Received 24 September 2023, accepted 4 October 2023, date of publication 9 October 2023, date of current version 13 October 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3322929

RESEARCH ARTICLE

Dynamic and Static Enhanced BIRCH for Functional Data Clustering

WANG LI, HANFANG LI, AND YOUXI LUO
School of Science, Hubei University of Technology, Wuhan 430068, China

Corresponding author: Youxi Luo (youxiluo@163.com)

This work was supported in part by the National Social Science Fund under Grant 17BJY210, in part by the Key Humanities and Social Science Fund of Hubei Provincial Department of Education under Grant 20D043, in part by the Doctor's Fund of Hubei University of Technology under Grant BSQD2020103, and in part by the Humanities and Social Science Fund of Hubei Provincial Department of Education under Grant 22Y059.

ABSTRACT Accurate and efficient clustering of large-scale functional data is of utmost importance in the era of big data. However, the current research falls short in fully considering the differentiability inherent in functional data. To tackle this significant challenge, we propose a novel method, namely Dynamic and Static Enhanced-BIRCH (DSE-BIRCH), which incorporates both the constant and derivate features to simultaneously measure the static and dynamic distances between functional samples. To this end, a novel matrix factorization-based approach is introduced to transform constant features, extracted through principal component analysis, into derivative features. Subsequently, these two sets of features are fused to form global clustering features with different weighting coefficients are assigned to each of them, reflecting their respective importance. Finally, an enhanced BIRCH algorithm is employed to handle both static and dynamic constraints, enabling hierarchical clustering from a more comprehensive perspective. The mathematical definition of the algorithm is rigorously provided. The superior empirical performance of our method on publicly available datasets and simulated datasets fully demonstrates its effective capture of dynamic information and its capability to achieve accurate clustering on real-world data. Further experiments involving noise and complexity attest to the algorithm's robustness and efficiency, highlighting its broad potential for applications in various complex scenarios involving large-scale functional data.

INDEX TERMS Functional data, clustering, BIRCH, dynamic and static information fusion.

I. INTRODUCTION

With the application of Internet of Things (IoT) technology and the development of big data, an increasing amount of discrete data with functional characteristics is being recorded and stored. In statistics, these continuous, dynamic data with functional characteristics are referred to as functional data. The concept of functional data was first proposed by Ramsay in 1982 [1], followed by a comprehensive exploration of its properties [2]. Since the publication of Ramsay and Silverman's monograph [3], functional data analysis has received extensive attention and recognition. With ongoing research, functional data has been widely utilized in various fields including medicine, biology, and finance. It encompasses the analysis of variables such as rainfall over a specific time period, stock trading prices in financial markets, and GDP indices, among many others [4], [5], [6], [7], [8]. Currently, research on functional data has become increasingly extensive, particularly in the areas of principal component analysis, regression analysis, clustering, and classification [9]. While there have been numerous studies on principal component analysis and regression analysis of functional data, research on clustering and classification, especially efficient clustering of large-scale high-dimensional functional data, still requires further exploration. According to the representation of the functional objects and the modeling approaches used, clustering methods for functional data can be categorized into four types.

The associate editor coordinating the review of this manuscript and approving it for publication was Pasquale De Meo.

Raw-data direct clustering is the most straightforward method for clustering, where the original data is directly used as discrete high-dimensional objects for analysis. Each observation time is treated as a one-dimensional feature, and conventional clustering methods are applied based on multi-dimensional distance measures [10], [11], [12]. For example, Boullé [11] proposed a method for functional data by modeling the probability distribution of the number of discrete observations falling into fixed-width intervals, thus avoiding the problem of selecting basis function. However, this method requires noise-free observations and uniform sampling at the same time points for all functions in the function set. Furthermore, discretizing functional data in this manner fails to leverage the continuous differentiability advantages of functional data.

Two-step tandem clustering involves fitting the discretely observed data to selected basis functions, and then using the fitted parameter coefficients as features for clustering [13], [14], [15], [16], [17], [18]. For instance, Yamamoto [15] proposed a method for clustering functional data in a low-dimensional subspace based on principal component analysis and k-means criterion, which enhances the applicability and effectiveness of the two-step sequential clustering framework. Zhang [18] proposed an adaptive weighted principal component-based k-medoids clustering method that combines adaptive weights and principal component analysis to address the clustering of functional data. Although TSTC adheres to the core principles of Functional Data Analysis (FDA), it suffers from a disconnect between feature selection and the clustering process, and the clustering results can be influenced by the choice of basis functions.

Nonparametric clustering of functional data relies primarily on derivative information extracted from the functions or through bayes nonparametric methods to quantify the similarity between curves [19], [20], [21], [22], [23], [24], [25], [26]. Chiou and Li [19] proposed a clustering method for longitudinal data, estimating the means and covariances of the classes through non-parametric iterative approaches, and subsequently predicting cluster membership probabilities for the members. Fortuna et al. [24] utilized the K-means algorithm and employed first and second derivatives as semi-metric methods to measure the similarity between functional data. Compared to clustering based solely on the static features of functions (e.g., base function similarity), these methods can more accurately differentiate the density relationships between functions by considering their derivative characteristics. Although they provide clustering information from multiple perspectives, the clustering results are unstable or sensitive to the choice of derivative order, and they fail to capture the hierarchical structure within the data.

In model-based clustering methods, the probability density of functional data is used to describe the distribution characteristics, typically assuming that the expansion coefficients of the basis functions follow a specific distribution as random variables. An alternative method to estimate the distribution of functional data is to use non-parametric kernel density estimation based on principal components analysis. This replaces the original probability density of the functions with a kernel density estimate constructed from the principal component scores [27], [28], [29], [30], [31], [32], [33], [34], [35], [36], [37]. For example, Wu et al. [35] incorporated non-parametric curve features and developed a probabilistic model based on Bayesian criteria for functional data clustering. These methods integrate data dimensionality reduction with the clustering process, performing both data reduction and clustering analysis simultaneously. However, the clustering results are highly sensitive to the initial parameter values, leading to poor result stability. Additionally, due to the complexity of the model itself, it is challenging to extract hierarchical structural information from the data.

In summary, existing clustering methods for functional data suffer from several limitations: 1. Most existing clustering algorithms do not fully consider the high-order differentiability and fail to leverage its advantages. While algorithms that utilize derivative features often exhibit sensitivity to the choice of derivative order. 2. Many algorithms are sensitive to parameter selection, leading to poor stability of clustering results. The robustness of the algorithms needs further improvement. 3. The algorithms have high computational complexity, making them less suitable for large-scale data analysis. They struggle to scale effectively for big data analysis. Therefore, there is a need to address these challenges and develop new clustering methods for functional data that can fully leverage the advantages of high-order differentiability, improve result stability and robustness, and enhance scalability for large-scale datasets.

To address the aforementioned clustering challenges, some researchers have attempted to develop new clustering methods, such as using novel metrics to replace traditional distance or similarity measures for improved clustering performance [38], [39], [40], [41], [42], [43]. For example, Sharma et al. [38] proposed an improved spectral clustering algorithm based on S-divergence for customer churn prediction and indicator clustering. They then presented clustering algorithms leveraging Kullback-Leibler divergence and Jeffreys divergence for uncertain data clustering, as well as an efficient density peak clustering method with adaptive mixed distances for detecting and diagnosing faults in mechanical gearboxes [39], [40]. Albert-Smet et al. [41] put forward an improved k-means algorithm with bootstrap Random Initialization to find optimal initial seeds for functional data. And some other studies have focused on enhancing the computational efficiency of large-scale data clustering algorithms. Hu et al. [42] proposed a more efficient k-means clustering approach, providing inspirational ideas for improving existing methods. While these algorithms successfully address clustering challenges for discrete data, they do not consider the temporal properties of time series data. Motivated by the above approaches and given certain limitations of current functional data clustering techniques, this paper aims to

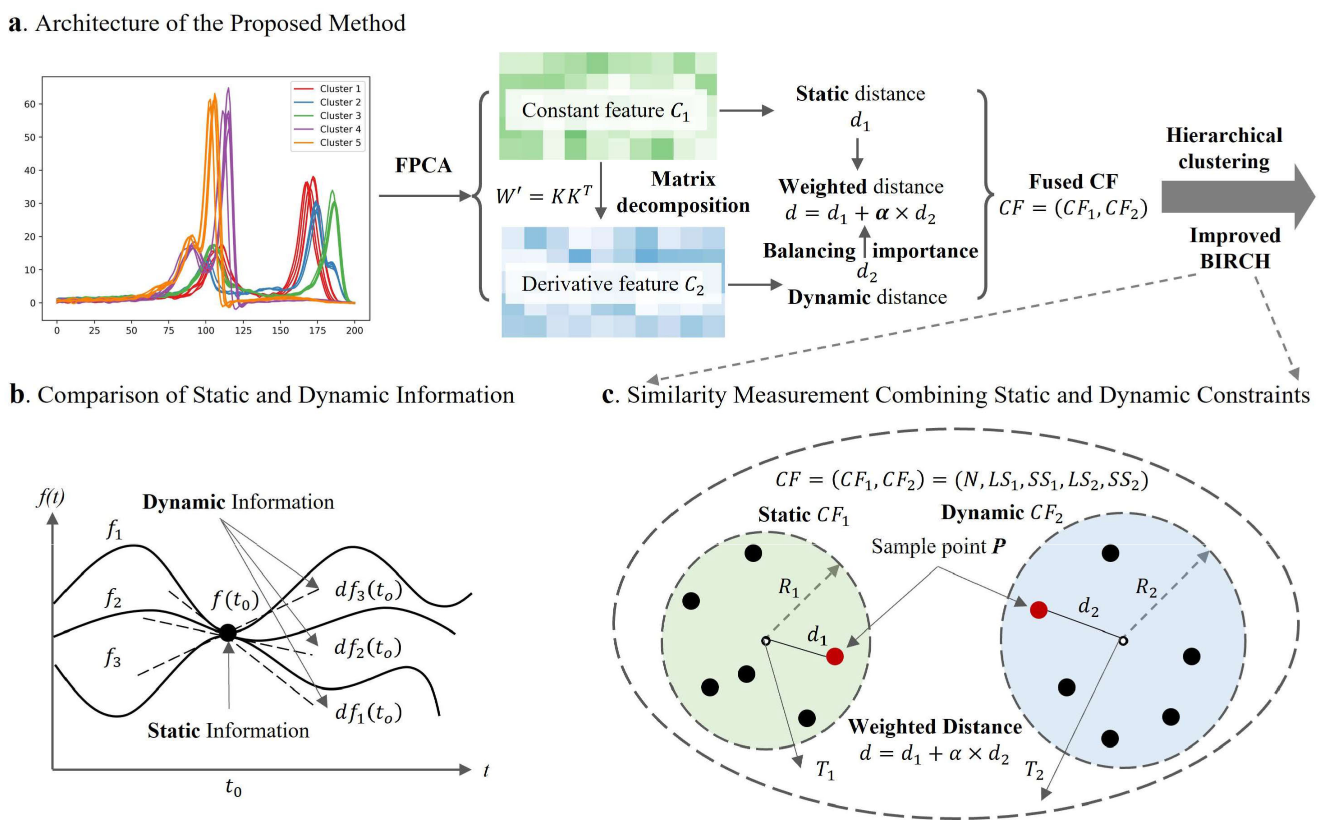

FIGURE 1. The framework of the proposed method in this paper. a, Method architecture, which consists of three main steps: Functional principal component analysis (FPCA), dynamic-static similarity weighted measurement, and improved BIRCH clustering. b, comparison of static and dynamic information in functional data. c, Key improvements of DSE-BIRCH. The left section represents the original BIRCH, where clustering is based solely on constant features. In contrast, the improved BIRCH incorporates both constant and derivative features, enabling the construction of CF trees while considering both static and dynamic constraints.

propose a clustering algorithm suitable for functional data with continuous characteristic.

According to the literature review, the BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) algorithm [44] tackles the challenge of managing large-scale clustering problems by incorporating explicit constraints on time and memory. It capitalizes on the observation that data points in the spatial domain frequently exhibit uneven distribution, implying that not all data points hold equal significance for clustering objectives. BIRCH significantly reduces the size of the dataset, making it much smaller than the original dataset. Studies [45], [46], [47] have shown that the BIRCH algorithm exhibits strong scalability and demonstrates notable efficiency advantages in clustering large-scale data while producing accurate clustering results. Therefore, based on the BIRCH algorithm, we propose a new approach to integrating static and dynamic information for measuring similarity in functional data clustering. Our goal is to develop an efficient, accurate clustering method for functional data that can uncover hierarchical structures.

To accurately distinguish the similarity and dissimilarity between functions, relying solely on static distance based on function values may not provide valuable information when functions have similar values in certain intervals but significant differences in their derivatives, as shown in Fig. 1b. Therefore, it is necessary to examine the dynamic shape features of functions. In order to leverage the high-order differentiability of functional data and address the issues mentioned earlier, this paper proposes a method based on the BIRCH algorithm that enhances the clustering of large-scale functional data by combining dynamic and static information. The architecture of our method is illustrated in Fig. 1a and mainly consists of three steps: functional principal component analysis (FPCA), similarity measurement based on weighted dynamic and static distances, and hierarchical clustering leveraging an improved BIRCH algorithm. Specifically, the method first employs FPCA to extract constant features from functional data, which are subsequently used to define the static similarity measure in the hierarchical clustering algorithm. When measuring the clustering similarity based on dynamic information, conventional multivariate feature clustering methods cannot be directly applied due to the involvement of distance calculations with non-unit matrices. To address this issue, this study introduces an additional matrix in conjunction with the original unit matrix through matrix factorization, thereby transforming the problem into a conventional multivariate clustering problem. As shown in Fig. 1c, the importance of constant and derivative features in clustering is balanced by setting optimal weights and creating a joint constraint that integrates static and dynamic similarity measurements. The improved BIRCH clustering algorithm explores the internal hierarchical relationships of functional data, leading to accurate and efficient clustering. To fulfill the requirements of functional data clustering, our approach extends the maintenance of a single clustering ball to the maintenance of two clustering spheres by introducing new triplets. The proposed method effectively compensates for the information loss in clustering based solely on static information, significantly reduces the complexity of the clustering algorithm, and enhances the efficiency of processing large-scale data.

II. METHOD

A. FUNCTIONAL PRINCIPAL COMPONENT ANALYSIS

Assuming that the function $x_i(t)$ represents square-integrable data in a Hilbert space that has been centered, functional principal component analysis (FPCA) seeks to identify a set of orthogonal unit bases in the Hilbert space. The main objective of FPCA is to capture maximum information from the original data while achieving dimensionality reduction.

Suppose $\phi_1(t), \phi_2(t), \ldots, \phi_p(t)$ are sets of orthonormal bases of $L^2(T)$, the original function can be expressed as a linear combination of these bases, $x_i(t) = \sum_{p=1}^{P} c_{ip}\phi_p(t)$, $p = 1, 2, \ldots, P$ where c_{ip} represents the score of the i-th sample in the p-th dimension. The first principal component in FPCA aims to find a function $\phi(t)$ that maximizes the variance of the projection of $x_i(t)$ onto $\phi(t)$, while satisfying the constraint that the norm of $\phi(t)$ is equal to 1.

The optimization problem for identifying the first principal component can be formulated as follows:

$$\begin{aligned} & max \sum_{i=1}^{n} c_i^2 = \max \frac{1}{n}\sum_{i=1}^{n} \left(\int_T \phi(t)\, x_i(t)dt\right)^2 \\ & s.t. \int_T \phi^2(t)dt = 1 \end{aligned} \tag{1}$$

To solve for the weight function of the j-th principal component, we need to satisfy the following optimization condition:

$$\begin{aligned} & \max \frac{1}{n}\sum_{i=1}^{n} \left(\int_T \phi_k(t)x_i(t)dt\right)^2 \\ & s.t. \int_T \phi_p^2(t)dt = 1, \\ & \quad \int_T \phi_k(t)\phi_p(t)\, dt = 0, \\ & \quad k = 1, 2, \ldots, p-1 \end{aligned} \tag{2}$$

Based on the Mercer theorem and the Karhunen-Loève (K-L) transform principle, the function $\phi(t)$ satisfies the characteristic equation [48]:

$$\int_T G(s,t)\phi(s) = \lambda\phi(t) \tag{3}$$

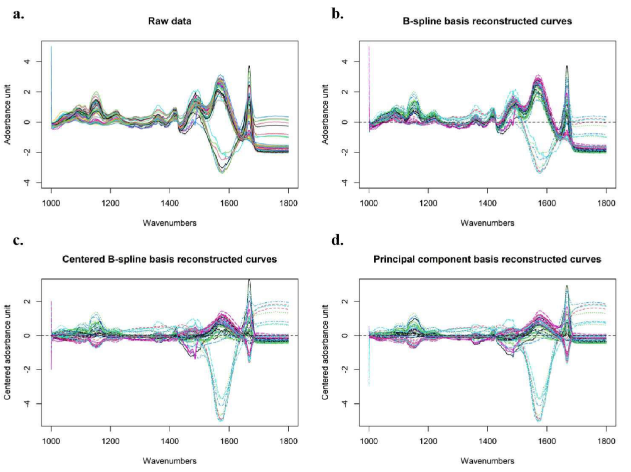

FIGURE 2. Reconstruction process of the Meat dataset. a, Raw curves constructed from the raw data points of the Meat dataset. b, Smoothing and denoising of the Meat dataset using B-spline basis functions. c, Centering the functional data obtained in Fig. 2b. d, Reconstruction of the centered functional data using the principal component bases.

where covariance function is $G(s,t) = \frac{1}{n}\sum_{i=1}^{n} x_i(s)\, x_i(t)$, and the value of $\phi(t)$ can be directly obtained through numerical integration.

Taking the Meat dataset [49] as an example, we demonstrated the process of utilizing FPCA for denoising and reconstruction of the data, as illustrated in Fig. 2. The dataset consists of 60 spectral curves categorized as chicken, pork, and turkey. These spectral curves were obtained through Attenuated Total Reflectance (ATR) sampling using Fourier Transform Infrared spectra. Each spectral curve is truncated within the wavenumber range of 1000 to 1800 cm^{-1}, resulting in 448 sampling points. Fig. 2 depicts the process of denoising and reconstruction on the first two principal component bases of the data. It can be observed that after denoising and reconstruction, the selected first two principal component bases effectively represent the function samples. The two principal component bases account for accumulative contribution variance of 94.7%, indicating their ability to explain a significant portion of the data variability and their importance as the most crucial components. Compared to the original data, the denoised and reconstructed data using principal component bases exhibit improved functional properties. The noise has been effectively removed, resulting in smoother and more regular reconstructed data. This enhanced representation of the data highlights the key features and provides better insights into the underlying characteristics.

B. THE SIMILARITY MEASUREMENT FOR FUNCTIONAL DATA CLUSTERING

After the function samples represented by the principal component bases, the mean function can be expressed using the following formula:

$$x_0(t) = \frac{1}{n}\sum_{i=1}^{n}\sum_{p=1}^{P} c_{ip}\phi_p(t) \tag{4}$$

At this point, the static distance between two function samples can be represented as follows

$$\begin{aligned} & d_1\left(x_i(t), x_j(t)\right) \\ & = (\int_T \left(x_i(t) - x_j(t)\right)^2 dt)^{\frac{1}{2}} \\ & = \left(\int_T \left(c_{ip}\phi_p(t) - c_{jp}\phi_p(t)\right)\left(c_{ip}\phi_p(t) - c_{jp}\phi_p(t)\right)^T\right)^{\frac{1}{2}} \\ & = \left(\left(c_i - c_j\right)\int_T \phi(t)\phi(t)^T \left(c_i - c_j\right)^T\right)^{\frac{1}{2}} \end{aligned} \tag{5}$$

where $\phi(t) = (\phi_1(t), \ldots, \phi_p(t))^T$, $c_i = (c_{i1}, c_{i2}, \ldots, c_{ip})^T$.

Let $\boldsymbol{W} = \int_T \phi(t)\phi(t)^T dt$, then

$$\boldsymbol{W} = \boldsymbol{I}_{P\times P} \tag{6}$$

$$d_1(x_i(t), x_j(t)) = ((c_i - c_j)(c_i - c_j)^T)^{\frac{1}{2}} \tag{7}$$

In this case, the static distance between function-based data is transformed into the Euclidean distance between coefficient vectors. Similarly, we can define the dynamic similarity measurement between two functional samples based on their derivatives as follows:

$$\begin{aligned} & d_2\left(x_i(t), x_j(t)\right) \\ & = (\int_T \left(Dx_i(t) - Dx_j(t)\right)^2 dt)^{\frac{1}{2}} \\ & = \left(\int_T \left(c_{ip}\phi'_p(t) - c_{jp}\phi'_p(t)\right)\left(c_{ip}\phi'{}_p(t) - c_{jp}\phi'{}_p(t)\right)^T\right)^{\frac{1}{2}} \\ & = \left(\left(c_i - c_j\right)\int_T \phi'(t)\phi'(t)^T \left(c_i - c_j\right)^T\right)^{\frac{1}{2}} \end{aligned} \tag{8}$$

where $Dx_i(t)$ and $\phi'_p(t)$ represent the firs-order derivative of the functional data and the basis functions respectively.

Let $\boldsymbol{W}' = \int_{\mathbf{T}} \phi'(t)\phi'(t)^T dt$, then

$$d_1(x_i(t), x_j(t)) = ((c_i - c_j)\boldsymbol{W}'(c_i - c_j)^T)^{\frac{1}{2}} \tag{9}$$

Meng et al. [50] have demonstrated that this distance satisfies the metric properties of a spatial distance measure on the function space $L^2(T)$.

C. THE CLUSTERING FEATURE (CF) IN THE ORIGINAL BIRCH

The CF triplet, which represents the clustering features, is the most crucial building element of the algorithm. By utilizing basic arithmetic operations to calculate the clustering features, the complexity is reduced and the computational speed is improved. The structure of CF is typically represented as a triplet in the following form:

$$CF = (N, LS, SS) \tag{10}$$

where N represents the number of points in the cluster, LS denotes the linear sum of the features for each sample, and SS is the sum of squares for each feature dimension of the data points. The calculation formulas for LS and SS are as follows:

$$LS = \sum\nolimits_{i=1}^{n} \vec{x}_i \tag{11}$$

$$SS = \sum\nolimits_{i=1}^{n} \vec{x}_i^2 \tag{12}$$

The CF features possess the property of linearity additivity. When merging two disjoint sub-clusters with CF features $CF_1 = (N_1, LS_1, SS_1)$ and $CF_2 = (N_2, LS_2, SS_2)$, a new cluster CF can be obtained that satisfies the additivity property. Mathematically, the additivity property is expressed as $CF = CF_1 + CF_2 = (N_1 + N_2, LS_1 + LS_2, SS_1 + SS_2)$.

For a cluster containing n samples of p dimensions, the calculation methods for cluster centroid x_0, cluster radius R, and cluster diameter D are as follows:

$$x_0 = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{13}$$

$$R = \sqrt{\frac{\sum_{i=1}^{n}(x_i - x_0)^2}{n}} = \sqrt{\frac{n * SS - LS^2}{n^2}} \tag{14}$$

$$D = \sqrt{\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\left(x_i - x_j\right)^2}{n(n-1)}} = \sqrt{\frac{2n * SS - 2LS^2}{n(n-1)}} \tag{15}$$

R represents the average distance from individual points within the cluster to the cluster centroid, while D represents the average pairwise distance between data points within the cluster. The proof of Equation 14 is provided in the Appendix. Given $X = \{x_1, x_2, x_3, \ldots, x_{n}1\}$, $Y = \{y_1, y_2, y_3, \ldots, y_{n2}\}$, the distance D_1 between two clusters, X and Y, can be calculated using the following formula:

$$D_1 = \left(\frac{\sum_{i=1}^{n_1}\sum_{j=n_1+1}^{n_1+n_2}\left(\vec{x}_i - \vec{y}_j\right)^2}{n_1 n_2}\right)^{\frac{1}{2}} \tag{16}$$

D. THE PRESENTATION OF CF FOR FUNCTIONAL DATA

When using the static distance d_1 as a similarity measure for clustering function samples, according to Equation 6, the matrix $\boldsymbol{W}$ is the identity matrix. Based on Equation 7, we can directly use the Euclidean distance between coefficient vectors, which represents the static distance between constant features, to cluster the functional data. This approach transforms functional data clustering into a conventional multivariate feature clustering problem. In this case, the representation of CF triplets remains consistent with the conventional CF representation method.

However, when using the dynamic distance d_2 as a similarity measure, $\boldsymbol{W}'$ is a non-identity matrix, which implies that generating the required CF triplets directly from the coefficient vectors is not feasible. Since $\boldsymbol{W}'$ is a symmetric matrix, we can perform a Cholesky decomposition of $\boldsymbol{W}'$ as $\boldsymbol{W}' = \boldsymbol{K}\boldsymbol{K}^T$. In this case, Equation 9 can be transformed as follows:

$$d_2\left(x_i(t), x_j(t)\right) = \left(\left(c_i - c_j\right) \boldsymbol{W}' \left(c_i - c_j\right)^T\right)^{\frac{1}{2}}$$
$$= \left(\left(c_i - c_j\right) \boldsymbol{K}\boldsymbol{K}^T (c_i - c_j)^T\right)^{\frac{1}{2}} \qquad (17)$$

*T*heorem 1. Assuming that CF_1 is the static information tuple generated from d_1, represented as $CF_1 = (N_1, LS_1, SS_1)$, the dynamic information triplet CF_2 generated from d_2 can be expressed as $CF_2 = (N, LS\boldsymbol{K}, \boldsymbol{K}SS\boldsymbol{K}^T)$.

Proof. Let $c_i = (c_{i1}, c_{i2}, \ldots c_{ip})$ be the projection coefficients of functional data on the principal component bases, and let $c_i' = c_i\mathbf{K}$. Denote $LS = \sum_{i=1}^{n} c_i$, $SS = \sum_{i=1}^{n} c_i^2$. According to Equation 17, we have:

$$d_2\left(x_i(t), x_j(t)\right) = \left(c_i\boldsymbol{K} - c_j\boldsymbol{K}\right)\left(\left(c_i - c_j\right)\boldsymbol{K}\right)^T$$
$$= \left(c_i' - c_j'\right)\left(c_i' - c_j'\right)^T \qquad (18)$$

This indicates that, similarly, function-based clustering using the d_2 distance measure can be transformed into a multivariate feature clustering based on the Euclidean distance. The information can be stored in CF triplets in a consistent manner. The original feature, represented by c_i, is transformed into the new feature $c_i\boldsymbol{K}$. In this case, CF_2 is obtained as $CF_2 = (N, LS\boldsymbol{K}, \boldsymbol{K}SS\boldsymbol{K}^T)$.

E. IMPROVED BIRCH WITH FUSION OF DYNAMIC AND STATIC INFORMATION

Intuitively, for functional data, conventional constant features reflect the static characteristics of the data, while derivative features capture the dynamic information from a different perspective. Dynamic information is crucial for functional data as it reveals the fundamental functional traits. In some cases, functions with approximately similar values in certain intervals may not provide valuable information to distinguish their differences. Therefore, relying solely on static distance based on functions may struggle to accurately differentiate the similarity and dissimilarity between functions. It is necessary to consider the dynamic shape features of functions, such as the velocity and acceleration of the derivative function. Incorporating supplementary dynamic information helps achieve more effective clustering.

By considering both constant and derivative information, the static distance d_1 and dynamic distance d_2 can be merged through a weighting parameter that balances their importance. This leads to the definition of a global distance:

$$d = d_1 + \alpha \times d_2 \qquad (19)$$

where α is the weight coefficient for the global distance measure. Let CF_1 and CF_2 be the CF vectors for the constant and derivative features, respectively. This global distance metric enables a comprehensive assessment of the similarity between functions. It can be represented as:

$$CF = (CF_1, CF_2)$$
$$= (N, LS_1, SS_1, LS_2, SS_2)$$
$$= \left(N, LS, SS, LS\boldsymbol{K}, \boldsymbol{K}SS\boldsymbol{K}^{\boldsymbol{T}}\right) \qquad (20)$$

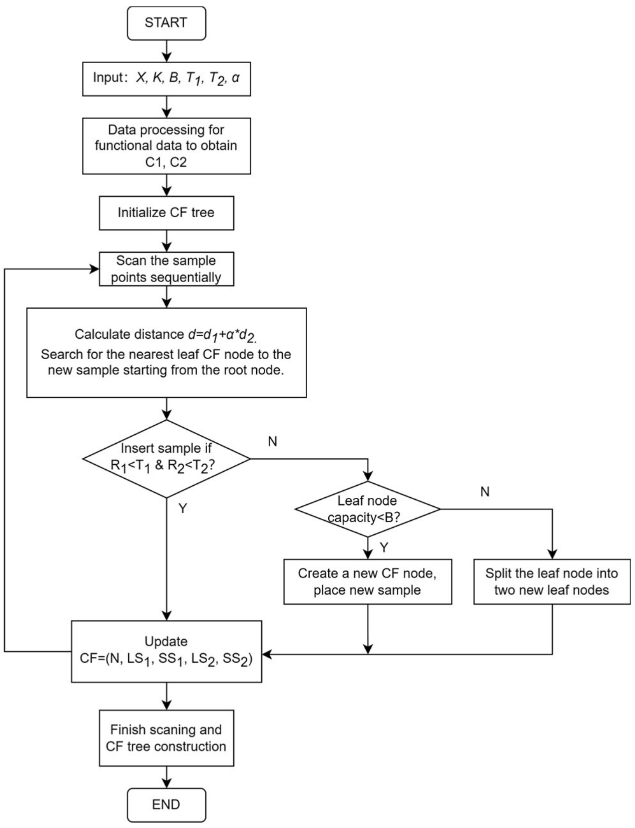
START
Input: X, K, B, T1, T2, α
Data processing for functional data to obtain C1, C2
Initialize CF tree
Scan the sample points sequentially
Calculate distance d=d1+α*d2. Search for the nearest leaf CF node to the new sample starting from the root node.
Insert sample if R1<T1 & R2<T2?
N
Y
Leaf node capacity<B?
N
Y
Create a new CF node, place new sample
Split the leaf node into two new leaf nodes
Update CF=(N, LS1, SS1, LS2, SS2)
Finish scaning and CF tree construction
END

FIGURE 3. **The workflow of the DSE-BIRCH algorithm.**

Given the relative magnitudes of constant and derivative quantities, the cluster sphere radius thresholds are defined as T_1 and T_2, respectively. The specific workflow of the DSE-BIRCH algorithm is illustrated in Fig. 3. DSE-BIRCH differs from the conventional BIRCH algorithm by concurrently maintaining spheres for both constant and derivative features. Supposing the two spheres are denoted as R_1 and R_2 to represent the respective radii. During the insertion of a new sample point, it will be placed in a leaf node if and only if $R_1 < T_1$ & $R_2 < T_2$ hold true.

Before constructing a CF tree, three parameters need to be defined in advance.

(1) The branch balance factor β, indicates the maximum permissible number of child nodes for every internal node, guaranteeing that the branching of internal nodes does not exceed this factor.

(2) The leaf balance factor λ, denotes the maximum capacity of each leaf node to CF.

(3) The threshold T, represents the maximum allowed sample radius or diameter for each CF within a leaf node.

It should be noted that the branch balance factor and the leaf balance factor can be set to the same value. The DSE-BIRCH algorithm is summarized in Algorithm 1. It operates on a matrix X, which contains m functional data samples of p dimensional principal components

Algorithm 1 Dynamic and Static Enhanced BIRCH Clustering Algorithm for Functional Data

```
Input: Data matrix $X$ of size $m \times n$, number of clusters $K$, branching factor $B$, thresholds $T_1$ and $T_2$, and weighting coefficient $\alpha$
Procedure:
1: Functional data processing
    a. Functional principal component analysis
    b. Compute the projections of the functional data onto the principal components to obtain $C_1$
    c. Compute the inner product $\mathbf{W}$ of the derivative of the basis functions, solving the equation $\mathbf{KK^T}= \mathbf{W}$, obtaining matrix
       $\mathbf{K}$ and the dynamic feature $C_2=C_1\mathbf{K}$
2: Initialize a CF tree with a single node
3: for $i = 0, 1, 2, \ldots$ do
    a. Compute the static and dynamic $d_1, d_2$ for sample $i$ based on corresponding features
    b. Measure the distance $d = d_1 + \alpha * d_2$, and find the leaf CF node closest to the new sample
    c. Insert the sample and check if it satisfies $R_1 < T_1 \& R_2 < T_2$. If true, update all CF triplets along the path and end the
       insertion. Otherwise, proceed to step d
    d. Check if the current leaf node has fewer CF nodes than $B$. If true, create a new CF node and place the new sample.
       Otherwise, proceed to step e
    e. Split the leaf node into two new leaf nodes, selecting the two CF nodes that are farthest apart as the initial CF nodes for
       the new leaf nodes
    f. Update all CF triplets along the path as $CF=(n, LS_1, SS_1, LS_2, SS_2)$
4: end for
5: All samples have been scanned, and the CF tree construction is complete.
Output: Cluster $Centroids = \{ct_1, ct_2, \ldots, ct_m$ for the $m$ samples and the classification results $Clusters = \{cl_1, cl_2, \ldots, cl_m\}$
```

projection coefficients. The algorithm takes several input parameters, including the maximum number of clusters K, the weighting coefficient α, and the radius thresholds T_1 and T_2, the branching factor $\beta = \lambda = B$. During the clustering process, the CF tree is constructed through iterative updates by sequentially scanning the dataset. Finally, the algorithm outputs the cluster partitioning results for m samples as $Clusters = \{cl_1, cl_2, \ldots, cl_m\}$, where each sample is assigned to a specific cluster represented by cl_i.

Additionally, the algorithm provides the centroids of each sample's corresponding cluster as $Centroids = \{ct_1, ct_2, \ldots, ct_m\}$. In our experiments, different hyperparameter ranges are considered:

$$\alpha \in \left\{10^{-5}, 10^{-4}, \ldots, 10^{4}, 10^{5}\right\},$$
$$T_2 \in \{1, 10, 100, T_1 = 0.2, B = 20.$$

The optimal hyperparameter settings may vary depending on the dataset. Notice that different values of α uniquely characterize the differences in importance between constant and derivative information across different datasets.

F. ALGORITHM COMPLEXITY ANALYSIS

Unlike most iterative clustering algorithms such as KMeans that require multiple iterations until convergence, the BIRCH clustering algorithm stands out by constructing the CF tree and completing the clustering process with just a single scan of the dataset. In this article, the primary motivation behind proposing DSE-BIRCH for clustering functional data is to harness the time complexity advantage of BIRCH, enabling fast and efficient clustering of large-scale functional data. Typically, the time complexity of KMeans is given by:

$$O(k * n * i * d)$$

where k represents the number of data points in the dataset, n denotes the size of the dataset, d signifies the number of iterations required for convergence, and d indicates the dimensionality of the data. Typically, the number of iterations in an algorithm is linearly related to the size of the dataset, which means that the variables i and n are of the same order. Therefore, the time complexity of the KMeans algorithm can be simplified as:

$$O(k{*}n{*}d)$$

In the case of the BIRCH algorithm, the data is compressed and clustered by the CF tree, leading to a significant reduction in the algorithm's time complexity to $O(nlogn)$, where n represents the size of the dataset.

During the construction process of the CF tree in DSE-BIRCH, the time complexity increases due to the calculation of distance measures for both constant and derivative features separately. This results in a doubling of the time overhead compared to BIRCH. However, despite this additional overhead, the time complexity of both DSE-BIRCH and BIRCH algorithms remains within the same order, ensuring that their overall time complexities remain unchanged as:

$$O(n\ log\ n)$$

III. RESULT

This section commences with a concise introduction to clustering evaluation metrics, providing a foundation for assessing the performance of clustering algorithms. Subsequently, the effectiveness of the DSE-BIRCH algorithm is evaluated by comparing it with the original KMeans and BIRCH algorithms on publicly available datasets. This comparison is aimed to validate the superiority of the DSE-BIRCH algorithm in terms of clustering quality and accuracy. Furthermore, the generalizability and robustness of the DSE-BIRCH algorithm are thoroughly examined through simulated experiments, using various synthetic datasets to assess its performance under diverse scenarios. Additionally, noise-injection experiments are conducted to investigate the algorithm's resilience against noisy data. Lastly, an in-depth analysis of the time complexity of the DSE-BIRCH algorithm is presented. This analysis sheds light on the algorithm's computational efficiency and scalability, providing valuable insights into its practical applicability for large-scale datasets.

A. CLUSTERING EVALUATION METRICS

Several commonly used clustering evaluation metrics are employed to assess the performance of the algorithm from multiple perspectives in our experiment. One of these metrics is the Rand Index (RI), which evaluates the consistency between the clustering results and a reference standard by measuring the allocation of samples to clusters. The Rand Index provides a value between 0 and 1, where a higher value indicates a higher degree of consistency between the clustering results and the reference standard. RI is defined as follows:

$$RI = \frac{a+b}{c_n^2} \tag{21}$$

where a is the number of correctly classified pairs, b is the number of incorrectly classified pairs, and c_n^2 represents the total number of samples pairs.

Similar to the RI, Purity also measures the consistency between the clustering results obtained by the algorithm and a reference standard in terms of sample allocation to clusters. Purity is defined as follows:

$$Purity = \frac{1}{N}\sum_{k=1}^{K} \max_{j=1}^{k} \mid c_k \cap t_j \mid \tag{22}$$

where N is the total number of samples, K is the number of clusters, c_k represents the sample set within each cluster, and t_j represents the sample set within each true class j.

The Davies-Bouldin Index (DB Index) measures the balance between the compactness and separation of clusters obtained by a clustering algorithm. It quantifies the average similarity between each cluster and its most similar cluster while considering the distances between their respective centroids. The DB index takes values in the range $[0, \infty)$, where a lower value indicates better clustering performance. It is defined as follows

$$DB \text{ Index } = \frac{1}{n}\sum_{i=1}^{n} \max_{j\neq i}\left\{\frac{s_i+s_j}{d_{ij}}\right\} \tag{23}$$

where n is the number of clusters, s_i represents the average distance between each sample in cluster i and its centroid, and d_{ij} represents the distance between the centroids of clusters i and j.

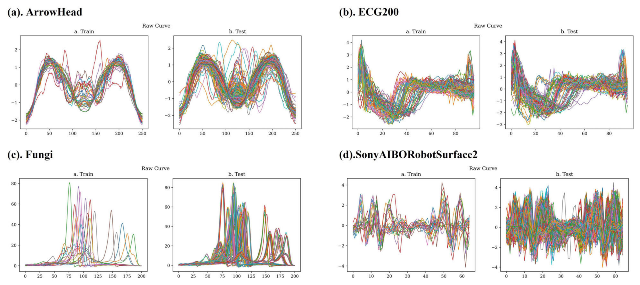

FIGURE 4. The curves of the four functional datasets. a-d are the curves of Arrowhead, ECG200, Fungi and SonyAIBORobotSurface2 dataset, respectively.

The Silhouette Coefficient (SC) measures the similarity and dissimilarity of a sample with its cluster compared to other clusters. The Silhouette Coefficient takes values in the range of [1, -1], where a higher value indicates a better clustering result. It is defined as follows:

$$SC = \frac{1}{N}\sum_{i=1}^{N}\frac{b(i)-a(i)}{\max\{a(i), b(i)\}} \tag{24}$$

where N is the total number of samples, a_i denotes the average distance between sample i and other samples within its cluster, and b_i represents the minimum average distance between sample i and all samples in other clusters.

B. EVALUATION ON PUBLIC DATASETS

In this study, a total of 10 time series datasets were selected from the UCR2018 Archive database [17] (available at www.cs.ucr.edu/∼eamonn/time_series_data_2018/).

These datasets were specifically chosen to encompass a wide range of real-life scenarios and were collected from diverse domains such as sensors, spectrum analyzers, images, and electrocardiograms. Table 1 presents a concise overview of these datasets. The inclusion of datasets from different domains ensures evaluating the algorithm's performance in handling various types of time series data, thus demonstrating its versatility and applicability across different real-world applications.

In Fig. 4, the time series plots of four datasets, namely ArrowHead, ECG200, Fungi, and SonyAIBORobotSurface2, are showcased. These plots provide a visual representation of the temporal patterns and distinctive features present in the respective datasets.

To examine the necessity and effectiveness of the design elements proposed in this study, we conducted ablation studies on the public datasets. Specifically, using the control variable method, we constructed the following baseline methods as control groups to assess the impact of introducing derivative features and the different approaches to incorporating them: KMeans-const, KMeans-comb, BIRCH-const, and BIRCH-comb. The methods with "const" as a suffix indicate the use of only constant features, while "comb" indicates that

TABLE 1. A summary of the 10 series datasets.

Name	Data Scale			Dataset Description
	Train	Test	Category	
ArrowHead	36	175	3	The dataset consists of 16 discrete points representing the outline of an arrow.
Beef	30	30	5	The dataset contains spectra of 60 dietary beef muscle samples.
ECG200	100	100	2	The dataset consists of 200 samples of electrocardiogram signals.
ECG5000	500	4500	5	The dataset comprises 5000 samples of electrocardiogram signals.
FaceFour	24	88	4	The dataset consists of four different directions of 2D facial contours.
Fungi	18	186	18	The dataset comprises 112 fungi images.
Herring	64	64	2	The dataset consists of 64 echograms for recognizing cracked herring.
SemgHandGenderCh2	300	600	2	The dataset contains EMG signals of hand muscles for both males and females.
SonyAIBORobotSurface2	27	953	2	The dataset consists of vibration signals from a robotic motion control system.
Strawberry	613	370	2	The dataset comprises 232 near-infrared spectral samples of strawberries.

TABLE 2. Clustering performance evaluation of different algorithms on the FaceFour dataset of 10 replicates.

Method/Metric	RI	Purity	DB Index	SC
KMeans-const	0.6951± 0.03	0.5644± 0.03	1.1911± 0.24	0.3147± 0.03
KMeans-comb	0.6895± 0.04	0.5518± 0.08	1.1657± 0.22	0.3032± 0.08
BIRCH-const	0.6790± 0.03	0.5442± 0.03	1.2133± 0.09	0.2403± 0.08
BIRCH-comb	0.6717± 0.06	0.5391± 0.08	1.2555± 0.27	0.2461± 0.09
DSE-BIRCH	0.7211± 0.02	0.6288± 0.01	1.0144± 0.10	0.3645± 0.03

TABLE 3. Clustering performance evaluation of different algorithms on the ECG5000 dataset of 10 replicates.

Method/Metric	RI	Purity	DB Index	SC
KMeans-const	0.7444± 0.02	0.9098± 0.00	1.5308± 0.07	0.3120± 0.01
KMeans-comb	0.7167± 0.00	0.9122± 0.00	1.2019± 0.03	0.3206± 0.01
BIRCH-const	0.7751± 0.04	0.9043± 0.01	1.4817± 0.22	0.3173± 0.04
BIRCH-comb	0.7413± 0.01	0.9087± 0.00	1.2749± 0.09	0.3064± 0.03
DSE-BIRCH	0.8671± 0.01	0.9211± 0.00	1.1484± 0.04	0.3885± 0.02

TABLE 4. Clustering performance evaluation of different algorithms on the SonyAIBORobotSurface2 dataset of 10 replicates.

Method/Metric	RI	Purity	DB Index	SC
KMeans-const	0.6350± 0.02	0.7594± 0.02	1.0055± 0.03	0.4459± 0.03
KMeans-comb	0.6288± 0.03	0.7524± 0.03	1.0265± 0.08	0.4344± 0.02
BIRCH-const	0.6030± 0.03	0.7258± 0.03	1.0863± 0.06	0.4181± 0.04
BIRCH-comb	0.6241± 0.03	0.7479± 0.03	1.0476± 0.03	0.4214± 0.04
DSE-BIRCH	0.6728± 0.02	0.7938± 0.02	0.9726± 0.02	0.4653± 0.01

the method incorporates both constant and derivative features. Regarding the approach to introducing derivative features, the baseline methods employed a simple weighted concatenation, where the constant features C_1 and derivative features C_2 are merged into the feature vector:

$$C = C_1|\alpha C_2$$

where α is the weight coefficient. "|" denotes the sequential concatenation of features along the feature dimension.

For each control groups, we randomly shuffled the train dataset with 10 different random seeds then conducted 10 repeated experiments. We reported the mean and standard deviation of the four performance evaluation metrics. The clustering results of the proposed DSE-BIRCH algorithm and the baseline algorithms on the 10 time series datasets can be found in Table 2 to Table 5, as well as in the Appendix Tables 13-18. The DSE-BIRCH algorithm outperforms the baseline algorithms in terms of various evaluation metrics, demonstrating its superior clustering capabilities. Higher values of the RI and Purity indicate better consistency between the clustering labels generated by DSE-BIRCH and the true labels. A lower DB Index suggests a higher ratio of compactness (intra-cluster distance) to separability (inter-cluster distance) in the clustering. Moreover, a higher Silhouette

TABLE 5. **Clustering performance evaluation of different algorithms on the Fungi dataset of 10 replicates.**

Method/Metric	RI	Purity	DB Index	SC
KMeans-const	0.8646± 0.02	0.4592± 0.03	0.9103± 0.06	0.1901± 0.05
KMeans-comb	0.8789± 0.02	0.5290± 0.03	0.8617± 0.12	0.3245± 0.06
BIRCH-const	0.8550± 0.02	0.4140± 0.03	0.8493± 0.06	0.3081± 0.05
BIRCH-comb	0.8789± 0.02	0.5296± 0.03	0.8055± 0.08	0.3429± 0.05
DSE-BIRCH	0.8809± 0.01	0.5387± 0.03	0.7879± 0.08	0.3693± 0.05

Coefficient reflects that data points within the same cluster are closer to each other and farther away from data points in other clusters. These metrics collectively demonstrate the effectiveness of the DSE-BIRCH algorithm from different perspectives.

The inclusion of derivative information improves the clustering performance on the 10 datasets, indicating the necessity of incorporating derivative features for clustering function-based data. For instance, on the Fungi dataset, simply combining the dynamic features with the static information resulted in a 1.62% mean improvement in RI for KMeans and a 2.72% mean improvement for BIRCH, with the values increasing from 0.8646 and 0.8550 to 0.8789 and 0.8789, respectively. The Silhouette Coefficients for both algorithms increased from 0.1901 and 0.3081 to 0.3245 and0.3429, respectively, exhibiting mean improvements of 62.1% and 11.3%. When the static and dynamic features are fused using the weighted combination approach in DSE-BIRCH and the global clustering hypersphere is maintained through the dual constraints, DSE-BIRCH achieves further significant mean improvements of 7.7% in SC, surpassing the results of the best baseline methods with values of 0.3429. For more experimental results, please refer to Tables 2-5 and the Appendix.

However, as shown on the majority of the comparative results, it is worth noting that simply weighted concatenation of derivative features and constant features may not necessarily improve the clustering performance (as shown in the Appendix). The effective fusion of derivative information is crucial for achieving more accurate clustering results. Compared to simple weighted concatenation, DSE-BIRCH iteratively maintains separate spheres for constant and derivative features, thereby preserving their individual properties while integrating their similarities.

The above results of 10 replicates characterize the magnitude of the performance improvements enabled by the effective derivate information incorporation on our model. To further investigate the significant level of the differences between our propose model and the baseline models, we conducted three statical hypothesis tests. Specifically, Wilcoxon's Signed-Rank Test is utilized to compare whether there is a significant difference in the medians of two paired performances from the proposed model and a baseline model. And the paired t-test is used to compare whether there is a significant difference in the means of paired performances. Furthermore, the one-tailed test is used to test whether the performance metrics of DSE-BIRCH are significantly better than the baseline models. We compared our model and the ablated models in pairs, resulting in four groups denoted as G1 to G4. We used the Rand Index and Silhouette Coefficient from the 10 replicates as paired samples, respectively. Their respective comparison results are summarized in Table 6 and Table 7. The results show that our proposed model exhibits $p\_values < 0.05$ in nearly all of the four comparison groups under the three statistical tests, indicating statistically significant differences between the full model and ablated models. Specifically, the statistics from Wilcoxon's Signed-Rank Test and paired T-test suggest significant performance differences between our proposed model and the ablated models, while the one-sided tests further indicate these significant differences are advantages of our model. Together, these statistical tests better characterize the significance of the performance improvements enabled by our approach.

In summary, the 10 repeated random experiments and statistical hypothesis testing jointly indicated that our proposed model demonstrates significantly superior performance over the baseline models, highlighting the advantages of our model and reflecting the efficacy of our innovations and model design.

To provide a more intuitive visualization of the clustering results, the data points in the clustering results were presented as scatter plots, where different colors represent different clusters. Fig. 5a presents the clustering results of the Fungi dataset after performing FPCA and retaining two principal components. The x and y axes represent the two dimensions of the constant features. On the other hand, Fig. 5b displays the clustering results of the SonyAIBORobotSurface2 dataset after performing FPCA and retaining ten principal components. The three dimensions of the coordinate axes were obtained by reducing the constant features using T-SNE. From these figures, it can be observed that there is a high consistency between the predicted clusters and the true clusters, indicating that the clustering results align well with the true distribution.

C. SIMULATION EXPERIMENTS

To further validate the advantages of the fusion approach in the DSE-BIRCH algorithm compared to the simple weighted concatenation approach, this section conducts simulation experiments using three synthetic datasets from the UCR Archive 2018 database: BME, SyntheticControl, and UMD. The time series curves of the UMD dataset in the training set are shown in Fig. 6.

The control groups for this experiment are DSE-BIRCH, KMeans-comb, and BIRCH-comb. The experimental results

TABLE 6. Statistical tests on rand index of 10 replicates.

Dataset	Wilcoxon's Rank-Sum Test				Paired T-Test				One-Tailed Test			
	G1	G2	G3	G4	G1	G2	G3	G4	G1	G2	G3	G4
ArrowHead	0.00407	0.00016	0.00029	0.00038	0.0666	2×10^{-14}	5×10^{-5}	0.00059	0.00015	9×10^{-15}	9×10^{-9}	2×10^{-13}
Beef	0.16200	0.00016	0.00171	0.00016	0.0562	6×10^{-5}	0.00152	8×10^{-8}	2×10^{-11}	1×10^{-11}	5×10^{-15}	2×10^{-14}
ECG200	0.00035	0.00049	0.00357	0.00035	0.00040	0.01370	0.01780	2×10^{-5}	8×10^{-7}	6×10^{-9}	7×10^{-6}	5×10^{-11}
ECG5000	0.00016	0.00016	0.00016	0.00016	3×10^{-9}	2×10^{-11}	2×10^{-5}	3×10^{-9}	8×10^{-11}	1×10^{-11}	1×10^{-9}	7×10^{-11}
FaceFour	0.05760	0.12200	0.00708	0.03410	0.00020	0.00622	0.00055	0.00409	0.00503	0.00179	0.00031	0.00011
Fungi	0.13100	0.40600	0.01560	0.40600	0.00211	4×10^{-6}	5×10^{-5}	3×10^{-6}	0.00351	0.34200	0.00019	0.34200
Herring	0.00016	0.00407	0.00016	0.00016	4×10^{-6}	4×10^{-5}	4×10^{-6}	7×10^{-5}	2×10^{-7}	0.00040	1×10^{-7}	6×10^{-7}
SemgHand..	0.00016	0.00016	0.00016	0.00016	3×10^{-10}	4×10^{-6}	4×10^{-6}	0.00025	6×10^{-13}	4×10^{-11}	2×10^{-11}	9×10^{-12}
SonyAIBO..	0.00194	0.00115	0.00029	0.00115	0.00133	0.00125	0.00014	0.00022	5×10^{-5}	2×10^{-5}	3×10^{-7}	7×10^{-6}
Strawberry	0.00016	0.00016	0.00051	0.00033	4×10^{-8}	2×10^{-7}	1×10^{-5}	7×10^{-5}	2×10^{-8}	9×10^{-8}	3×10^{-6}	1×10^{-6}

TABLE 7. Statistical tests on silhouette coefficient of 10 replicates.

Dataset	Wilcoxon's Rank-Sum Test				Paired T-Test				One-Tailed Test			
	G1	G2	G3	G4	G1	G2	G3	G4	G1	G2	G3	G4
ArrowHead	0.001940	0.000157	0.000157	0.000157	0.001920	4×10^{-7}	3×10^{-6}	4×10^{-8}	4×10^{-7}	1×10^{-11}	1×10^{-10}	4×10^{-11}
Beef	0.015600	0.000212	0.002500	0.000157	0.017700	4×10^{-5}	0.000762	9×10^{-10}	0.000180	8×10^{-8}	7×10^{-6}	2×10^{-10}
ECG200	0.004110	0.00232	0.00268	0.000487	6×10^{-6}	0.055200	0.002380	0.001620	9×10^{-5}	1×10^{-7}	9×10^{-6}	8×10^{-11}
ECG5000	0.000157	0.000157	0.000670	0.000157	2×10^{-6}	7×10^{-7}	0.000350	6×10^{-5}	1×10^{-7}	3×10^{-7}	2×10^{-7}	6×10^{-8}
FaceFour	0.001720	0.057600	0.000349	0.007080	0.001060	0.020800	0.002910	0.004800	0.000553	0.000142	8×10^{-7}	1×10^{-6}
Fungi	0.000157	0.049400	0.010200	0.131000	2×10^{-8}	0.000258	0.000139	4×10^{-14}	3×10^{-7}	0.006360	0.001110	0.050500
Herring	0.000285	0.000285	0.000285	0.000157	0.00024	0.000981	2×10^{-5}	1×10^{-5}	2×10^{-6}	0.000412	2×10^{-6}	7×10^{-7}
SemgHand..	0.000157	0.000212	0.000157	0.000157	3×10^{-5}	1×10^{-5}	3×10^{-6}	0.089600	1×10^{-10}	9×10^{-7}	1×10^{-8}	3×10^{-11}
SonyAIBO..	0.112000	0.000246	0.000157	0.000157	0.095100	0.002030	0.004720	0.006010	5×10^{-7}	8×10^{-9}	2×10^{-10}	4×10^{-10}
Strawberry	0.000157	0.000157	0.000507	0.000507	7×10^{-7}	2×10^{-7}	9×10^{-6}	0.001740	3×10^{-7}	4×10^{-8}	2×10^{-5}	3×10^{-11}

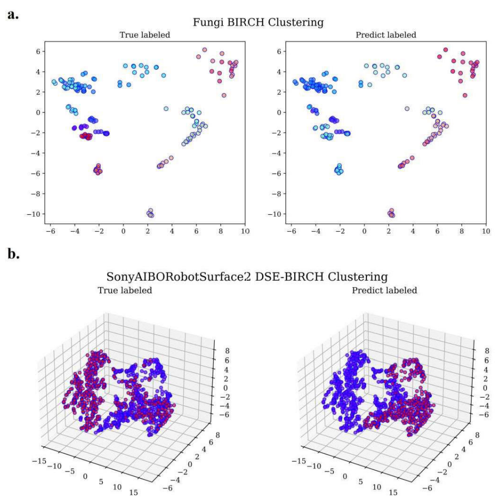

FIGURE 5. Visualization of clustering results on the fungi and SONYAIBOROBOTSURFACE2 datasets.

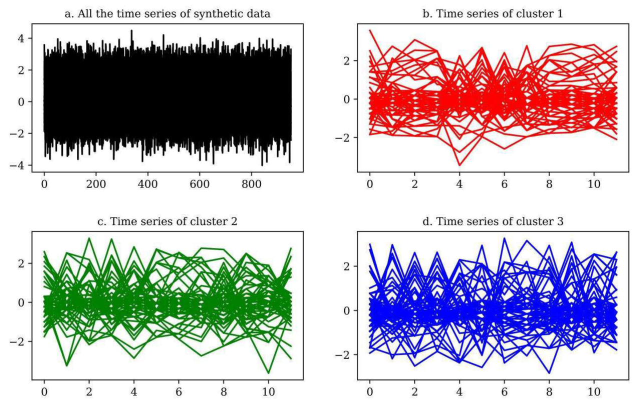

FIGURE 6. The time series curves of the UMD synthetic dataset.

of the three algorithms are shown in Table 8. Taking the BME dataset as an example, DSE-BIRCH outperforms the two baseline models with a lead of 31.9% and 32.4% in terms of RI, and achieves improvements of 40.2% and 63.8% in terms of SC. The outstanding performance of DSE-BIRCH on the three synthetic datasets demonstrates its superior ability to integrate derivative information, further confirming

TABLE 8. Clustering results of combining constant and derivative information on synthetic time series datasets.

Dataset	Method	RI	Purity	DB Index	SC
BME	DSE-BIRCH	**0.8674**	**0.6600**	**1.5155**	**0.2680**
	KMeans-comb	0.6577	0.4233	1.3483	0.1911
	BIRCH-comb	0.6553	0.4400	1.4222	0.1636
Synthetic-Control	DSE-BIRCH	**0.6486**	**0.6200**	**1.2353**	**0.3210**
	KMeans-comb	0.6313	0.5333	1.2181	0.3207
	BIRCH-comb	0.6203	0.5133	1.1411	0.3376
UMD	DSE-BIRCH	**0.6458**	**0.5486**	**1.4792**	**0.2553**
	KMeans-comb	0.5901	0.5694	1.7467	0.2548
	BIRCH-comb	0.5385	0.5278	1.3502	0.2871

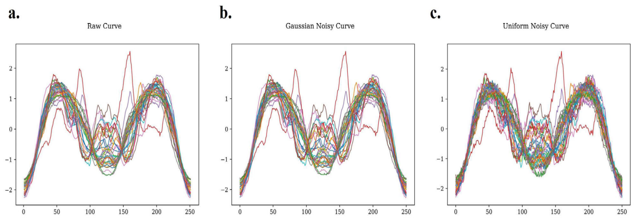

FIGURE 7. Time series curves of the arrowHead training dataset before and after noise addition. a, Raw curve. b, Curve after adding Gaussian noise. c, Curve after adding uniform noise.

its effectiveness in incorporating and utilizing derivative information.

D. ROBUSTNESS ANALYSIS WITH NOISE ADDITION

To investigate the robustness of the DSE-BIRCH algorithm, two types of noise, Gaussian noise and uniform noise, were introduced to the 10 datasets used in the evaluation section. In our experiment, Gaussian noise was generated from a Gaussian distribution with a mean of 0 and a standard deviation of 0.01, while uniform noise was generated from a uniform distribution with values ranging from -1 to 1. The noisy data was obtained by adding these noises to the original data. Taking the ArrowHead dataset as an example, the time series curves of the ArrowHead dataset before and after adding noises are shown in Fig. 7.

The experimental results are shown in Fig. 8. It can be observed that after adding Gaussian noise and uniform noise, the metrics RI and Purity only slightly decrease overall, while the changes in DB Index and SC are relatively small. In absolute terms, higher values of RI, Purity, and SC, and lower values of DB Index, indicate high consistency between the predicted clusters and the true clusters, with compact within-cluster structure and well-separated between-cluster structure. The experimental results demonstrate that the addition of noise did not significantly affect the clustering performance of the DSE-BIRCH algorithm. The DSE-BIRCH algorithm exhibits good robustness, maintaining high clustering quality even in the presence of noise, suggesting its wide applicability in real-life scenarios.

E. ALGORITHM COMPLEXITY EXPERIMENT

To explore the computational efficiency of the DSE-BIRCH algorithm under different experimental scenarios, we conducted controlled experiments in this subsection. Initially, we constructed a high-dimensional function-based dataset comprising 7000 samples. By applying FPCA, we retained 2 to 10 principal components, yielding corresponding constant features C_1 and derivative features C_2. Subsequently, using the controlled variables method, we measured the algorithm's computational time under two experimental setups: (1) fixed feature dimensions and varying sample sizes (Setup 1), and (2) fixed sample sizes and varying feature dimensions (Setup 2).

In Setup 1, we held the feature dimensions constant at 10 and systematically increased the sample sizes from 1000 to 7000, repeating the experiment for 5 rounds. The average time with standard deviation as the confidence interval is shown in Fig. 9a. It can be observed that the algorithm based on BIRCH had significantly lower computational time cost compared to the algorithm based on KMeans, and the DSE-BIRCH algorithm had approximately twice the computational time of the BIRCH-comb baseline. Moving to Setup 2, we maintained the sample size at 7000 while linearly increasing the feature dimensions from 2 to 10, repeating the experiment five times. Fig. 9b reveals a similar trend to Fig. 9a, underscoring the computational efficiency of the DSE-BIRCH algorithm. Although the computational time of DSE-BIRCH is an integer multiple of the two baseline algorithms based on BIRCH, it remains significantly smaller than that of the baseline method relying on KMeans.

The experimental findings highlight the DSE-BIRCH algorithm's effective trade-off between clustering accuracy and computational cost. Despite incurring a computational time twice that of the BIRCH-comb baseline, the DSE-BIRCH algorithm exhibits substantial improvements in clustering performance compared to the KMeans-based baseline method. Moreover, it demonstrates efficiency and scalability when handling large-scale datasets, ensuring accurate clustering. These results establish the practical feasibility and effectiveness of the DSE-BIRCH algorithm.

F. EMPIRICAL APPLICATION ANALYSIS

To further verify the practicality and generalizability of the DSE-BIRCH algorithm, we conducted experiments on three real-world datasets of three different scales. Specifically, a dataset with a small number of records but a

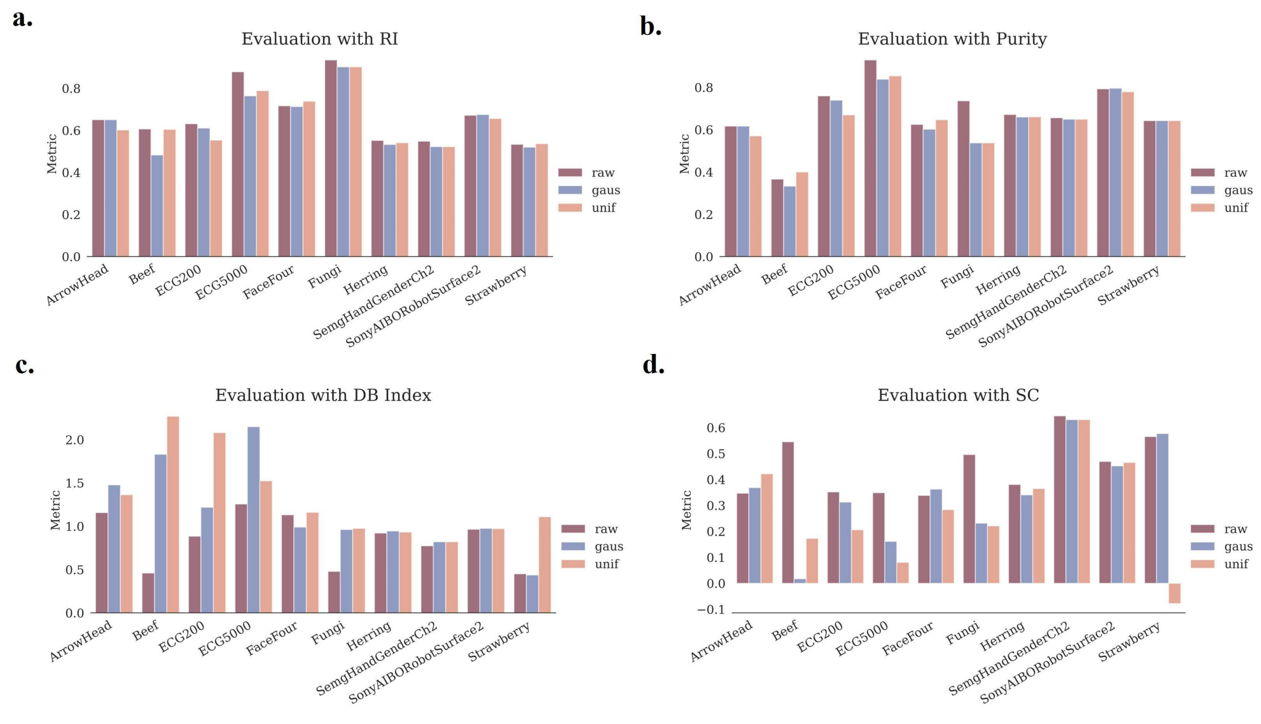

FIGURE 8. Evaluation of DSE-BIRCH clustering performance before and after noise addition. a-d are the comparisons of evaluation results for metrics RI, Purity, DB Index, and SC, respectively.

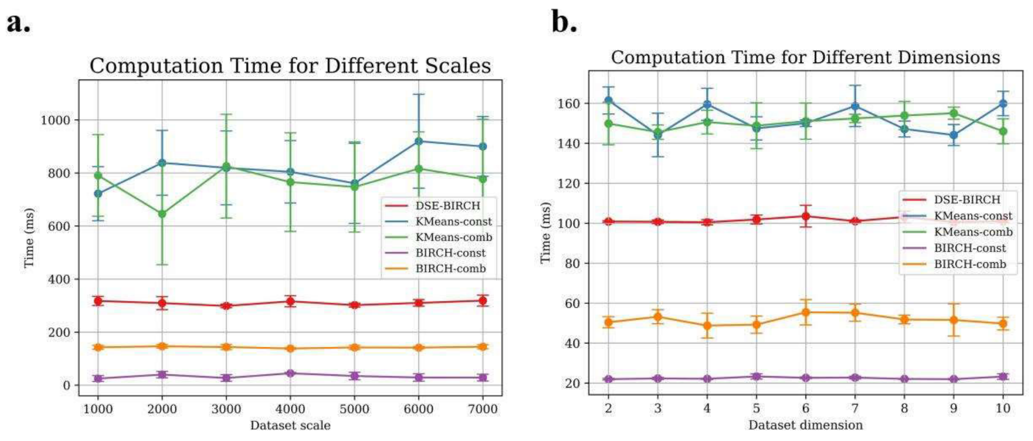

FIGURE 9. The variation of computational cost for DSE-BIRCH clustering concerning changes in the scale of dataset samples and feature dimensions. a, Comparison of computational cost for different dataset scales across five algorithms. b, Comparison of computational cost for different dimensions across five algorithms.

large number of attributes, a dataset with a small number of attributes but a large number of records, and a dataset with a large number of records and a large number of attributes are collected from standard UCI repository (https://archive.ics.uci.edu/datasets). Correspondingly, the three datasets are Musk (Version 1) [51], Online Shoppers Purchasing Intention [52], and Multiple Features. The summary information for the three is presented in Table 9. Specifically, the Musk dataset describes a set of 92 molecules of which 47 are judged by human experts to be musks and the remaining 45 molecules are judged to be non-musks. The goal is to learn to predict whether new molecules will be musks or non-musks. The Online Shoppers Purchasing Intention dataset contains 12330 sessions over time. And the

TABLE 9. Summary of the three real-world datasets.

Dataset	Samples	Features	Clusters
Musk (Version 1)	476	169	2
Online Shoppers Purchasing Intention	12330	17	2
Multiple Features	2000	649	10

TABLE 10. Clustering performance comparison on the musk dataset of 10 replicates.

Method/Metric	RI	Purity	DB Index	SC
KMeans-const	0.5013± 0.00	0.5635± 0.00	0.7035± 0.01	0.5611± 0.01
KMeans-comb	0.5017± 0.00	0.5635± 0.00	0.7048± 0.01	0.5597± 0.00
BIRCH-const	0.5005± 0.00	0.5635± 0.00	0.7078± 0.03	0.5573± 0.02
BIRCH-comb	0.4976± 0.00	0.5635± 0.00	0.7604± 0.03	0.4999± 0.04
DSE-BIRCH	0.5032± 0.00	0.5646± 0.00	0.6830± 0.01	0.5701± 0.01

TABLE 11. Clustering performance comparison on the online shoppers purchasing intention dataset of 10 replicates.

Method/Metric	RI	Purity	DB Index	SC
KMeans-const	0.7019± 0.02	0.8452± 0.00	0.4986± 0.06	0.7737± 0.02
KMeans-comb	0.7158± 0.01	0.8452± 0.00	0.4459± 0.06	0.7936± 0.02
BIRCH-const	0.7015± 0.04	0.8452± 0.00	0.4468± 0.13	0.7872± 0.06
BIRCH-comb	0.7015± 0.03	0.8452± 0.00	0.4666± 0.12	0.7811± 0.05
DSE-BIRCH	0.7392± 0.00	0.8460± 0.00	0.2206± 0.02	0.8631± 0.01

TABLE 12. Clustering performance comparison on the multiple features dataset of 10 replicates.

Method/Metric	RI	Purity	DB Index	SC
KMeans-const	0.8813± 0.01	0.5312± 0.03	1.2277± 0.08	0.3095± 0.01
KMeans-comb	0.8792± 0.00	0.5350± 0.01	1.1526± 0.07	0.3189± 0.00
BIRCH-const	0.8749± 0.01	0.5125± 0.03	1.2514± 0.05	0.2958± 0.01
BIRCH-comb	0.8800± 0.01	0.5437± 0.04	1.1246± 0.02	0.3108± 0.03
DSE-BIRCH	0.8944± 0.00	0.5769± 0.02	1.0635± 0.03	0.3518± 0.01

Multiple Features dataset consists of features of handwritten numerals extracted from a collection of Dutch utility maps. The clustering experiments on the three datasets are conducted 10 times with 10 random seeds. The comparative performance of our proposed model and the baseline models on the three datasets are demonstrated in Table 10-12, respectively. DSE-BIRCH consistently outperformed the baseline models with significant advantages, demonstrating its generalizability in complicated application scenarios and its broad applicability in the real world.

IV. CONCLUSION AND DISCUSSION

In the era of big data, the application of functional data in various fields such as medicine, biology, and finance has gained significant traction. However, clustering algorithms specifically designed for large-scale functional data remain a research challenge. Many existing methods in this field tend to overlook the dynamic information present in functional data and primarily rely on conventional static features, failing to fully exploit the continuous and differentiable characteristics of functional data. Additionally, investigating the integration of static and dynamic information in a complementary manner is also an important research question. Addressing these issues, this study proposes DSE-BIRCH, which aims to integrate both static and dynamic information and establish a comprehensive measure of similarity. The DSE-BIRCH algorithm takes into account the similarity between functions from both static and dynamic perspectives, thereby enabling a more comprehensive utilization of the inherent characteristics of functional data. The algorithm first performs principal component analysis on the functional data to extract the static features, which are then transformed into dynamic features using our newly proposed matrix factorization-based method. By integrating these two types of features, a global feature is obtained, and static and dynamic similarity measures are defined based on these features. The importance of these features in clustering is balanced by assigning weights to the global similarity measure. Building upon an improved BIRCH algorithm, DSE-BIRCH performs hierarchical clustering, leveraging the algorithm's capacity to capture hierarchical information and handle large-scale datasets. As a result, DSE-BIRCH effectively captures the hierarchical structure within high-dimensional functional data and enables efficient clustering of large datasets.

The fundamental differences between DSE-BRICH from the other similar algorithms are uniformity, simplicity, efficiency, and theoretical basis. We ingeniously implemented the simultaneous measurement of static and dynamic distance metrics in a unified framework through a novel matrix decomposition approach, while in other methods the two are separated. This makes the heterogeneity between the constant and derivative features constructed by our algorithm lower, thus enabling better integration and performance when fusing them. Additionally, the unified framework makes our algorithm more concise, meaning stronger robustness and better generalization.

Experimental results on the public and synthetic datasets demonstrate that DSE-BIRCH significantly outperforms baseline methods in terms of multiple evaluation metrics. By fully leveraging the differentiability of functional data, DSE-BIRCH achieves superior clustering accuracy, efficiency, and granularity compared to existing methods. Moreover, the robustness of DSE-BIRCH against noise interference is demonstrated through a noise experiment. The algorithm maintains high clustering quality even in the presence of noise, making it highly applicable in real-world scenarios. Furthermore, complexity experiments highlight the favorable trade-off between time complexity and clustering accuracy achieved by DSE-BIRCH, showcasing its practical feasibility and effectiveness in clustering large-scale functional data. The empirical study on the three real-world datasets demonstrates the generalizability of DSE-BIRCH in complicated application scenarios and its broad applicability in the real world. The superior performance of the model on these datasets of varying scales demonstrates the adaptability of our algorithm across different application scenarios. Thanks to the lower time complexity of the BIRCH, our algorithm can be effectively extended to large-scale functional data clustering scenarios. We hope that our method can inspire more researchers to explore clustering studies that fully exploit the inherent characteristics of functional data. To facilitate further research, our code is publicly available at https://github.com/SallyLi0606/DSE-BIRCH.

Although this study focuses on first-order derivatives as the derivative information, future research directions include exploring higher-order derivatives and effectively integrating them into the clustering process. Additionally, further investigation into non-empirical methods for accurate weighting fusion of constant and derivative features, such as determining the optimal weighting coefficient, would contribute to enhancing the performance of the DSE-BIRCH algorithm. While this study verifies that fusing dynamic and static information can enhance clustering performance, and further shows the value of higher order derivative data for functional data clustering, it remains unclear if adding more higher order derivative information will continue improving results, and how many orders of derivatives are optimal. Further investigation is still needed to determine the impact of including increasing orders of derivatives on clustering performance. In addition, the method proposed in this study primarily addresses the fusion of dynamic and static information for univariate functional data, and does not consider the fusion approach for multivariate functional data. As future work, principal component analysis on multivariate functional data could be incorporated to extend the application of DSE-BIRCH to multivariate functional data clustering.

APPENDIX A
RROOF OF THEOREM

Theorem 2. $R = \sqrt{\frac{\sum_{i=1}^{n}(x_i - x_0)^2}{n}} = \sqrt{\frac{NSS - LS^2}{N^2}}$

Proof. Assuming $x_i = (x_{i1}, x_{i2}, \ldots, x_{iq})$, if a cluster contains n samples and its centroid is denoted as x_0, then it follows that $R = \sqrt{\frac{\sum_{i=1}^{n}(x_i - x_0)^2}{n}}$. The proof is as follows:

$$\begin{aligned}
&\sum_{i=1}^{n}(x_i - x_0)^2 \\
&= \sum_{i=1}^{n}(x_{i1} - x_{01})^2 + (x_{i2} - x_{02})^2 + \ldots + \left(x_{ip} - x_{0p}\right)^2 \\
&= \sum_{i=1}^{n}\left(x_{i1}^2 + nx_{o1}^2 - 2nx_{i1}x_{01}\right) \\
&\quad + \sum_{i=1}^{n}\left(x_{i2}^2 + nx_{o2}^2 - 2nx_{i1}x_{02}\right) \\
&\quad + \ldots + \sum_{i=1}^{n}\left(x_{ip}^2 + nx_{op}^2 - 2nx_{i1}x_{0p}\right) \\
&= \sum_{i=1}^{n}\left(x_{i1}^2 + nx_{o1}^2\right) + \sum_{i=1}^{n}\left(x_{i2}^2 + nx_{o2}^2\right) + \sum_{i=1}^{n}\left(x_{ip}^2 + nx_{op}^2\right) \\
&= \sum_{i=1}^{n}\left(x_{i1}^2 + x_{i2}^2 + \ldots + x_{ip}^2\right) \\
&\quad - n\sum_{i=1}^{n}\left(x_{01}^2 + x_{02}^2 + \ldots + x_{0p}^2\right) \\
&= SS - nx_o x_o^T \\
&= SS - \frac{LS^2}{n}
\end{aligned}$$

TABLE 13. Clustering performance evaluation of different algorithms on the arrowHead dataset of 10 replicates.

Method/Metric	RI	Purity	DI	SC
KMeans-const	0.6275± 0.03	0.5789± 0.02	1.1743± 0.01	0.2974± 0.03
KMeans-comb	0.3466± 0.01	0.3983± 0.00	1.4556± 0.17	0.1998± 0.04
BIRCH-const	0.5837± 0.03	0.5509± 0.02	1.1945± 0.03	0.2360± 0.03
BIRCH-comb	0.4309± 0.14	0.4554± 0.09	1.2879± 0.02	0.2208± 0.02
DSE-BIRCH	0.6473± 0.01	0.6034± 0.01	1.1573± 0.01	0.3399± 0.01

APPENDIX B
EXPERIMENT RESULT

See Tables 13–18.

APPENDIX C
PROOF OF CONVERGENCE

BIRCH is a heuristic aggregation-based algorithm that determines merging by iteratively computing "clustering feature distances" without an objective function. Since the iteration process of the BIRCH algorithm is a non-convex optimization problem and does not yield a global optimum, its convergence

TABLE 14. Clustering performance evaluation of different algorithms on the beef dataset of 10 replicates.

Method/Metric	RI	Purity	DB Index	SC
KMeans-const	0.5618± 0.07	0.3634± 0.01	0.5611± 0.08	0.4812± 0.04
KMeans-comb	0.5586± 0.02	0.3233± 0.03	0.5845± 0.06	0.4060± 0.06
BIRCH-const	0.4851± 0.09	0.3667± 0.00	0.6051± 0.09	0.4564± 0.04
BIRCH-comb	0.5050± 0.02	0.3200± 0.02	0.7495± 0.06	0.2828± 0.03
DSE-BIRCH	0.6101± 0.00	0.3777± 0.01	0.4805± 0.05	0.5278± 0.03

TABLE 15. Clustering performance evaluation of different algorithms on the ECG200 dataset of 10 replicates.

Method/Metric	RI	Purity	DB Index	SC
KMeans-const	0.6056± 0.02	0.7333± 0.02	1.2444± 0.15	0.3156± 0.02
KMeans-comb	0.5773± 0.06	0.7089± 0.05	1.8958± 1.38	0.2675± 0.11
BIRCH-const	0.6134± 0.02	0.7411± 0.02	1.2729± 0.19	0.3032± 0.03
BIRCH-comb	0.5273± 0.04	0.6578± 0.03	2.0395± 0.97	0.1462± 0.12
DSE-BIRCH	0.6375± 0.01	0.7656± 0.01	0.9547± 0.11	0.3491± 0.02

TABLE 16. Clustering performance evaluation of different algorithms on the herring dataset of 10 replicates.

Method/Metric	RI	Purity	DB Index	SC
KMeans-const	0.5090± 0.01	0.5985± 0.01	0.9532± 0.02	0.3616± 0.01
KMeans-comb	0.5273± 0.01	0.6312± 0.01	0.9266± 0.00	0.3758± 0.00
BIRCH-const	0.5084± 0.01	0.5985± 0.01	0.9553± 0.02	0.3615± 0.01
BIRCH-comb	0.5136± 0.01	0.6047± 0.01	0.9421± 0.03	0.3577± 0.01
DSE-BIRCH	0.5380± 0.01	0.6500± 0.01	0.9076± 0.02	0.3901± 0.01

TABLE 17. Clustering performance evaluation of different algorithms on the SemgHandGenderCh2 dataset of 10 replicates.

Method/Metric	RI	Purity	DB Index	SC
KMeans-const	0.5191± 0.00	0.6500± 0.00	0.8332± 0.01	0.6221± 0.01
KMeans-comb	0.5295± 0.01	0.6500± 0.00	0.7996± 0.01	0.6374± 0.00
BIRCH-const	0.5277± 0.01	0.6500± 0.00	0.8236± 0.01	0.6320± 0.00
BIRCH-comb	0.5255± 0.01	0.6500± 0.00	0.8247± 0.05	0.6181± 0.05
DSE-BIRCH	0.5473± 0.00	0.6550± 0.00	0.7709± 0.02	0.6460± 0.00

TABLE 18. Clustering performance evaluation of different algorithms on the strawberry dataset of 10 replicates.

Method/Metric	RI	Purity	DB Index	SC
KMeans-const	0.4995± 0.00	0.6432± 0.00	0.8431± 0.02	0.4112± 0.01
KMeans-comb	0.5037± 0.00	0.6432± 0.00	0.9600± 0.18	0.3764± 0.03
BIRCH-const	0.5125± 0.01	0.6432± 0.00	0.6988± 0.06	0.4591± 0.02
BIRCH-comb	0.5100± 0.01	0.6432± 0.00	2.0108± 0.72	0.1837± 0.25
DSE-BIRCH	0.5294± 0.01	0.6625± 0.01	0.5335± 0.06	0.5325± 0.03

cannot be rigorously proven [53]. The original BIRCH paper [54] and numerous subsequent improvement papers, such as [47], did not provide a strict convergence proof. Our algorithm, DSE-BIRCH, as an enhanced version of BIRCH, improves the clustering feature distance measure to make it suitable for functional data. Similarly, it is a heuristic algorithm. Although we cannot provide a rigorous mathematical proof of the convergence of the DSE-BIRCH algorithm, we can demonstrate that it can complete clustering within a finite number of iterations and converge to a local optimal solution:

Let $n$be the initial number of sample data points.

1. Initially, each data point is treated as a cluster, resulting in $m = n$ clusters.

2. DSE-BIRCH incrementally constructs the CF tree. At each step, under the constraints of both static and dynamic distance measures, a new data point is assigned to the most suitable sub-cluster. Each leaf node can contain a maximum of L clusters, where L is a predefined threshold.

3. The maximum number of non-leaf nodes is n/L, based on rounding down.

4. The sum of clusters of non-leaf nodes per layer is upper bounded by B, where B is a preset threshold.

5. From root to leaf, the height of the CF Tree is upper bounded by $h = \lceil logB(n/L) \rceil$, which is a constant.

6. In each merge iteration, the number of clusters decreases by 1 or remains the same.

7. After at most $m-1$iterations, the number of clusters at root reaches 1.

8. Therefore, the maximum number of iterations for the entire CF tree construction and clustering merge is upper

bounded by $O(m)$. Since $m = n$ is the sample size, the upper bound on the number of iterations is also a constant, denoted as N.

9. Each merge greedily improves the current clustering density of samples.

10. After N greedy merge iterations, the sample distribution tends to stabilize.

11. When iterations can no longer improve clustering, i.e., reduce the entropy of sample distribution, DSE-BIRCH is considered converged. The sample distribution reaches a locally stable state.

Through the analysis of controlling the tree height and node size during the construction process, it can be proven that the number of iterations for DSE-BIRCH clustering is upper-bounded by the number of samples n, ensuring clustering is completed within a finite number of times and converges to a locally optimal solution. The above is only an intuitive description of the convergence of algorithms based on heuristic aggregation, and more rigorous mathematical proofs are still needed through further exploration.

REFERENCES

[1] J. O. Ramsay, "When the data are functions," *Psychometrika*, vol. 47, no. 4, pp. 379–396, Dec. 1982.

[2] J. O. Ramsay and C. J. Dalzell, "Some tools for functional data analysis," *J. Roy. Stat. Soc. Ser. B, Stat. Methodol.*, vol. 53, no. 3, pp. 539–561, Dec. 2018.

[3] J. O. Ramsay and B. W. Silverman, "Life course data in criminology," in *Applied Functional Data Analysis: Methods and Case Studies*. New York, NY, USA: Springer, 2002, pp. 17–40.

[4] J. Suhaila and Z. Yusop, "Spatial and temporal variabilities of rainfall data using functional data analysis," *Theor. Appl. Climatol.*, vol. 129, nos. 1–2, pp. 229–242, Jul. 2017.

[5] P. Kokoszka, H. Miao, and X. Zhang, "Functional dynamic factor model for intraday price curves," *J. Financial Econ.*, vol. 13, no. 2, pp. 456–477, Feb. 2014.

[6] T. Chen, J. DeJuan, and R. Tian, "Distributions of GDP across versions of the Penn World Tables: A functional data analysis approach," *Econ. Lett.*, vol. 170, pp. 179–184, Sep. 2018.

[7] M. Ashkartizabi and M. Aminghafari, "Functional data clustering using K-means and random projection with applications to climatological data," *Stochastic Environ. Res. Risk Assessment*, vol. 32, no. 1, pp. 83–104, Jan. 2018.

[8] S. Ullah and C. F. Finch, "Applications of functional data analysis: A systematic review," *BMC Med. Res. Methodol.*, vol. 13, p. 43, Mar. 2013.

[9] J.-L. Wang, J.-M. Chiou, and H.-G. Müller, "Functional data analysis," *Annual Rev. Stat. Appl.*, vol. 3, pp. 257–295, Jun. 2016.

[10] J. Ren and S.-L. Shi, "Multivariable panel data ordinal clustering and its application in competitive strategy identification of appliance-wiring listed companies," in *Proc. Int. Conf. Manag. Sci. Eng.*, 2009, pp. 253–258.

[11] M. Boullé, "Functional data clustering via piecewise constant nonparametric density estimation," *Pattern Recognit.*, vol. 45, no. 12, pp. 4389–4401, Dec. 2012.

[12] C. Bouveyron and C. Brunet-Saumard, "Model-based clustering of high-dimensional data: A review," *Comput. Statist. Data Anal.*, vol. 71, pp. 52–78, Mar. 2014.

[13] N. Serban and L. Wasserman, "CATS: Clustering after transformation and smoothing," *J. Amer. Stat. Assoc.*, vol. 100, no. 471, pp. 990–999, Sep. 2005.

[14] S. Dabo-Niang, F. Ferraty, J. M. Chiou, and P. L. Li, "Functional clustering of longitudinal data," in *Functional and Operatorial Statistics*. Berlin, Germany: Physica-Verlag HD, 2008, pp. 103–107.

[15] M. Yamamoto, "Clustering of functional data in a low-dimensional subspace," *Adv. Data Anal. Classification*, vol. 6, no. 3, pp. 219–247, Oct. 2012.

[16] M. Yamamoto and H. Hwang, "Dimension-reduced clustering of functional data via subspace separation," *J. Classification*, vol. 34, no. 2, pp. 294–326, Jul. 2017.

[17] A. Delaigle, P. Hall, and T. Pham, "Clustering functional data into groups by using projections," *J. Roy. Stat. Soc. Ser. B, Stat. Methodol.*, vol. 81, no. 2, pp. 271–304, Feb. 2019.

[18] J. Zhang, "Functional clustering based on weighted partitioning around medoid algorithm with estimation of number of clusters," in *Proc. IEEE 6th Int. Conf. Cloud Comput. Big Data Analytics (ICCCBDA)*, Apr. 2021, pp. 200–204.

[19] J.-M. Chiou and P.-L. Li, "Functional clustering and identifying substructures of longitudinal data," *J. Roy. Stat. Soc. Ser. B, Stat. Methodol.*, vol. 69, no. 4, pp. 679–699, Aug. 2007.

[20] F. Miller, J. Neill, and H. Wang, "Nonparametric clustering of functional data," *Statist. Interface*, vol. 1, no. 1, pp. 47–62, 2008.

[21] F. Ieva, A. M. Paganoni, D. Pigoli, and V. Vitelli, "Multivariate functional clustering for the morphological analysis of electrocardiograph curves," *J. Roy. Stat. Soc. Ser. C, Appl. Statist.*, vol. 62, no. 3, pp. 401–418, Oct. 2012.

[22] M. Boullé, R. Guigourès, and F. Rossi, "Nonparametric hierarchical clustering of functional data," in *Advances in Knowledge Discovery and Management*, vol. 4. Cham, Switzerland: Springer, 2014, pp. 15–35.

[23] M. Ciollaro, C. R. Genovese, and D. Wang, "Nonparametric clustering of functional data using pseudo-densities," *Electron. J. Statist.*, vol. 10, no. 2, pp. 2922–2972, Jan. 2016.

[24] F. Fortuna, F. Maturo, and T. Di Battista, "Clustering functional data streams: Unsupervised classification of soccer top players based on Google trends," *Qual. Rel. Eng. Int.*, vol. 34, no. 7, pp. 1448–1460, Nov. 2018.

[25] N. Margaritella, V. Inácio, and R. King, "Parameter clustering in Bayesian functional principal component analysis of neuroscientific data," *Statist. Med.*, vol. 40, no. 1, pp. 167–184, Jan. 2021.

[26] G. Hu, J. Geng, Y. Xue, and H. Sang, "Bayesian spatial homogeneity pursuit of functional data: An application to the US income distribution," *Bayesian Anal.*, vol. 18, no. 2, pp. 579–605, Jun. 2023.

[27] A. Delaigle and P. Hall, "Defining probability density for a distribution of random functions," *Ann. Statist.*, vol. 38, no. 2, pp. 1171–1193, Apr. 2010.

[28] A. Samé, F. Chamroukhi, G. Govaert, and P. Aknin, "Model-based clustering and segmentation of time series with changes in regime," *Adv. Data Anal. Classification*, vol. 5, no. 4, pp. 301–321, Dec. 2011.

[29] J. Jacques and C. Preda, "Funclust: A curves clustering method using functional random variables density approximation," *Neurocomputing*, vol. 112, pp. 164–171, Jul. 2013.

[30] M. Giacofci, S. Lambert-Lacroix, G. Marot, and F. Picard, "Wavelet-based clustering for mixed-effects functional models in high dimension," *Biometrics*, vol. 69, no. 1, pp. 31–40, Mar. 2013.

[31] J. Jacques and C. Preda, "Model-based clustering for multivariate functional data," *Comput. Statist. Data Anal.*, vol. 71, pp. 92–106, Mar. 2014.

[32] E. Devijver, "Model-based regression clustering for high-dimensional data: Application to functional data," *Adv. Data Anal. Classification*, vol. 11, no. 2, pp. 243–279, Jun. 2017.

[33] F. Chamroukhi and H. D. Nguyen, "Model-based clustering and classification of functional data," *Wiley Interdiscipl. Rev., Data Mining Knowl. Discovery*, vol. 9, no. 4, p. e1298, Jul. 2019.

[34] A. Sharp and R. Browne, "Functional data clustering by projection into latent generalized hyperbolic subspaces," *Adv. Data Anal. Classification*, vol. 15, no. 3, pp. 735–757, Sep. 2021.

[35] R. Wu, B. Wang, and A. Xu, "Functional data clustering using principal curve methods," *Commun. Statist.-Theory Methods*, vol. 51, no. 20, pp. 7264–7283, Oct. 2022.

[36] H. Wu and Y.-F. Li, "Clustering spatially correlated functional data with multiple scalar covariates," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jan. 12, 2022, doi: 10.1109/TNNLS.2021.3137795.

[37] N. Pronello, R. Ignaccolo, L. Ippoliti, and S. Fontanella, "Penalized model-based clustering of complex functional data," *Statist. Comput.*, vol. 33, no. 6, p. 122, Aug. 2023.

[38] K. Kumar Sharma, A. Seal, E. Herrera-Viedma, and O. Krejcar, "An enhanced spectral clustering algorithm with S-distance," *Symmetry*, vol. 13, no. 4, p. 596, Apr. 2021.

[39] K. K. Sharma, A. Seal, A. Yazidi, A. Selamat, and O. Krejcar, "Clustering uncertain data objects using Jeffreys-divergence and maximum bipartite matching based similarity measure," *IEEE Access*, vol. 9, pp. 79505–79519, 2021.

[40] K. K. Sharma, A. Seal, A. Yazidi, and O. Krejcar, "A new adaptive mixture distance-based improved density peaks clustering for gearbox fault diagnosis," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–16, 2022.
[41] J. Albert-Smet, A. Torrente, and J. Romo, "Band depth based initialization of K-means for functional data clustering," *Adv. Data Anal. Classification*, vol. 17, no. 2, pp. 463–484, Jun. 2023.
[42] H. Hu, J. Liu, X. Zhang, and M. Fang, "An effective and adaptable K-means algorithm for big data cluster analysis," *Pattern Recognit.*, vol. 139, Jul. 2023, Art. no. 109404.
[43] Y. Wang, D. Wang, Y. Zhou, X. Zhang, and C. Quek, "VDPC: Variational density peak clustering algorithm," *Inf. Sci.*, vol. 621, pp. 627–651, Apr. 2023.
[44] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: A new data clustering algorithm and its applications," *Data Mining Knowl. Discovery*, vol. 1, no. 2, pp. 141–182, 1997.
[45] Z. Chu, W. Wang, B. Li, W. Jin, S. Liu, B. Zhang, and Z. Lin, "An operation health status monitoring algorithm of special transformers based on BIRCH and Gaussian cloud methods," *Energy Rep.*, vol. 7, pp. 253–260, Apr. 2021.
[46] I. D. M. Ventorim, D. Luchi, A. L. Rodrigues, and F. M. Varejão, "BIRCHSCAN: A sampling method for applying DBSCAN to large datasets," *Exp. Syst. Appl.*, vol. 184, Dec. 2021, Art. no. 115518.
[47] A. Lang and E. Schubert, "BETULA: Fast clustering of large data with improved BIRCH CF-trees," *Inf. Syst.*, vol. 108, Sep. 2022, Art. no. 101918.
[48] D. Bosq, *Linear processes in function spaces: Theory and applications* (Lecture Notes in Statistics), vol. 149. New York, NY, USA: Springer, 2000.
[49] H. A. Dau, A. Bagnall, K. Kamgar, C. M. Yeh, Y. Zhu, S. Gharghabi, C. A. Ratanamahatana, and E. Keogh, "The UCR time series archive," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 6, pp. 1293–1305, Nov. 2019.
[50] Y. Meng, J. Liang, F. Cao, and Y. He, "A new distance with derivative information for functional K-means clustering algorithm," *Inf. Sci.*, vol. 463, pp. 166–185, Oct. 2018.
[51] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez, "Solving the multiple instance problem with axis-parallel rectangles," *Artif. Intell.*, vol. 89, nos. 1–2, pp. 31–71, Jan. 1997.
[52] C. O. Sakar, S. O. Polat, M. Katircioglu, and Y. Kastro, "Real-time prediction of online shoppers' purchasing intention using multilayer perceptron and LSTM recurrent neural networks," *Neural Comput. Appl.*, vol. 31, no. 10, pp. 6893–6908, Oct. 2019.
[53] K. G. Murty and S. N. Kabadi, "Some NP-complete problems in quadratic and nonlinear programming," *Math. Program.*, vol. 39, no. 2, pp. 117–129, Jun. 1987.
[54] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: An efficient data clustering method for very large databases," *ACM SIGMOD Rec.*, vol. 25, no. 2, pp. 103–114, Jun. 1996.

WANG LI was born in Jingzhou, Hubei, China. She is currently pursuing the master's degree in applied statistics with the School of Science, Hubei University of Technology. Her current research interests include functional data analysis, data mining, and machine learning.

HANFANG LI received the Ph.D. degree in statistics from Huazhong Normal University, with a focus on econometric modeling, MCMC algorithms, Bayesian analysis, and related fields. She has led and participated in five projects funded by the National Natural Science Foundation of China, the National Social Science Foundation, the Humanities and Social Sciences Fund of the Ministry of Education, and the Wuhan Municipal Federation of Social Science Associations.

YOUXI LUO received the Ph.D. degree in statistics from the Renmin University of China. He was a Postdoctoral Researcher with the Institute of Computing Technology, Chinese Academy of Sciences. He has been a Visiting Scholar with Emory University, USA, and Nanyang Technological University, Singapore. His current research interests include econometric modeling, high-dimensional data mining, MCMC algorithms, Bayesian analysis, and related fields