Fitness $E(\alpha, \beta, \lambda, m, N_{clus})$.
Begin
$Frac - FCM(\alpha, \beta, \lambda, m, N_{clus})$;
Calculate the fitness function E correspond to Equation (13);
Return $E(\alpha, \beta, \lambda, m, N_{clus})$.

Algorithm 3: Frac-FCM Pseudo code

Input: Input data A, maximum number of iterations N_{iter}, derivative order α, gradient descent rates β, λ, fuzziness m, and number of cluster N_{clus}
Output: Center matrix c, membership matrix u.
Begin
Initialize:
$k = 0$, $k = 1$, and $k = 2$; set the initial parameters c and membership u using the standard FCM ;
for $k = 3$ *to* N_{iter} **do**
for $i = 1$ *to* N **do**
for $j = 1$ *to* N_{cls} **do**
Calculate the remaining parameters c^k_j, u^k_{ij} correspond to Equations (9), (12) respectively.
Apply the projection operator P_F to the new u^k_{ij} corresponding to Equation (10).
end
end
end
Return c^k_j, u^k_{ij}.

6. Experimental Results and Discussion

In this study, we utilized a variety of academic datasets to analyze the genetic algorithm optimization results, focusing on the optimal values and performance across several metrics,

including the Xie–Beni index and Fukuyama–Sugeno index. These metrics provided a comprehensive evaluation of the algorithm's effectiveness. Additionally, we compared our proposed Frac-FCM method with other popular clustering techniques, such as K-means, self-organizing maps, fuzzy C-means, and Gaussian mixture models, using the Silhouette and Dunn indices for performance comparison purposes. Also, we employed the Friedman test to assess the performance of various techniques. This extensive evaluation aimed to offer a deeper understanding of the Frac-FCM algorithm's performance relative to established clustering approaches.

6.1. Description of Datasets

To evaluate and enhance the performance of the proposed methods, we conducted experiments on a variety of datasets from different domains, each offering distinct characteristics. Specifically, we utilized several academic datasets, including Abalone, Haberman, and Pima, among others. These datasets were chosen for their unique properties, which are summarized in Table 1, which provides an overview of their specific attributes and features.

Table 1. Description of the experimental datasets.

Dataset	Features	Samples	Domain Area
Abalone	8	4177	Marine Biology
Balance	4	625	Cognitive Psychology
Concrete Compressive Strength	8	1030	Physics, Chemistry
Ecoli	7	336	Biology
Haberman	3	306	Medical
Iris	4	150	Botany
Libra	90	360	Physics
Liver	6	345	Medical
Pageblock	10	5473	Document Processing
Pima	8	768	Medical
Seed	7	210	Agriculture
Segment	16	2310	Image Processing
Statlog (Australian Credit Approval)	14	690	Business
Wine	13	178	Chemistry

6.2. Clustering Validity Measurement Indexes

Evaluating clustering results is crucial for determining the effectiveness and quality of clusters. Various indices serve to assess cluster homogeneity and separation, providing insights into the performance of clustering algorithms. Metrics such as the FS index [67] and XB index [68], featured in the clustering validity index based on the geometric information of the dataset and membership degree, are essential for measuring clustering quality, including aspects like compactness and cohesion.

- Fukuyama and Sugeno index (FS)

The FS index, introduced by Fukuyama and Sugeno, evaluates clustering validity by considering both membership degrees and the geometric distribution of the data. This index is expressed in summation form, where $J_m(u,c)$ represents the similarity measure, and $K_m(u,c)$ indicates the separation measure, which should be maximized. Therefore, a lower FS index value signifies better clustering quality. The FS index is calculated as follows in Equation (18).

$$V_{\mathrm{FS}} = J_m(u,c) - K_m(u,c) = \sum_{i=1}^{N}\sum_{j=1}^{N_{clus}} u_{ij}^m \|x_i - c_j\|^2 - \sum_{i=1}^{N}\sum_{j=1}^{N_{clus}} u_{ij}^m \|c_j - \hat{c}\|^2. \tag{18}$$

where $\hat{c} = \frac{1}{N_{clus}} \sum_{j=1}^{N_{clus}} c_j$.

- Xie and Beni index (XB)

The XB index, proposed by Xie and Beni, is a widely used clustering validity measure. It is defined as the ratio of the compactness within clusters to the minimum separation between cluster centers. A lower XB index value indicates better clustering results. The XB index can be computed as follows:

$$V_{\mathrm{XB}} = \frac{\sum_{i=1}^{N} \sum_{j=1}^{N_{clus}} u_{ij}^{m} \|x_i - c_j\|^2}{N \times \min_{p \neq q} \|c_p - c_q\|^2}. \tag{19}$$

- Silhouette index

The Silhouette index evaluates clustering quality by comparing cluster cohesion with cluster separation [69]. The Silhouette ratios are stated as follows:

$$S(i) = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \right), \tag{20}$$

where:

- $a(i)$ denotes the average distance from sample i to all other samples within the same cluster.
- $b(i)$ represents the smallest average distance from sample i to any other cluster. The Silhouette coefficient ranges from -1 to 1, where:
- A value of -1 indicates that the sample is likely assigned to the wrong cluster;
- A value of 0 suggests overlapping clusters;
- A value of 1 indicates that the sample is in a well-separated, compact cluster.

- Dunn index

The Dunn index is another metric used to evaluate clustering by considering the separation between clusters and the compactness within clusters [70]. It is defined as:

$$DI = \frac{\min_{i \neq j} \left\{ \min_{\forall x_k \in c_i, \forall x_{k'} \in c_j} \left\| x_k^{(i)} - x_{k'}^{(j)} \right\|_2 \right\}}{\max_{r=1,\ldots,c} \left\{ \max_{k \neq k'} \left\| x_k^{(r)} - x_{k'}^{(r)} \right\|_2 \right\}}. \tag{21}$$

where $d_{\min}$ is the smallest distance between data points in different clusters and $d_{\max}$ represents the largest intra-cluster distance. A higher Dunn index indicates better clustering, as it reflects well-separated and compact clusters, implying superior clustering quality.

6.3. Genetic Algorithm: Analyzing Performance and Results

The genetic algorithm is instrumental in enhancing the optimized Frac-FCM method by optimizing five distinct parameters to achieve superior clustering accuracy. By employing evolutionary principles, the GA systematically explores the solution space to identify parameter values that minimize the fitness function associated with the given optimization problem (P'). Table 2 details the parameter settings utilized in the GA. To ensure effective convergence and computational efficiency, two stopping criteria are employed: a maximum of generations (M_{gnr}) and a function tolerance criterion, which terminates the algorithm when it reaches a specified performance threshold. These criteria are designed to balance computational resource constraints with the need for accurate and reliable solutions.
The genetic algorithm's performance is visualized through several key curves that track its progress and behavior. The functions used in this work include:

- Best fitness: shows the best and mean fitness values across generations.
- Best individuals: monitors the fitness of the best individual in each generation.
- Distance average: displays the distances between individuals, indicating population diversity.
- Genealogy: visualizes the relationships between parents and offspring.

- Score diversity: represents the diversity of fitness scores within the population.
- Score: shows the distribution of fitness scores across generations.

Table 2. Genetic algorithm configuration.

GA Option	Values
Initialization	Random uniform
Population size p_s	$50 < p_s < 200$
Selection function	Stochastic uniform
Crossover function	Scattered crossover
Crossover probability p_c	$p_c = 0.8$
Mutation function	Gaussian mutation
Max generations	$M_{gnr} = 100$
Constraint tolerance	1×10^{-3}
Function tolerance	1×10^{-6}
Nonlinear constraint algorithm	Augmented Lagrangian

This section presents the optimization results and provides a comprehensive analysis of the genetic algorithm's performance. By examining the outcomes, we aim to gain insights into how the GA adapts the optimized Frac-FCM technique to the unique characteristics of the dataset. The fitness function ranks potential solutions, determining which ones will advance to the next generation. It evaluates the overall performance of the Optimized Frac-FCM model by combining FS and XB indices, focusing on the model's collective efficiency rather than individual parameters. Furthermore, the final scores include penalty values for population members that the evolutionary algorithm minimizes. This ensures that solutions not only perform optimally but also satisfy the imposed constraints. The process guides optimization by efficiently exploring the solution space while adhering to the given limitations.

In this part, we examine the behavior of our genetic algorithm by visualizing the evolution of the fitness function, which serves as the objective function for problem (P'). For each dataset, we present various performance indicators, including best and mean fitness values (top-left subplot), the most optimal individuals (top-right subplot), average distance (left-middle subplot), genealogy (right-middle subplot), score diversity histogram (left-bottom subplot), and score values (right-bottom subplot). These metrics are illustrated in Figures 3–6.

The figures offer a clear visualization of the genetic algorithm's performance during the optimization of the fractional fuzzy C-means (Frac-FCM) method. The best fitness value is achieved early and remains stable throughout the generations, indicating the algorithm quickly finds a near-optimal solution. However, the mean fitness of the population remains higher than the best fitness, highlighting a disparity between the best individual and the population as a whole. Additionally, the average distance between individuals decreases steadily across generations, signifying population convergence and reduced diversity, which is expected in later stages of optimization but may also suggest premature convergence. Furthermore, the score histogram and individual fitness show that while many individuals approach optimal fitness, some variability remains, reflecting ongoing exploration of the search space. The population history reveals that while the best individual remains stable, the other individuals fluctuate significantly, demonstrating the algorithm's exploratory capabilities. Overall, the genetic algorithm shows strong optimization performance, consistently finding effective solutions while exploring the search space to ensure well-tuned Frac-FCM parameters.

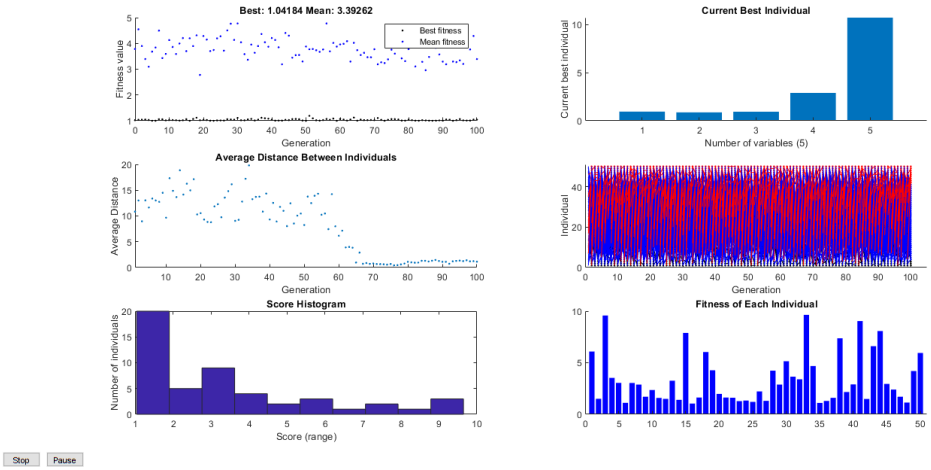

Figure 3. Genetic algorithm performance indicators for Haberman dataset optimization.

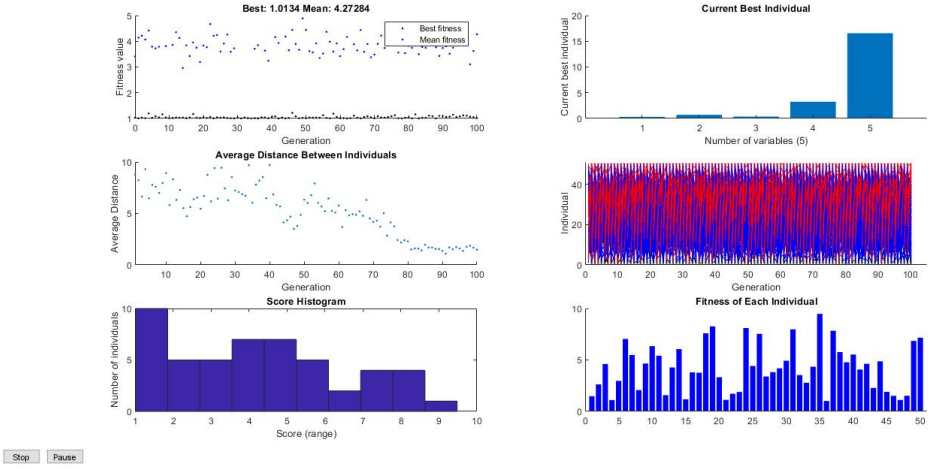

Figure 4. Genetic algorithm performance indicators for Libra dataset optimization.

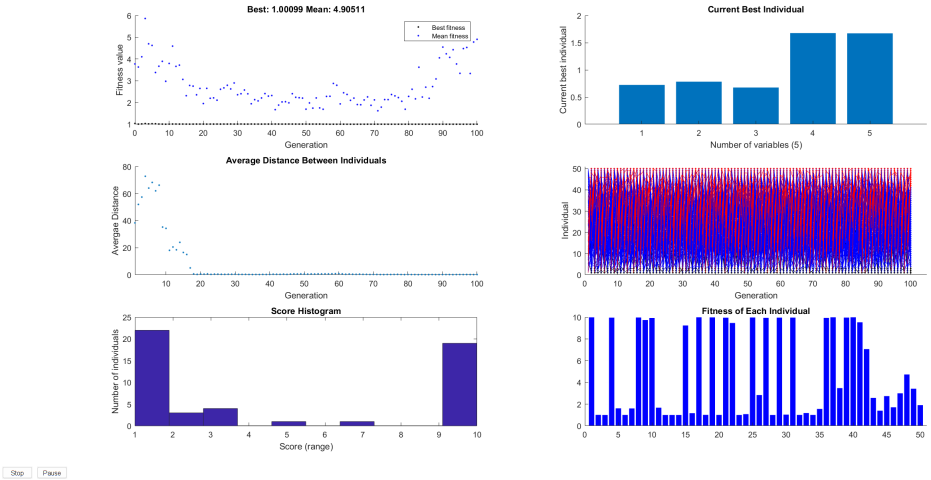

Figure 5. Genetic algorithm performance indicators for Wine dataset optimization.

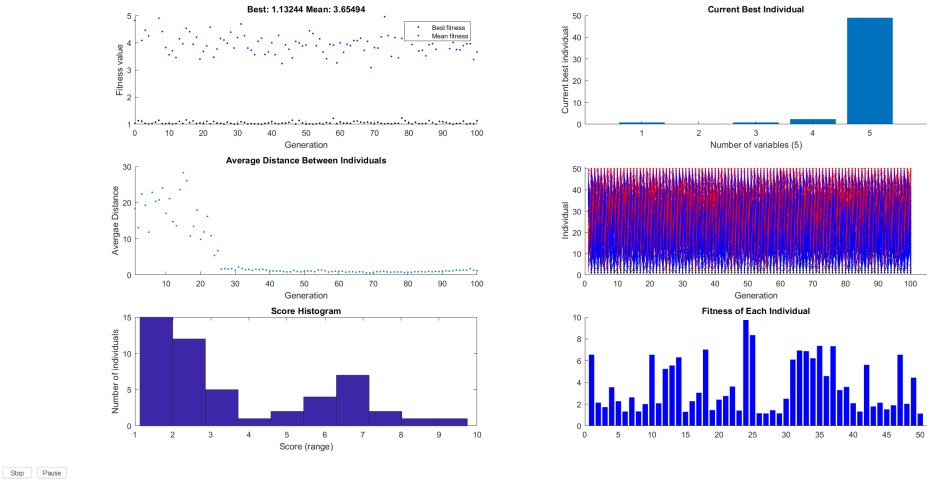

Figure 6. Genetic algorithm performance indicators for Pima dataset optimization.

The analysis focuses on optimizing fuzzy partitions based on geometric criteria, including model complexity and cluster separation, to enhance clustering performance across generations. Frac-FCM refines its quality by selecting the optimal fractional derivative order (α), gradient descent rates (β, λ), fuzziness parameter (m), and number of clusters (N_{clus}), as shown in Table 3. Additionally, it includes performance metrics such as the XB index (V_{XB}), which measures the compactness and separation of clusters, and the FS index (V_{FS}), indicating cluster validity. Across the datasets, the parameters vary significantly, reflecting the adaptive nature of the Frac-FCM approach to different data characteristics. The performance metrics show consistent improvements in clustering quality, as evidenced by low V_{XB} values, especially in datasets like Wine and Pageblock, which have minimal inter-cluster overlap and high intra-cluster cohesion.

In summary, the genetic algorithm efficiently determines the optimal parameters for Frac-FCM without requiring a large number of generations. The stability of the GA-Frac-FCM system is highlighted by the absence of fluctuations in the fitness curves, indicating consistent and reliable performance.

Table 3. Optimal parameters for the optimized Frac-FCM method along with associated performance metrics across various datasets.

Data Set	α^*	β^*	λ^*	m^*	N^*_{clus}	V_{XB}	V_{FS}
Abalone	0.8347	0.55588	0.46174	2.5917	22	1.8869×10^{-4}	1.0014
Balance	0.99012	0.58211	0.62506	4.9998	50	0.0071	0.0212
Concrete Compressive Strength	0.95369	0.13018	0.88148	1.7576	181	0.32236	0.77474
Ecoli	0.74087	0.81775	0.83244	4.7027	224	0.28282	0.73210
Haberman	0.9508	0.86634	0.93594	2.8985	11	6.0193×10^{-5}	1.0418
Iris	0.991	0.92502	0.75972	3.7868	17	0.6414	0.94329
Libra	0.3125	0.71263	0.36938	3.2398	17	1.1736×10^{-5}	1.0134
Liver	0.40721	0.45748	0.97339	4.6291	20	1.0059	1.8658×10^{-2}
Pageblock	0.97405	0.42475	0.61145	3.1203	5	3.1426×10^{-9}	1.0224
Pima	0.83677	0.13917	0.83677	2.3475	49	1.8048×10^{-4}	1.1323
Seed	0.98438	0.75347	0.56306	1.7567	185	0.11035	0.91225
Segment	0.37799	0.061637	0.51809	4.2317	4	0.50740	0.55712
Statlog (Australian Credit Approval)	0.016843	0.74695	0.57621	3.8614	3	2.6667×10^{-5}	1.0002
Wine	0.72374	0.78268	0.67563	1.677	2	2.4298×10^{-11}	1.0010

The number of clusters is an important parameter in clustering algorithms as it has a significant impact on the final clustering outcomes. The proposed technique, like standard FCM, cannot determine the number of clusters independently. But using a genetic algorithm

to improve parameters, including the number of clusters, effectively solves this problem. The GA explores multiple possibilities, and the optimal number of clusters is selected based on quality measures embedded in the objective function. A well-chosen number of clusters boosts clustering performance by balancing intra-cluster compactness and inter-cluster separation. The variation in the number of clusters (N_{clus}) between datasets emphasizes the Frac-FCM algorithm's ability to adapt datasets. For example, some datasets do best with a small number of clusters, but others require a greater number of clusters. This adaptability to different datasets is critical for producing high-quality clusters, as seen by improvements in the XB and FS indices.

The Frac-FCM model introduces a memory effect through the fractional derivative order α, which influences how previous updates affect current cluster centers and memberships. This parameter balances the impact of short-term and long-term updates, with higher α values illustrating recent changes for faster convergence, while lower values incorporate a longer history, capturing non-local dependencies and long memory effects to enhance robustness [71]. The evolutionary GA optimizes α for each dataset, ensuring that the method adapts to the specific characteristics of the problem. By adjusting α to match the complexity of diverse datasets, the Frac-FCM model surpasses traditional methods in clustering performance.

6.4. Comparison Study

In this section, we perform a comparative evaluation of the Frac-FCM (ω^*) algorithm against several clustering techniques, including K-means, self-organizing maps, fuzzy C-means, and Gaussian mixture models, across various datasets. The comparison is based on both Dunn's index and the Silhouette index, which assess cluster compactness, separation, and overall clustering quality.

The Silhouette index assesses how well an object matches its cluster compared with other clusters, with higher values indicating better-defined and more distinct clusters. FCM and GMM generally exhibit lower performance in most cases, while SOM and K-means perform relatively well across various datasets. Notably, Frac-FCM achieves higher Silhouette scores, particularly on datasets like Iris (0.64413 ± 0.02630), Seed (0.52985 ± 0.03334), and Statlog (0.84676 ± 0.00238). These results suggest that Frac-FCM produces more distinct and well-separated clusters than traditional methods, enhancing cluster quality and interoperability. Dunn's index, which evaluates the compactness and separation of clusters, demonstrates that Frac-FCM consistently outperforms other methods across most datasets. This indicates that Frac-FCM generates clusters that are both more compact and better separated. Although K-means and SOM perform competitively in some cases, FCM and GMM tend to score lower. The clustering results outlined above are presented in Table 4, highlighting Silhouette and Dunn's indices for the evaluated methods across 15 different datasets. Overall, Frac-FCM demonstrates superior performance, particularly in terms of cluster separation and compactness, making it a strong option for complex datasets.

To determine whether the differences in the Silhouette and Dunn indices across datasets are statistically significant, we employed the Friedman test. This non-parametric test evaluates the performance ranks of each algorithm on each dataset. The null hypothesis (H_0) of the Friedman test states: "All algorithms perform equivalently, and there is no significant difference in their ranks".

The Friedman test analyzes the average ranks of the algorithms under the null hypothesis. The test results demonstrate that Frac-FCM significantly outperforms the other clustering algorithms, indicating its superior performance. With an average rank of 1.533 for the Silhouette index and 2.07 for the Dunn index, Frac-FCM consistently outperformed the rest algorithms. The closest opponents were SOM and K-means. At a significance level of 0.05 and with four degrees of freedom, the null hypothesis was rejected due to the extremely small p-values (2.736×10^{-9} for Silhouette and 1.093×10^{-3} for Dunn), further confirming the superiority of Frac-FCM. Figures 7 and 8 illustrate the average clustering performance of the algorithms based on the Silhouette and Dunn indices, respec-

tively. These figures clearly highlight the superior performance of the Frac-FCM algorithm compared with the other methods, as evidenced by its consistently higher scores across the datasets.

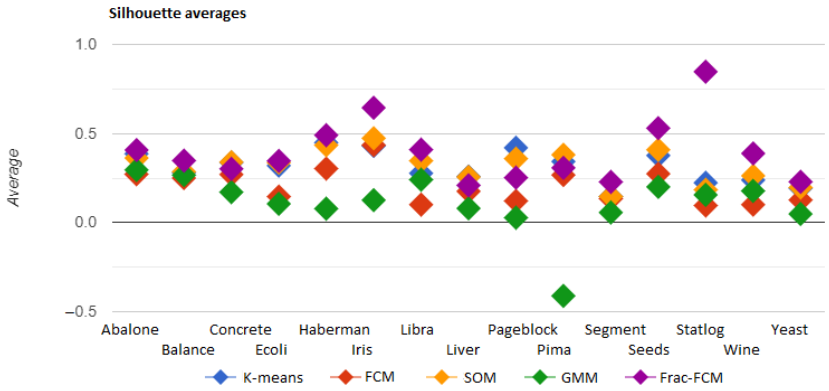

Figure 7. Average performance of clustering based on the Silhouette index.

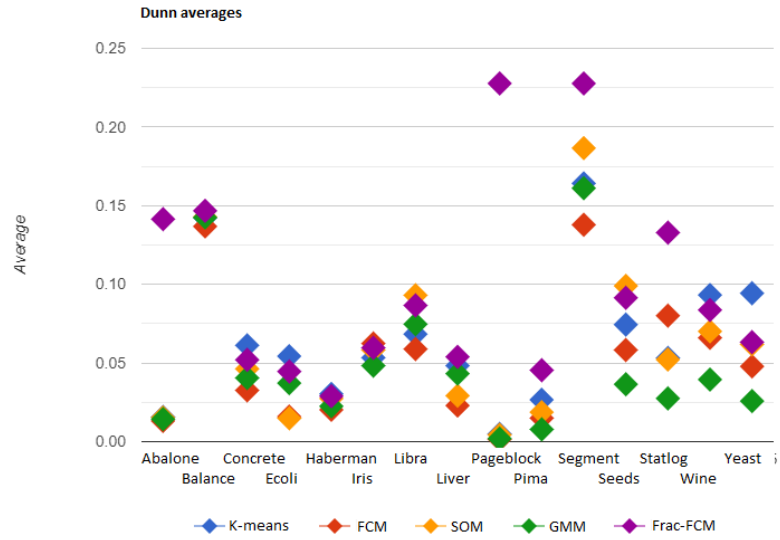

Figure 8. Average clustering performance based on the Dunn index.

The results presented in the table are well-supported by the existing literature, highlighting the benefits of using genetic algorithms in clustering approaches. Ref. [72] demonstrates that optimizing Silhouette indices using genetic algorithms can enhance the clarity of cluster separation by addressing non-informative variables, leading to improved Silhouette scores. This aligns with the findings in the table, where the Frac-FCM algorithm achieves the highest Silhouette indices, indicating clearer group separation. Similarly, ref. [73] studied the use of genetic algorithms to address K-means challenges, such as random seed selection and the need to specify the number of clusters in advance. GA also helps avoid local minima, improving clustering accuracy before fine-tuning the results using K-means. Furthermore, ref. [74] points out that clustering algorithms are often sensitive to noise and outliers, but the use of genetic algorithms can mitigate these issues effectively. The table's results show a robust performance for GA-based methods, particularly for complex datasets, where Frac-FCM exhibits enhanced resilience to noise. The optimization of the FCM algorithm through genetic algorithms leads to improved performance compared with traditional FCM methods [39].

Table 4. Evaluation of K-means, FCM, SOM, GMM, and Frac-FCM performance using Silhouette and Dunn indices across multiple datasets.

Dataset	K-Means		FCM		SOM		GMM		Frac-FCM	
	Silhouette	Dunn	Silhouette	Dunn	Silhouette	Dunn	Silhouette	Dunn	Silhouette	Dunn
Abalone	0.38636 ± 0.00346	0.01564 ± 0.00092	0.27144 ± 0.00190	0.01316 ± 0.00027	0.36265 ± 0.00053	0.01534 ± 0.00016	0.29536 ± 0.00393	0.01435 ± 0.00047	0.40710 ± 0.00016	0.14147 ± 0.00100
Balance	0.28038 ± 0.00110	0.14224 ± 0.00030	0.25000 ± 0.00422	0.13687 ± 0.00066	0.28543 ± 0.00031	0.14282 ± 0.00008	0.26650 ± 0.00789	0.14253 ± 0.00028	0.34781 ± 0.02031	0.14678 ± 0.00102
Concrete Compressive Strength	0.33747 ± 0.00268	0.06122 ± 0.00380	0.26990 ± 0.00153	0.03260 ± 0.00280	0.33902 ± 0.00019	0.04630 ± 0.00022	0.17075 ± 0.00820	0.04049 ± 0.00232	0.30151 ± 0.00711	0.05201 ± 0.00217
Ecoli	0.32003 ± 0.00403	0.05432 ± 0.00235	0.14606 ± 0.00334	0.01594 ± 0.00024	0.33644 ± 0.00192	0.01486 ± 0.00035	0.10528 ± 0.00556	0.03721 ± 0.00192	0.34648 ± 0.00153	0.04458 ± 0.00349
Haberman	0.44908 ± 0.00612	0.03038 ± 0.00247	0.30311 ± 0.00466	0.02026 ± 0.00126	0.43432 ± 0.00151	0.02743 ± 0.00200	0.07845 ± 0.00989	0.02253 ± 0.00057	0.49020 ± 0.00713	0.02893 ± 0.00318
Iris	0.43004 ± 0.00613	0.05338 ± 0.00205	0.43494 ± 0.00433	0.06237 ± 0.00356	0.47320 ± 0.00515	0.05824 ± 0.00097	0.12628 ± 0.01339	0.04829 ± 0.00376	0.64413 ± 0.02630	0.05964 ± 0.00727
Libra	0.27642 ± 0.00362	0.06838 ± 0.00228	0.10037 ± 0.01690	0.05884 ± 0.00363	0.34675 ± 0.00132	0.09306 ± 0.00349	0.24073 ± 0.00488	0.07470 ± 0.00354	0.40913 ± 0.01802	0.08658 ± 0.00969
Liver	0.25858 ± 0.00389	0.04828 ± 0.00254	0.17583 ± 0.00254	0.02289 ± 0.00131	0.25605 ± 0.00396	0.02917 ± 0.00170	0.07958 ± 0.00851	0.04327 ± 0.00095	0.20920 ± 0.00570	0.05392 ± 0.00375
Pageblock	0.41982 ± 0.01017	0.00488 ± 0.00020	0.12146 ± 0.00232	0.00148 ± 0.00005	0.35856 ± 0.00184	0.00438 ± 0.00053	0.02751 ± 0.00381	0.00192 ± 0.00019	0.25191 ± 0.03312	0.22770 ± 0.00144
Pima	0.34238 ± 0.00250	0.02662 ± 0.00134	0.26679 ± 0.00230	0.01500 ± 0.00140	0.38023 ± 0.00161	0.01873 ± 0.00058	−0.41161 ± 0.01702	0.00781 ± 0.00021	0.30758 ± 0.03119	0.04539 ± 0.00139
Segment	0.14660 ± 0.00533	0.16413 ± 0.00707	0.13416 ± 0.00430	0.13784 ± 0.00535	0.14656 ± 0.00319	0.18663 ± 0.00564	0.05638 ± 0.00730	0.16114 ± 0.00502	0.22832 ± 0.00201	0.22770 ± 0.00144
Seeds	0.37685 ± 0.00448	0.07439 ± 0.00332	0.27361 ± 0.00709	0.05824 ± 0.00184	0.40877 ± 0.00233	0.09903 ± 0.00292	0.20063 ± 0.00843	0.03649 ± 0.00133	0.52985 ± 0.03334	0.09142 ± 0.00356
Statlog	0.22353 ± 0.01004	0.05315 ± 0.00436	0.09590 ± 0.01129	0.08007 ± 0.00206	0.18435 ± 0.00462	0.05221 ± 0.00030	0.15568 ± 0.00805	0.02747 ± 0.00277	0.84676 ± 0.00238	0.02747 ± 0.00277
Wine	0.23868 ± 0.00787	0.17824 ± 0.00397	0.10090 ± 0.00524	0.14632 ± 0.00091	0.26251 ± 0.00272	0.19142 ± 0.00540	0.17767 ± 0.00855	0.16556 ± 0.00432	0.38808 ± 0.00607	0.17917 ± 0.00850
Yeast	0.19260 ± 0.00114	0.05970 ± 0.00049	0.12671 ± 0.00436	0.04059 ± 0.00093	0.19699 ± 0.00055	0.03649 ± 0.00133	0.04848 ± 0.00699	0.04059 ± 0.00093	0.22892 ± 0.01705	0.06522 ± 0.00297
Average Rank	2.4	2.53	4.266	4.2	2.133	2.6	4.666	3.6	1.533	2.07

Moreover, ref. [75] integrates an improved artificial bee colony (IABC) algorithm and fractional-order modeling with KFCM [39]. This approach enhances the clustering performance by improving parameter estimation, effectively capturing non-linear dynamics, resulting in faster convergence, and increased robustness against local optima. In addition, ref. [76] highlights a hybrid model combining fuzzy C-means clustering with fractional-order methods to improve prediction accuracy, which directly supports the performance of the fractional clustering methods shown in the table. The high results for the Frac-FCM method in most datasets demonstrate that fractional-order models lead to better clustering accuracy and separation, highlighting their benefits in clustering.

The proposed Frac-FCM method may face challenges from the computational complexity of the genetic algorithm and the limitations of fuzzy C-means when dealing with large datasets. While genetic algorithms improve clustering accuracy, they can be computationally demanding, particularly with high-dimensional data [77]. Additionally, FCM struggles with large datasets [78], making it less suitable for such cases. These factors underscore the importance of efficiency and data suitability when applying these techniques.

7. Conclusions

In this paper, we introduced a novel clustering model that combines the fuzzy C-means (FCM) algorithm with fractional derivatives, resulting in the fractional fuzzy C-means (Frac-FCM) algorithm. This model effectively addresses the limitations of traditional FCM, which include its sensitivity to initial cluster assignments, slow convergence, and challenges in handling non-linear and overlapping clusters. By incorporating fractional derivatives, Frac-FCM captures non-local dependencies and long memory effects, enabling a more precise representation of data relationships in complex datasets. The integration of a genetic algorithm (GA) for optimizing the least squares objective function, with a focus on geometric cluster properties like the Fukuyama–Sugeno and Xie–Beni indices, further improves clustering performance by achieving a better balance between compactness and separation. Experimental evaluations across multiple datasets demonstrate that Frac-FCM consistently outperforms traditional clustering methods in terms of both the Silhouette and Dunn indices. These results confirm the robustness and superiority of the Frac-FCM algorithm, making it a valuable tool for clustering complex data structures more accurately and efficiently.

Although Frac-FCM shows promising results, there are several areas for further research. These include exploring hybrid optimization techniques to improve the accuracy of the model and addressing the scalability of Frac-FCM for large datasets. Additionally, research could focus on enhancing Frac-FCM's ability to handle high-dimensional data and more complex data structures, as well as investigating its performance in real-time dynamic data environments.

Author Contributions: Conceptualization, S.S. and K.E.M.; methodology, K.E.M.; validation, A.-M.P.; investigation, S.S. and A.-M.P.; writing—review and editing, S.S. and A.-M.P.; visualization, S.S. and K.E.M.; supervision, K.E.M. All authors have read and agreed to the published version of the manuscript.

Funding: The APC was funded by Dunărea de Jos University of Galaţi, Romania.

Data Availability Statement: The dataset is available on request from the authors.

Acknowledgments: This work was partially supported by the Ministry of National Education, Professional Training, Higher Education and Scientific Research (MENFPESRS) and the Digital Development Agency (DDA) of Morocco (Nos.Alkhawarizmi/2020/23).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Moshtaghi, M.; Havens, T.C.; Bezdek, J.C.; Park, L.; Leckie, C.; Rajasegarar, S.; Keller, J.M.; Palaniswami, M. Clustering ellipses for anomaly detection. *Pattern Recognit.* **2011**, *44*, 55–69. [CrossRef]

2. Sharma, P.; Suji, J. A review on image segmentation with its clustering techniques. *Int. J. Signal Process. Image Process. Pattern Recognit.* **2016**, *9*, 209–218. [CrossRef]
3. Ramasubbareddy, S.; Srinivas, T.A.S.; Govinda, K.; Manivannan, S.S. Comparative study of clustering techniques in market segmentation. In *Innovations in Computer Science and Engineering: Proceedings of 7th ICICSE*; Springer: Singapore, 2020; pp. 117–125.
4. Li, Q.; Kim, B.M. Clustering approach for hybrid recommender system. In Proceedings of the IEEE/WIC International Conference on Web Intelligence (WI 2003), Halifax, NS, Canada, 13–17 October 2003; pp. 33–38.
5. Zadeh, L.A. Fuzzy sets. *Inf. Control* **1965**, *8*, 338–353. [CrossRef]
6. Dunn, J.C. A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *J. Cybern.* **1973**, *3*, 32–57. [CrossRef]
7. Suganya, R.; Shanthi, R. Fuzzy c-means algorithm—A review. *Int. J. Sci. Res. Publ.* **2012**, *2*, 1.
8. Machado, J.A. System modeling and control through fractional-order algorithms. *Nonlinear Dyn. Chaos Control Their Appl. Eng. Sci.* **2002**, *4*, 99–116.
9. Halkidi, M.; Batistakis, Y.; Vazirgiannis, M. Cluster validity methods: Part I. *ACM Sigmod Rec.* **2002**, *31*, 40–45. [CrossRef]
10. Halkidi, M.; Batistakis, Y.; Vazirgiannis, M. Clustering validity checking methods: Part II. *ACM Sigmod Rec.* **2002**, *31*, 19–27. [CrossRef]
11. Rezaei, M.; Fränti, P. Set matching measures for external cluster validity. *IEEE Trans. Knowl. Data Eng.* **2016**, *28*, 2173–2186. [CrossRef]
12. Vendramin, L.; Campello, R.J.; Hruschka, E.R. Relative clustering validity criteria: A comparative overview. *Stat. Anal. Data Min. ASA Data Sci. J.* **2010**, *3*, 209–235. [CrossRef]
13. Cebeci, Z. Comparison of internal validity indices for fuzzy clustering. *Agrárinformatika/J. Agric. Inform.* **2019**, *10*, 1–14. [CrossRef]
14. Bezdek, J.C. *Pattern Recognition with Fuzzy Objective Function Algorithms*, 2nd ed.; Springer: New York, NY, USA, 1987.
15. Liew, A.W.C.; Leung, S.H.; Lau, W.H. Fuzzy image clustering incorporating spatial continuity. *IEE Proc.-Vis. Image Signal Process.* **2000**, *147*, 185–192. [CrossRef]
16. Forouzanfar, M.; Forghani, N.; Teshnehlab, M. Parameter optimization of improved fuzzy c-means clustering algorithm for brain MR image segmentation. *Eng. Appl. Artif. Intell.* **2010**, *23*, 160–168. [CrossRef]
17. Krishnapuram, R.; Keller, J.M. A possibilistic approach to clustering. *IEEE Trans. Fuzzy Syst.* **1993**, *1*, 98–110. [CrossRef]
18. Duda, R.O.; Hart, P.E.; Stork, D.G. *Pattern Classification and Scene Analysis*; Wiley: New York, NY, USA, 1973; Volume 3, pp. 731–739.
19. Jafar, O.M.; Sivakumar, R. A study on possibilistic and fuzzy possibilistic c-means clustering algorithms for data clustering. In Proceedings of the 2012 International Conference on Emerging Trends in Science, Engineering and Technology (INCOSET), Tiruchirappalli, India, 13–14 December 2012; pp. 90–95.
20. Pal, N.R.; Pal, K.; Keller, J.M.; Bezdek, J.C. A new hybrid c-means clustering model. In Proceedings of the 2004 IEEE International Conference on Fuzzy Systems (IEEE Cat. No. 04CH37542), Budapest, Hungary, 25–29 July 2004; Volume 1, pp. 179–184.
21. Pal, N.R.; Pal, K.; Keller, J.M.; Bezdek, J.C. A possibilistic fuzzy c-means clustering algorithm. *IEEE Trans. Fuzzy Syst.* **2005**, *13*, 517–530. [CrossRef]
22. El Moutaouakil, K.; Palade, V.; Safouan, S.; Charroud, A. FP-Conv-CM: Fuzzy probabilistic convolution C-means. *Mathematics* **2023**, *11*, 1931. [CrossRef]
23. Yu, J.; Yang, M.S. A generalized fuzzy clustering regularization model with optimality tests and model complexity analysis. *IEEE Trans. Fuzzy Syst.* **2007**, *15*, 904–915. [CrossRef]
24. Chiang, J.H.; Hao, P.Y. A new kernel-based fuzzy clustering approach: Support vector clustering with cell growing. *IEEE Trans. Fuzzy Syst.* **2003**, *11*, 518–527. [CrossRef]
25. Huang, H.C.; Chuang, Y.Y.; Chen, C.S. Multiple kernel fuzzy clustering. *IEEE Trans. Fuzzy Syst.* **2011**, *20*, 120–134. [CrossRef]
26. Chen, L.; Chen, C.P.; Lu, M. A multiple-kernel fuzzy c-means algorithm for image segmentation. *IEEE Trans. Syst. Man Cybern. Part B (Cybern.)* **2011**, *41*, 1263–1274. [CrossRef]
27. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN'95-International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948.
28. Alam, S.; Dobbie, G.; Koh, Y.S.; Riddle, P.; Rehman, S.U. Research on particle swarm optimization based clustering: A systematic review of literature and techniques. *Swarm Evol. Comput.* **2014**, *17*, 1–13. [CrossRef]
29. Silva Filho, T.M.; Pimentel, B.A.; Souza, R.M.; Oliveira, A.L. Hybrid methods for fuzzy clustering based on fuzzy c-means and improved particle swarm optimization. *Expert Syst. Appl.* **2015**, *42*, 6315–6328. [CrossRef]
30. Wang, G.; Yin, X.; Pang, Y.; Zhang, M.; Zhao, W.; Zhang, Z. Studies on fuzzy c-means based on ant colony algorithm. In Proceedings of the 2010 International Conference on Measuring Technology and Mechatronics Automation, Changsha, China, 13–14 March 2010; Volume 3, pp. 515–518.
31. Doğan, B.; Korürek, M. A new ECG beat clustering method based on kernelized fuzzy c-means and hybrid ant colony optimization for continuous domains. *Appl. Soft Comput.* **2012**, *12*, 3442–3451. [CrossRef]
32. Cheng, X.; Gong, X. An image segmentation of fuzzy C-means clustering based on the combination of improved Ant Colony Algorithm and Genetic Algorithm. In Proceedings of the 2008 International Workshop on Education Technology and Training, 2008 International Workshop on Geoscience and Remote Sensing, Shanghai, China, 21–22 December 2008; Volume 2, pp. 804–808.
33. Karaboga, D.; Basturk, B. A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. *J. Glob. Optim.* **2007**, *39*, 459–471. [CrossRef]

34. Krishnamoorthi, M.; Natarajan, A.M. Artificial bee colony algorithm integrated with fuzzy c-mean operator for data clustering. *J. Comput. Sci.* **2013**, *9*, 404. [CrossRef]
35. Kumar, A.; Kumar, D.; Jarial, S.K. A hybrid clustering method based on improved artificial bee colony and fuzzy C-Means algorithm. *Int. J. Artif. Intell.* **2017**, *15*, 40–60.
36. Alomoush, W.; Alrosan, A.; Almomani, A.; Alissa, K.; Khashan, O.A.; Al-Nawasrah, A. Spatial information of fuzzy clustering based mean best artificial bee colony algorithm for phantom brain image segmentation. *Int. J. Electr. Comput. Eng. (IJECE)* **2021**, *11*, 4050–4058. [CrossRef]
37. Alata, M.; Molhim, M.; Ramini, A. Optimizing of fuzzy c-means clustering algorithm using GA. *Int. J. Comput. Inf. Eng.* **2008**, *2*, 670–675.
38. Wikaisuksakul, S. A multi-objective genetic algorithm with fuzzy c-means for automatic data clustering. *Appl. Soft Comput.* **2014**, *24*, 679–691. [CrossRef]
39. Ding, Y.; Fu, X. Kernel-based fuzzy c-means clustering algorithm based on genetic algorithm. *Neurocomputing* **2016**, *188*, 233–238. [CrossRef]
40. Krishna, K.; Murty, M.N. Genetic K-means algorithm. *IEEE Trans. Syst. Man Cybern. Part B (Cybern.)* **1999**, *29*, 433–439. [CrossRef] [PubMed]
41. Kohonen, T. The self-organizing map. *Proc. IEEE* **1990**, *78*, 1464–1480. [CrossRef]
42. Dempster, A.P.; Laird, N.M.; Rubin, D.B. Maximum likelihood from incomplete data via the EM algorithm. *J. R. Stat. Soc. Ser. B (Methodol.)* **1977**, *39*, 1–22. [CrossRef]
43. El Moutaouakil, K.; Roudani, M.; El Ouissari, A. Optimal entropy genetic fuzzy-C-means SMOTE (OEGFCM-SMOTE). *Knowl.-Based Syst.* **2023**, *262*, 110235. [CrossRef]
44. El Moutaouakil, K.; El Ouissari, A.; Palade, V.; Charroud, A.; Olaru, A.; Baïzri, H.; Chellak, S.; Cheggour, M. Multi-objective optimization for controlling the dynamics of the diabetic population. *Mathematics* **2023**, *11*, 2957. [CrossRef]
45. El Moutaouakil, K.; Yahyaouy, A.; Chellak, S.; Baizri, H. An optimized gradient dynamic-neuro-weighted-fuzzy clustering method: Application in the nutrition field. *Int. J. Fuzzy Syst.* **2022**, *24*, 3731–3744. [CrossRef]
46. Herzallah, M.A.; Baleanu, D. Fractional-order Euler–Lagrange equations and formulation of Hamiltonian equations. *Nonlinear Dyn.* **2009**, *58*, 385–391. [CrossRef]
47. Magin, R.; Feng, X.; Baleanu, D. Solving the fractional order Bloch equation. *Concepts Magn. Reson. Part A Educ. J.* **2009**, *34*, 16–23. [CrossRef]
48. Tarasov, V.E.; Zaslavsky, G.M. Fokker–Planck equation with fractional coordinate derivatives. *Phys. A Stat. Mech. Its Appl.* **2008**, *387*, 6505–6512. [CrossRef]
49. Compte, A.; Metzler, R. The generalized Cattaneo equation for the description of anomalous transport processes. *J. Phys. A Math. Gen.* **1997**, *30*, 7277. [CrossRef]
50. Podlubny, I. *Fractional Differential Equations*; Mathematics in Science and Engineering; Academic Press: San Diego, CA, USA, 1999.
51. Ostalczyk, P. *Discrete Fractional Calculus: Applications in Control and Image Processing*; World Scientific: Singapore, 2015; Volume 4.
52. Samko, S.G.; Kilbas, A.A.; Marichev, O.I. *Fractional Integrals and Derivatives*; Gordon and Breach Science Publishers: Yverdon-les-Bains, Switzerland, 1993; Volume 1.
53. Scherer, R.; Kalla, S.L.; Tang, Y.; Huang, J. The Grünwald–Letnikov method for fractional differential equations. *Comput. Math. Appl.* **2011**, *62*, 902–917. [CrossRef]
54. Meerschaert, M.M.; Mortensen, J.; Scheffler, H.P. Vector Grunwald formula for fractional derivatives. *Fract. Calc. Appl. Anal.* **2004**, *7*, 61–82.
55. Meerschaert, M.M.; Tadjeran, C. Finite difference approximations for fractional advection–dispersion flow equations. *J. Comput. Appl. Math.* **2004**, *172*, 65–77. [CrossRef]
56. Wei, Y.; Yin, W.; Zhao, Y.; Wang, Y. A new insight into the Grünwald–Letnikov discrete fractional calculus. *J. Comput. Nonlinear Dyn.* **2019**, *14*, 041008. [CrossRef]
57. Hajipour, M.; Jajarmi, A.; Baleanu, D. An efficient nonstandard finite difference scheme for a class of fractional chaotic systems. *J. Comput. Nonlinear Dyn.* **2018**, *13*, 021013. [CrossRef]
58. Wang, H.; Du, N. A fast finite difference method for three-dimensional time-dependent space-fractional diffusion equations and its efficient implementation. *J. Comput. Phys.* **2013**, *253*, 50–63. [CrossRef]
59. Abdelouahab, M.S.; Hamri, N.E. The Grünwald–Letnikov fractional-order derivative with fixed memory length. *Mediterr. J. Math.* **2016**, *13*, 557–572. [CrossRef]
60. Mathew, T.V. Genetic Algorithm. 2012. Available online: https://datajobs.com/data-science-repo/Genetic-Algorithm-Guide-[Tom-Mathew].pdf (accessed on 12 September 2024).
61. Bhoskar, M.T.; Kulkarni, M.O.K.; Kulkarni, M.N.K.; Patekar, M.S.L.; Kakandikar, G.; Nandedkar, V. Genetic algorithm and its applications to mechanical engineering: A review. *Mater. Today Proc.* **2015**, *2*, 2624–2630. [CrossRef]
62. Omara, F.A.; Arafa, M.M. Genetic algorithms for task scheduling problem. *J. Parallel Distrib. Comput.* **2010**, *70*, 13–22. [CrossRef]
63. Metawa, N.; Hassan, M.K.; Elhoseny, M. Genetic algorithm based model for optimizing bank lending decisions. *Expert Syst. Appl.* **2017**, *80*, 75–82. [CrossRef]
64. Manning, T.; Sleator, R.D.; Walsh, P. Naturally selecting solutions: The use of genetic algorithms in bioinformatics. *Bioengineered* **2013**, *4*, 266–278. [CrossRef] [PubMed]

65. Hauswirth, A.; Bolognani, S.; Hug, G.; Dörfler, F. Projected gradient descent on Riemannian manifolds with applications to online power system optimization. In Proceedings of the 2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton), Monticello, IL, USA, 27–30 September 2016; pp. 225–232.
66. El Moutaouakil, K.; Touhafi, A. A new recurrent neural network fuzzy mean square clustering method. In Proceedings of the 2020 5th International Conference on Cloud Computing and Artificial Intelligence: Technologies and Applications (CloudTech), Marrakesh, Morocco, 24–26 November 2020; pp. 1–5.
67. Fukuyama, Y. A new method of choosing the number of clusters for fuzzy c-means method. In Proceedings of the 5th Fuzzy System Symposium, Tokyo, Japan, 25–27 October 1989; Volume 5, pp. 247–250.
68. Xie, X.L.; Beni, G. A validity measure for fuzzy clustering. *IEEE Trans. Pattern Anal. Mach. Intell.* **1991**, *13*, 841–847. [CrossRef]
69. Starczewski, A.; Krzyżak, A. Performance evaluation of the silhouette index. In *Artificial Intelligence and Soft Computing: 14th International Conference, ICAISC 2015, Zakopane, Poland, June 14–18. 2015, Proceedings, Part II*; Springer International Publishing: Cham, Switzerland, 2015; pp. 49–58.
70. Dunn, J.C. Well-separated clusters and optimal fuzzy partitions. *J. Cybern.* **1974**, *4*, 95–104. [CrossRef]
71. Wang, J.; Wen, Y.; Gou, Y.; Ye, Z.; Chen, H. Fractional-order gradient descent learning of BP neural networks with Caputo derivative. *Neural Netw.* **2017**, *89*, 19–30. [CrossRef] [PubMed]
72. Lletí, R.; Ortiz, M.C.; Sarabia, L.A.; Sánchez, M.S. Selecting variables for k-means cluster analysis by using a genetic algorithm that optimises the silhouettes. *Anal. Chim. Acta* **2004**, *515*, 87–100. [CrossRef]
73. Rahman, M.A.; Islam, M.Z. A hybrid clustering technique combining a novel genetic algorithm with K-Means. *Knowl.-Based Syst.* **2014**, *71*, 345–365. [CrossRef]
74. Kuo, R.J.; Lin, J.Y.; Nguyen, T.P.Q. Genetic Algorithm Based Fuzzy c-Ordered-Means to Cluster Analysis. In Proceedings of the 2019 IEEE 6th International Conference on Industrial Engineering and Applications (ICIEA), Tokyo, Japan, 12–15 April 2019; pp. 918–922.
75. Zhou, S.; Xu, X.; Xu, Z.; Chang, W.; Xiao, Y. Fractional-order modeling and fuzzy clustering of improved artificial bee colony algorithms. *IEEE Trans. Ind. Inform.* **2019**, *15*, 5988–5998. [CrossRef]
76. Chen, Y.; Gao, Q.; Su, X.; Yu, C. Research on consumption prediction of spare parts based on fuzzy C-means clustering algorithm and fractional order model. *Vibroeng. Procedia* **2017**, *16*, 129–133. [CrossRef]
77. Oliveto, P.S.; Witt, C. Improved time complexity analysis of the simple genetic algorithm. *Theor. Comput. Sci.* **2015**, *605*, 21–41. [CrossRef]
78. Hashemi, S.E.; Gholian-Jouybari, F.; Hajiaghaei-Keshteli, M. A fuzzy C-means algorithm for optimizing data clustering. *Expert Syst. Appl.* **2023**, *227*, 120377. [CrossRef]