Hindawi
Journal of Advanced Transportation
Volume 2022, Article ID 5978704, 23 pages
https://doi.org/10.1155/2022/5978704

Research Article

Trajectory Clustering in an Intersection by GDTW

Lei Gao,[1,2] **Lu Wei**,[1] **Jian Yang**,[1,2] **and Jinhong Li**[1,2]

[1]*Beijing Key Lab of Urban Road Traffic Intelligent Technology, North China University of Technology, Beijing 100144, China*
[2]*School of Information Science and Technology, North China University of Technology, Beijing 100144, China*

Correspondence should be addressed to Lei Gao; gaolei@ncut.edu.cn

Received 23 May 2022; Revised 23 June 2022; Accepted 30 June 2022; Published 8 August 2022

Academic Editor: Eneko Osaba

GPS trajectory data in intersections are series data with different lengths. Dynamic time wrapping (DTW) is good to measure the similarity between series with different lengths, however, traditional DTW could not deal with the inclusive relationship well between series. We propose a unified generalized DTW algorithm (GDTW) by extending the boundary constraint and continuity constraint of DTW and using the weighted local distance to normalize the cumulative distance. Based on the density peak clustering algorithm DPCA using asymmetric GDTW to measure the similarity of two trajectories, we propose an improved DPCA algorithm (ADPC) to adopt this asymmetric similarity measurement. In experiments using the proposed method, the number of clusters is reduced.

1. Introduction

With the development of intelligent and connected vehicle technology, tremendous vehicle trajectory data is increasingly available in many applications, such as urban traffic planning and management, traffic analysis, etc. Mining and analyzing vehicle trajectory data are of great significance for intelligent transportation, point of interest recommendation, and location-based services [1]. Trajectory clustering can explore different spatial-temporal features of vehicle trajectory data and has been a research hotspot in recent years [2, 3].

Many studies have been attempted for vehicle trajectory clustering. How to define and measure trajectory distance appropriately is quite a challenging work. There are three main distance metrics used in the existing literature, i.e., Euclidean distance, shape-based distance, and warping-based distance. Traditional clustering methods using Euclidean distance often lead to inaccurate results because of different lengths of trajectories. To overcome this limitation, the warping distance is used in several methods. Besides, shape-based distances, such as Hausdorff and Frechet distances could also apply to trajectories. On the other hand, deep learning-based methods have shown an impressive ability to process multidimensional sequence data [4–6].

Our work is similar to that presented in paper [7], however, we focus on improving the DTW algorithm and clustering algorithm. We propose a unified generalized DTW algorithm called GDTW. Moreover, we propose a clustering algorithm called ADPC by improving DPCA to adopt the asymmetric similarity measurement.

In our application scenario, namely a road intersection, some short trajectory is included by a long trajectory. Thus, the short trajectory is a subsequence of a long trajectory. We assume that the short trajectory and long trajectory have the same pattern, which means belonging to the same cluster, and the long trajectory can represent a short trajectory. In similarity measurement, this inclusive relationship should be considered. Therefore, we generalize DTW to match this pattern and propose a clustering algorithm to adjust the generalized DTW.

Dynamic time warping is a classic dynamic programming algorithm to measure the similarity between series with different lengths. It is first proposed in the last century. At that time, the research hotspot is reducing computation complexity and constrained search area [8]. At the beginning of the 21st century, research on the lower bound function and other techniques to accelerate DTW in the application is popular [9–11]. From beginning to end,

improving the calculation efficiency in the application is always the key spot of DTW-related research. However, on the other hand, there is research on a variety of DTW to generalize its formation [12, 13], and our work about DTW is in this area.

The clustering is also a widely studied area. There are many classic algorithms, from k-means FCM to GMM, from graph-based hierarchical algorithms to spectrum clustering, and various density-based algorithms. The density-based clustering methods can automatically acquire the number of clusters and form clusters of arbitrary geometric shapes. There are several classic density-based algorithms, such as mean-shift [14], DBSCAN [15], and DPCA [16]. Based on DPCA, we introduce an asymmetric distance clustering algorithm to find the superinclusive trajectory of each cluster.

In this paper, we proposed a method for an unsupervised clustering of vehicle trajectories at intersection areas. The contributions of this work are as follows:

(1) We proposed a generalized DTW algorithm to calculate the similarity between two trajectories that can handle trajectories with inclusion relationships. First of all, we extend the definition of DTW by the concept of the matched scheme. Then, we generalized various conditions of DTW among boundary and continuity conditions. We summarized the scale of the problem for each generalized condition for DTW.

(2) We proved the validity of generalized DTW and proposed a unified algorithm to calculate all kinds of generalized DTW, and we proved that this unified algorithm's complexity is the same as traditional DTW's dynamical programming algorithm. Then, we use asymmetric GDTW as the similarity measurement to match the inclusive relationship between trajectories.

(3) We discussed different normalization methods for cumulative DTW and proposed a weighted normalization method. To adopt the proposed generalized asymmetric DTW as distance measurement, we also proposed a clustering algorithm ADPC, which is based on DPCA. This proposed clustering algorithm can extract the local max trajectories as the cluster representative.

2. Methodology

In this section, we first give the traditional dynamic time warping's definition in a matched scheme formation and describe the traditional DTW's conditions in this matched scheme formation. Second, we define the generalized DTW conditions. Third, we evaluate the scale of the problem for generalized DTW. Fourth, we give the definition and prove the recursive formation of various generalized DTW, and we propose a unified dynamical programming algorithm to calculate generalized DTW. Fifth, we give a unified normalization for generalized DTW. Sixth, based on DPCA, we propose an asymmetric distance density peak clustering algorithm to adopt the generalized DTW.

2.1. Matched Scheme for Traditional DTW. In this part, we recall the traditional DTW and proposed a concept named matched scheme. We use matched scheme to define the traditional DTW and its three conditions.

The dynamic time warping (DTW) is used to measure the similarity of two series with different lengths. Supposing there are two series $\mathbf{a} = [a_1, a_2, ..., a_M]$ and $\mathbf{b} = [b_1, b_2, \ldots, b_N]$, $a_i, b_j \in \mathbb{R}^2$ are the elements of series **a** and series **b**, respectively. The DTW distance between series **a** and series **b** is defined as (1).

$$\mathrm{dtw}(\mathbf{a}, \mathbf{b}) = \min_{\mathbf{P}} \left(\sum_{(i,j)\subset \mathbf{P}} \delta_{i,j} \right), \tag{1}$$

where $\delta_{i,j}$ is the direct local distance between the i^{th} element of series **a** and the j^{th} element of series **b**. The local distance could be measured by any distance measurement, e.g., the Euclidean distance $\delta_{i,j} = {}^{2}_{\|a_i - b_j\|}$. In this formation, it is obvious that $\delta_{i,j} \neq \delta_{j,i}$.

In (1), $\mathbf{P} = [(i_1, j_1), (i_2, j_2), \ldots, (i_K, j_K)]$ is the matched index pair sequence, for each pair $(i, j) \in \mathbf{P}$ means that the index values of the element are about series **a** and **b**, respectively, and there is no same index pair, namely $(i_u, j_u) \neq (i_v, j_v)$, where $u \neq v$. The matched index pair sequence is denoting the matched pair of elements between the two series. Therefore, the matched index pair sequence **P** could be assumed as a matched scheme. The DTW is to find a matched scheme **P** to minimize (1) among all valid matched schemes.

For the traditional DTW, the matched scheme must satisfy the three conditions mentioned below.

2.1.1. Monotonic Condition. Let $k \in [2, 3, \ldots, K]$, for each matched index pair (i_k, j_k) of a matched scheme **P**, must satisfy $i_{k-1} \leq i_k$ and $j_{k-1} \leq j_k$. The monotonic condition means both index values in the matched index pair are monotonically increasing and there is no inverse matching.

2.1.2. Continuity Condition. Let $k \in [2, 3, \ldots, K]$, for each matched index pair (i_k, j_k) of a matched scheme **P**, must satisfy $i_k - i_{k-1} \leq 1$ and $j_k - j_{k-1} \leq 1$. Continuity condition indicates the limitation on the growing amount of both index values for each matched index pair.

2.1.3. Boundary Condition. Let $\mathbf{P} = [(i_1, j_1), (i_2, j_2), \ldots, (i_K, j_K)]$, which must satisfy $(i_1, j_1) = (1, 1)$ and $(i_K, j_K) = (M, N)$. Condition $(i_1, j_1) = (1, 1)$ means both of the first elements of series **a** and series **b** must be matched, and we call it the front boundary condition. Condition $(i_K, j_K) = (M, N)$ means both of the last elements of series **a** and series **b** must be matched, and we call it the back boundary condition. The boundary condition of traditional DTW meets both front and back boundary conditions.

Therefore, (1) can be explained as the traditional DTW that can find the lowest cumulative local distance among all matched schemes that meet monotonic, continuity, and boundary conditions. We call matched schemes that meet all these three conditions as DTW-matched schemes. The number of DTW-matched schemes saw an exponential increase with the length of the series. It is not realistic to iterate all matched schemes to find the solution. In practice, the DTW is calculated by dynamic programming.

Definition 1. The DTW distance can be calculated recursively by equation (2)

$$\Delta_{i,j} = \min\left(\Delta_{i-1,j}, \Delta_{i,j-1}, \Delta_{i-1,j-1}\right) + \delta_{i,j}, \quad (2)$$

where $\delta_{i,j}$ is the same as (1), and it denotes the direct local distance between the i^{th} element of series a and the j^{th} element of series b. $\Delta_{i,j}$ denotes the DTW distance between $\mathbf{a}$'s subseries $\mathbf{a}_{1\ldots i} = [a_1, a_2, \ldots, a_i]$ and $\mathbf{b}$'s subseries $\mathbf{b}_{1\ldots j} = [b_1, b_2, \ldots, b_j]$, and it has $\text{dtw}(\mathbf{a}, \mathbf{b}) = \Delta_{M,N}$. When $i < 1 \vee j < 1$, it means the index pair does not exist. Therefore, let the nonexistent local distance $\Delta_{i,j} = \infty$.

For example, there are two-dimensional trajectory series a and b.

As shown in Figure 1, trajectory series a has 6 sampling points, and trajectory series b has 14 sampling points. The matched pairs in the DTW context are linked by a dotted line. In this paper, we call the figure that is similar to Figure 1 a matched pairs diagram.

In Figure 2, the y axis is the index of the element of series a, and the x axis is the index of the element of series b. In this paper, we call the figure that is similar to Figure 2 a cumulative distance diagram. Each grid (x, y) is represented the partial cumulative DTW distance $\Delta_{y,x}$ by different gray depth levels. The path from the left bottom to the upper right is linked by each matched indexes pair is the search path. The start point and end point of the search path represent the boundary condition of DTW. The step of the search path represents the continuity condition of DTW, which is not greater than one on both of y and x axes. The monotonic condition of DTW is intuitively shown by the direction of the search path. In fact, DTW is to find a search path from the left bottom to the upper right in the cumulative distance diagram.

It must be aware that although DTW could measure the similarity between the two series, DTW is not exactly a distance. DTW is symmetric, namely $\text{dtw}(\mathbf{a}, \mathbf{b}) = \text{dtw}(\mathbf{b}, \mathbf{a})$. However, DTW is not an identity, because in most cases, it is hard to say that series a is series b when $\text{dtw}(\mathbf{a}, \mathbf{b}) = 0$. DTW does not meet triangle inequality, namely, there may exist three series, namely a, b, and c, having $\text{dtw}(\mathbf{a}, \mathbf{b}) + \text{dtw}(\mathbf{b}, \mathbf{c}) < \text{dtw}(\mathbf{a}, \mathbf{c})$. Therefore, DTW is not suitable to calculate a transitive similarity between many series by accumulating them.

2.2. Generalized DTW Conditions. The traditional DTW needs to meet three conditions, which are monotonic, continuity, and boundary. In this paper, we keep the monotonic condition and generalize DTW by removing the boundary condition and extending the continuity condition.

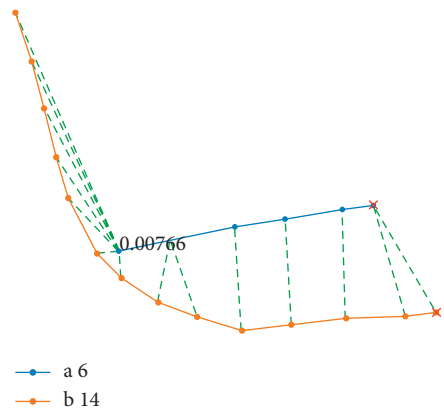


Figure 1: Spatial distribution and matched pairs diagram of DTW between two series a and b.

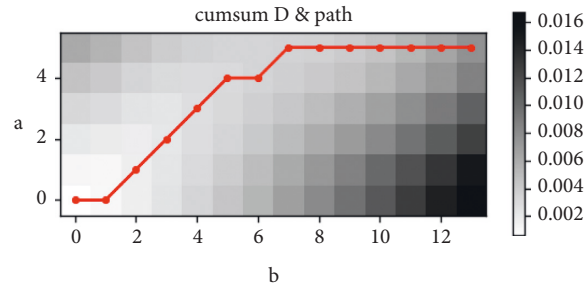


Figure 2: The cumulative local distances and search path diagram of DTW between two series a and b.

First of all, we introduce some new generalized conditions. Let $\mathbf{P} = [(i_1, j_1), (i_2, j_2), \ldots (i_K, j_K)]$ be a matched scheme.

We divide the boundary condition into four parts.

(1) The left-front-boundary condition, which must meet $i_1 = 1$. It means the first element of series a (left series) must be in the first match pair, which is the start point of the search path of the cumulative distance diagram.

(2) The right-front-boundary condition, which must meet $j_1 = 1$. It means the first element of series b (right series) must be in the first match pair, which is the start point of the search path of the cumulative distance diagram.

(3) The left-back-boundary condition, which must meet $i_K = M$. It means the last element of series a (left series) must be in the last match pair, which is the end point of the search path of the cumulative distance diagram.

(4) The right-back-boundary condition, which must meet $j_K = N$. It means the last element of series b (right series) must be in the first match pair, which is the end point of the search path of the cumulative distance diagram.

When meeting both left and right front boundary conditions, we call it to meet the left-boundary condition. When meeting both left and right back boundary conditions, we call it to meet the right-boundary condition. When meeting both left and right boundary conditions, it must meet the original boundary condition. When meeting only one of the left or right boundary conditions, we call it meeting the half-boundary condition.

The relationship between these boundary conditions is shown in Table 1. The boundary condition of traditional DTW needs to meet all four generalized boundary conditions. The four new generalized boundary conditions are a partition of the original boundary condition, respectively. In application, the partition of boundary conditions can make partial matching between series.

We extend the continuity condition and divide it into two parts. Let $k \in [2, 3, ..., K]$ and (i_k, j_k) be any matched index pair of a matched scheme **P**, except the first match pair. For any number $n \in \mathbb{N}$,

(1) The left-n-step continuity condition, which must meet $i_k - i_{k-1} \leq n$ and $j_k - j_{k-1} \leq 1$. Compared with the continuity condition of DTW, the left n-step continuity condition extends the growth amount of index value for each matched index pair of series a (left series) to n from the original 1. It would lead to some elements of series a to not be in the matched scheme, which means those elements would be neglected. In the application, we can use this feature to ignore some outliers.

(2) The right-n-step continuity condition, which must meet $i_k - i_{k-1} \leq 1$ and $j_k - j_{k-1} \leq n$. The formation of the right n-step continuity condition is similar to the left n-step continuity condition. It only just neglects some elements of series b.

We also give a full-match condition and can divide it into two parts. Let $|\{i\}|$ and $|\{j\}|$ be the cardinal of sets in matched scheme **P** for the two series, respectively.

(1) The left-full-match condition, which must meet $|\{i\}| = M$, means each element of series a (left series) must be in the matched scheme.

(2) The right-full-match condition, which must meet $|\{j\}| = N$, means each element of series b (left series) must be in the matched scheme.

When meeting both left and right full-match conditions, we call it meeting the dual-full-match condition. When meeting one of the left or right full-match conditions, we call it meeting the half-full-match condition.

The relationship between various generalized conditions is shown in Table 2, which let n-step greater than one. In application, the half-full-match condition should be met to let the DTW make sense in most cases. When only meeting the one left- or right-half condition, the generalized DTW is not symmetric.

Table 1: The relationship between various boundary conditions.

Condition name			Condition
Boundary condition	Front	Left	$i_1 = 1$
		Right	$j_1 = 1$
	Back	Left	$i_K = M$
		Right	$j_K = N$

Table 2: The relationship between various generalized conditions.

Half condition name	Condition	Left-full	Right-full
Left-boundary	$i_1 = 1 \wedge i_K = M$	Meet	
Right-boundary	$j_1 = 1 \wedge j_K = N$		Meet
Left-n-step-continuity	$i_k - i_{k-1} \leq n \wedge j_k - j_{k-1} \leq 1$		Meet
Right-n-step-continuity	$i_k - i_{k-1} \leq 1 \wedge j_k - j_{k-1} \leq n$	Meet	

2.3. The Scale of Problem for GDTW. In this paper, we generalized dynamic time warping to extend its application range by eliminating some conditions. However, along with the elimination of conditions, the number of valid matched schemes will be increased, leading to an increase in the scale of the problem. The number of traditional DTW-matched schemes is approximate $e^{(N-1)(e-1)}$ when $M = N$. In an extreme case, when all conditions are removed, the structure can be assumed as a binary graph between series a's M elements and series b's N elements, and there are $2^{M \times N}$ matched schemes.

Let $N_{dtw}(M, N)$ be valid matched schemes of traditional DTW for series a with M elements and series b with N elements. By eliminating some conditions, the generalized DTW has a different number of the matched schemes, which are expressed as $N_o\, bA(M, N) N_{oeA}(M, N)\, N_{obB}(M, N) N_{oe}$ $B(M, N) N_{ob}(M, N) N_{oe}(M, N) N$ $obeA(M, N) N$ $obeB(M, N) N_{obef}(M, N) N_{obAeAB}(M, N) N_{obBeAB}(M, N) N$ $obe(M, N)$. We would explain the meaning and give the calculation equation for these matched schemes, respectively.

$N_{dtw}(M, N)$ is symmetric

$$N_{dtw}(M, N) = N_{dtw}(N, M). \tag{3}$$

$N_{obA}(M, N)$ is the opened begin boundary condition for series A, which means the matched scheme could ignore some elements in front of series a. $N_{oeA}(M, N)$ is the opened end boundary condition for the series, which means the matched scheme could ignore some elements in the tail of series a. $N_{obB}(M, N)$ and $N_{oeB}(M, N)$ are similar to $N_{obA}(M, N)$ and $N_{oeA}(M, N)$, except those for series b.

$$\begin{aligned} N_{\mathrm{obA}}(M,N) &= N_{\mathrm{oeA}}(M,N) \\ &= N_{\mathrm{dtw}}(M,N) + \sum_{i=1}^{M-1} N_{\mathrm{dtw}}(i,N) \\ &= \sum_{i=1}^{M} N_{\mathrm{dtw}}(i,N), \\ N_{\mathrm{obB}}(M,N) &= N_{\mathrm{oeB}}(M,N) \\ &= N_{\mathrm{dtw}}(M,N) + \sum_{j=1}^{N-1} N_{\mathrm{dtw}}(M,j) \\ &= \sum_{j=1}^{N} N_{\mathrm{dtw}}(M,j), \end{aligned} \tag{4}$$

$N_{obA}(M,N)N_{oeA}(M,N)N_{obB}(M,N)N_{oeB}(M,N)$ is asymmetric. From fd3(3) and (4), we have the following:

$$\begin{aligned} N_{\mathrm{obA}}(M,N) &= N_{\mathrm{oeA}}(M,N) \\ &= N_{\mathrm{obB}}(N,M) \\ &= N_{\mathrm{oeB}}(N,M). \end{aligned} \tag{5}$$

$N_{ob}(M,N)$ and $N_{oe}(M,N)$ are symmetric upon removing front or back boundary conditions, respectively. The symmetric feature $N_{ob}(M,N) = N_{oe}(M,N)$ can be deduced by (5).

$$\begin{aligned} N_{oe}(M,N) &= N_{oeA}(M,N) + N_{oeB}(M,N) - N_{dtw}(M,N), \\ N_{ob}(M,N) &= N_{obA}(M,N)N_{obB}(M,N) - N_{dtw}(M,N). \end{aligned} \tag{6}$$

$N_{obeA}(M,N)$ means the left-boundary condition is removed but meets the right-full condition, namely every element of series b must be in the matched scheme. In application, when series b is a subsequence of series a, this form of generalized DTW could be zero. The $N_{obeB}(M,N)$ is the same as $N_{obeA}(M,N)$ but meets the left-full condition instead of the right-full condition.

$$\begin{aligned} N_{obeA}(M,N) &= \sum_{i=1}^{M} \sum_{k=1}^{i} N_{dtw}(k,N) \\ = \sum_{i=1}^{M} N_{oeA}(k,N)N_{obeB}(M,N) &= \sum_{j=1}^{N} \sum_{k=1}^{j} N_{dtw}(M,k) \\ = \sum_{j=1}^{N} N_{oeB}(M,k). \end{aligned} \tag{7}$$

From (5) and (7), we have $N_{obeA}(M,N) = N_{obeB}(N,M)$, and it proved that they are asymmetric. $N_{obef}(M,N)$ is symmetric.

$$\begin{aligned} N_{obef}(M,N) &= N_{obeA}(M,N) + N_{obeB}(M,N) \\ &\quad - N_{dtw}(M,N). \end{aligned} \tag{8}$$

$N_{obAeAB}(M,N)$ and $N_{obBeAB}(M,N)$ are GDTW with the opened begin boundary condition based on removing the back-boundary condition.

$$\begin{aligned} N_{obAeAB}(M,N) &= \sum_{i=1}^{M} N_{oe}(i,N), \\ N_{obBeAB}(M,N) &= \sum_{j=1}^{N} N_{oe}(M,j). \end{aligned} \tag{9}$$

$N_{obe}(M,N)$ removed all boundary conditions, and we have the following:

$$\begin{aligned} N_{\mathrm{obe}}(M,N) &= N_{\mathrm{obAeAB}}(M,N) + N_{\mathrm{obBeAB}}(M,N) \\ &\quad - N_{\mathrm{oe}}(M,N). \end{aligned} \tag{10}$$

The number of matched schemes can be seen as the search path count in the cumulative distance diagram. We compared the numbers of different generalized DTW with various sizes of series a and b.

As shown in Figure 3, the growth proportion is stable for various size ratios of series. The scale of the problem is linear growth between generalized DTW.

2.4. Generalized DTW. In this part, we introduce the recursive formation for generalized DTW. Then, we give the recursive equation for each type of generalized DTW. Also, we propose a unified algorithm to calculate all these generalized DTW.

Let $\mathbf{a}_{u\ldots i} = [a_u, a_{u+1}, \ldots, a_i]$ denote the subsequence of series a, which is from the u^{th} element to the i^{th} element. Let $\mathbf{b}_{v\ldots j} = [b_v, b_{v+1}, \ldots, b_j]$ denote the subsequence of series b, which is from the v^{th} element to the j^{th} element. Let $D_{u,v,i,j}$ denote the traditional DTW between the subsequence series $\mathbf{a}_{u\ldots i}$ and $\mathbf{b}_{v\ldots j}$. The traditional DTW $\Delta_{M,N}$ between series a and b can be denoted as $D_{1,1,M,N}$. The recursive (2) can be rewritten.

Definition 2. The generalized formation of recursive calculation for DTW distance.

$$D_{u,v,i,j} = \min\left(D_{u,v,i-1,j}, D_{u,v,i,j-1}, D_{u,v,i-1,j-1}\right) + \delta_{i,j}. \tag{11}$$

In (11), $D_{u,v,i,j}$ must meet the monotonic condition, which is $1 \le u \le i \le M$ and $1 \le v \le j \le N$. When the monotonic condition is not met, it means the search path in the cumulative distance diagram does not exist, thus letting the nonexistent DTW $D_{u,v,i,j} = \infty$.

The Open-End DTW for series a and b are denoted, respectively, by $D_{M,N}^{oeA}$ and $D_{M,N}^{oeB}$, and it is obvious that,

$$\begin{aligned} D_{M,N}^{oeA} &= \min_{1\le i\le M} D_{1,1,i,N} \\ &= \min_{1\le i\le M} \Delta_{i,N}, \\ D_{M,N}^{oeB} &= \min_{1\le j\le N} D_{1,1,M,j} \\ &= \min_{1\le j\le N} \Delta_{M,j}. \end{aligned} \tag{12}$$

In (12), the Open-End DTW for series a is just to find the minimum value in the cumulative distance diagram's right

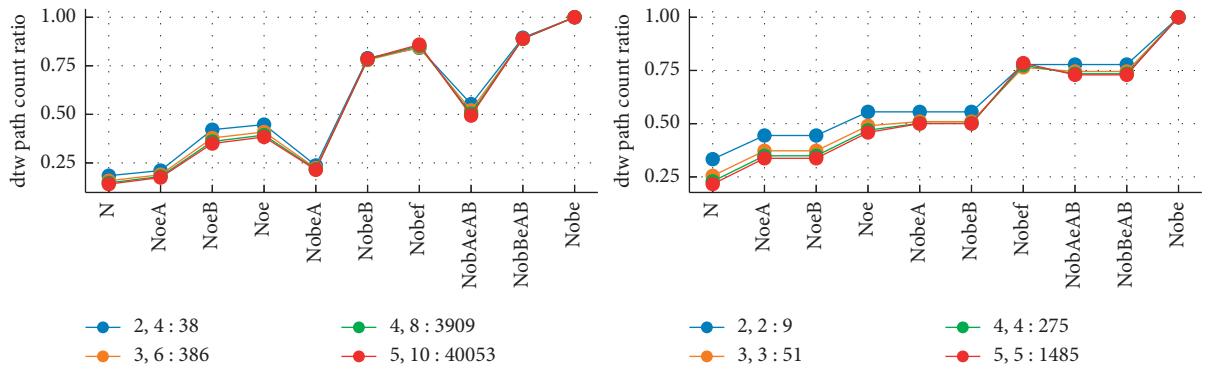


Figure 3: Number of matched schemes between different generalized DTW.

edge. On the other hand, the Open-End DTW for series b is just to find the minimum value in the cumulative distance diagram's top edge. The computation complexity is the same as traditional DTW. $D^{oeA}_{M,N} \neq D^{oeB}_{M,N}$ are asymmetric but have $D^{oeA}_{M,N} = D^{oeB}_{N,M}$.

The Open-Begin DTW for series a and b is denoted, respectively, by $D^{obA}_{M,N}$ and $D^{obB}_{M,N}$, and it is obvious that,

$$\begin{aligned} D^{obA}_{M,N} &= \min_{1\le u\le M} D_{u,1,M,N}, \\ D^{obB}_{M,N} &= \min_{1\le v\le N} D_{1,v,M,N}. \end{aligned} \tag{13}$$

Equation (13) only gives the relationship between $D^{obA}_{M,N}$ and subsequence DTW $D_{u,v,i,j}$. Furthermore, from (11) and (13), $D^{obA}_{M,N}$ can be expressed as follows:

$$\begin{aligned} D^{obA}_{M,N} &= \min_{1\le u\le M}\left(\min\left(D_{u,1,M-1,N}, D_{u,1,M,N-1}, D_{u,1,M-1,N-1}\right) + \delta_{M,N}\right) \\ &= \min_{1\le u\le M}\left(\min\left(D_{u,1,M-1,N}, D_{u,1,M,N-1}, D_{u,1,M-1,N-1}\right)\right) + \delta_{M,N} \\ &= \min\begin{pmatrix} \min_{1\le u\le M} D_{u,1,M-1,N} \\ \min_{1\le u\le M} D_{u,1,M,N-1} \\ \min_{1\le u\le M} D_{u,1,M-1,N-1} \end{pmatrix} + \delta_{M,N} \\ &= \min\begin{pmatrix} \min_{1\le u\le M-1} D_{u,1,M-1,N} \\ \min_{1\le u\le M} D_{u,1,M,N-1} \\ \min_{1\le u\le M-1} D_{u,1,M-1,N-1} \end{pmatrix} + \delta_{M,N} \\ &= \min\left(D^{obA}_{M-1,N}, D^{obA}_{M,N-1}, D^{obA}_{M-1,N-1}\right) + \delta_{M,N}. \end{aligned} \tag{14}$$

The recursive end margin for $D^{obA}_{M,N}$ is $D^{obA}_{1,j} = \Delta_{1,j}$, which is the same as traditional DTW using the cumulative distance. On the other hand, because of $D^{obA}_{i,1} = \min_{1\le u\le M} D_{u,1,M,1} = D_{u,1,M,1} = \delta_{i,1}$, the computation complexity is less than traditional DTW using local distance.

$D^{obB}_{M,N}$ has the same recursive formation as $D^{obA}_{M,N}$, and therefore, Open-Begin DTW's recursive formation is as follows:

$$\begin{aligned} D^{obA}_{M,N} &= \min\left(D^{obA}_{M-1,N}, D^{obA}_{M,N-1}, D^{obA}_{M-1,N-1}\right) + \delta_{M,N}, \\ D^{obB}_{M,N} &= \min\left(D^{obB}_{M-1,N}, D^{obB}_{M,N-1}, D^{obB}_{M-1,N-1}\right) + \delta_{M,N}. \end{aligned} \tag{15}$$

Equation (15) has the same recursive formation as traditional DTW, and the only difference is the recursive end margin $D^{obA}_{i,1} = \delta_{i,1}$ and $D^{obB}_{1,j} = \delta_{1,j}$. Also, the computation complexity is the same as traditional DTW, and in fact, it is a little less than the traditional DTW because the start margin is not a cumulative distance. $D^{obA}_{M,N} \neq D^{obB}_{M,N}$ are asymmetric but have $D^{obA}_{M,N} = D^{obB}_{N,M}$.

Also, there is another formation. Let M' denote series a's reverse sequence $\mathbf{a}_{M'} = [a_M, a_{M-1}, \ldots, a_1]$. Let N' denote series b's reverse sequence $\mathbf{b}_{N'} = [b_N, b_{N-1}, \ldots, b_1]$. We have $D^{obA}_{M,N} = D^{oeA}_{M',N'}$ and $D^{obB}_{M,N} = D^{oeB}_{M',N'}$.

The Open-Begin-End DTW for series a and b are denoted, respectively, by $D^{obeA}_{M,N}$ and $D^{obeB}_{M,N}$, and it is obvious that,

$$\begin{aligned} D^{obeA}_{M,N} &= \min_{1\le u\le i\le M} D_{u,1,i,N}, \\ D^{obeB}_{M,N} &= \min_{1\le v\le j\le N} D_{1,v,M,j}. \end{aligned} \tag{16}$$

In (16), $D^{obeA}_{M,N}$ can be expressed as follows:

$$D_{M,N}^{obeA} = \min_{1\le u\le i\le M} D_{u,1,i,N} = \min_{1\le u\le M} \min_{u\le i\le M} D_{u,1,i,N}. \tag{17}$$

Because of (11)'s definition, when $u > i$, the monotonic condition is not met and $D_{u,v,i,j} = \infty$, having $\min_{u\le i\le M} D_{u,1,i,N} = \min_{1\le i\le M} D_{u,1,i,N}$. Thus,

$$D_{M,N}^{obeA} = \min_{1\le u\le M} \min_{1\le i\le M} D_{u,1,i,N} = \min_{1\le i\le M} \min_{1\le u\le M} D_{u,1,i,N}. \tag{18}$$

Furthermore, when $u > i$, having $\min_{1\le u\le M} D_{u,1,i,N} = \min_{1\le u\le i} D_{u,1,i,N}$. Therefore,

$$D_{M,N}^{obeA} = \min_{1\le i\le M} \min_{1\le u\le i} D_{u,1,i,N} = \min_{1\le i\le M} D_{i,N}^{obA}. \tag{19}$$

$D_{M,N}^{obeB}$ has the same recursive formation as $D_{M,N}^{obeA}$. Therefore, the Open-Begin-End DTW can be expressed by $D_{M,N}^{obA}$ and $D_{M,N}^{obB}$, respectively.

$$D_{M,N}^{obeA} = \min_{1\le i\le M} D_{i,N}^{obA}, \quad D_{M,N}^{obeB} = \min_{1\le j\le N} D_{M,j}^{obB}. \tag{20}$$

Equation (20) has the same form as (12) of $D_{M,N}^{oeA}$ and $D_{M,N}^{oeB}$, except for replacing $\Delta_{i,N}\Delta_{M,j}$ with replacing $D_{i,N}^{obA} D_{M,j}^{obB}$. Therefore, the computation complexity is the same as traditional DTW. $D_{M,N}^{obeA} \ne D_{M,N}^{obeB}$ are asymmetric but have $D_{M,N}^{obeA} = D_{N,M}^{obeB}$.

The symmetric Open-End DTW for series a and b is denoted by $D_{M,N}^{oe}$ and is defined as follows:

$$D_{M,N}^{oe} = \min\left(D_{M,N}^{oeA}, D_{M,N}^{oeB}\right) = \min\left(\min_{1\le i\le M}\Delta_{i,N}, \min_{1\le j\le N}\Delta_{M,j}\right). \tag{21}$$

Equation (21) is showing the symmetric property with $D_{M,N}^{oe} = D_{N,M}^{oe}$. Compared to (12), the symmetric Open-End DTW is just to find the minimum value in the cumulative distance diagram's right and top edge. The computation complexity is the same as traditional DTW.

The symmetric Open-Begin DTW for series a and b is denoted by $D_{M,N}^{ob}$, and for $D_{M,N}^{oe}$, it defined as

$$D_{M,N}^{ob} = \min\left(D_{M,N}^{obA}, D_{M,N}^{obB}\right). \tag{22}$$

From (15) and (22), we have the following:

$$\begin{aligned} D_{M,N}^{ob} &= \min\begin{pmatrix} \min\left(D_{M-1,N}^{obA}, D_{M,N-1}^{obA}, D_{M-1,N-1}^{obA}\right) + \delta_{M,N} \\ \min\left(D_{M-1,N}^{obB}, D_{M,N-1}^{obB}, D_{M-1,N-1}^{obB}\right) + \delta_{M,N} \end{pmatrix} \\ &= \min\begin{pmatrix} D_{M-1,N}^{obA}, D_{M,N-1}^{obA}, D_{M-1,N-1}^{obA} \\ D_{M-1,N}^{obB}, D_{M,N-1}^{obB}, D_{M-1,N-1}^{obB} \end{pmatrix} + \delta_{M,N} \\ &= \min\begin{pmatrix} \min\left(D_{M-1,N}^{obA}, D_{M-1,N}^{obB}\right) \\ \min\left(D_{M,N-1}^{obA}, D_{M,N-1}^{obB}\right) \\ \min\left(D_{M-1,N-1}^{obA}, D_{M-1,N-1}^{obB}\right) \end{pmatrix} + \delta_{M,N}. \end{aligned} \tag{23}$$

Therefore, we have the recursive formation as follows:

$$D_{M,N}^{ob} = \min\left(D_{M-1,N}^{ob}, D_{M,N-1}^{ob}, D_{M-1,N-1}^{ob}\right) + \delta_{M,N}. \tag{24}$$

Equation (24) is symmetric and has $D_{M,N}^{ob} = D_{N,M}^{ob}$. Compared to (15), the symmetric Open-Begin DTW has the same formation. The difference is the recursive end margin $D_{i,1}^{ob} = \delta_{i,1}$ and $D_{1,j}^{ob} = \delta_{1,j}$, which means the start margin is not the cumulative distance. Therefore, the computation complexity is the same as traditional DTW, and the calculated quantity is a little less than asymmetric Open-Begin DTW's, which is a little less than traditional DTW.

The symmetric Open-Begin-End-half-Full DTW for series a and b is denoted by $D_{M,N}^{obef}$, and it is defined as follows:

$$D_{M,N}^{obef} = \min\left(D_{M,N}^{obeA}, D_{M,N}^{obeB}\right). \tag{25}$$

Because of $D_{i,j}^{obeA} \ne D_{i,j}^{obeB}$, there is no recursive formation same as traditional DTW. Hence, it must search both asymmetric Open-Begin-End DTW for series a and series b. Therefore, the computation complexity is two times the traditional DTW.

The asymmetric Open-Begin-for-series-A-End-for-series-A-B DTW is denoted by $D_{M,N}^{obAeAB}$. On the other hand, the asymmetric Open-Begin-for-series-B-End-for-series-A-B DTW is denoted by $D_{M,N}^{obBeAB}$. They are defined as follows:

$$D_{M,N}^{obAeAB} = \min\left(\min_{1\le u\le i\le M} D_{u,1,i,N}, \min_{1\le u\le M} \min_{1\le j\le N} D_{u,1,M,j}\right), \quad D_{M,N}^{obBeAB} = \min\left(\min_{1\le v\le j\le N} D_{1,v,M,j}, \min_{1\le v\le N} \min_{1\le i\le M} D_{1,v,i,N}\right). \tag{26}$$

From equation (17)–(19), we have the following:

$$D_{M,N}^{obAeAB} = \min\left(\min_{1\le i\le M} D_{i,N}^{obA}, \min_{1\le j\le N} D_{M,j}^{obA}\right), \quad D_{M,N}^{obBeAB} = \min\left(\min_{1\le j\le N} D_{M,j}^{obB}, \min_{1\le i\le M} D_{i,N}^{obB}\right). \tag{27}$$

Intuitively, $D_{M,N}^{obAeAB}$ and $D_{M,N}^{obBeAB}$ search the minimized value in the cumulative distance diagram's right and top margin, and the element was calculated by $D_{i,N}^{obA}$ and $D_{M,j}^{obB}$, respectively. Their symmetric formation of Open-Begin-End DTW is denoted by $D_{M,N}^{obABeAB}$ and defined as follows:

$$D_{M,N}^{obABeAB} = \min\left(D_{M,N}^{obAeAB}, D_{M,N}^{obBeAB}\right). \tag{28}$$

From (27), we have the following:

$$\begin{aligned} D_{M,N}^{obABeAB} &= \min\begin{pmatrix} \min_{1\le i\le M} D_{i,N}^{obA}, \min_{1\le j\le N} D_{M,j}^{obA} \\ \min_{1\le j\le N} D_{M,j}^{obB}, \min_{1\le i\le M} D_{i,N}^{obB} \end{pmatrix} \\ &= \min\begin{pmatrix} \min_{1\le i\le M}\left(\min\left(D_{i,N}^{obA}, D_{i,N}^{obB}\right)\right) \\ \min_{1\le j\le N}\left(\min\left(D_{M,j}^{obA}, D_{M,j}^{obB}\right)\right) \end{pmatrix}. \end{aligned} \tag{29}$$

From (22), we have the following:

$$D_{M,N}^{obABeAB} = \min\left(\min_{1\le i\le M} D_{i,N}^{ob}, \min_{1\le j\le N} D_{M,j}^{ob}\right), \tag{30}$$

$D_{M,N}^{obAeAB}$, $D_{M,N}^{obBeAB}$, and $D_{M,N}^{obABeAB}$ do not meet the half-full-match condition and are hard to make sense of in the application. We mention them just for completeness. They have the same formation as $D_{M,N}^{oe}$, except with different cumulative distances. Therefore, the computation complexity is the same as traditional DTW.

The left-n-step continuity DTW for series a and b is denoted by $D_{M,N}^{nA}$, which means the growth step for series a is expanded to n from 1. Its recursive formation is defined as follows:

$$D_{M,N}^{nA} = \min\begin{pmatrix} \min_{1\le d\le n} D_{M-d,N}^{nA} \\ \min_{0\le d\le n} D_{M-d,N-1}^{nA} \end{pmatrix} + \delta_{M,N}. \tag{31}$$

On the other hand, the right-n-step continuity DTW for series a and b is denoted by $D_{M,N}^{nB}$, which means the growth step for series b is expanded to n from 1. The recursive formation is the same as $D_{M,N}^{nA}$, and it is defined as follows:

$$D_{M,N}^{nB} = \min\begin{pmatrix} \min_{1\le d\le n} D_{M,N-d}^{nB} \\ \min_{0\le d\le n} D_{M-1,N-d}^{nB} \end{pmatrix} + \delta_{M,N}. \tag{32}$$

In (31) and (32), for both $D_{i,j}^{nA}$ and $D_{i,j}^{nB}$, when $i < 1$ or $j < 1$, it means the matched scheme does not exist, thus having $D_{i,j}^{nA} = \infty$ and $D_{i,j}^{nB} = \infty$. It is obvious that they are asymmetric, which is expressed by $D_{M,N}^{nA} \ne D_{M,N}^{nB}$. On the other hand, same as other asymmetric DTW, it has $D_{M,N}^{nA} = D_{N,M}^{nB}$. The recursive formation is the same as traditional DTW, except for the needs to search $2(n-1)$ more elements for each step, and the computation complexity is as traditional DTW.

We summarize all these generalized DTW by their features, such as symmetric, half-full, local distance for start margin, and whether search-all-end margin

In Table 3, the star marker ($*$) means that the feature cannot be ensured because of the symmetric feature. The n-step continuity DTW $D_{M,N}^{nA}$ and $D_{M,N}^{nB}$ can be combined with other boundary-free DTW. The symmetric front-back-boundary free DTW $D_{M,N}^{obe}$ needs to calculate both asymmetric $D_{M,N}^{obeA}$ and $D_{M,N}^{obeB}$.

Therefore, we proposed a generalized asymmetric algorithm to calculate all these generalized DTW. This proposed algorithm's structure is the same as traditional DTW's dynamic programming algorithm.

Algorithm 1 is the asymmetric generalized DTW Search (AGDTW_Search). It is used to search the generalized DTW for two series A and B. The parameters oba, obb, oea, and oeb are bool switchers to disable the boundary condition. The parameters istep and jstep control the moving step for generalizing the continuity condition. Default 1 is equal to traditional DTW's continuity condition. In the traditional DTW's dynamic programming implementation, (1) it initializes the cumulative distance matrix and traceback matrix and (2) forward passes to set each element of the cumulative distance matrix and traceback matrix. After these two processes, the cumulative distance matrix is generated. Then, the traditional DTW just chooses the last element of both matrices as the end position of the matched scheme.

The differences between AGDTW and traditional DTW are in the three following aspects: (1) the initialization of cumulative distance matrix, depending on parameter oba and obb whether set. AGDTW would initialize the cumulative distance matrix's first column or first row by local distance instead of the cumulative distance. (2) In the forward pass process, the back-search range depends on parameter isetp and jsetp, which are to control the continuity condition. (3) After the forward pass process is over, the whole cumulative distance matrix is generated, depending on parameters oea and oeb whether set. The end position of the matched scheme is searched in the cumulative distance matrix's last column and last row.

In algorithm 1 AGDTW_Search, the first row is the initialization of the cumulative distance matrix and traceback matrix. The 2^{nd} to 6^{th} rows are the forward pass processes, which are used to set all elements of the cumulative distance matrix D and traceback matrix K. The 7^{th} to 12^{th} rows are used to search the end position of the matched scheme.

Algorithm 2 is Init_cumulative_and_trace_matrix. It is used to initialize the cumulative distance matrix D and traceback matrix K. The 1^{st} and 2^{nd} rows have generated two empty matrices for D and K. From the 3^{rd} to 7^{th} rows, initialize the two matrices as removing the front-boundary condition, which directly uses the local distance to set the start edges of the cumulative distance matrix D, and all elements of the traceback matrix are set to (0,0), which means the traceback end. From the 8^{th} and 13^{th} rows, the first column and first row of D from the local distance to cumulative distance are adjusted, depending on the parameters oba and obb whether set. In the meantime, the traceback matrix K would be adjusted accordingly.

When the execution of algorithm 1 AGDTW_Search is completed, we get the cumulative value alone. In practice, we need to trace back to get the index pair of the matched scheme.

Algorithm 3 is Trace_back_index_pair_list. It is the same as traditional DTW's path traceback process. We trace back K starting from the position index pair (i_{end}, j_{end}), which returned from algorithm 1 AGDTW_Search. The 5^{th}

AGDTW_Search
Input: A, B: two series
istep, jstep: n-step for series A and B, default 1
oba, obb, oea, oeb: boundary-free switcher
Return: dtw: cumulative DTW value
iend, jend: end index pair of matched scheme
D, K: cumulative distance & traceback index matrix
(1) D, K = **Init_cumulative_and_trace_mat** (A, B, oba, obb)
(2) for i in 1 . . . A.length-1
(3) for j in 1. . . B.length-1
(4) $i_{\min}$, $j_{\min}$ = find min in D[i-i_{step}. . .i, j-j_{step}. . .j]
(5) D[i, j] = D[$i_{\min}$, $j_{\min}$] + local_distance(A[i],B[j])
(6) K[i, j] = ($i_{\min}$, $j_{\min}$)
(7) i_{end}, j_{end} = A.length-1, B.length-1
(8) if oea
(9) i_{end}, j_{end} = index of minimal in D's last column
(10) if oeb
(11) i_{end}, j_{end} = A.length-1, index of minimal in D's last row
(12) dtw = D[i_{end}, j_{end}]
(13) return dtw, i_{end}, j_{end}, D, K

Algorithm 1: (AGDTW_Search).

row means the traceback end condition is index pair (0,0). The 8th row means the idx_pair_list needs to be reversed because the traceback process is from the end position to the start position. Finally, we generated the index pair list of the matched scheme. Furthermore, we can get the local distances of all index pairs.

Trace_back_index_pair_list
Input: iend, jend: end index pair of matched scheme
K: traceback index matrix
Return: idx_pair_list: index pair list of matched scheme
(1) idx_pair_list = []
(2) i, j = iend, jend
(3) while True:
(4) idx_pair_list.append([i, j])
(5) if i = = 0 and j = = 0
(6) break
(7) i, j = K[i, j]
(8) idx_pair_list.reverse()
(9) return idx_pair_list

2.5. Normalization of Cumulative Distance. The generalized DTW search process is to find a matched scheme $\mathbf{P}$ that minimizes the cumulative DTW value under some specific conditions. The cumulative DTW value is the sum of all local distances of this matched scheme $\mathbf{P}$. Let the matched scheme $\mathbf{P} = [(i_1, j_1), (i_2, j_2), \ldots, (i_K, j_K)]$. We define $d_k = \delta_{i_k, j_k}$ and denote the local distance of $\mathbf{P}$'s k^{th} index pair ((i_k, j_k)), which is the local distance between the i_k^{th} element of series A and j_k^{th} element of series B. Therefore, we define the local distance sequence of the matched scheme $\mathbf{P}$ as $\mathbf{d} = [d_1, d_2, \ldots, d_K]$.

The cumulative DTW value can be expressed as follows:

$$d_{\text{sum}} = \sum_{i=1}^{K} d_i. \tag{33}$$

In application, it is hard to directly use the cumulative DTW because of the various sampling frequencies between different trajectories. More sampling amount would lead to more matched index pairs and a greater cumulative distance, even if each local distance is small.

Therefore, the cumulative distance is needed to be normalized to solve this problem. The basic normalization is average, which is expressed as follows:

$$d_{\text{avg}} = \frac{1}{C} \sum_{i=1}^{K} d_i. \tag{34}$$

The normalization parameter C of the average normalized distance can use a different value, such as series A's length M, series B's length N, the max length of matched scheme $M + N - 1$, the actual length of the matched scheme K, or the max(M, N). Each type of average normalization

parameter would lead to a minor effect on the final normalized result. In general, it is reasonable to set C to the length of matched scheme K.

Overall, the average normalization prefers lower results. Assume a matched scheme with K pairs, and only one local distance is nonzero and has a very large value. When K is big enough, the average normalization value is close to zero, which means the two series are very close. However, in fact, there is an extremely different element in these two series. The average normalization ignored this difference. Therefore, the average normalization prefers to make a lesser result, which is not good for reflecting an extremely different element between the two series.

On the contrary, there is max normalization, which is expressed as follows:

$$d_{max} = \max_{1 \le i \le K} (d_i). \tag{35}$$

Max normalization is choosing the max distance among all matched pairs. It is sensitive to the extreme local distance difference position between the two series. Accompany with the DTW search process, it first finds the minimal cumulative distance among all possible matched schemes, then finds the maximal local distance among all index pairs of that minimal cumulative distance matched scheme.

On the contrary, to average normalization, max normalization is sensitive to an extreme position. Therefore, we propose weighted normalization to balance the average and max normalization. The weighted normalization is expressed as follows:

$$d_w = \sum_{i=1}^{K} w_i d_i, \tag{36}$$

where $\sum_{i=1}^{K} w_i = 1$ and $w_i \ge 0$. For weighted normalization, the greater the local distance, the larger the weight. Therefore, weighted normalization can reflect the larger difference among all matched pairs and is not too sensitive. Weighted normalization with norm1 is expressed as follows:

$$d_{\text{norm }1} = \frac{\sum_{i=1}^{K} d_i d_i}{\sum_{J=1}^{K} d_j} = \sum_{i=1}^{K} \frac{d_i}{\|\mathbf{d}\|_1}, \qquad d_i = \frac{\|\mathbf{d}\|_2^2}{\|\mathbf{d}\|_1^1}. \tag{37}$$

For complementary, we propose norm-P weighted normalization, which is expressed as follows:

$$d_{\text{norm }P} = \sum_{i=1}^{K} \frac{d_i^{\,P}}{\|\mathbf{d}\|_P^P}, \qquad d_i = \frac{\|\mathbf{d}\|_{P+1}^{P+1}}{\|\mathbf{d}\|_P^P}. \tag{38}$$

From the definition, we have $d_{\text{avg}} \le d_{\text{norm }1} \le d_{\text{norm }2} \cdots \le d_{\text{norm }\infty} = d_{\max}$. In this paper, we use norm1 weighted normalization.

2.6. Asymmetric Distance DPCA Clustering. In this paper, the similarity measurement for clustering is asymmetric generalized DTW. We apply Open-Begin-End-boundary, extend an n-step continuity for series A, and use norm-1 normalization for these cumulative distances. In the matched scheme, series B must be fully matched. This measurement can match the pattern in which series A contains series B. Therefore, the longest trajectory that contains more short trajectories would have greater density.

We adjust DPCA to find the longest trajectory using asymmetric generalized DTW. We name it asymmetric density peak clustering (ADPC). Similar to the original DPCA, ADPC's most work is to calculate the local density and minimal distance to other samples that have greater density. The main difference is how to calculate the minimal distance.

Let D be the distance matrix among all trajectories. D_{ij} is norm-1 normalized asymmetric generalized DTW from the i^{th} trajectory to j^{th} trajectory. As we use asymmetric DTW, the distance matrix D is asymmetric, with D_{ij} not equal to D_{ji}. When D_{ij} is small, it means that the i^{th} trajectory contains the j^{th} trajectory.

The calculation of local density for each trajectory is the same as DPCA, with a hyperparameter cutoff distance and Gaussian kernel bandwidth. The calculation of the minimal distance to the greater density sample is a little different from DPCA. To calculate the i^{th} trajectory's minimal distance, we search it from all D_{ji}, namely the j^{th} column of matrix D. The minimal distance means the minimal distance from its supertrajectories to itself. The super trajectory mean density is greater than itself.

Algorithm 4 is ADPC_make_decision_graph. It is to calculate the local density list rho and minimal distance list delta. After rho and delta are calculated, same as DPCA, samples with greater rho and delta would be assumed as cluster centers.

As we use asymmetric distance and DTW does not meet triangle inequality, the process of allocating the sample to its cluster is just only one pass and needs to consider the difference between the two asymmetric distances.

Algorithm 5 is ADPC_allocate_sample. It needs two thresholds to decide whether a sample belongs to a cluster. The distance from cluster center to all cluster members must be less than the threshold thre_c2m. The distance from all cluster members to the cluster center must be less than threshold thre_m2c. In the 13^{th} row, when a sample does not meet the threshold conditions, it would be set as the outlier. In the 15^{th} row, when allocating a sample to a cluster, we consider both asymmetric distances by adding them. Then, we assign the sample to a cluster in which the distance sum is the least.

3. Experiment Results

In this section, we apply our proposed generalized DTW algorithm to a real GPS trajectory dataset to illustrate the ability to recognize the inclusive relationship between trajectories. We compare different DTW normalization

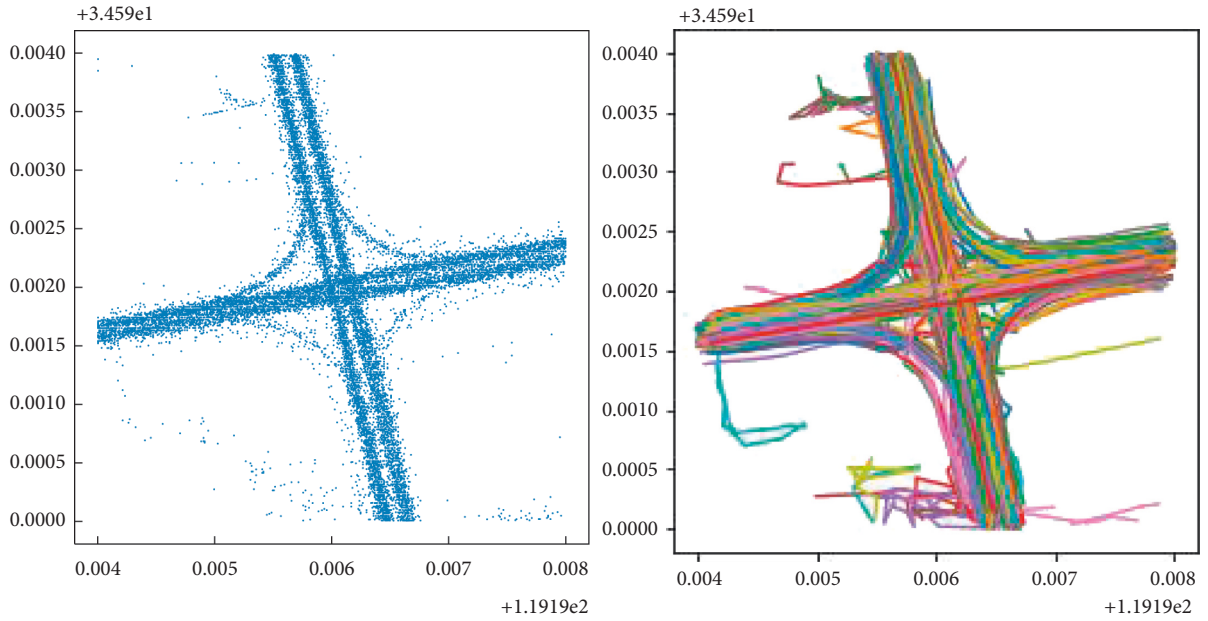


Figure 4: Scatter of GPS trajectory for experiment dataset.

methods, and the asymmetric generalized DTW clustering ADPC result is given.

The experiment dataset is an intersection area GPS trajectory dataset with 18,813 sampling points and 1654 trajectories, as shown in Figure 4.

The trajectory data consists of samplings located by longitude and latitude. When calculating two samples' direct local distance, we use the arc distance of the Earth. Let $p_1 = (\text{lon}_1, \text{lat}_1)$, $p_2 = (\text{lon}_2, \text{lat}_2)$, and lon_1, lat_1, lon_2, and lat_2 be measured by radian, and the local distance δ is defined as follows:

$$\delta = R \cdot \arccos\left(\cos\left(\text{lat}_1\right)\cos\left(\text{lat}_2\right)\cos\left(\text{lon}_1 - \text{lon}_2\right) - \cos\left(\text{lat}_1\right)\cos\left(\text{lat}_2\right)\right), \tag{39}$$

where R is the average radius of Earth between lat1 and lat2, which is defined as follows:

$$R = \frac{1}{2}\left(\sqrt{\left(R_E \cos\left(\text{lat}_1\right)\right)^2 + \left(R_P \sin\left(\text{lat}_1\right)\right)^2} + \sqrt{\left(R_E \cos\left(\text{lat}_2\right)\right)^2 + \left(R_P \sin\left(\text{lat}_2\right)\right)^2}\right), \tag{40}$$

where RE is Earth radius on the equator, which is approximately 6,378,137 meters, and R_P is the Earth radius on the polar, which is approximately 6,356,725 meters. We use this arc distance to reduce local distance variation by different latitudes and because of the use of the meter unit so that it is good for setting the threshold intuitively later.

3.1. GDTW Comparison. For various generalized DTW, we compare the results in the same trajectory pair. For the boundary condition extending, we compared OBA, OBB, OEA, OEB, OBEA, OBEB, OE, OB, and OBEF. Also, we compared different gap steps for continuity condition extending.

The traditional DTW's matching result is shown in Figure 5. In traditional DTW, the start and end points for both trajectory A and B are matched.

Next, we demonstrate the asymmetric generalized DTW of the boundary condition extending. They are Open Begin for series A (OBA), Open Begin for series B (OBB), Open End for series A (OEA), Open End for series B (OEB), Open Begin and End for series A (OBEA), Open Begin, and End for series B (OBEB).

In Figure 6, DTW results with different generalized boundary conditions are demonstrated. The start or end points for trajectories A and B are not matched necessary. Therefore, it could be recognized as the inclusive relationship between trajectories.

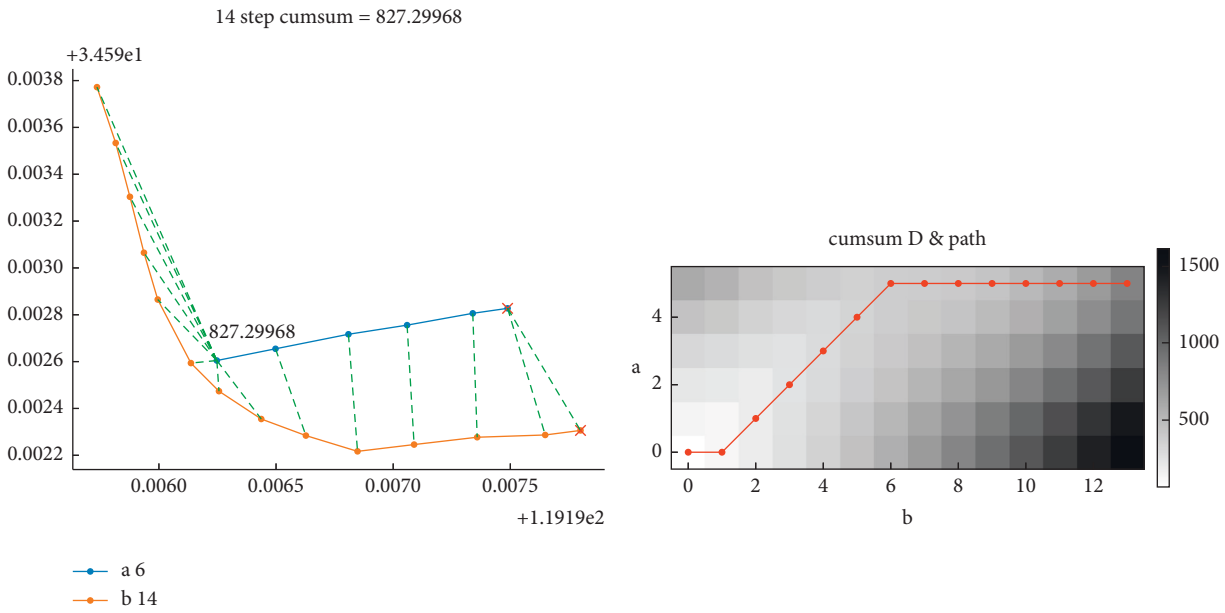


Figure 5: Spatial distribution and matched pairs diagram of traditional DTW between two series a and b.

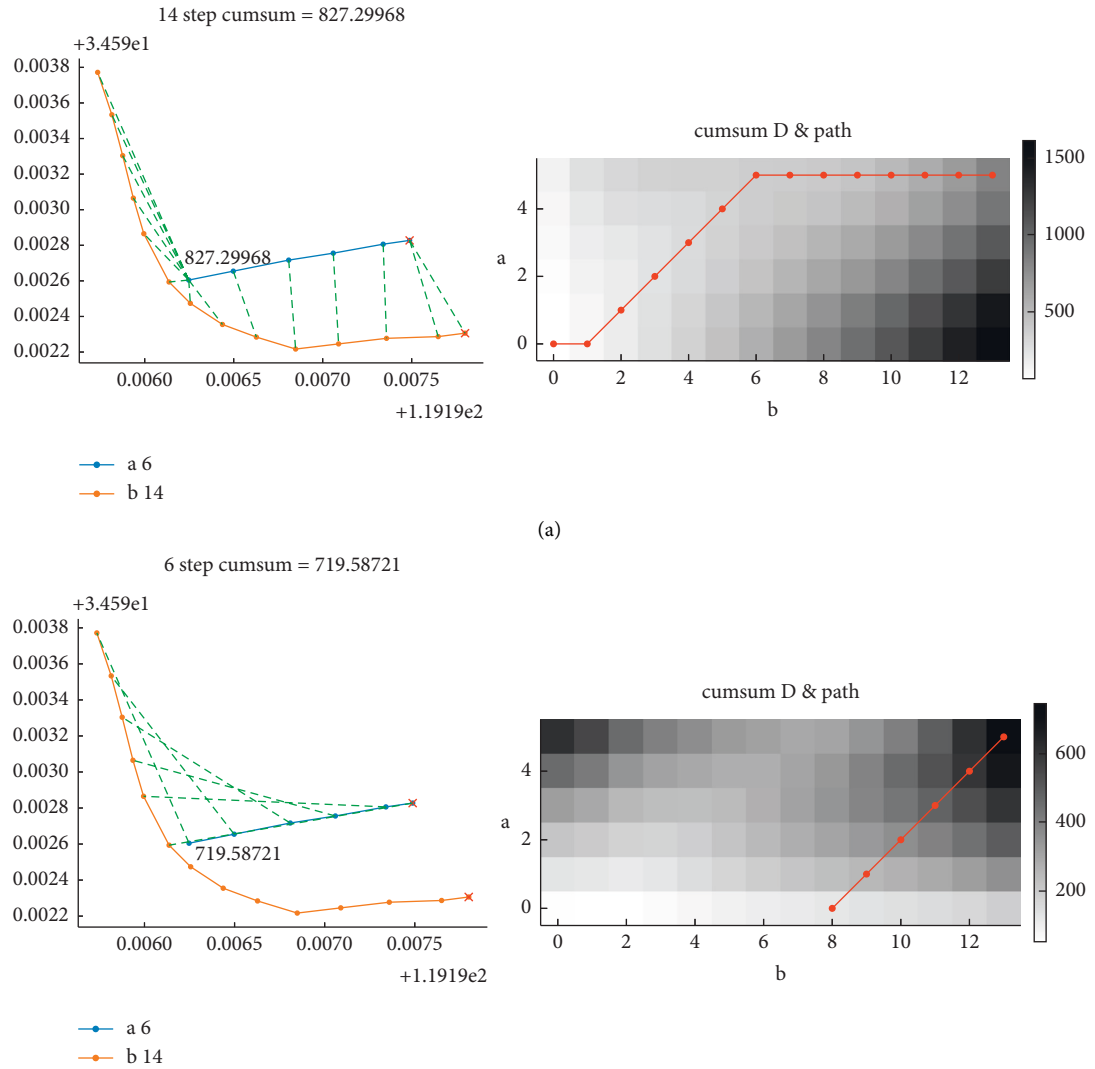
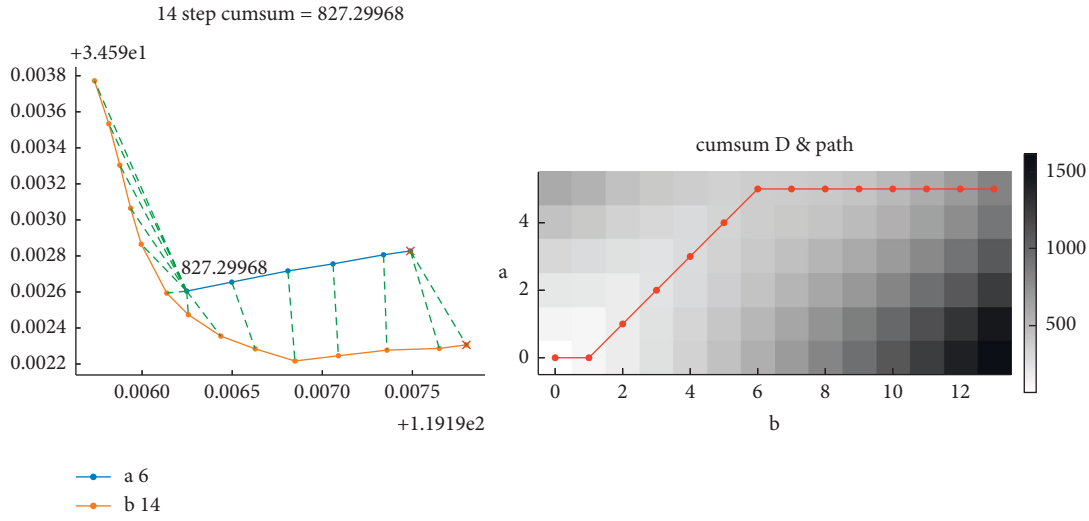
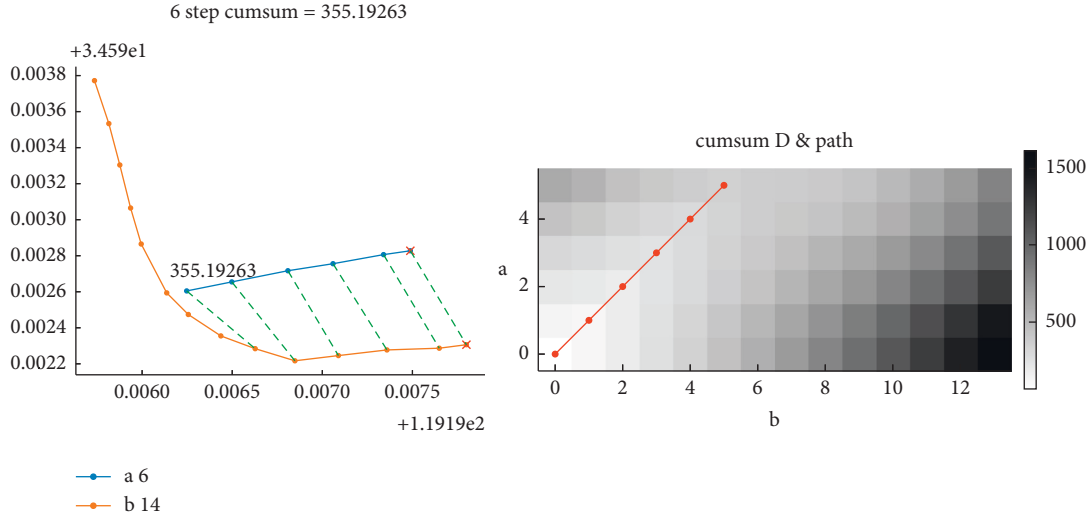


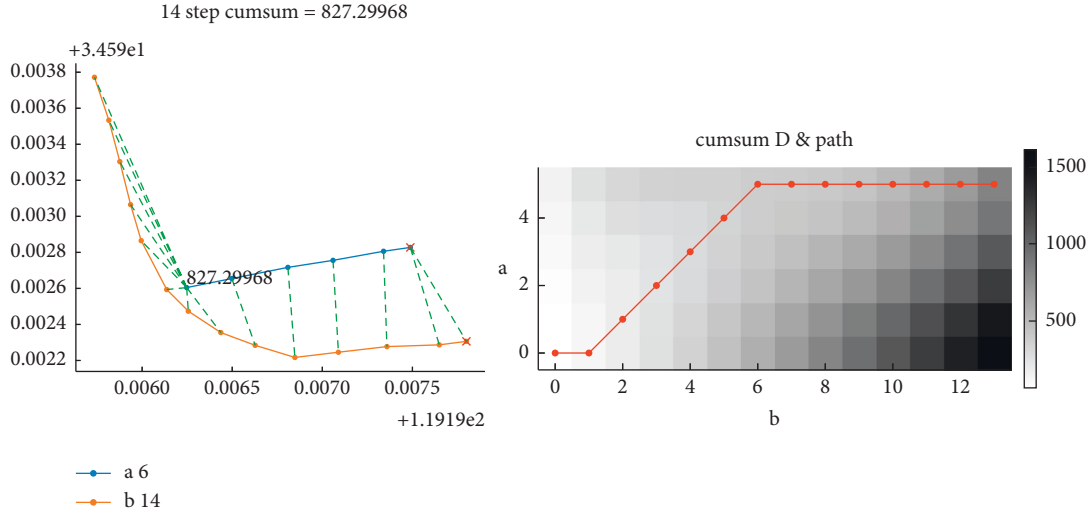
Figure 6: Continued.



(c)

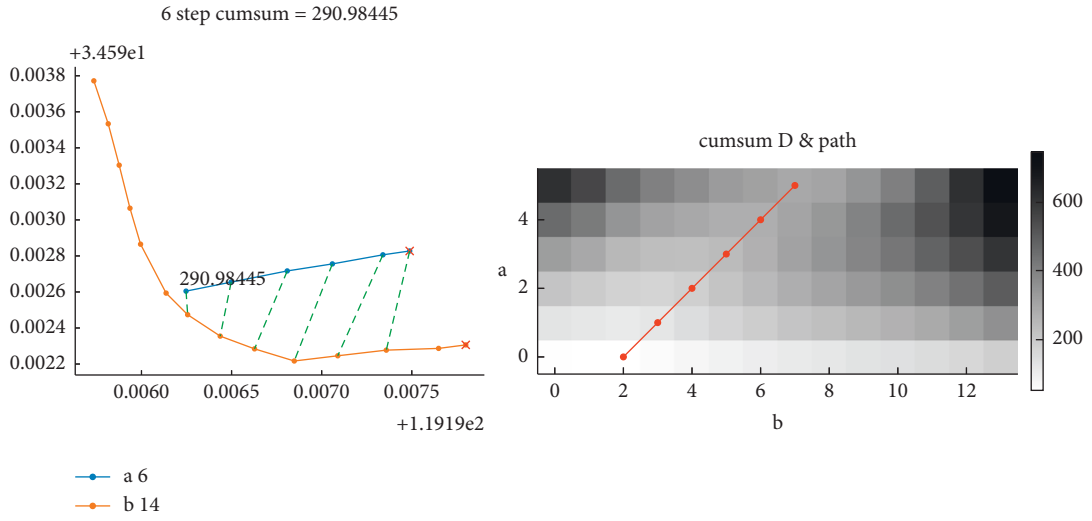


(d)



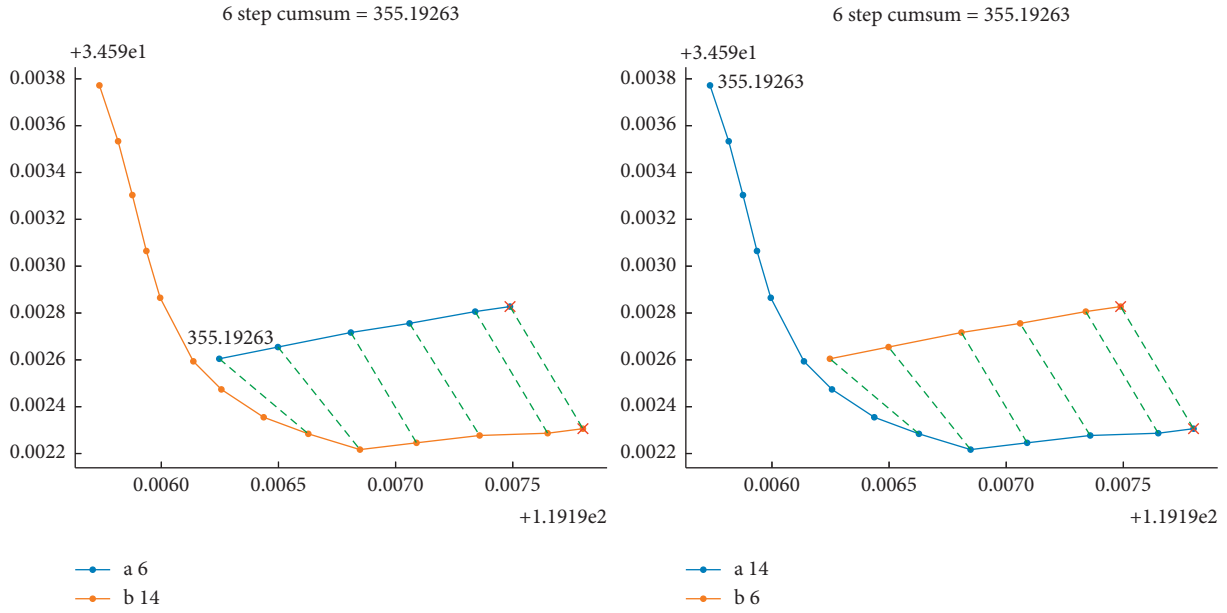
(e)

Figure 6: Continued.



(f)

Figure 6: Asymmetric generalized DTW for boundary condition. (a) OBA. (b) OBB. (c) OEA. (d) OEB. (e) OBEA. (f) OBEB.



(a)

Figure 7: Continued.

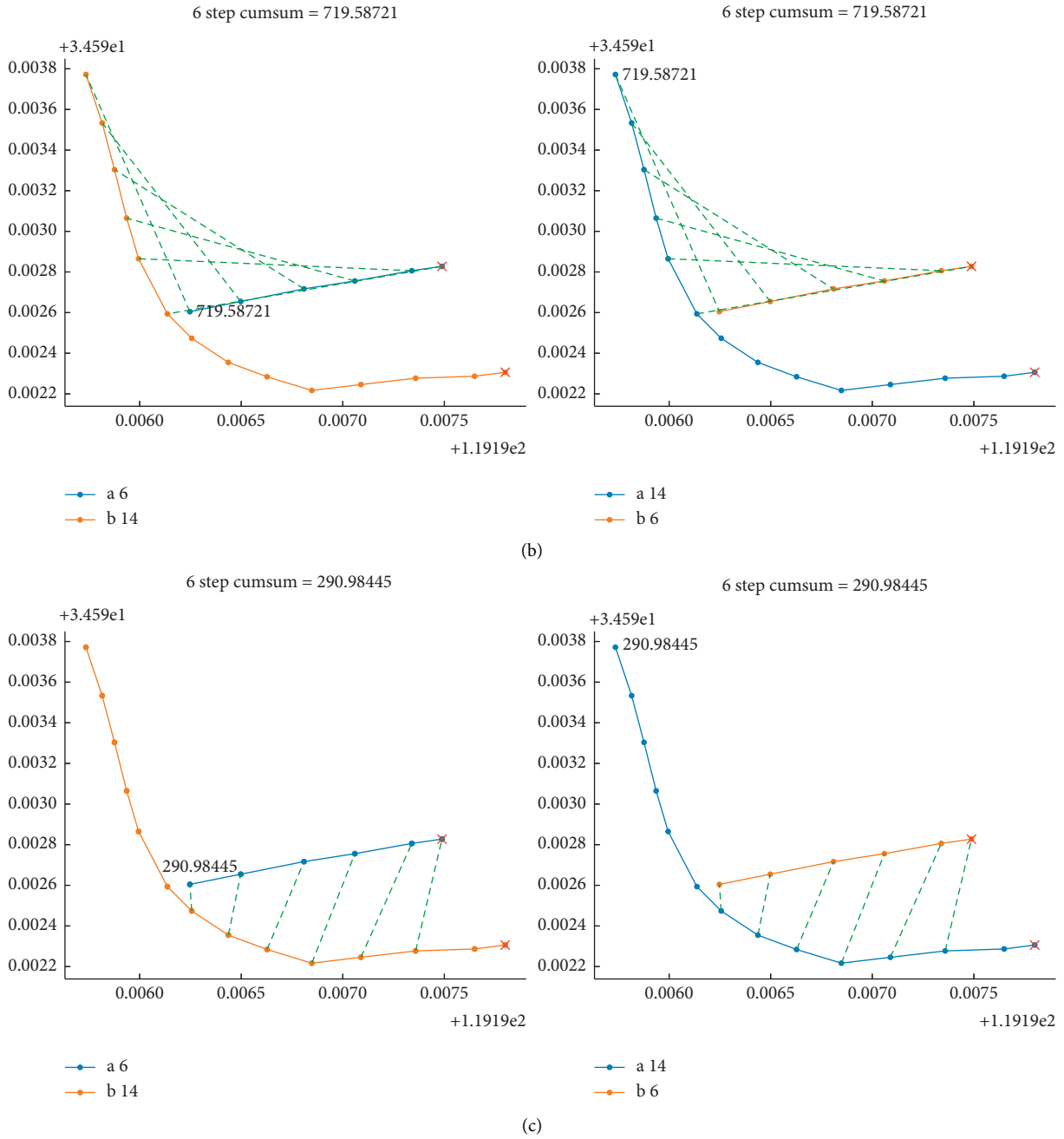


Figure 7: Symmetric generalized DTW for boundary condition. (a) OE. (b) OB. (c) OBEF.

For completeness, we demonstrate the symmetric form of boundary condition generalization. They are Open End (OE), Open Begin (OB), Open Begin End, and full-match on a series (OBEF).

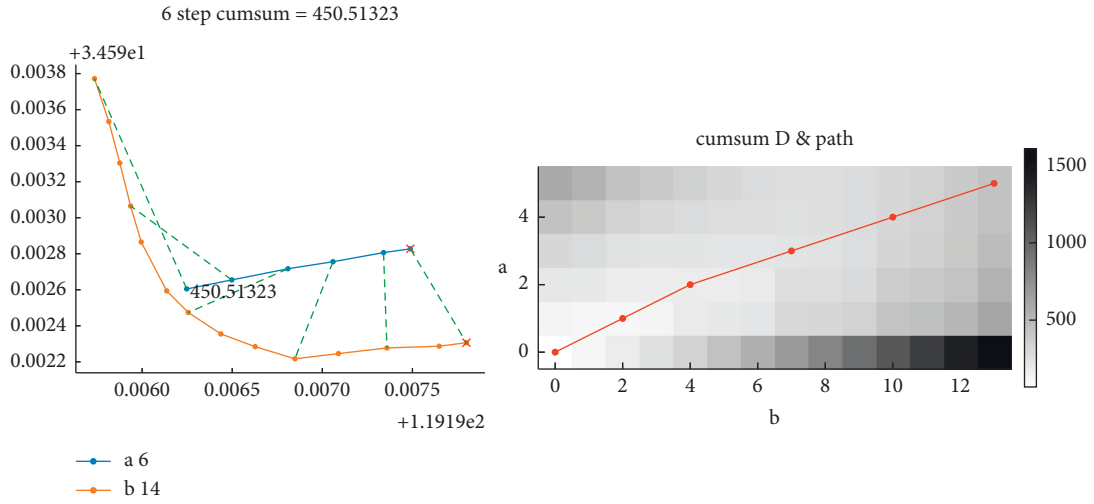
In Figure 7, DTW results in the symmetric form with different generalized boundary conditions are demonstrated. The start or end points for trajectories A and B are not matched necessarily.

Next, we compared DTW results for extending continuity conditions with different step gaps for series B. The compared step gaps are 3-step, 5-step, and 7-step.

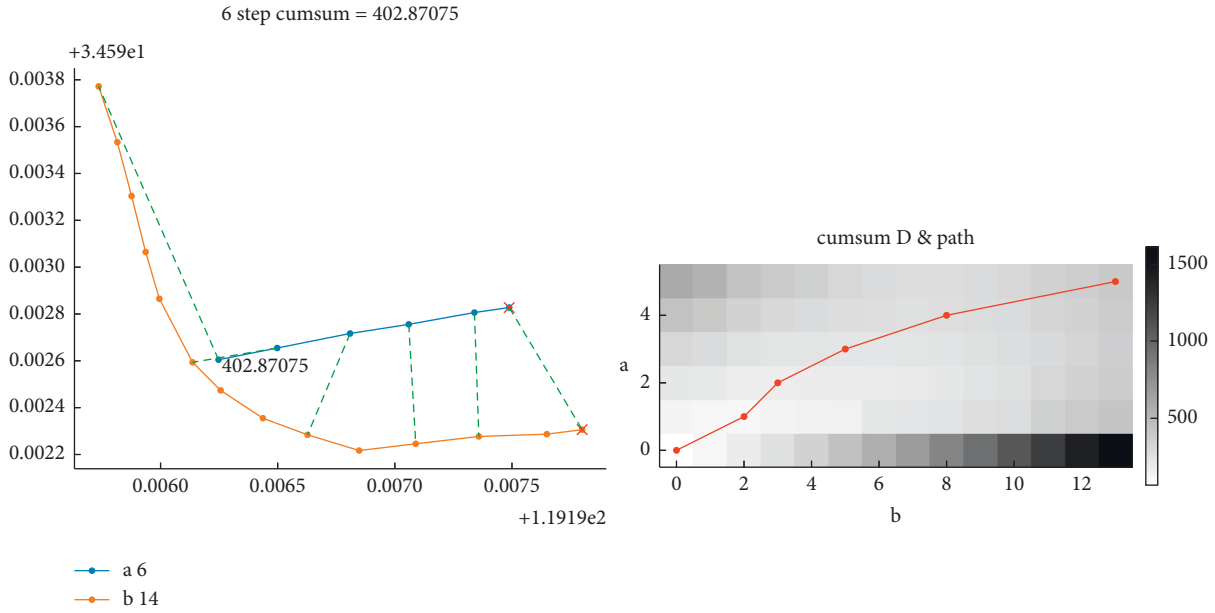
In Figure 8, we compared different step results. The bigger the step gap, the lower the cumulative distance. When the step is big enough to a critical value, because of the single full-match condition, the cumulative distance is not reduced more. Finally, we use OBEA5 as an asymmetric similarity measurement, which is an open begin and end boundary and 5-step continuity for trajectory A, and it leads to assuming series A as the supertrajectory.

3.2. Normalization Comparison. We compare the affection of various normalization methods by different sampling amounts for the same trajectories using traditional DTW.

In Figure 9, trajectory A is resampling in different amounts. The more the sampling amount, the bigger the

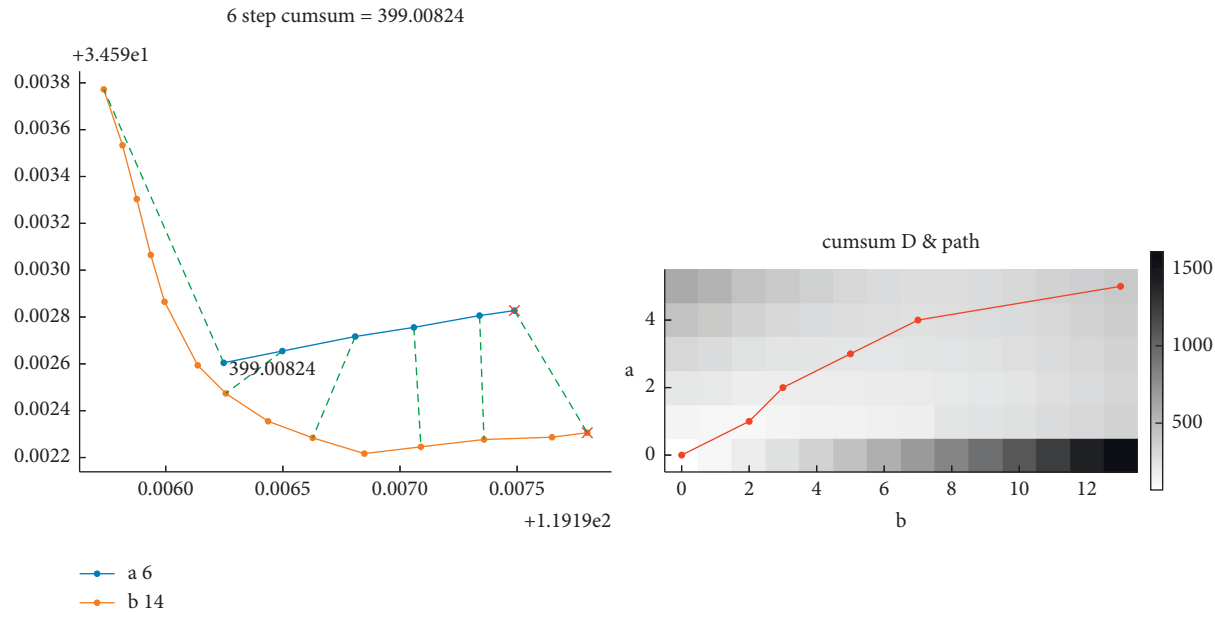


(a)



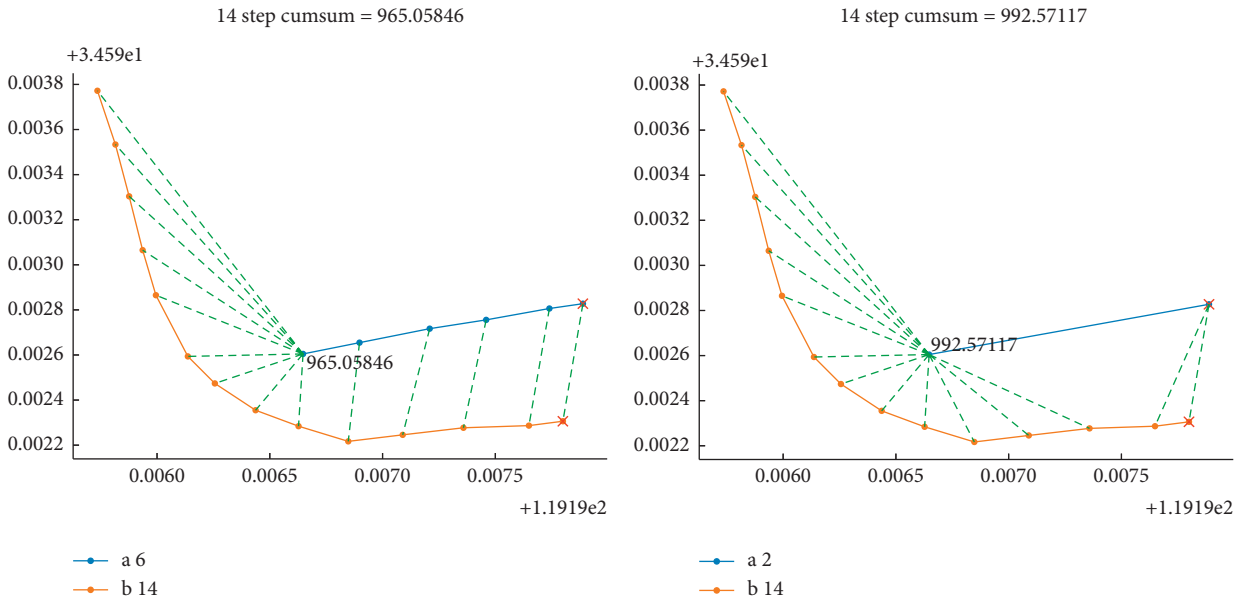
(b)

Figure 8: Continued.



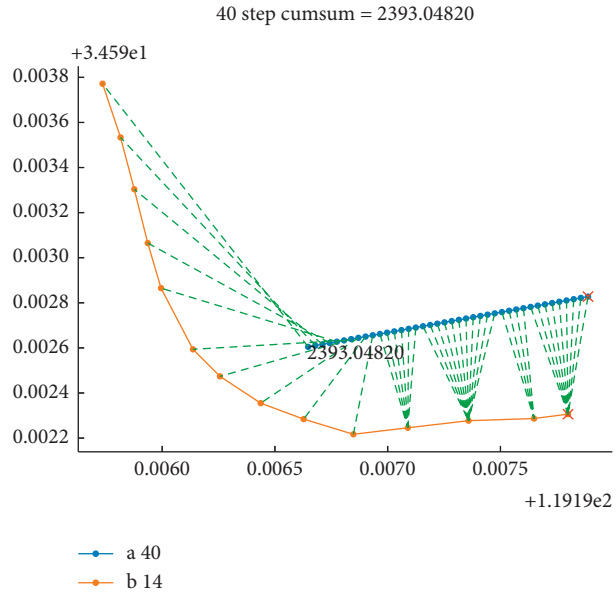
(c)

Figure 8: Comparison of DTW with different n-step continuity. (a) 3-step continuity for trajectory a. (b) 5-step continuity for trajectory a. (c) 7-step continuity for trajectory a.



(a) (b)

Figure 9: Continued.



(c)

Figure 9: Spatial distribution and matched pairs diagram of DTW between two series *a* and *b*. (a) Original samplings for trajectory *a*, (b) 2 samplings for trajectory *a*, and (c) 40 samplings for trajectory *a*.

```
    Init_cumulative_and_trace_matrix
    Input: series A and B, oba, obb
    Return: D: cumulative distance matrix
            K: traceback index matrix
(1)   allocate cumulative distance matrix D[A.length, B.length]
(2)   allocate traceback matrix K with same size to D
(3)   D[0, 0] = local_distance(A[0], B[0])
(4)   for i in 1 ... A.length-1
(5)         K[i,0], D[i, 0] =  (0,0), local_distance(A[i], B[0])
(6)   for j in 1 ... B.length-1
(7)         K[0,j], D[0, j] = (0,0), local_distance(A[0], B[j])
(8)   if not oba: # use cumulative distance for series A
(9)         for i in 1 ... A.length-1
(10)          K[i, 0], D[i, 0] = (i − 1, 0), D[i − 1, 0] +D[i, 0]
(11)  if not obb: # use cumulative distance for series B
(12)        for j in 1 ... B.length-1
(13)          K[0, j], D[0, j]  =  (0, j − 1), D[0, j − 1] +D[0, j]
(14)  return D, K
```

Algorithm 2: (Init_cumulative_and_trace_matrix).

cumulative distance. The comparison for different normalization results is shown in Table 4.

M is the sampling amount of trajectory A. **K** is the length of the matched scheme, which represents the number of matched pairs. The **sum** is the cumulative distance for all matched pairs. Therefore, it increases greatly along with the sampling amount. **Nmax** is the max local distance among matched pairs. It reflects the most difference between two trajectories but lost most information and is seriously affected by noise. **Naverage** is calculated by the sum dividing **K**, which is an easy and good normalization, however, it flattens the local difference. **Nmedian** flattens more and is a little complex to calculate, just for comparison. **Nnorm-1** is good as **Naverage** and can distinguish the effect of greater local distances. Therefore, we use **Norm-1** normalization in clustering.

3.3. ADPC Clustering. Before ADPC clustering, the similarity matrix is calculated by GDTW (open begin and end boundary and 5-step continuity for trajectory A, Nnorm-1 normalization) for 1654 trajectories. However, the

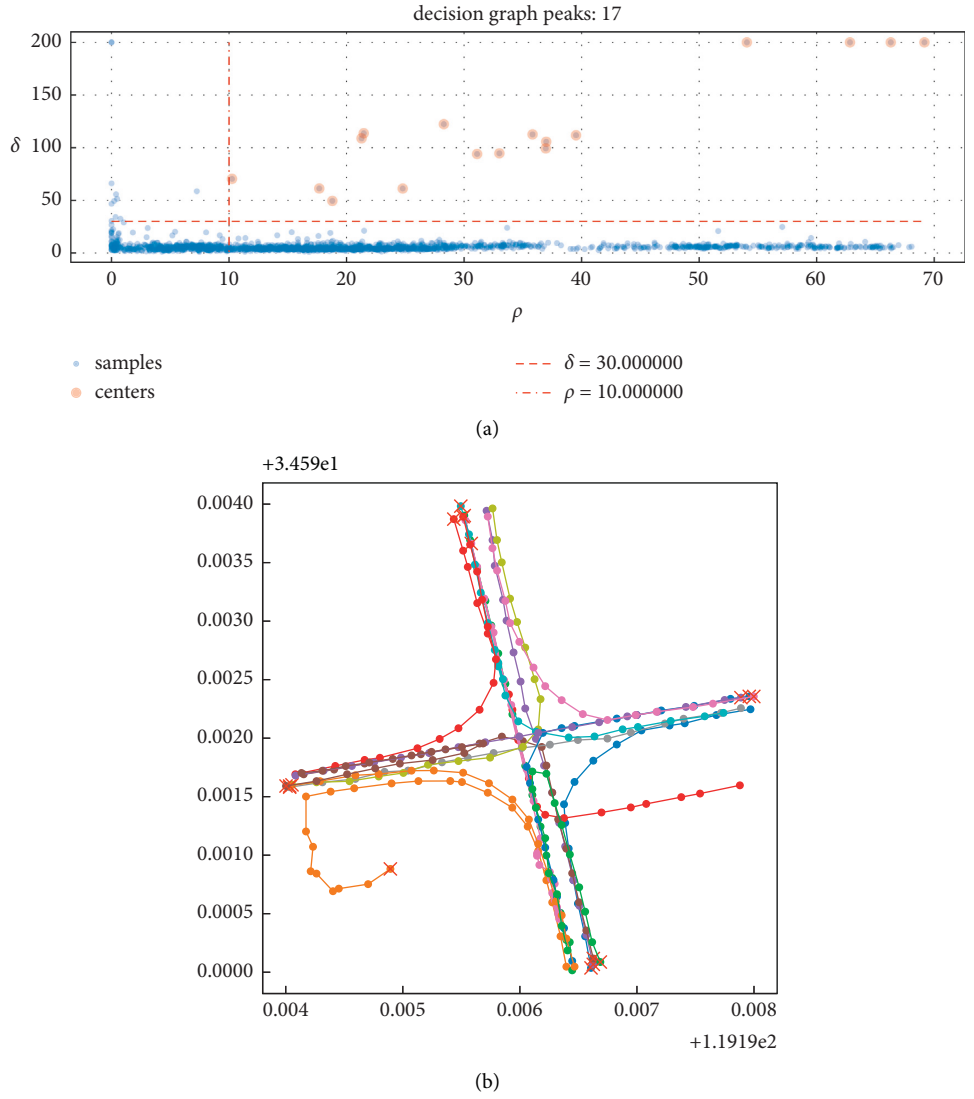


Figure 10: Decision graph and cluster center trajectories. (a) Decision graph. (b) Cluster center trajectories.

calculation efficiency is not satisfactory. In our implementation by python, it takes 2 hours to calculate the similarity matrix.

In ADCP clustering, the hyperparameter cutoff distance is set to 15 meters, and Gaussian kernel bandwidth is also set to 15.

In Figure 10(a), there are 17 clustering centers in the upper right area, which are divided by two dotted lines. The horizontal dotted line is the threshold for min-distance δ, which is 30. The vertical dotted line is the threshold for local density ρ, which is 10. The corresponding trajectories of cluster centers are demonstrated in Figure 10(b).

APDC is designed to find the maximum inclusive trajectory as the cluster center, however, in reality, there are some long singular trajectories that would be clustered as centers. Therefore, the sample assignment procedure of ADPC is worked to filter those singular cluster centers.

The clustering result after sample assignment is shown in Figure 11. The subtitle like "p1 247 26" means the cluster center 1, 247 is the trajectory sample's serial number, and 26 is the number of trajectories belonging to this cluster. The long singular trajectories, such as p2 399 and p4 578 are regarded as singular trajectories and are not assigned any other trajectories.

3.4. Discussion. We use the arc distance of the Earth to measure the local distance of two GPS sampling points. It can reflect the truth distance better than the Euclidean distance of the latitude and longitude, although the improvement is minor.

We compared various GDTW proposed in this section by the same two trajectories. It proved the validity of GDTW and illustrated the asymmetric boundary-free DTW with 5-step, which is good for recognizing the inclusive relationship between the two trajectories.

By comparing various normalization methods, we proved that Norm-1 is good to reserve more information on difference, while, in the meantime, we reflect the similarity as K average normalization.

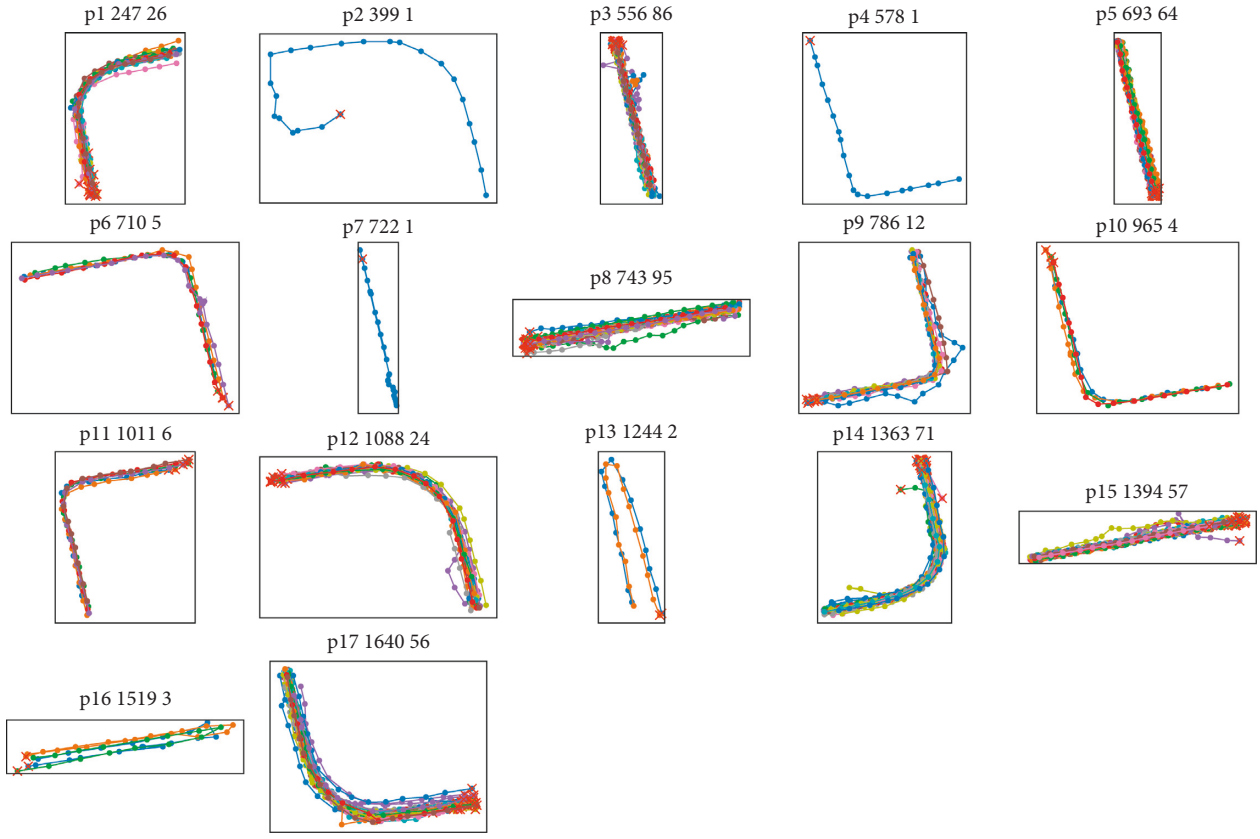


Figure 11: Clustering results of ADPC.

Table 3: Generalized DTW.

GDTW	Local-distance		Search-all-end		Half-full		Symmetric
	a	b	a	b	a	b	
$D^{oeA}_{M,N}$	No	No	Yes	No	No	Yes	No
$D^{oeB}_{M,N}$	No	No	No	Yes	Yes	No	No
$D^{obA}_{M,N}$	Yes	No	No	No	No	Yes	No
$D^{obB}_{M,N}$	No	Yes	No	No	Yes	No	No
$D^{oe}_{M,N}$	No	No	Yes	Yes	*	*	Yes
$D^{ob}_{M,N}$	Yes	Yes	No	No	*	*	Yes
$D^{obeA}_{M,N}$	Yes	No	Yes	No	No	Yes	No
$D^{obeB}_{M,N}$	No	Yes	No	Yes	Yes	No	No
$D^{obe}_{M,N}$	*	*	*	*	*	*	Yes
$D^{nA}_{M,N}$	No	No	No	No	No	Yes	No
$D^{nB}_{M,N}$	No	No	No	No	Yes	No	No

```
    Trace_back_index_pair_list
    Input: iend, jend: end index pair of matched scheme
           K: traceback index matrix
    Return: idx_pair_list: index pair list of matched scheme
(1)     idx_pair_list = []
(2)     i, j = iend, jend
(3)     while True:
(4)        idx_pair_list.append([i, j])
(5)        if i = = 0 and j = = 0
(6)           break
(7)        i, j = K[i, j]
(8)     idx_pair_list.reverse()
(9)     return idx_pair_list
```

Algorithm 3: (Trace_back_index_pair_list).

```
    ADPC_make_decision_graph
    Input: D: normalized asymmetric generalized DTW
           r: cutoff-distance
    Return: rho: local density list
           delta: minimal distance list.
(1)     rho, delta = list[len(D)], list[len(D)]
(2)     for i = 0. . .len(D)-1
(3)        neighbors = D[i][D[I]<=r]4
(4)        rho[i] = gaussan_kernel(neighbors)
(5)     sortedidx = argsort(rho)
(6)     for i = 0. . .len(D)-1
(7)        r i = sortedidx[i]
(8)        if i = = len(D)-1
(9)           delta[ri] = D[:, ri].max()
(10)       else:
(11)          idx = sortedidx[i + 1:end]
(12)          delta[ri] = D[:, ri][idx].min()
(13)    return rho, delta
```

Algorithm 4: (ADPC_make_decision_graph).

```
    ADPC_allocate_sample
    Input: D: normalized asymmetric generalized DTW
           centeridxs: candidate cluster center index list
           thre_c2m: threshold from center to member
           thre_m2c: threshold from member to center
    Return: centeridx_to_memberidxs. outlieridxs:
           outlier index list.
(1)     cm_mat = D[centeridxs,:]
(2)     mc_mat = D[:, centeridxs]
(3)     outlieridxs = []
(4)     centeridx_to_memberidxs = dict((i, []) for i in centeridxs)
(5)     for i = 0. . .len(D)-1:
(6)        if i in centeridx_to_memberidxs:
(7)           centeridx_to_memberidxs[i].append(i)
(8)           continue
(9)        br_c2m = cm_mat[:, i] < thre_c2m
```

Algorithm 5: Continued.

```
(10)        br_m2c = mc_mat[i] < thre_m2c
(11)        ii = true_idxs(br_c2m & br_m2c)
(12)        if len(ii) = = 0:
(13)            outlieridxs.append(i)
(14)            continue
(15)        cj = argmin(cm_mat[j, i] + mc_mat[i, j] for j in ii)
(16)        ci = centeridxs[ ii[cj] ]
(17)        centeridx_to_memberidxs[ci].append(i)
(18)    return centeridx_to_memberidxs, outlieridxs
```

Algorithm 5: (ADPC_allocate_sample).

Table 4: Normalization comparison (unit: meter).

M	K	Sum	Nmax	Nnorm-1	Naverage	Nmedian
6	14	965	154	87	69	56
2	14	993	154	88	71	61
40	40	2393	154	68	60	54

In ADPC clustering, we use asymmetric DTW to acquire the max inclusive trajectory as the clustering center and let it as the representation of the cluster. The result shows that our sample allocation algorithm is good to distinguish noise.

4. Conclusions

The conclusions can be summarized as follows:

(1) A unified DTW algorithm (GDTW) is proposed to calculate the similarity between two trajectories and can recognize inclusive relationships.

(2) To overcome the influence of unbalanced sampling quantity on the trajectories, a weighted normalization method is proposed.

(3) We use the generalized asymmetric DTW as a similarity measurement and proposed a clustering algorithm (ADPC). This clustering algorithm can extract the local max trajectories as cluster representatives to reduce the number of clusters caused by short trajectories.

The limitation of this method is calculation efficiency. In our implementation of GDTW, it takes 2 hours to calculate the similarity matrix for 1654 trajectories. An improvement on this limitation will be conducted in future work.

Data Availability

The data used to support the findings of the study can be obtained from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study was supported by Beijing Social Science Foundation 21XCC013.

References

[1] C. Wang, S. Zourlidou, J. Golze, and M. Sester, "Trajectory analysis at intersections for traffic rule identification," *Geo-Spatial Information Science*, vol. 24, no. 1, pp. 75–84, 2021.

[2] M. Liang, R. W. Liu, S. Li, Z. Xiao, X. Liu, and F. Lu, "An unsupervised learning method with convolutional auto-encoder for vessel trajectory similarity computation," *Ocean Engineering*, vol. 225, Article ID 108803, 2021.

[3] R. W. Liu, M. Liang, J. Nie, W. Y. B. Lim, Y. Zhang, and M. Guizani, "Deep learning-powered vessel trajectory prediction for improving smart traffic services in maritime internet of things," *IEEE Transactions on Network Science and Engineering*, vol. 1, 1 page, 2022.

[4] K. Chen, L. Yao, D. Zhang, X. Wang, X. Chang, and F. Nie, "A semisupervised recurrent convolutional attention model for human activity recognition," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 5, pp. 1747–1756, 2020.

[5] M. Luo, X. Chang, L. Nie, Y. Yang, A. G. Hauptmann, and Q. Zheng, "An adaptive semisupervised feature analysis for video semantic recognition," *IEEE Transactions on Cybernetics*, vol. 48, no. 2, pp. 648–660, 2018.

[6] D. Zhang, L. Yao, K. Chen, S. Wang, X. Chang, and Y. Liu, "Making sense of spatio-temporal preserving representations for EEG-based human intention recognition," *IEEE Transactions on Cybernetics*, vol. 50, no. 7, pp. 3033–3044, 2020.

[7] W. Wang, F. Xia, H. Nie et al., "Vehicle trajectory clustering based on dynamic representation learning of internet of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 6, pp. 3567–3576, 2021.

[8] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Transactions on Acoustics, Speech, & Signal Processing*, vol. 26, no. 1, pp. 43–49, 1978.

[9] E. Keogh and C. A. Ratanamahatana, "Exact indexing of dynamic time warping," *Knowledge and Information Systems*, vol. 7, no. 3, pp. 358–386, 2005.

[10] D. Lemire, "Faster retrieval with a two-pass dynamic-time-warping lower bound," *Pattern Recognition*, vol. 42, no. 9, pp. 2169–2180, 2009.

[11] T. Rakthanmanon, B. Campana, A. Mueen et al., "Searching and mining trillions of time series subsequences under dynamic time warping," in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '12. Presented at the the 18th ACM SIGKDD International Conference*, p. 262, August 2012.

[12] T. Giorgino, "Computing and visualizing dynamic time warping alignments in R: the dtw package," *Journal of Statistical Software*, vol. 31, no. 7, 2009.

[13] L. Rabiner, A. Rosenberg, and S. Levinson, "Considerations in dynamic time warping algorithms for discrete word recognition," *IEEE Transactions on Acoustics, Speech, & Signal Processing*, vol. 26, no. 6, pp. 575–582, 1978.

[14] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 8, pp. 790–799, 1995.

[15] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96 6)*, Portland, Oregon, August 1996.

[16] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, 2014.