Pattern Recognition 44 (2011) 678–693

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

A global averaging method for dynamic time warping, with applications to clustering

François Petitjean [a,b,c,*], Alain Ketterlin [a,b], Pierre Gançarski [a,b]

[a] University of Strasbourg, 7 rue René Descartes, 67084 Strasbourg Cedex, France
[b] LSIIT–UMR 7005, Pôle API, Bd Sébastien Brant, BP 10413, 67412 Illkirch Cedex, France
[c] Centre National d'Études Spatiales, 18 avenue Edouard Belin, 31401 Toulouse Cedex 9, France

ARTICLE INFO

Article history:
Received 13 January 2010
Received in revised form
11 August 2010
Accepted 18 September 2010

Keywords:
Sequence analysis
Time series clustering
Dynamic time warping
Distance-based clustering
Time series averaging
DTW barycenter averaging
Global averaging
Satellite image time series

ABSTRACT

Mining sequential data is an old topic that has been revived in the last decade, due to the increasing availability of sequential datasets. Most works in this field are centred on the definition and use of a distance (or, at least, a similarity measure) between sequences of elements. A measure called dynamic time warping (DTW) seems to be currently the most relevant for a large panel of applications. This article is about the use of DTW in data mining algorithms, and focuses on the computation of an average of a set of sequences. Averaging is an essential tool for the analysis of data. For example, the K-MEANS clustering algorithm repeatedly computes such an average, and needs to provide a description of the clusters it forms. Averaging is here a crucial step, which must be sound in order to make algorithms work accurately. When dealing with sequences, especially when sequences are compared with DTW, averaging is not a trivial task.

Starting with existing techniques developed around DTW, the article suggests an analysis framework to classify averaging techniques. It then proceeds to study the two major questions lifted by the framework. First, we develop a global technique for averaging a set of sequences. This technique is original in that it avoids using iterative pairwise averaging. It is thus insensitive to ordering effects. Second, we describe a new strategy to reduce the length of the resulting average sequence. This has a favourable impact on performance, but also on the relevance of the results. Both aspects are evaluated on standard datasets, and the evaluation shows that they compare favourably with existing methods. The article ends by describing the use of averaging in clustering. The last section also introduces a new application domain, namely the analysis of satellite image time series, where data mining techniques provide an original approach.

1. Introduction

Time series data have started to appear in several application domains, like biology, finance, multimedia, image analysis, etc. Data mining researchers and practitioners are thus adapting their techniques and algorithms to this kind of data. In exploratory data analysis, a common way to deal with such data consists in applying clustering algorithms. Clustering, *i.e.*, the unsupervised classification of objects into groups, is often an important first step in a data mining process. Several extensive reviews of clustering techniques in general have been published [1–4] as well as a survey on time series clustering [5].

Given a suitable similarity measure between sequential data, most classical learning algorithms are readily applicable.

A similarity measure on time series data (also referred to as sequences hereafter) is more difficult to define than on classical data, because the order of elements in the sequences has to be considered. Accordingly, experiments have shown that the traditional Euclidean distance metric is not an accurate similarity measure for time series. A similarity measure called dynamic time warping (DTW) has been proposed [6,7]. Its relevance was demonstrated in various applications [8–13].

Given this similarity measure, many distance-based learning algorithms can be used (*e.g.*, hierarchical or centroid-based ones). However, many of them, like the well-known K-MEANS algorithm, or even Ascendant Hierarchical Clustering, also require an averaging method, and highly depend on the quality of this averaging. Time series averaging is not a trivial task, mostly because it has to be consistent with the ability of DTW to realign sequences over time. Several attempts at defining an averaging method for DTW have been made, but they provide an inaccurate

* Corresponding author at: LSIIT–UMR 7005, Pôle API, Bd Sébastien Brant, BP 10413, 67412 Illkirch Cedex, France. Tel.: +33 3 68 85 45 78; fax: +33 3 68 85 44 55.
E-mail addresses: fpetitjean@unistra.fr (F. Petitjean), alain@unistra.fr (A. Ketterlin), gancarski@unistra.fr (P. Gançarski).

0031-3203/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2010.09.013

notion of average [14], and perturb the convergence of such clustering algorithms [15]. That is mostly why several time series clustering attempts prefer to use the K-MEDOIDS algorithm instead (see [16–18] for examples combining DTW and the K-MEDOIDS algorithm). Throughout this article, and without loss of generality, we use some times the example of the K-MEANS algorithm, because of its intensive use of the averaging operation, and because of its applicability to large datasets.

In this article, we propose a novel method for averaging a set of sequences under DTW. The proposed method avoids the drawbacks of other techniques, and is designed to be used, among others, in similarity-based methods (*e.g.*, K-MEANS) to mine sequential datasets. Section 2 introduces the DTW similarity measure on sequences. Then Section 3 considers the problem of finding a consensus sequence from a set of sequences, providing theoretical background and reviewing existing methods. Section 4 introduces the proposed averaging method, called DTW barycenter averaging (DBA). It also describes experiments on standard datasets from the UCR time series classification and clustering archive [19] in order to compare our method to existing averaging methods. Then, Section 5 looks deeper into the sufficient number of points to accurately represent of a set of sequences. Section 6 describes experiments conducted to demonstrate the applicability of DBA to clustering, by detailing experiments carried out with the K-MEANS algorithm on standard datasets as well as on an application domain, namely satellite image time series. Finally, Section 7 concludes the article and presents some further work.

2. Dynamic time warping (DTW)

In this section, we recall the definition of the euclidean distance and of the DTW similarity measure. Throughout this section, let $A=\langle a_1,\ldots,a_T\rangle$ and $B=\langle b_1,\ldots,b_T\rangle$ be two sequences, and let δ be a distance between elements (or *coordinates*) of sequences.

Euclidean distance: This distance is commonly accepted as the simplest distance between sequences. The distance between A and B is defined by

$$D(A,B)=\sqrt{\delta(a_1,b_1)^2+\cdots+\delta(a_T,b_T)^2}$$

Unfortunately, this distance does not correspond to the common understanding of what a sequence really is, and cannot capture flexible similarities. For example, $X=\langle a,b,a,a\rangle$ and $Y=\langle a,a,b,a\rangle$ are different according to this distance even though they represent similar trajectories.

Dynamic time warping: DTW is based on the Levenshtein distance (also called edit distance) and was introduced in [6,7], with applications in speech recognition. It finds the optimal alignment (or coupling) between two sequences of numerical values, and captures flexible similarities by aligning the coordinates inside both sequences. The cost of the optimal alignment can be recursively computed by

$$D(A_i,B_j)=\delta(a_i,b_j)+\min\begin{Bmatrix}D(A_{i-1},B_{j-1})\\D(A_i,B_{j-1})\\D(A_{i-1},B_j)\end{Bmatrix} \quad (1)$$

where A_i is the subsequence $\langle a_1,\ldots,a_i\rangle$. The overall similarity is given by $D(A_{|A|},B_{|B|})=D(A_T,B_T)$.

Unfortunately, a direct implementation of this recursive definition leads to an algorithm that has exponential cost in time. Fortunately, the fact that the overall problem exhibits overlapping subproblems allows for the memoization of partial results in a matrix, which makes the minimal-weight coupling computation a process that costs $|A|\cdot|B|$ basic operations. This measure has thus a time and a space complexity of $O(|A|\cdot|B|)$.

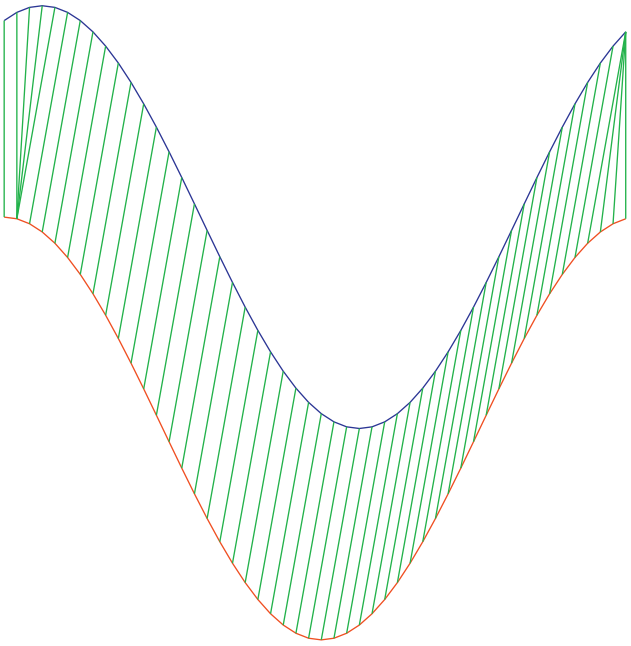


Fig. 1. Two 1D sequences aligned with dynamic time warping. Coordinates of the top and bottom sequences have been, respectively, computed by $\cos(t)$ and $\cos(t+\alpha)$. For visualization purpose, the top sequence is drawn vertically shifted.

DTW is able to find optimal global alignment between sequences and is probably the most commonly used measure to quantify the dissimilarity between sequences [9–13]. It also provides an overall real number that quantifies similarity. An example DTW-alignment of two sequences can be found in Fig. 1: it shows the alignment of points taken from two sinusoids, one being slightly shifted in time. The numerical result computed by DTW is the sum of the heights[1] of the associations. Alignments at both extremities on Fig. 1 show that DTW is able to correctly realign one sequence with the other, a process which, in this case, highlights similarities that Euclidean distance is unable to capture. Algorithm 1 details the computation.

Algorithm 1. DTW

Require: $A=\langle a_1,\ldots,a_S\rangle$
Require: $B=\langle b_1,\ldots,b_T\rangle$
Let δ be a distance between coordinates of sequences
Let $m[S,T]$ be the matrix of couples (*cost*,*path*)
  $m[1,1]\leftarrow(\delta(a_1,b_1),(0,0))$
  for $i\leftarrow 2$ to S **do**
    $m[i,1]\leftarrow(m[i-1,1,1]+\delta(a_i,b_1),(i-1,1))$
  end for
  for $j\leftarrow 2$ to T **do**
    $m[1,j]\leftarrow(m[1,j-1,1]+\delta(a_1,b_j),(1,j-1))$
  end for
  for $i\leftarrow 2$ to S **do**
    for $j\leftarrow 2$ to T **do**
      $minimum\leftarrow minVal(m[i-1,j],m[i,j-1],m[i-1,j-1])$
      $m[i,j]\leftarrow(first(minimum)+\delta(a_i,b_j),second(minimum))$
    end for
  end for
  return $m[S,T]$

[1] In fact, the distance $\delta(a_i,b_j)$ computed in Eq. (1) is the distance between two coordinates without considering the time distance between them.

Algorithm 2. first

Require: $p = (a, b)$: *couple*
  return a

Algorithm 3. second

Require: $p = (a, b)$: *couple*
  return b

Algorithm 4. minVal

Require: v_1, v_2, v_3: couple
  if $first(v_1) \leq min(first(v_2), first(v_3))$ **then**
    return v_1
  else if $first(v_2) \leq first(v_3)$ **then**
    return v_2
  else
    return v_3
  end if

3. Related work

In the context of classification, many algorithms require a method to represent in one object, informations from a set of objects. Algorithms like K-MEDOIDS are using the medoid of a set of objects. Some others like K-MEANS need to average a set of objects, by finding a mean of the set. If these "consensus" representations are easily definable for objects in the Euclidean space, this is much more difficult for sequences under DTW.

Finding a consensus representation of a set of sequences is even described by [20], chapter 14, as the Holy Grail. In this section, we first introduce the problem of finding a consensus sequence from a set of sequences, inspired by theories developed in computational biology. Then, we make the link between these theories on strings and methods currently used to average sequences of numbers under dynamic time warping.

To simplify explanations, we use the term *coordinate* to designate an element, or point, or component of a sequence. Without loss of generality, we consider that sequences contain T coordinates that are one-dimensional data. We note $A^{(\beta)}$ a sequence of length β. In the following, we consider a set $\mathbb{S} = \{\mathbb{S}_1, \cdots, \mathbb{S}_N\}$ of N sequences from which we want to compute a consensus sequence $\mathcal{C}$.

3.1. The consensus sequence problem

As we focus on DTW, we will only detail the consensus sequence problem from the edit distance side. (The problem for DTW is almost the same, and we will detail the differences in next subsection.) The term *consensus* is subjective, and depends on the needs. In the context of sequences, this term is used with three meanings: the longest common subsequence of a set, the medoid sequence of the set, or the average sequence of the set.

The longest common subsequence generally permits to visualize a summary of a set of sequences. It is, however, generally not used in algorithms because the resulting common subsequence does not cover the whole data.

The two other concepts refer to a more formal definition, corresponding to the sequence in the *center* of the set of sequences. We have to know what the *center* notion means. The commonly accepted definition is the object minimizing the sum of squared distances to objects of the set. When the center must be found in the dataset, the center is called "medoid sequence". Otherwise, when the search space of the center is not restricted, the most widely used term is "average sequence".

As our purpose is the definition of a center minimizing the sum of squared distances to sequences of a set, we focus on the definition of an average sequence when the corresponding distance is DTW.

Definition. Let E be the space of the coordinates of sequences. By a minor abuse of notation, we use E^T to designate the space of all sequences of length T. Given a set of sequences $\mathbb{S} = \{\mathbb{S}_1, \cdots, \mathbb{S}_N\}$, the average sequence $\mathcal{C}^{(T)}$ must fulfill

$$\forall X \in E^T, \quad \sum_{n=1}^{N} DTW^2(\mathcal{C}^{(T)}, \mathbb{S}_n) \leq \sum_{n=1}^{N} DTW^2(X, \mathbb{S}_n) \tag{2}$$

Since no information on the length of the average sequence is available, the search cannot be limited to sequences of a given length, so all possible values for T have to be considered. Note that sequences of $\mathbb{S}$ have a fixed length T. $\mathcal{C}$ has hence to fulfill

$$\forall t \in [1, +\infty[, \quad \left(\forall X \in E^t, \sum_{n=1}^{N} DTW^2(\mathcal{C}, \mathbb{S}_n) \leq \sum_{n=1}^{N} DTW^2(X, \mathbb{S}_n) \right) \tag{3}$$

This definition relies in fact on the *Steiner trees* theory; $\mathcal{C}$ is called *Steiner sequence* [20]. Note that the sums in Eqs. (2) and (3), is often called *Within Group Sum of Squares* (WGSS), or *discrepancy distance* in [21]. We will also use the simple term *inertia*, used in most works on clustering.

3.2. Exact solutions to the Steiner sequence problem

As shown in [22], when considered objects are simple points in an α- dimensional space, the minimization problem corresponding to Eq. (2) can be solved by using the property of the arithmetic mean. As the notion of arithmetic mean is not easily extendable to semi-pseudometric spaces (*i.e.*, spaces induced by semi-pseudometrics like DTW), we need to detail this *Steiner sequence problem*, *i.e.*, the problem to find an average sequence. To solve this problem, there are two close families of methods.

The first one consists in computing the global multiple alignment [23] of the N sequences of $\mathbb{S}$. This multiple alignment is computable by extending DTW for aligning N sequences. For instance, instead of computing DTW by comparing three values in a square, one have to compare seven values in a cube for three sequences. This idea can be generalized by computing DTW in a N-dimensional hypercube. Given this global alignment, $\mathcal{C}$ can be found by averaging *column by column* the multiple alignment. However, this method presents two major difficulties, that prevent its use. First, the multiple alignment process takes $\Theta(T^N)$ operations [24], and is not tractable for more than a few sequences. Second, the global length of the multiple alignment can be on the order of T^N, and requires unrealistic amounts of memory.

The second family of methods consists in searching through the solution space, keeping those minimizing the sum (Eq. (2)). In the discrete case (*i.e.*, sequences of characters or of symbolical values), as the alphabet is generally finite, this scan is easy. However, as the length of the global alignment is potentially $T^N - 2T$, scanning the space takes $\Theta(T^N)$ operations. In the continuous case (*i.e.*, sequences of numbers or of number vectors), scanning all solutions is impossible. Nevertheless, we will explain in the next paragraph how this search can be guided towards potential solutions, even though this strategy also exhibits exponential complexity.

In fact, we need a method to generate all potential solutions. Each solution corresponds to a coupling between $\mathcal{C}$ and each sequence of $\mathbb{S}$. As all coordinates of each sequence of $\mathbb{S}$ must be associated to at least one coordinate of $\mathcal{C}$, we have to generate all possible groupings between coordinates of each sequence of $\mathbb{S}$

and coordinates of $\mathcal{C}$. Each coordinate of $\mathcal{C}$ must be associated to a non-empty non-overlapping set of coordinates of sequences of $\mathbb{S}$. To generate all possible groupings, we can consider that each sequence is split in as many subsequences as there are coordinates in $\mathcal{C}$. Thus, the first coordinate of $\mathcal{C}$ will be associated to the coordinates appearing in the first subsequences of sequences of $\mathbb{S}$, the second coordinate to coordinates appearing in the second subsequences of sequences of $\mathbb{S}$, and so on. Then, for $\mathcal{C}^{(p)}$ ($\mathcal{C}$ of length p), the N sequences of $\mathbb{S}$ have to be cut into p parts. There are $\binom{T}{p}$ ways to cut a sequence of length T into p subsequences. Thus, for N sequences there are $\binom{T}{p}^N$ possibilities. The time complexity of this scan is therefore in $\Theta(\binom{T}{p}^N)$.

One can note in Eq. (3) that $p > T$ does not make sense. It would mean that sequences have to be split in more subsequences than they contain coordinates. Thus, we can limit the search conditions of $\mathcal{C}$ to $p \in [1,T]$. But even with this tighter bound, the overall computation remains intractable.

3.3. Approximating the exact solution

As we explained in the previous subsection, finding the average sequence is deeply linked to the multiple alignment problem. Unfortunately, 30 years of well-motivated research did not provide any exact scalable algorithm, neither for the multiple alignment problem, nor for the consensus sequence problem. Given a sequence standing for the solution, we even cannot check if the potential solution is optimal because we rely on a subset of the search space. A number of heuristics have been developed to solve this problem (see [25–29] for examples). We present in this subsection the most common family of methods used to approximate the average sequence: iterative pairwise averaging. We will also link this family to existing averaging method for DTW.

Iterative pairwise averaging consists in iteratively merging two average sequences of two subsets of sequences into a single average sequence of the union of those subsets. The simplest strategy computes an average of two sequences and iteratively incorporates one sequence to the average sequence. Differences between existing methods are the order in which the merges are done, and the way they compute an average of two sequences.

3.3.1. Ordering schemes

Tournament scheme. The simplest and most obvious averaging ordering consists in pairwise averaging sequences following a tournament scheme. That way, $N/2$ average sequences are created at first step. Then those $N/2$ sequences, in turn, are pairwise averaged into $N/4$ sequences, and so on, until one sequence is obtained. In this approach, the averaging method (between two sequences) is applied N times.

Ascendant hierarchical scheme. A second approach consists in averaging at first the two sequences whose distance (DTW) is minimal over all pairs of sequences. This works like Ascendant Hierarchical Clustering, computing a distance matrix before each average computation. In that way, the averaging method is also called $N-1$ times. In addition, one has to take into account the required time to compute N times the distance matrix.

3.3.2. Computing the average sequence

Regarding the way to compute an average from two sequences under DTW, most methods are using associations (coupling) computed with DTW.

One coordinate by association. Starting from a coupling between two sequences, the average sequence is built using the center of each association. Each coordinate of the average sequence will thus be the center of each association created by DTW. The main problem of this technique is that the resulting mean can grow substantially in length, because up to $|A|+|B|-2$ associations can be created between two sequences A and B.

One coordinate by connected component. Considering that the coupling (created by DTW) between two sequences forms a graph, the idea is to associate each connected component of this graph to a coordinate of the resulting mean, usually taken as the barycenter of this component. Contrary to previous methods, the length of resulting mean can decrease. The resulting length will be between 2 and $\min(|A|,|B|)$.

3.4. Existing algorithms

The different ordering schemes and average computations just described are combined in the DTW literature to make up algorithms. The two main averaging methods for DTW are presented below.

Nonlinear alignment and averaging filters: NLAAF was introduced in [30] and rediscovered in [15]. This method uses the *tournament scheme* and the *one coordinate by association* averaging method. Its main drawback lies in the growth of its resulting mean. As stated earlier, each use of the averaging method can almost double the length of the average sequence. The entire NLAAF process could produce, over all sequences, an average sequence of length $N \times T$. As classical datasets comprise thousands of sequences made up on the order of hundred coordinates, simply storing the resulting mean could be impossible. This length problem is moreover worsened by the complexity of DTW, that grows bi-linearly with lengths of sequences. That is why NLAAF is generally used in conjunction with a process reducing the length of the mean, leading to a loss of information and thus to an unsatisfactory approximation.

Prioritized shape averaging: PSA was introduced in [21] to resolve shortcomings of NLAAF. This method uses the *Ascendant hierarchical scheme* and the *one by connected component* averaging method. Although this *hierarchical* averaging method aims at preventing the error to propagate too much, the length of average sequences remains a problem. If one alignment (with DTW) between two sequences leads to two connected components (*i.e.*, associations are forming two *hand-held fans*), the overall resulting mean will be composed of only two coordinates. Obviously, such a sequence cannot represent a full set of potentially long sequences. This is why authors proposed to replicate each coordinate of the average sequence as many times as there were associations in the corresponding connected component. However, this repetition of coordinates causes the problem already observed with NLAAF, by potentially doubling the number of coordinates of each intermediate average sequence. To alleviate this problem, the authors suggest using a process in order to reduce the length of the resulting mean.

3.5. Motivation

We have seen that most of the works on averaging sets of sequences can be analysed along two dimensions: first, the way they consider the individual sequences when averaging, and second, the way they compute the elements of the resulting sequences. These two characteristics have proved useful to classify the existing averaging techniques. They are also useful angles under which new solutions can be elaborated.

Regarding the averaging of individual sequences, the main shortcoming of all existing methods is their use of pairwise averaging. When computing the mean of N sequences by pairwise averaging, the order in which sequences are taken influences the quality of the result, because neither NLAAF nor PSA are associative functions. Pairwise averaging strategies are intrinsically sensitive to the order, with no guarantee that a different order would lead to the same result. *Local* averaging strategies like PSA or NLAAF may let an

initial approximation error propagate throughout the averaging process. If the averaging process has to be repeated (*e.g.*, during K-MEANS iterations), the effects may dramatically alter the quality of the result. This is why a *global* approach is desirable, where sequences would be averaged all together, with no sensitivity to their order of consideration. The obvious analogy to a global method is the computation of the barycenter of a set of points in a Euclidean space. Section 4 follows this line of reasoning in order to introduce a global averaging strategy suitable for DTW, and provides empirical evidence of its superiority over existing techniques.

The second dimension along which averaging techniques can be classified is the way they select the elements of the mean. We have seen that a naive use of the DTW-computed associations may lead to some sort of "overfit", with an average covering almost every detail of every sequence, whereas simpler and smoother averages could well provide a better description of the set of sequences. Moreover, long and detailed averages have a strong impact on further processing. Here again, iterative algorithms like K-MEANS are especially at risk: every iteration may lead to a longer average, and because the complexity of DTW is directly related to the length of the sequences involved, later iterations will take longer than earlier ones. In such cases, unconstrained averaging will not only lead to an inadequate description of clusters, it will also cause a severe performance degradation. This negative effect of sequence averaging is well-known, and corrective actions have been proposed. Section 5 builds on our averaging strategy to suggest new ways of shortening the average.

4. A new averaging method for DTW

To solve the problems of existing pairwise averaging methods, we introduce a global averaging strategy called DTW *barycenter averaging* (DBA). This section first defines the new averaging method and details its complexity. Then DBA is compared to NLAAF and PSA on standard datasets [19]. Finally, the robustness and the convergence of DBA are studied.

4.1. Definition of DBA

DBA stands for DTW barycenter averaging. It consists in a heuristic strategy, designed as a global averaging method. DBA is an averaging method which consists in iteratively refining an initially (potentially arbitrary) average sequence, in order to minimize its squared distance (DTW) to averaged sequences.

Let us provide an intuition on the mechanism of DBA. The aim is to minimize the sum of squared DTW distances from the average sequence to the set of sequences. This sum is formed by single distances between each coordinate of the average sequence and coordinates of sequences associated to it. Thus, the contribution of one coordinate of the average sequence to the total sum of squared distance is actually a sum of euclidean distances between this coordinate and coordinates of sequences associated to it during the computation of DTW. Note that a coordinate of one of the sequences may contribute to the new position of several coordinates of the average. Conversely, any coordinate of the average is updated with contributions from one or more coordinates of each sequence. In addition, minimizing this partial sum for each coordinate of the average sequence is achieved by taking the barycenter of this set of coordinates. The principle of DBA is to compute each coordinate of the average sequence as the barycenter of its associated coordinates of the set of sequences. Thus, each coordinate will minimize its part of the total WGSS in order to minimize the total WGSS. The updated average sequence is defined once all barycenters are computed.

Technically, for each refinement *i.e.*, for each iteration, DBA works in two steps:

1. Computing DTW between each individual sequence and the temporary average sequence to be refined, in order to find associations between coordinates of the average sequence and coordinates of the set of sequences.
2. Updating each coordinate of the average sequence as the barycenter of coordinates associated to it during the first step.

Let $\mathbb{S}=\{\mathbb{S}_1,\cdots,\mathbb{S}_N\}$ be the set of sequences to be averaged, let $\mathcal{C}=\langle \mathcal{C}_1,\ldots,\mathcal{C}_T\rangle$ be the average sequence at iteration i and let $\mathcal{C}'=\langle \mathcal{C}_1',\ldots,\mathcal{C}_{T'}\rangle$ be the update of $\mathcal{C}$ at iteration $i+1$, of which we want to find coordinates. In addition, each coordinate of the average sequence is defined in an arbitrary vector space E (*e.g.*, usually a Euclidean space):

$$\forall t\in[1,T],\quad \mathcal{C}_t\in E \tag{4}$$

We consider the function *assoc*, that links each coordinate of the average sequence to one or more coordinates of the sequences of $\mathbb{S}$. This function is computed during DTW computation between $\mathcal{C}$ and each sequence of $\mathbb{S}$. The tth coordinate of the average sequence $\mathcal{C}_t'$ is then defined as

$$\mathcal{C}_t'=barycenter(assoc(\mathcal{C}_t)) \tag{5}$$

where

$$barycenter\{X_1,\ldots,X_\alpha\}=\frac{X_1+\cdots+X_\alpha}{\alpha} \tag{6}$$

(the addition of X_i is the vector addition). Algorithm 5 details the complete DBA computation.

Then, by computing again DTW between the average sequence and all sequences of $\mathbb{S}$, the associations created by DTW may change. As it is impossible to anticipate how these associations will change, we propose to make $\mathcal{C}$ iteratively converge. Fig. 2 shows four iterations (*i.e.*, four updates) of DBA on an example with two sequences.

Algorithm 5. DBA

Require: $\mathcal{C}=\langle \mathcal{C}_1,\ldots,\mathcal{C}_{T'}\rangle$ the initial average sequence
Require: $\mathbb{S}_1=\langle s_{1_1},\ldots,s_{1_T}\rangle$ the 1st sequence to average
⋮
Require: $\mathbb{S}_n=\langle s_{n_1},\ldots,s_{n_T}\rangle$ the nth sequence to average
Let T be the length of sequences
Let *assocTab* be a table of size T' containing in each cell a set of coordinates associated to each coordinate of $\mathcal{C}$
Let $m[T,T]$ be a temporary DTW (cost,path) matrix
  $assocTab\leftarrow[\emptyset,\ldots,\emptyset]$
  for *seq* in $\mathbb{S}$ **do**
    $m\leftarrow DTW(\mathcal{C},seq)$
    $i\leftarrow T'$
    $j\leftarrow T$
    while $i\geq 1$ and $j\geq 1$ **do**
      $assocTab[i]\leftarrow assocTab[i]\cup seq_j$
      $(i,j)\leftarrow second(m[i,j])$
    end while
  end for
  for $i=1$to T **do**
    $\mathcal{C}_i'=barycenter(assocTab[i])$ {see Eq. (6)}
  end for
  return $\mathcal{C}'$

As a summary, the proposed averaging method for dynamic time warping is a global approach that can average a set of sequences all together. The update of the average sequence

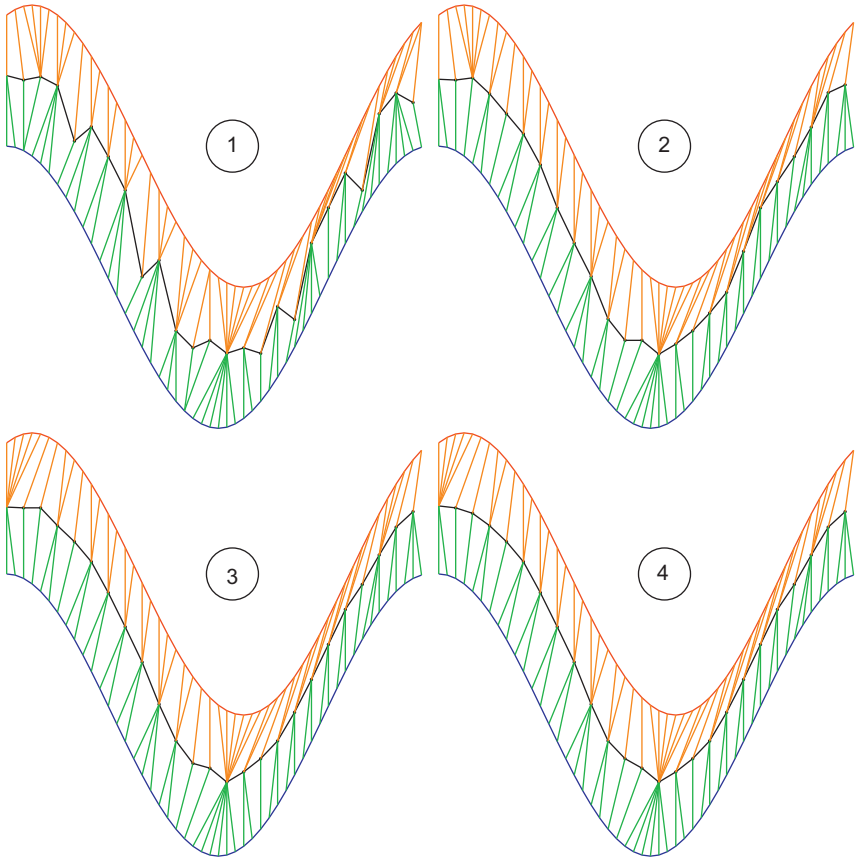


Fig. 2. DBA iteratively adjusting the average of two sequences.

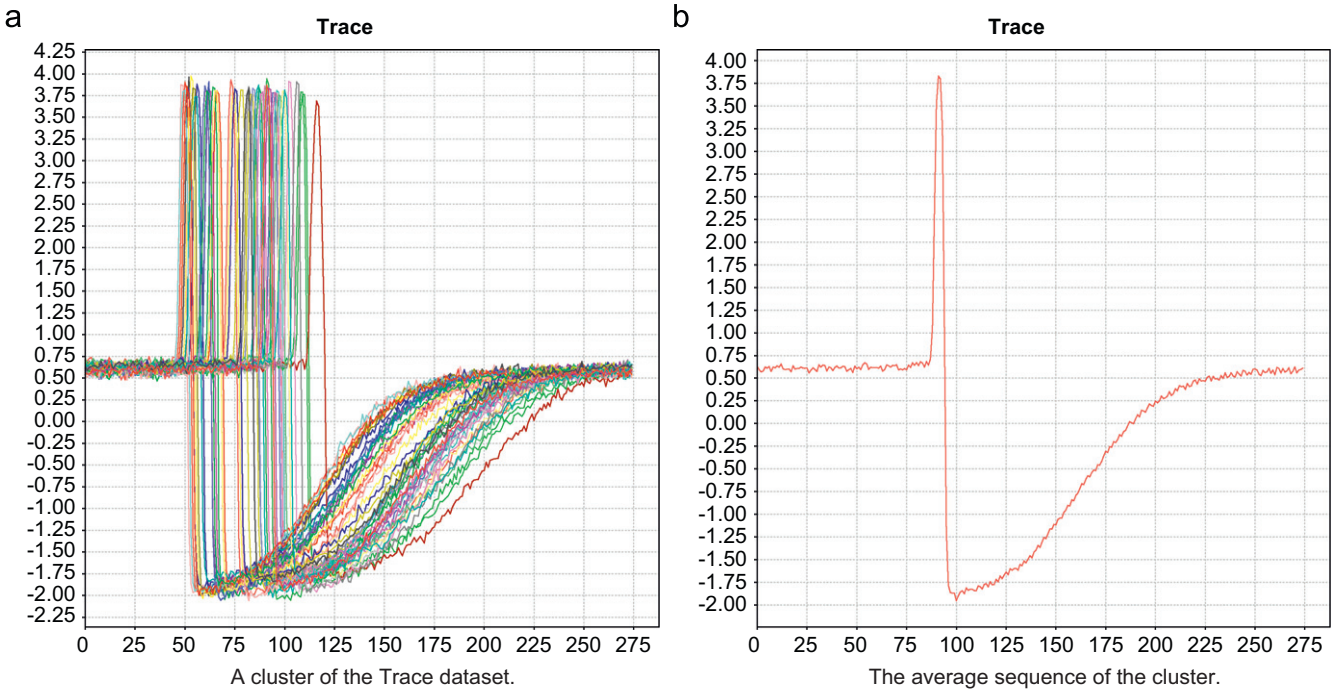


Fig. 3. An example of the DBA averaging method on one cluster from the "Trace" dataset from [19]. (a) A cluster of the *Trace* dataset, (b) The average sequence of the cluster.

between two iterations is independent of the order with which the individual sequences are used to compute their contribution to the update in question. Fig. 3 shows an example of an average sequence computed with DBA, on one dataset from [19]. This figure shows that DBA preserves the ability of DTW, identifying time shifts.

4.2. Initialization and convergence

The DBA algorithm starts with an initial averaging and refines it by minimizing its WGSS with respect to the sequences it averages. This section examines the effect of the initialisation and the rate of convergence.

Initialization: There are two major factors to consider when priming the iterative refinement:

- first the length of the starting average sequence,
- and second the values of its coordinates.

Regarding the length of the initial average, we have seen in Section 3.2 that its upper bound is T^N, but that such a length cannot reasonably be used. However, the inherent redundancy of the data lets one except that a much shorter sequence can be adequate. We will observe in Section 4.5 that a length of around T (the length of the sequences to average) performs well.

Regarding the values of the initial coordinates, it is theoretically impossible to determine the optimal values, otherwise the whole averaging process would become useless. In methods that require an initialisation, *e.g.*, K-MEANS clustering, a large number of heuristics have been developed. We will describe in Section 4.5 experiments with the most frequent techniques: first, a randomized choice, and second, using an element of the set of sequences to average. We will show empirically that the latter gives an efficient initialisation.

Convergence. As explained previously, DBA is an iterative process. It is necessary, once the average is computed, to update it several times. This has the property of letting DTW refine its associations. It is important to note that at each iteration, inertia can only decrease, since the new average sequence is closer (under DTW) to the elements it averages. If the update does not modify the alignment of the sequences, so the Huygens' theorem applies; barycenters composing the average sequence will get closer to coordinates of $\mathbb{S}$. In the other case, if the alignment is modified, it means that DTW calculates a better alignment with a smaller inertia (which decreases in that case also). We thus have a guarantee of convergence. Section 4.6 details some experiments in order to quantify this convergence.

4.3. Complexity study

This section details the time complexity of DBA. Each iteration of the iterative process is divided into two parts:

1. Computing DTW between each individual sequence and the temporary (*i.e.*, current) average sequence, to find associations between its coordinates and coordinates of the sequences.
2. Updating the mean according to the associations just computed.

Finding associations. The aim of Step 1 is to determine the set of associations between each coordinate of $\mathcal{C}$ and coordinates of sequences of $\mathbb{S}$. Therefore we have to compute DTW once per sequence to average, that is N times. The complexity of DTW is $\Theta(T^2)$. The complexity of Step 1 is therefore $\Theta(N \cdot T^2)$.

Updating the mean: After Step 1, each coordinate $\mathcal{C}_t$ of the average sequence has a set of coordinates $\{p_1, \ldots, p_{\alpha_t}\}$ associated to it. The process of updating $\mathcal{C}$ consists in updating each coordinate of the average sequence as the barycenter this set of coordinates. Since the average sequence is associated to N sequences, its T coordinates are, overall, associated to $N \cdot T$ coordinates, *i.e.*, all coordinates of sequences of $\mathbb{S}$. The update step will thus have a time complexity of $\Theta(N \cdot T)$.

Overall complexity: Because DBA is an iterative process, let us note I the number of iterations. The time complexity of the averaging process of N sequences, each one containing T coordinates, is thus

$$\Theta(DBA) = \Theta(I(N \cdot T^2 + N \cdot T)) = \Theta(I \cdot N \cdot T^2) \tag{7}$$

Comparison with PSA and NLAAF: To compute an average sequence from two sequences, PSA and NLAAF need to compute DTW between these two sequences, which has a time complexity of $\Theta(T^2)$. Then, to compute the temporary average sequence, PSA and NLAAF require $\Theta(T)$ operations. However, after having computed this average sequence, it has to be shorten to the length of averaged sequences. The classical averaging process used is uniform scaling which requires $\Theta(T+2T+3T+\cdots+T^2) = \Theta(T^3)$. The computation of the average sequence of two sequences requires $\Theta(T^3+T^2+T) = \Theta(T^3)$. The overall NLAAF averaging of a set of N sequences then requires

$$\Theta((N-1) \cdot (T^3+T^2+T)) = \Theta(N \cdot T^3) \tag{8}$$

Moreover, as PSA is using a hierarchical strategy to order sequences, it has at least to compute a dissimilarity matrix, which requires $\Theta(N^2 \cdot T^2)$ operations. The overall PSA averaging of a set of N sequences then requires

$$\Theta((N-1) \cdot (T^3+T^2+T)+N^2 \cdot T^2) = \Theta(N \cdot T^3+N^2 \cdot T^2) \tag{9}$$

As $I \ll T$, the time complexity of DBA is thus smaller than PSA and NLAAF ones.

4.4. Experiments on standard datasets

Evaluating an average sequence is not a trivial task. No ground truth of the expected sequence is available and we saw in Section 3 that many meanings are covered by the "average" (or consensus) notion. Most experimental and theoretical works use the WGSS to quantify the relative quality of an averaging technique. Thus, to assess the performance of DBA by comparison with existing averaging methods, we compare DBA to NLAAF and PSA in terms of WGSS over datasets from the UCR classification/clustering archive [19] (see Fig. 4).

Let us briefly remind what NLAAF and PSA are. NLAAF works by placing each coordinate of the average sequence of two sequences, as the center of each association created by DTW. PSA associates each connected component of the graph (formed by the coupling between two sequences) to a coordinate of the average sequence. Moreover, to average N sequences, it uses a hierarchical method to average at first closest sequences.

Experimental settings. To make these experiments reproducible, we provide here the details about our experimental settings:

- all programs are implemented in Java and run on a Core 2 Duo processor running at 2.4 GHz with 3 GB of RAM;
- the distance used between two coordinates of sequences is the squared Euclidean distance. As the square function is a strictly increasing function on positive numbers, and because we only use comparisons between distances, it is unnecessary to compute square roots. The same optimization has been used in [31], and seems rather common;
- sequences have been normalized with Z-score: for each sequence, the mean $\overline{x}$ and standard deviation σ of the coordinate values are computed, and each coordinate y_i is replaced by

$$y'_i = \frac{y_i - \overline{x}}{\sigma} \tag{10}$$

- we compare DBA to NLAAF on all datasets from [19] and to PSA using results given by [21];
- as we want to test the capacity of DBA to minimize the WGSS, and because we do not focus on supervised methods, we put all sequences from both *train* and *test* dataset together;
- for each set of sequences under consideration, we report the inertia under DBA and other averaging techniques. To provide quantitative evaluation, we indicate the ratio between the inertia with respect to the average computed by DBA and those computed by NLAAF and/or PSA (see the tables). To provide an overall evaluation, the text also sometimes mentions geometric averages of these ratios.

Intraclass inertia comparison: In this first set of experiments, we compute an average for each class in each dataset. Table 1 shows the global WGSS obtained for each dataset. We notice that, for all datasets, DBA reduces/improves the intraclass inertia. The geometric average of the ratios shown in Table 1 is 65%.

Table 2 shows a comparison between results of DBA and PSA. Here again, for all results published in [21], DBA outperforms PSA, with a geometric average of inertia ratios of 65%.

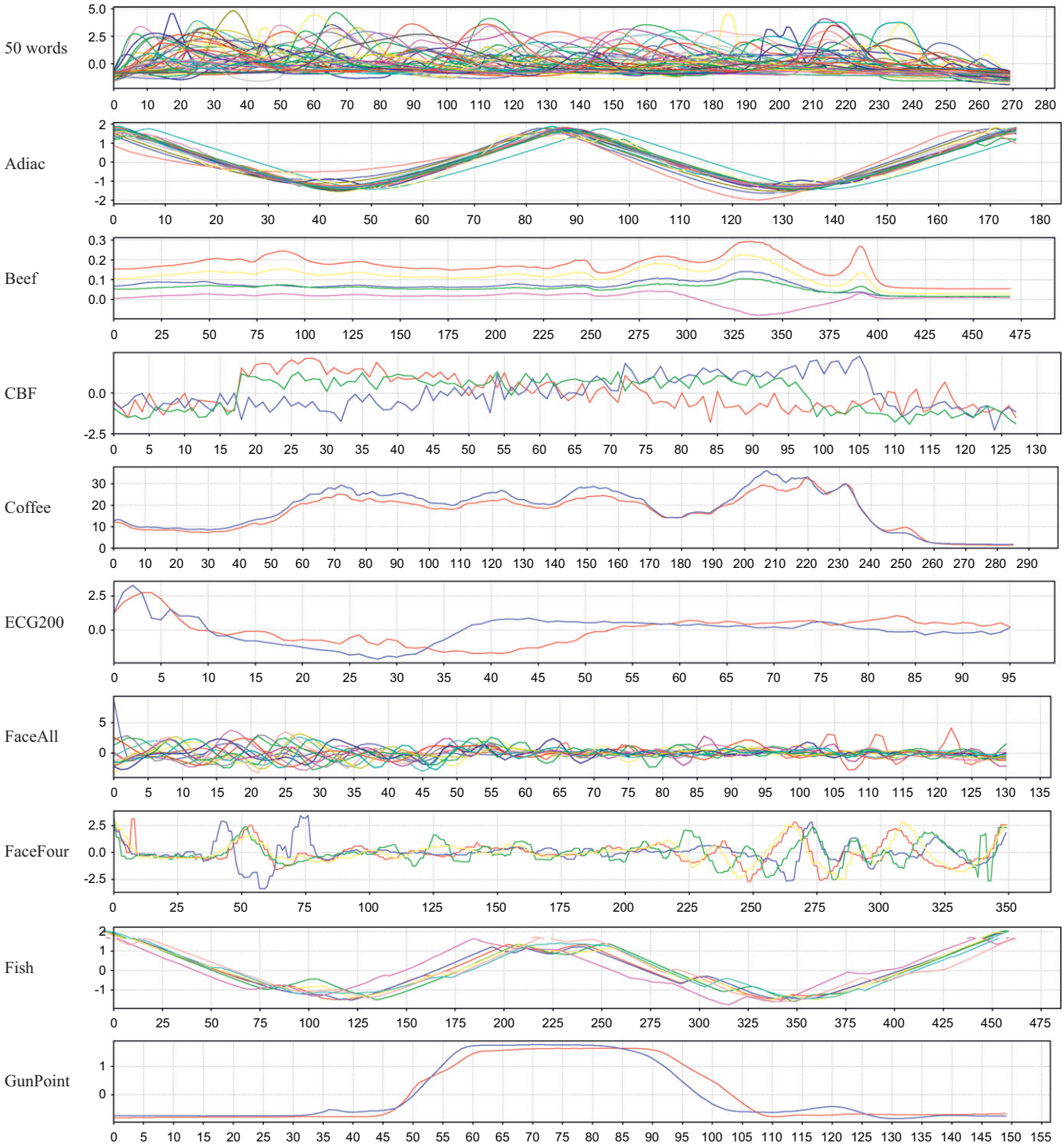


Fig. 4. Sample instances from the test datasets. One time series from each class is displayed for each dataset.

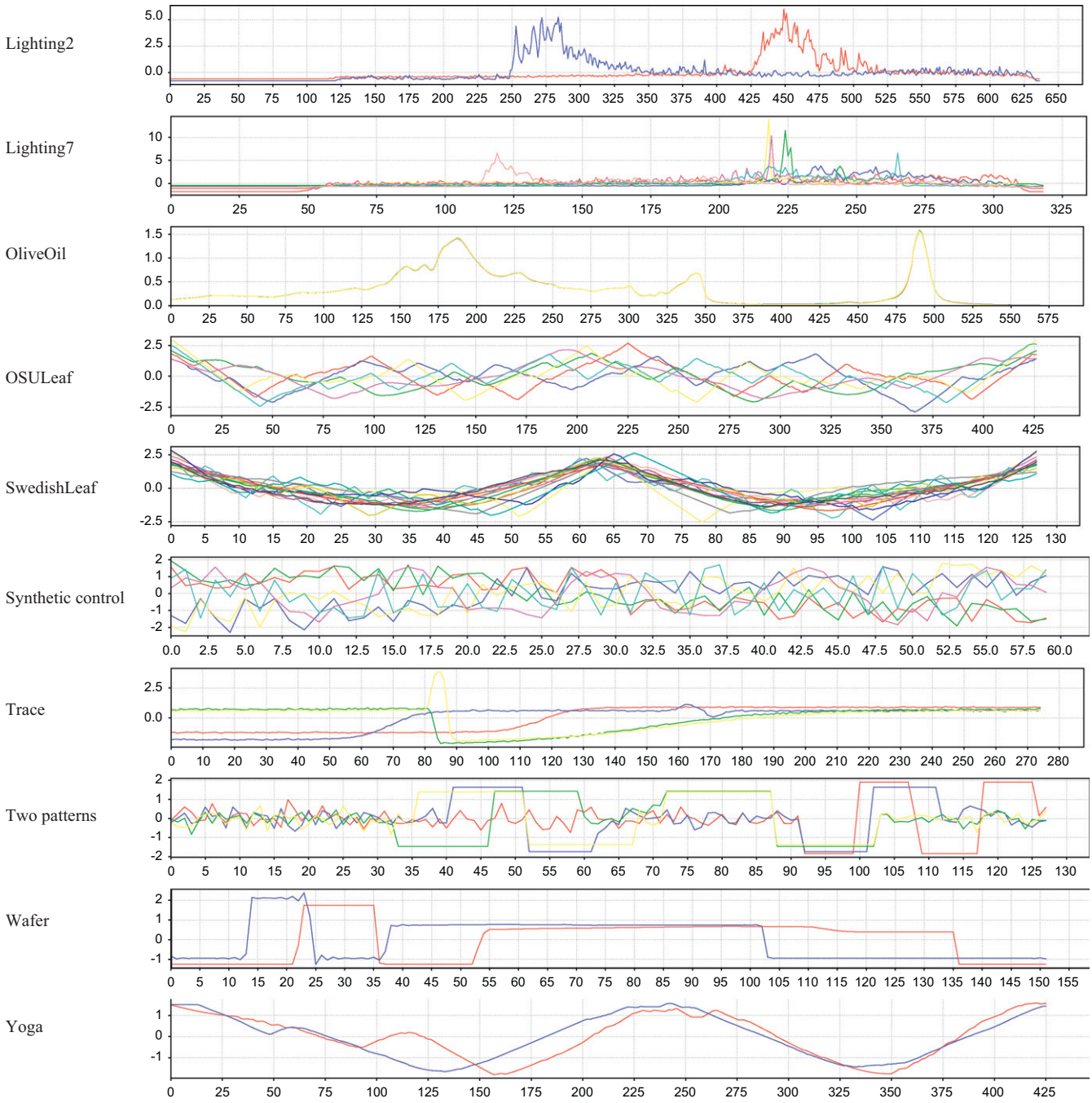


Fig. 4. (*Continued*)

Actually, such a decreases of inertia show that old averaging methods could not be seriously considered for machine learning use.

Global dataset inertia: In the previous paragraph, we computed an average for each class in each dataset. In this paragraph, the goal is to test robustness of DBA with more data variety. Therefore, we average all sequences of each dataset. This means that only one average sequence is computed for a whole dataset. Thats way, we compute the global dataset inertia under DTW with NLAAF and DBA, to compare their capacity to summarize mixed data.

As can be seen in Table 3, DBA systematically reduces/improves the global dataset inertia with a geometric average ratio of 56%. This means that DBA not only performs better than NLAAF (Table 1), but also more robust to diversity.

4.5. Impact of initialisation

DBA is deterministic once the initial average sequence C is chosen. It is thus important to study the impact of the choice of the initial mean on the results of DBA. When used with K-MEANS, this choice must be done at each iteration of the algorithm, for example by taking as the initialisation the average sequence C obtained at the previous iteration. However, DBA is not limited to this context.

We have seen in Section 4.2 that two aspects of initialisation have to be evaluated empirically: first, the length of the initial average sequence, and second the values of its coordinates. We have designed three experiments on some of the datasets

Table 1
Comparison of intraclass inertia under DTW between NLAAF and DBA.

Dataset	Intraclass inertia		
	NLAAF	DBA	$\frac{DBA}{NLAAF}$ (%)
50words	11.98	**6.21**	52
Adiac	0.21	**0.17**	81
Beef	29.90	**9.50**	32
CBF	14.25	**13.34**	94
Coffee	0.72	**0.55**	76
ECG200	11.34	**6.95**	61
FaceAll	17.77	**14.73**	83
FaceFour	34.46	**24.87**	72
Fish	1.35	**1.02**	75
GunPoint	7.24	**2.46**	34
Lighting2	194.07	**77.57**	40
Lighting7	48.25	**28.77**	60
OliveOil	0.018261	**0.018259**	100
OSULeaf	53.03	**22.69**	43
SwedishLeaf	2.50	**2.21**	88
Synthetic control	9.71	**9.28**	96
Trace	1.65	**0.92**	56
Two patterns	9.19	**8.66**	94
Wafer	54.66	**30.40**	56
Yoga	40.07	**37.27**	93

The best scores are shown in boldface.

Table 2
Comparison of intraclass inertia under DTW between PSA and DBA.

Dataset	Intraclass inertia		
	PSA	DBA	$\frac{DBA}{PSA}$ (%)
Beef	25.65	**9.50**	37
Coffee	0.72	**0.55**	76
ECG200	9.16	**6.95**	76
FaceFour	33.68	**24.87**	74
Synthetic control	10.97	**9.28**	85
Trace	1.66	**0.92**	56

The best scores are shown in boldface.

Table 3
Comparison of global dataset inertia under DTW between NLAAF and DBA.

Dataset	Global dataset inertia		
	NLAAF	DBA	$\frac{DBA}{PSA}$ (%)
50words	51 642	**26 688**	52
Adiac	647	**470**	73
Beef	3154	**979**	31
CBF	21 306	**18 698**	88
Coffee	61.60	**39.25**	64
ECG200	2190	**1466**	67
FaceAll	72 356	**63 800**	88
FaceFour	6569	**3838**	58
Fish	658	**468**	71
GunPoint	1525	**600**	39
Lighting2	25 708	**9673**	38
Lighting7	14 388	**7379**	51
OliveOil	2.24	**1.83**	82
OSULeaf	30 293	**12 936**	43
SwedishLeaf	5590	**4571**	82
Synthetic control	17 939	**13 613**	76
Trace	22 613	**4521**	20
Two patterns	122 471	**100 084**	82
Wafer	416 376	**258 020**	62
Yoga	136 547	**39 712**	29

The best scores are shown in boldface.

from [19]. Because these experiments require heavy computations, we have not repeated the computation on all datasets.

1. The first experiment starts with randomly generated sequences, of lengths varying from 1 to 2T, where T is the length of the sequences in the data set. Once the length is chosen, the coordinates are generated randomly with a normal distribution of mean zero and variance one. The curves on Fig. 5 show the variation of inertia with the length.
2. Because the previous experiment shows that the optimal inertia is attained with an initial sequence of length in the order of T, we have repeated the computation 100 times with different, randomly generated initial sequences of length T: the goal of this experiment is to measure how stable this heuristic is. Green triangles on Fig. 5 show the inertias with the different random initialisations of length T.
3. Because priming DBA with a sequence of length T seems to be an adequate choice, we have tried to replace the randomly generated sequence with one drawn (randomly) from the dataset. We have repeated this experiment 100 times. Red triangles on Fig. 5 show the inertias with the different initial sequences from the dataset.

Our experiments on the choice of the initial average sequence lead to two main conclusions. First, an initial average of length T (the length of the data sequences) is the most appropriate. It almost always leads to the minimal inertia. Second, randomly choosing an element of the dataset leads to the least inertia on almost all cases. Using some data to prime an iterative algorithm is part of the folklore. DBA is another case where it performs well. We have used this strategy in all our experiments with DBA.

Moreover, in order to compare the impact of the initialisation on DBA and on NLAAF, we perform 100 computations of the average on few datasets (because of computing times), by choosing a random sequence from the dataset as an initialisation of DBA. As NLAAF is sensitive to the initialisation too, the same method is followed, in order to compare results. Fig. 6 presents the mean and standard deviation of the final inertia. The results of DBA are not only better than NLAAF (shown on the left part of Fig. 6), but the best inertia obtained by NLAAF is even worse as the worst inertia obtained by DBA (see the table on the right of Fig. 6).

4.6. Convergence of the iterative process

As explained previously, DBA is an iterative process. It is necessary, once the average is computed, to update it several times. This has the property of letting DTW refine its associations. Fig. 7(a) presents the average convergence of the iterative process on the *50words* dataset. This dataset is diverse enough (it contains 50 different classes) to test the robustness of the convergence of DBA.

Besides the overall shape of the convergence curve in Fig. 7(a), it is important to note that in some cases, the convergence can be *uneven* (see Fig. 7(b) for an example). Even if this case is somewhat unusual, one has to keep in mind that DTW makes nonlinear distortions, which cannot be predicted. Consequently, the convergence of DBA, based on alignments, cannot be always smooth.

5. Optimization of the mean by length shortening

We mentioned in Section 3.4 that algorithms such as NLAAF need to reduce the length of a sequence. Actually, this problem is more global and concerns the scaling problem of a sequence under time warping. Many applications working with subsequences or

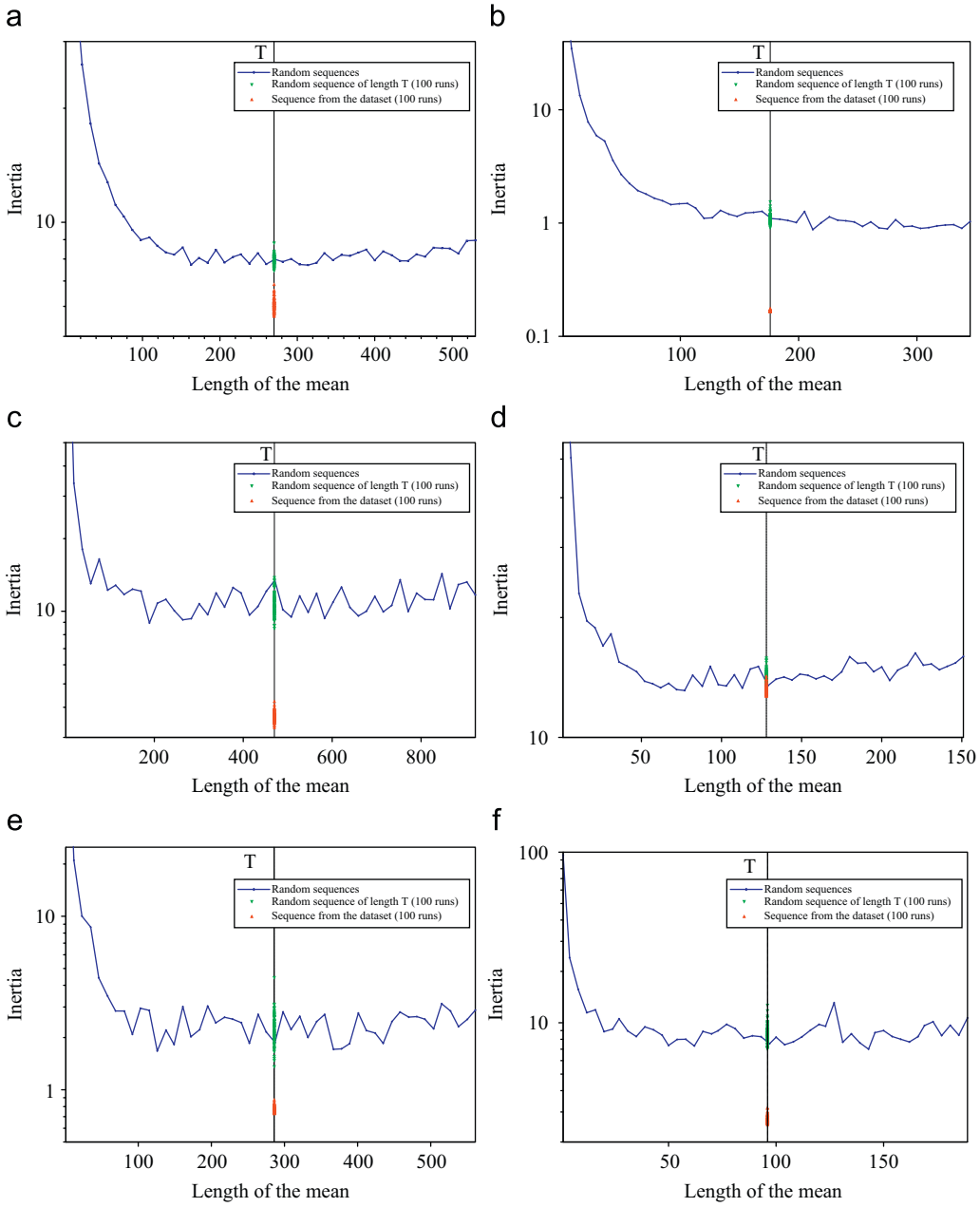


Fig. 5. Impact of different initialisation strategies on DBA. Note that the Inertia is displayed with logarithmic scale. (a) 50words, (b) Adiac, (c) Beef, (d) CBF, (e) Coffee, (f) ECG200.

even with different resolutions[2] require a method to uniformly scale a sequence to a fixed length. This kind of methods is generally called uniform scaling; further details about its inner working can be found in [31].

Unfortunately, the use of uniform scaling is not always coherent in the context of DTW, which computes non-uniform distortions. To avoid the use of uniform scaling with DTW, as done in [31,15,21], we propose here a new approach specifically designed for DTW. It is called adaptive scaling, and aims at reducing the length of a sequence with respect to one or more other sequences. In this section, we first recall the definition of uniform scaling, then we detail the proposed approach and finally its complexity is studied and discussed.

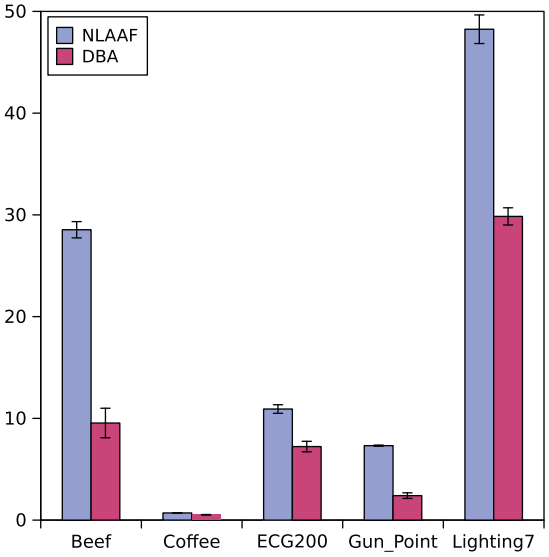
[2] The resolution in this case is the number of samples used to describe a phenomenon. For instance, a music can be sampled with different bit rates.

5.1. Uniform scaling

Uniform scaling is a process that reduces the length of a sequence with regard to another sequence. Let A and B be two sequences. Uniform scaling finds the prefix A_{sub} of A such that, scaled up to B, $DTW(A_{sub},B)$ is minimal. The subsequence A_{sub} is defined by

$$A_{sub} = \underset{i \in [1,T]}{\operatorname{argmin}}\{DTW(A_{1,i},B)\} \tag{11}$$

uniform scaling has two main drawbacks: one is directly linked to the method itself, and one is linked to its use with DTW. First, while uniform scaling considers a prefix of the sequence (*i.e.*, a subsequence), the representativeness of the resulting mean using such a reduction process can be discussed. Second, uniform scaling is a uniform reduction process, whereas DTW makes non-uniform alignments.



Dataset	NLAAF Best score	DBA Worst score
Beef	26.7	13.2
Coffee	0.69	0.66
ECG200	9.98	8.9
Gun_Point	7.2	3.1
Lighting7	45	32

Fig. 6. Effect of initialisation on NLAAF and DBA.

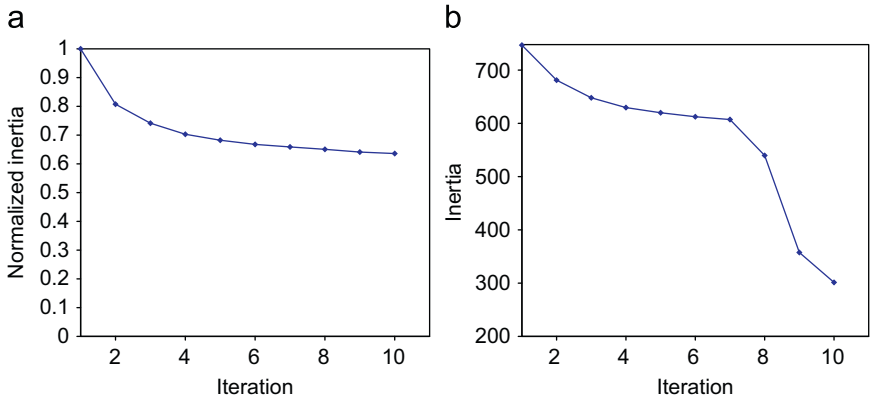


Fig. 7. Convergence of the iterative process for the 50words dataset. (a) Average convergence of the iterative process over 50 clusters, (b) example of an uneven convergence (cluster 4).

5.2. *Adaptive scaling*

We propose to make the scaling adaptive. The idea of the proposed adaptive scaling process is to answer to the following question: "How can one remove a point from a sequence, such as the distance to another sequence does not increase much?" To answer this question, adaptive scaling works by merging the two closest successive coordinates.

To explain how adaptive scaling works, let us start with a simple example. If two consecutive coordinates are identical, they can be merged. DTW is able to stretch the resulting coordinate and so recover the *original* sequence. This fusion process is illustrated in Fig. 8. Note that in this example, DTW gives the same score in Fig. 8(a) as in Fig. 8(b).

This article focuses on finding an average sequence consistent with DTW. Performances of DBA have been demonstrated on an average sequence of length arbitrarily fixed to T. In this context, the question is to know if this average sequence can be shortened, without making a less representative mean (*i.e.*, without increasing inertia). We show in the first example, that the constraint on inertia is respected. Even if uniform scaling could be used to reduce the length of the mean, an adaptive scaling would give

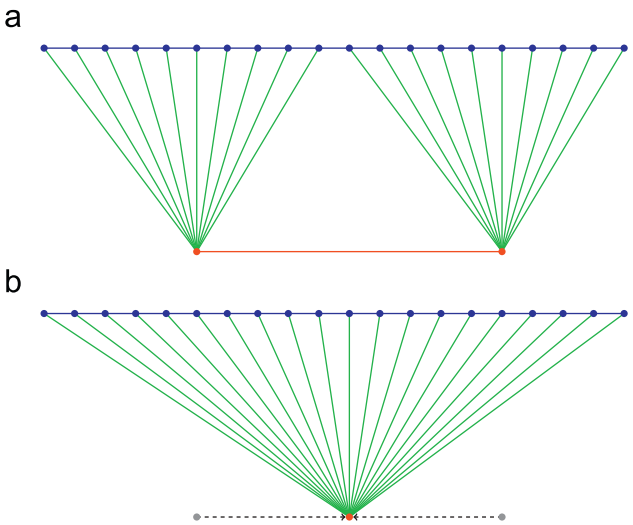


Fig. 8. Illustration of the length reduction process. (a) Alignment of two sequences: sequence below is composed of two same coordinates, (b) alignment of two sequences: sequence below is composed of only one coordinate.

better results, because DTW is able to make deformations on the time axis. Adaptive scaling is described in Algorithm 6.

Algorithm 6. Adaptive Scaling

Require: $A = \langle A_1, \ldots, A_T \rangle$
  while Need to reduce the length of A
    $(i,j) \leftarrow$ successive coordinates with minimal distance
    merge A_i with A_j
  end while
  return A

Let us now explain how the inertia can also decrease by using adaptive scaling. Fig. 9 illustrates the example used below. Imagine now that the next to last coordinate C_{T-1} of the average sequence is perfectly aligned with the last α coordinates of the N sequences of $\mathbb{S}$. In this case, the last coordinate C_T of the average sequence must still be, at least, linked to all last coordinates of the N sequences of $\mathbb{S}$. Therefore, as the next to last coordinate was (in this example) *perfectly aligned*, aligning the last coordinate will increase the total inertia. This is why adaptive scaling is not only able to shorten the average sequence, but also to reduce the inertia. Moreover, by checking the evolution of inertia after each merging, we can control this length reduction process, and so guarantee the improvement of inertia. Thus, given the resulting mean of DBA, coordinates of the average sequence can be successively merged as long as inertia decreases.

5.3. Experiments

Table 4 gives scores obtained by using adaptive scaling after the DBA process on various datasets. It shows that adaptive scaling alone always reduces the intraclass inertia, with a geometric average of 84%. Furthermore, the resulting average sequences are much shorter, by almost two thirds. This is an interesting idea of the minimum necessary length for representing a tim behaviour.

In order to demonstrate that adaptive scaling is not only designed for DBA, Fig. 10 shows its performances as a reducing process in NLAAF. Adaptive scaling is here used to reduce the length of a temporary pairwise average sequence (see Section 3.4). Fig. 10 shows that adaptive scaling used in NLAAF leads to scores similar to the ones achieved by PSA.

5.4. Complexity

Adaptive scaling (AS) consists in merging the two closest successive coordinates in the sequence. If we know ahead of time the number K of coordinates that must be merged, for example using AS for NLAAF to be scalable, it requires a time complexity of $\Theta(K \cdot T)$, and is thus tractable.

One could, however, need to guarantee that adaptive scaling does not merge too many coordinates. That is why we suggest to control the dataset inertia, by computing it and by stopping the adaptive scaling process if inertia increases. Unfortunately, computing the dataset inertia under DTW takes $\Theta(N \cdot T^2)$. Its complexity may prevent its use in some cases.

Nevertheless, we give here some interesting cases, where the use of adaptive scaling could be beneficial, because having shorter sequences means spending less time in computing DTW. In databases, the construction of indexes is an active research domain. The aim of indexes is to represent data in a better way while being fast queryable. Using adaptive scaling could be used here, because it correctly represents data and reduces the DTW complexity for further queries. The construction time of the index is here negligible compared to the millions of potential queries.

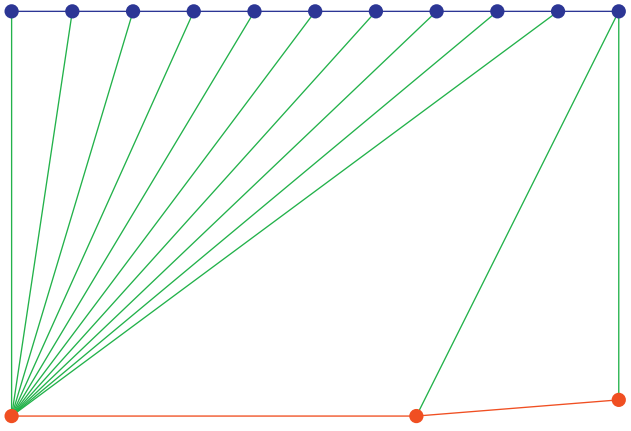


Fig. 9. Average sequence is drawn at the bottom and one sequence of the set is drawn at the top.

Table 4
Inertia comparison of intraclass inertias and lengths of resulting means with or without using the adaptive scaling (AS) process.

Dataset	Intraclass inertia		Length of the mean	
	DBA	DBA+AS	DBA	DBA+AS
50words	6.21	**6.09**	270	151
Adiac	0.17	**0.16**	176	162
CBF	13.34	**12.11**	128	57
FaceAll	14.73	**14.04**	131	95
Fish	1.02	**0.98**	463	365
GunPoint	2.46	**2.0**	150	48
Lighting2	77.57	**72.45**	637	188
Lighting7	28.77	**26.97**	319	137
OliveOil	0.018259	**0.01818**	570	534
OSULeaf	22.69	**21.96**	427	210
SwedishLeaf	2.21	**2.07**	128	95
Two patterns	8.66	**6.99**	128	59
Wafer	30.40	**17.56**	152	24
Yoga	37.27	**11.57**	426	195
Beef	9.50	**9.05**	470	242
Coffee	0.55	**0.525**	286	201
ECG200	6.95	**6.45**	96	48
FaceFour	24.87	**21.38**	350	201
Synthetic control	9.28	**8.15**	60	33
Trace	0.92	**0.66**	275	108

The best scores are shown in boldface.

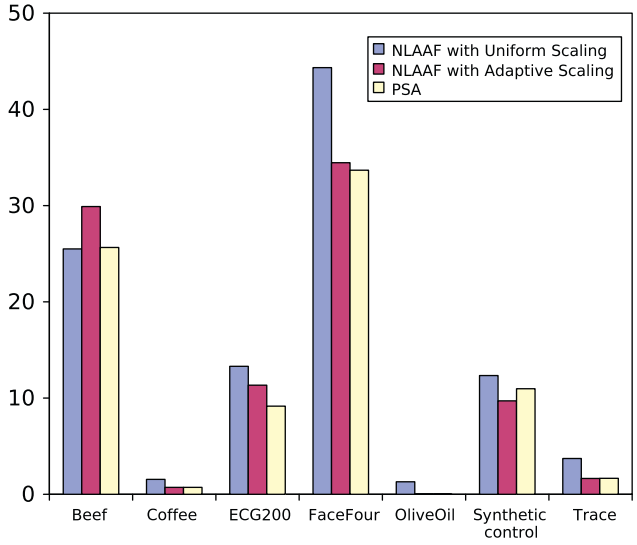


Fig. 10. Adaptive scaling: benefits of the reduction process for NLAAF.

Another example is (supervised or unsupervised) learning, where the learning time is often negligible compared to the time spent in classification. adaptive scaling could be very useful in such contexts, and can be seen in this case as an optimization process, more than an alternative to uniform scaling.

6. Application to clustering

Even if DBA can be used in association to DTW out of the context of K-means (and more generally out of the context of clustering), it is interesting to test the behaviour of DBA with K-means because this algorithm makes a heavy use of averaging. Most clustering techniques with DTW use the K-medoids algorithm, which does not require any computation of an average [15–18]. However, K-medoids has some problems related to its use of the notion of median: K-medoids is not idempotent, which means that its results can oscillate. Whereas DTW is one of the most used similarity on time series, it was not possible to use it reliably with well-known clustering algorithms.

To estimate the capacity of DBA to summarize clusters, we have tested its use with the K-means algorithm. We present here different tests on standard datasets and on a satellite image time series. Here again, result of DBA are compared to those obtained with NLAAF.

6.1. On UCR datasets

Table 5 shows, for each dataset, the global WGSS resulting from a K-means clustering. Since K-means requires initial centers, we place randomly as many centers as there are classes in each dataset. As shown in the table, DBA outperforms NLAAF in all cases except for *Fish* and *OliveOil* datasets. Including these exceptions, the inertia is reduced with a geometric average of 72%.

Table 5
Comparison of intracluster inertia under DTW between NLAAF and DBA.

Dataset	Intracluster inertia		
	NLAAF	DBA	$\frac{DBA}{NLAAF}$ (%)
50words	5920	**3503**	59
Adiac	86	**84**	98
Beef	393	**274**	70
CBF	12 450	**11 178**	90
Coffee	39.7	**31.5**	79
ECG200	1429	**950**	66
FaceAll	34 780	**29 148**	84
FaceFour	3155	**2822**	89
Fish	**221**	324	147
GunPoint	408	**180**	44
Lighting2	16 333	**6519**	40
Lighting7	6530	**3679**	56
OliveOil	**0.55**	0.80	146
OSULeaf	13 591	**7213**	53
SwedishLeaf	2300	**1996**	87
Synthetic control	5993	**5686**	95
Trace	387	**203**	52
Two patterns	45 557	**40 588**	89
Wafer	157 507	**108 336**	69
Yoga	73 944	**24 670**	33

The best scores are shown in boldface.

Let us try to explain the seemingly negative results that appear in Table 5. First, on *OliveOil*, the inertia over the whole dataset is very low (*i.e.*, all sequences are almost identical; see Fig. 4), which makes it difficult to obtain meaningful results. The other particular dataset is *Fish*. We have seen, in Section 4.4, that DBA outperforms NLAAF provided it has meaningful clusters to start with. However, here, the K-means algorithm tries to minimize this inertia in grouping elements in "centroid form". Thus, if clusters to identify are not organized around "centroids", this algorithm may converge to any local minima. In this case, we explain this exceptional behaviour on *Fish* as due to non-centroid clusters. We have shown in Section 4.4 that, if sequences are averaged per class, DBA outperforms all results, even those of these two datasets. This means that these two seemingly negative results are linked to the K-means algorithm itself, that converges to a less optimal local minimum even though the averaging method is better.

6.2. On satellite image time series

We have applied the K-means algorithm with DTW and DBA in the domain of satellite image time series analysis. In this domain, each dataset (*i.e.*, sequence of images) provides thousands of relatively short sequences. This kind of data is the opposite of sequences commonly used to validate time sequences analysis. Thus, in addition to evaluate our approach on small datasets of long sequences, we test our method on large datasets of short sequences.

Our data are sequences of numerical values, corresponding to radiometric values of pixels from a satellite image time series (SITS). For every pixel, identified by its coordinates (x,y), and for a sequence of images $\langle I_1,\ldots,I_n\rangle$, we define a sequence as $\langle I_1(x,y),\ldots,I_n(x,y)\rangle$. That means that a sequence is identified by coordinates x and y of a pixel (not used in measuring similarity), and that the values of its coordinates are the vectors of radiometric values of this pixel in each image. Each dataset contains as many sequences as there are pixels in one image.

We have tested DBA on one SITS of size 450×450 pixels, and of length 35 (corresponding to images sensed between 1990 and 2006). The whole experiment thus deals with 202,500 sequences of length 35 each, and each coordinate is made of three radiometric values. This SITS is provided by the Kalideos database [32] (see Fig. 11 for an extract).

We have applied the K-means algorithm on this dataset, with a number of clusters of 10 or 20, chosen arbitrarily. Then we computed the sum of intraclass inertia after five or 10 iterations. Table 6 summarizes results obtained with different parameters.

We can note that scores (to be minimized) are always ordered as $NLAAF > DBA >$ DBA+AS, which tends to confirm the behaviour of DBA and adaptive scaling. As we could expect, adaptive scaling

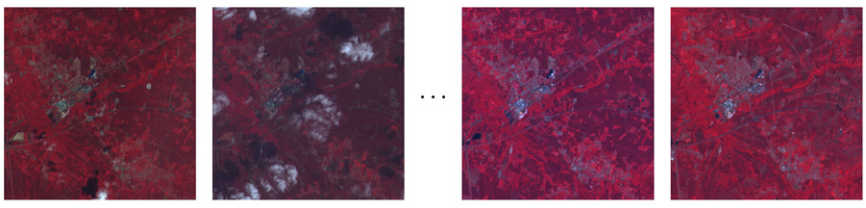


Fig. 11. Extract of the satellite image time series of Kalideos used. © CNES 2009—distribution spot image.

Table 6
Comparison of intracluster inertia of K-means with different parameterizations.

Nb seeds	Nb iterations	Inertia		
		NLAAF	DBA	DBA and AS
10	5	2.82×10^7	2.73×10^7	2.59×10^7
20	5	2.58×10^7	2.52×10^7	2.38×10^7
10	10	2.79×10^7	2.72×10^7	2.58×10^7
20	10	2.57×10^7	2.51×10^7	2.37×10^7

Distance used is DTW while averaging methods are NLAAF, DBA and DBA followed by adaptive scaling (AS).

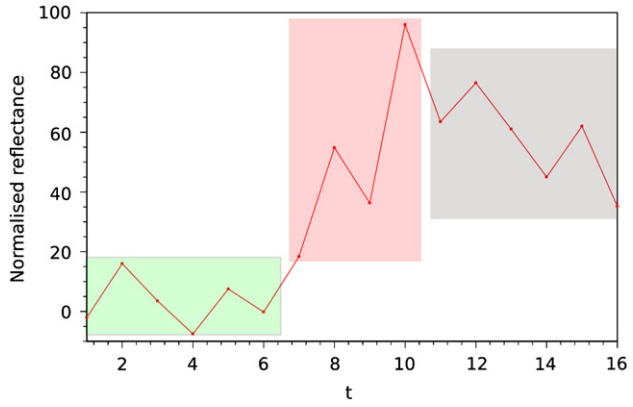


Fig. 12. The average of one of the clusters produced by K-means on the satellite image time series. This sequence corresponds to the thematical behaviour of the urban growth (construction of buildings, roads, etc.). The three rectangles corresponds to three phases: vegetation or bare soil, followed by new roofs and roads, followed by damaged and dusty roofs and roads.

permits to significantly reduce the score of DBA. We can see that even if the improvement seems less satisfactory, than those obtained on synthetic data, it remains, however, better than NLAAF.

Let us try to explain why the results of DBA are close to those of NLAAF. One can consider that when clusters are close to each other, then the improvement is reduced. The most likely explanation is that, by using so short sequences, DTW has not much material to work on, and that the alignment it finds early have little chance to change over the successive iterations. In fact, the shorter the sequences are, the closer DTW is from the Euclidean distance. Moreover, NLAAF makes less errors when there is no re-alignment between sequences. Thus, when sequences are small, NLAAF makes less errors. For that reason, even if DBA is in this case again better than NLAAF, the improvement is smaller.

Let us now explain why adaptive scaling is so useful here. In SITS, there are several evolutions which can be considered as random perturbations. Thus, the mean may not need to represent these perturbations, and we think that shorter means are sometimes better, because they can represent a perturbed constant subsequence by a single coordinate. Actually, this is often the case in SITS. As an example, a river can stay almost the same over a SITS and one or two coordinates can be sufficient to represent the evolution of such an area.

From a thematic point of view, having an average for each cluster of radiometric evolution sequences highlights and describes typical ground evolutions. For instance, the experiment just described provides a cluster representing a typical urban growth behaviour (appearance of new buildings and urban densification). This is illustrated on Fig. 12. Combining DTW with DBA has led to the extraction of urbanizing zones, but has also provided a symbolic description of this particular behaviour. Using euclidean distance instead of DTW, or any other averaging technique instead of DBA, has led to inferior results. Euclidean distance was expected to fail somehow, because urbanization has gone faster in some zones than in others, and because the data sampling is nonuniform. The other averaging methods have also failed to produce meaningful results, probably because of the intrinsic difficulty of the data (various sensors, irregular sampling, etc.), which leads to difficultly separable objects. In such cases, less precise averaging tends to blur cluster boundaries.

7. Conclusion

The DTW similarity measure is probably the most used and useful tool to analyse sets of sequences. Unfortunately, its applicability to data analysis was reduced because it had no suitable associated averaging technique. Several attempts have been made to fill the gap. This article proposes a way to classify these averaging methods. This "interpretive lens" permits to understand where existing techniques could be improved. In light of this contextualization, this article defines a global averaging method, called Dtw barycenter averaging (DBA). We have shown that DBA achieves better results on all tested datasets, and that its behaviour is robust.

The length of the average sequence is not trivial to choose. It has to be as short as possible, but also sufficiently long to represent the data it covers. This article also introduces a shortening technique of the length of a sequence called adaptive scaling. This process is shown to shorten the average sequence in adequacy to DTW and to the data, but also to improve its representativity.

Having a sound averaging algorithm lets us apply clustering techniques to time series data. Our results show again a significant improvement in cluster inertia compared to other techniques, which certainly increases the usefulness of clustering techniques.

Many application domains now provide time-based data and need data mining techniques to handle large volumes of such data. DTW provides a good similarity measure for time series, and DBA complements it with an averaging method. Taken together, they constitute a useful foundation to develop new data mining systems for time series. For instance, satellite imagery has started to produce satellite image time series, containing millions of sequences of multi-dimensional radiometric data. We have briefly described preliminary experiments in this domain.

We believe this work opens up a number of research directions. First, because it is, as far as we know, the first global approach to the problem of averaging a set of sequences, it raises interesting questions on the topology of the space of sequences, and on how the mean relates to the individual sequences.

Regarding Dtw barycenter averaging proper, there are still a few aspects to be studied. One aspect could be the choice of the initial sequence where sequences to be averaged do not have the same length. Also we have provided an empirical analysis of the rate of convergence of the averaging process. More theoretical or empirical work is needed to derive a more robust strategy, able to adjust the number of iterations to perform. An orientation of this work could be the study of the distribution of coordinates contributing to a coordinate of the average sequence.

Adaptive scaling has important implications on performance and relevance. Because of its adaptive nature, it ensures that average sequences have "the right level of detail" on appropriate sequence segments. It currently considers only the coordinates of the average sequence. Incorporating averaged sequences may lead to a more precise scaling, but would require more computation time. Finding the right balance between cost and precision requires further investigation.

When combining DBA with adaptive scaling, *e.g.*, when building reduced average sequences, we have often noticed that

they provide short summaries, that are at the same time easy to visualize *and* truly representative of the underlying phenomenon. For instance, the process of length reduction builds an average sequence around the major states of the data. It thus provides a sampling of the dataset built from the data themselves. Exploiting and extending this property is a promising research direction.

References

[1] A.K. Jain, M.N. Murty, P.J. Flynn, Data clustering: a review, ACM Computing Surveys 31 (3) (1999) 264–323.

[2] A. Rauber, E. Pampalk, J. Paralic, Empirical evaluation of clustering algorithms, Journal of Information and Organizational Sciences (JIOS) 24 (2) (2000) 195–209.

[3] P. Berkhin, Survey of clustering data mining techniques, Technical Report, Accrue Software, San Jose, CA, 2002.

[4] R. Xu, D. Wunsch, Survey of clustering algorithms, IEEE Transactions on Neural Networks 16 (3) (2005) 645–678.

[5] T.W. Liao, Clustering of time series data—a survey, Pattern Recognition 38 (11) (2005) 1857–1874.

[6] H. Sakoe, S. Chiba, A dynamic programming approach to continuous speech recognition, Proceedings of the Seventh International Congress on Acoustics, vol. 3, 1971, pp. 65–69.

[7] H. Sakoe, S. Chiba, Dynamic programming algorithm optimization for spoken word recognition, IEEE Transactions on Acoustics, Speech and Signal Processing 26 (1) (1978) 43–49.

[8] A.P. Shanker, A. Rajagopalan, Off-line signature verification using DTW, Pattern Recognition Letters 28 (12) (2007) 1407–1414.

[9] D. Sankoff, J. Kruskal, The symmetric time-warping problem: from continuous to discrete, in: Time Warps, String Edits and Macromolecules: The Theory and Practice of Sequence Comparison, Addison Wesley Publishing Company1983, pp. 125–161.

[10] J. Aach, G.M. Church, Aligning gene expression time series with time warping algorithms, Bioinformatics 17 (6) (2001) 495–508.

[11] Z. Bar-Joseph, G. Gerber, D.K. Gifford, T.S. Jaakkola, I. Simon, A new approach to analyzing gene expression time series data, in: RECOMB: Proceedings of the Sixth Annual International Conference on Computational Biology, ACM, New York, NY, USA2002, pp. 39–48.

[12] D.M. Gavrila, L.S. Davis, Towards 3-D model-based tracking and recognition of human movement: a multi-view approach, in: IEEE International Workshop on Automatic Face- and Gesture-Recognition., 1995, pp. 272–277.

[13] T. Rath, R. Manmatha, Word image matching using dynamic time warping, IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, 2003, pp. 521–527.

[14] V. Niennattrakul, C.A. Ratanamahatana, Inaccuracies of shape averaging method using dynamic time warping for time series data, in: S. Berlin (Ed.), Computational Science–ICCS, Lecture Notes in Computer Science, vol. 4487, 2007.

[15] V. Niennattrakul, C.A. Ratanamahatana, On clustering multimedia time series data using K-means and dynamic time warping, in: International Conference on Multimedia and Ubiquitous Engineering, 2007, pp. 733–738.

[16] T. Liao, B. Bolt, J. Forester, E. Hailman, C. Hansen, R. Kaste, J. O'May, Understanding and projecting the battle state, in: 23rd Army Science Conference, 2002.

[17] T.W. Liao, C.-F. Ting, P.-C. Chang, An adaptive genetic clustering method for exploratory mining of feature vector and time series data, International Journal of Production Research 44 (2006) 2731–2748.

[18] V. Hautamaki, P. Nykanen, P. Franti, Time-series clustering by approximate prototypes, in: 19th International Conference on Pattern Recognition, 2008, pp. 1–4.

[19] E. Keogh, X. Xi, L. Wei, C.A. Ratanamahatana, The UCR time series classification/clustering homepage, 〈http://www.cs.ucr.edu/∼eamonn/time_series_data/〉, 2006.

[20] D. Gusfield, Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology, Cambridge University Press, 1997.

[21] V. Niennattrakul, C.A. Ratanamahatana, Shape averaging under time warping, in: International Conference on Electrical Engineering/Electronics, Computer, Telecommunications, and Information Technology, 2009.

[22] E. Dimitriadou, A. Weingessel, K. Hornik, A combination scheme for fuzzy clustering, International Journal of Pattern Recognition and Artificial Intelligence 16 (7) (2002) 901–912.

[23] S.B. Needleman, C.D. Wunsch, A general method applicable to the search for similarities in the amino acid sequence of two proteins, Journal of Molecular Biology 48 (3) (1970) 443–453.

[24] L. Wang, T. Jiang, On the complexity of multiple sequence alignment, Journal of Computational Biology 1 (4) (1994) 337–348.

[25] R.C. Edgar, MUSCLE: a multiple sequence alignment method with reduced time and space complexity, BMC Bioinformatics 5 (1) (2004) 1792–1797.

[26] J. Pei, R. Sadreyev, N.V. Grishin, PCMA: fast and accurate multiple sequence alignment based on profile consistency, Bioinformatics 19 (3) (2003) 427–428.

[27] T. Lassmann, E.L.L. Sonnhammer, Kalign—an accurate and fast multiple sequence alignment algorithm, BMC Bioinformatics 6 (1) (2005) 298–306.

[28] C. Notredame, D.G. Higgins, J. Heringa, T-coffee: a novel method for fast and accurate multiple sequence alignment, Journal of Molecular Biology 302 (1) (2000) 205–217.

[29] J. Pei, N.V. Grishin, PROMALS: towards accurate multiple sequence alignments of distantly related proteins, Bioinformatics 23 (7) (2007) 802–808.

[30] L. Gupta, D. Molfese, R. Tammana, P. Simos, Nonlinear alignment and averaging for estimating the evoked potential, IEEE Transactions on Biomedical Engineering 43 (4) (1996) 348–356.

[31] A.W.-C. Fu, E.J. Keogh, L.Y.H. Lau, C.A. Ratanamahatana, R.C.-W. Wong, Scaling and time warping in time series querying, VLDB Journal 17 (4) (2008) 899–921.

[32] CNES, Kalideos, Distribution Spot Image, 〈http://kalideos.cnes.fr〉, 2009.

François Petitjean is a Ph.D. student in Computer Science at the University of Strasbourg, France. His research interests include complex time series clustering and remote sensing image analysis.

Alain Ketterlin holds a Ph.D. in Computer Science from the University Louis Pasteur, Strasbourg, France. Since 1995, he is Assistant Professor at University of Strasbourg. His research interests include data mining and remote sensing image processing.

Pierre Gançarski received his Ph.D. degree in 1989 and his Habilitation in 2007 in Computer Science from the University Louis Pasteur, Strasbourg, France. He is currently an Associate Professor at the Department of Computer Science from the University of Strasbourg. His current research interests include collaborative multistrategical clustering with applications to complex data mining and remote sensing image analysis.