Elastic Partial Matching of Time Series

L. J. Latecki[1], V. Megalooikonomou[1], Q. Wang[1], R. Lakaemper[1],
C. A. Ratanamahatana[2], and E. Keogh[2]

[1] Computer and Information Sciences Dept.
Temple University, Philadelphia, PA 19122
{ latecki, vasilis, qwang, lakaemper }@temple.edu
[2] Computer Science and Engineering Dept.
University of California, Riverside, CA 92521
{ ratana, eamonn }@cs.ucr.edu

Abstract. We consider a problem of elastic matching of time series. We propose an algorithm that automatically determines a subsequence b' of a target time series b that best matches a query series a. In the proposed algorithm we map the problem of the best matching subsequence to the problem of a cheapest path in a DAG (directed acyclic graph). Our experimental results demonstrate that the proposed algorithm outperforms the commonly used Dynamic Time Warping in retrieval accuracy.

1 Motivation

For many datasets we can easily and accurately extract the beginning and ending of patterns of interest. However in some domains it is non-trivial to define the exact beginning and ending of a pattern within a longer sequence. This is a problem because if the endpoints are incorrectly specified they can swamp the distance calculation in otherwise similar objects. For concreteness we will consider an example of just such a domain and show that Minimal Variance Matching (MVM), proposed in this paper, can be expected to outperform Dynamic Time Warping (DTW) and Euclidean distance. There is increasing interest in indexing sports data, both from sports fans who may wish to find particular types of shots or moves, and from coaches who are interested in analyzing their athletes performance over time. Let us consider the high jump. We can automatically collect the athletes center of mass information from video and convert to time series. In Fig. 1, we see 3 time series automatically extracted from 2 athletes.

Both sequence **A** and **B** are from one individual, a tall male, and **C** is from a (relatively) short female with a radically different style. The difference in their technique is obvious even to a non-expert, however **A** and **C** where automatically segmented in such a way that the bounce from the mat is visible, whereas in **B** this bounce was truncated. In Fig. 1(*middle*) we can see that DTW is forced to map this bounce section to the end of sequence **B**, even though that sequence clearly does not have a truly corresponding section. In contrast MVM is free to ignore the sections that do not have a natural correspondence. It is this difference that enables MVM to produce the more natural clustering shown in

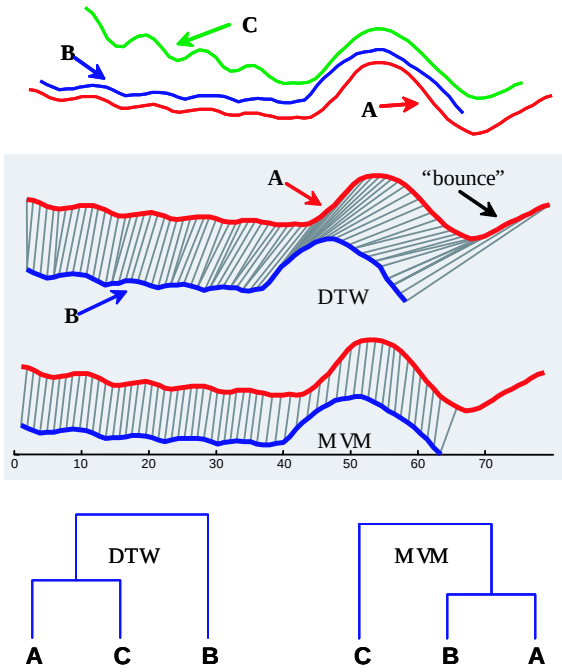

Fig. 1. (*top*) Three examples of athletes trajectories as they attempt a high jump. The sequence shows the height of their center of mass (with possible parallax effects). Reading left to right we can see their bounding run followed by the takeoff and landing. (*middle*) The alignment achieved by DTW and MVM on two of the sequences. (*bottom*) The clustering achieved by DTW and MVM.

Fig. 1(*bottom*). While this is a somewhat contrived example on a specialized domain, similar remarks apply to many commercially important domains including medical data mining and oil exploration.

2 Related Work

Because time series are a ubiquitous and increasingly prevalent type of data, there has been much research effort devoted to time series data mining in recent years. Many data mining algorithms have similarity measurement at their core. Examples include motif discovery [1], anomaly detection [2], rule discovery [3], classification [4] and clustering [5]. In this paper we deal with computation of time series distances based on elastic time series matching.

As many researchers have mentioned in their work [3, 4, 6], the Euclidean distance is not always the optimal distance measure for similarity searches. For example, in some time series, different parts have different levels of significance in their meaning. Also, the Euclidean distance does not allow shifting in time axis, which is not unusual in real life applications.

To solve the problem of time scaling in time series, Dynamic Time Warping (DTW) [7, 8] aligns the time axis prior to the calculation of the distance. DTW distance between time series is the sum of distances of their corresponding elements. Dynamic programming is used to find corresponding elements so that this distance is minimal. The DTW distance has been shown to be superior to the Euclidean in many cases [5, 9–11]. See [12] for a detailed discussion of DTW. As illustrated in Section 1, DTW requires the matched time series to be well aligned, and it is particularly sensitive to outliers, since it is not able to skip any elements of the target series. DTW always matches the query time series to the whole target time series.

The Longest Common Subsequence (LCSS) measure has been used in time series [13, 14] to deal with the alignment and outliers problems. Given a query and a target series, LCSS determines their longest common subsequence, i.e., LCSS finds subsequences of the query and target (of the same length) that best correspond to each other. The distance is based on the ratio between the length of longest common subsequence and the length of the whole sequence. The subsequence does not need to consist of consecutive points, the order of points is not rearranged, and some points can remain unmatched. When LCSS is applied to time series of numeric values, one needs to set a threshold that determines when values of corresponding points are treated as equal [14]. The performance of LCSS heavily depends on correct setting of this threshold, which may be a particularly difficult problem for some applications.

The proposed MVM (Minimal Variance Matching) computes the distance value between two time series directly based on the distances of corresponding elements, just as DTW does, and it allows the query sequence to match to only subsequence of the target sequence, just as LCSS does. The main difference between LCSS and MVM is that LCSS optimizes over the length of the longest common subsequence (which requires the distance threshold), while MVM directly optimizes the sum of distances of corresponding elements (without any distance threshold). The main difference between DTW and MVM is that MVM can skip some elements of the target series when computing the correspondence.

While DTW requires that each point of the query sequence is matched to each element of the target sequence, MVM allows skipping elements of the target sequence. LCSS allows skipping elements of both query and target sequence. Therefore, MVM should be used when one is interested in finding the best matching part of the target sequence for a given query sequence, since it guarantees that the whole query sequence will be matched. This is, for example, the case, when the query is a model sequence, one wants to find in a given data set. However, when the query sequence contains outliers and skipping them is allowed, then LCSS should be used.

3 Minimal Variance Matching

We now present an algorithm for elastic matching of two time series of different lengths m and n, which we will call **Minimal Variance Matching (MVM)**.

More specifically, for two finite sequences of real positive numbers $a = (a_1, ..., a_m)$ and $b = (b_1, ..., b_n)$ with $m < n$, the goal is to find a subsequence b' of b of length m such that a best matches b'. Thus, we want to find the best possible correspondence of sequence a to a subsequence b' of b. Formally we define a **correspondence** as a monotonic injection $f : \{1, ..., m\} \rightarrow \{1, ..., n\}$, (i.e., a function f such that $f(i) < f(i + 1)$) such that a_i is mapped to $b_{f(i)}$ for all $i \in \{1, ..., m\}$. The set of indices $\{f(1), ..., f(m)\}$ defines the subsequence b' of b. Recall that in the case of DTW, the correspondence is a relation on the set of indices $\{1, ..., m\} \times \{1, ..., n\}$, i.e., a one-to-many and many-to-one mapping.

Once the correspondence is known, it is easy to compute the distance between the two sequences. We do not have any restrictions on distance functions, i.e., any distance function is possible. To allow for comparison to the existing time series matching techniques, we use the Euclidean distance in this paper:

$$d(a, b, f) = \sqrt{\sum_{i=1}^{m} (b_{f(i)} - a_i)^2}. \tag{1}$$

Our goal is to find a correspondence f so that $d(a, b, f)$ is minimal. More precisely, an optimal correspondence $\hat{f}$ of numbers in series a to numbers in series b is defined as the one that yields the global minimum of $d(a, b, f)$ over all possible correspondences f:

$$\hat{f} = \text{argmin}\{d(a, b, f) : f \text{ is a correspondence}\}. \tag{2}$$

Finally, the optimal distance is obtained as $d(a, b) = d(a, b, \hat{f})$, i.e., $d(a, b)$ is the global minimum over all possible correspondences.

We can also state the correspondence problem in a statistical framework. Let us assume that there is a subsequence b' of b that is a noisy version of a such that $a \sim b' + \mathcal{N}(0, v)$, where $\mathcal{N}(0, v)$ denotes a zero-mean Gaussian noise variable with variance v, i.e., $b' = (b_{f(i)})_i$ for $i \in \{1, ..., m\}$. Since the mean of the differences $(b_{f(i)} - a_i)_i$ is zero, i.e., $b' - a \sim \mathcal{N}(0, v)$, the variance σ^2 of difference sequence $(b_{f(i)} - a_i)_i$ is given by

$$\sigma^2(a, b, f) = \frac{1}{m} \sum_{i=1}^{m} (b_{f(i)} - a_i)^2. \tag{3}$$

Clearly, $\sigma^2(a, b, f) = v$ (the variance of the Gaussian noise). Observe that in this case the variance corresponds to the Euclidean distance (1). Thus, the variance of the difference sequence is minimal when mapping f establishes a correct correspondence of elements of both sequences.

Now we describe the method used to minimize (3). We first form the difference matrix

$$r = (r_{ij}) = (b_j - a_i).$$

It is a matrix with m rows and n columns with $m < n$. For example, the difference matrix for two time series t_1=(1, 2, 8, 6, 8) and t_2=(1, 2, 9, 3, 3, 5, 9) is shown

$$r = \begin{bmatrix} \boxed{0} & 1 & 8 & 2 & 2 & 4 & 8 \\ -1 & \boxed{0} & 7 & 1 & 1 & 3 & 7 \\ -7 & -6 & \boxed{1} & -5 & -5 & -3 & 1 \\ -5 & -4 & 3 & -3 & -3 & \boxed{-1} & 3 \\ -7 & -6 & 1 & -5 & -5 & -3 & \boxed{1} \end{bmatrix}$$

Fig. 2. In order to compute $\hat{f}$ for t_1=(1, 2, 8, 6, 8) and t_2=(1, 2, 9, 3, 3, 5, 9), we first form the difference matrix with rows corresponding to elements of t_1 and columns to elements of t_2.

in Fig. 2. Observe that t_1 and t_2 are similar if we ignore the two elements in t_2 with value 3.

Clearly, (r_{ij}) can be viewed as a surface over a rectangle of size m by n, where the height at point (i, j) is the value r_{ij}. We obtain the correspondence with minimal variance by solving the least-value path problem on the difference matrix. To obtain the solution, we treat (r_{ij}) as a directed graph with the links: r_{ij} is directly linked to r_{kl} if and only if **(1)** $k - i = 1$ and **(2)** $j < l$. When traversing the obtained directed graph, the meaning of both conditions is as follows. For any two consecutive points r_{ij}, r_{kl} in each path (1) means that we always go to the next row, while (2) means that we can skip some columns, but cannot go backwards.

Our goal is to have a least-value path with respect to the following cost function for each directed link: $linkcost(r_{ij}, r_{kl}) = (r_{kl})^2$. Each path can start in first row, between columns 1 and $n - m$, i.e., at r_{1j} for $j = 1, ..., n - m$ and the path can end at r_{mj} for $j = n - m, ..., n$. The conditions (1) and (2) imply that we can obtain a DAG (directed acyclic graph) G whose nodes are the elements of $(r_{ij})_{ij}$ and weights are defined by the function $linkcost$. It is well known that we can solve the least-value path problem using the shortest path algorithm on G. The obtained least-value path defines exactly correspondence $\hat{f}$, which minimizes (3) in accordance with (2).

The shortest path for the example matrix in Fig. 2 is marked with boxes. Following the boxes, the optimal correspondence $\hat{f}$ is given by

$$\hat{f}(1) = 1,\ \hat{f}(2) = 2,\ \hat{f}(3) = 3,\ \hat{f}(4) = 6,\ \hat{f}(5) = 7.$$

Finally, from (1) we obtain the distance $d(t_1, t_2) = \sqrt{3} \approx 1.732$.

The obtained optimal correspondence $\hat{f}$ automatically determines a subsequence $b' = \hat{f}(a)$ of a target time series b that best matches a query series a. In particular, two cases are possible: **Whole Sequence Matching:** Subsequence b' is dense in b, which indicates a similarity of a to b. **Subsequence Matching:** Subsequence b' is not dense in b but is dense in part of b which indicates a similarity of a to part of b.

4 Experimental Results

We compare the results of MVM to the DTW results on three data sets Face, Leaf, and Gun from [12]. A detailed description of these data sets is given in [12]. We briefly mention that Face dataset is composed of 112 sequences representing head profiles of 4 different individuals. The length of each sequence ranges from 107 to 240 points. Leaf dataset is composed of 442 sequences representing contours of six different leaf species. The length of each sequence ranges from 22 to 475 points. Gun dataset is composed of 200 sequences representing gun drawing events by two different actors. The length of each sequence is 150 points.

Following [12], we measure the classification accuracy of 1-NN (Nearest Neighbor) classifier applied to the distance matrices obtained by the evaluated methods. The obtained results are shown in Table 1. As can be seen MVM systematically outperforms DTW. The DTW results are cited from [12], where all possible sizes of warping windows for DTW were examined and the optimal warping window size was determined for each data set. We did not use any warping or correspondence window bound for MVM.

	Face	**Gun**	**Leaf**
MVM	98.21	100	97.29
DTW	96.43	99.00	96.38

Table 1. 1-NN classification accuracy. The DTW results are cited from [12].

Although the proposed method does not require any length normalization, we used length normalized time series in order to allow for a comparison to the results in [12]. When calculating the distance between a pair of time series with MVM, we resampled the query series so that its length is approximately 75% of the length of the target series. This means that the total elasticity amount for MVM is about 25% of the length of the second time series. We obtained nearly identical results with elasticity varying from 25% to 50%. The values of each time series were zero-mean normalized in the standard way. That is, each time series X is normalized as: $X = \frac{(X-\mu(X))}{\sigma(X)}$, where $\mu(X)$ is the mean value of X and $\sigma(X)$ is its standard deviation.

The superior performance of MVM reported in Table 1 is due to MVM ability to correctly align matched sequences in that *bad matching* elements of the target sequence are excluded from the correspondence. One example of this fact is given in Section 1. Here we use the Face dataset in order to directly link this fact to the superior classification accuracy of MVM. The face dataset is a particularly good dataset on which to demonstrate this fact. It consists of head profiles converted to time series representing the curvature at sample points. Because the face is intrinsically elastic, as the subject smiles or grimaces, Euclidean distance in unsuitable here, and we therefore would consider a more elastic distance measure such as DTW or MVM.

As one might imagine, the parts of the signal that correspond to the face contain the most useful information, and the parts of the signal that correspond to the back of the head contain much less information. In fact the parts of the signal that correspond to the back of the head may actually contain misleading information, since the texture of hair causes problems for the time series generation algorithm. The problem with DTW is that it is forced to align everything, thus may be forced to align this poor quality data from the back of the head in one signal to poor quality data in another. A small amount of such poor quality data can rapidly swamp the distance calculation. In contrast, MVM has the ability to simply ignore the poor quality data, as shown in Fig. 3. Note that in this simple contrived example we might be able to achieve better results simply by truncating the back of the head section of the signals. However this begs a nontrivial question of finding a good segmentation algorithm. In any case in data mining we generally assume that we do not have such a priori knowledge about the domain in question.

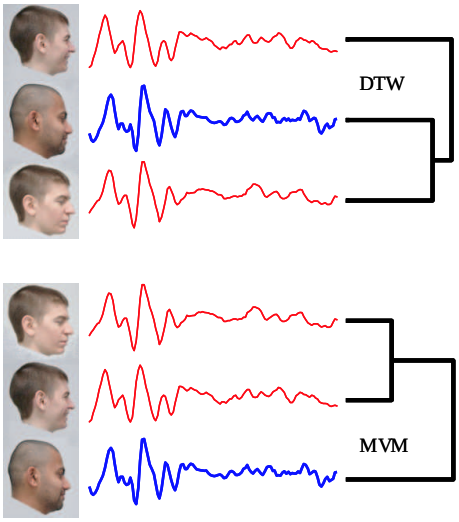

Fig. 3. Three time series from Face dataset, derived from profiles, compared using DTW and MVM. Two time series of the same person are correctly identified by MVM which is not the case for DTW.

5 Conclusions

The proposed new method for time series matching, called MVM, performs the following tasks simultaneously **(1)** automatically determines whether the query sequence best matches the whole target sequence or only part of the target

sequence, **(2)** automatically skips outliers that are present in the target sequence, **(3)** minimizes the statistical variance of dissimilarities of corresponding elements. The reported experiments show that this method outperforms DTW. By mapping the problem of elastic matching of sequences to finding a cheapest path in a DAG, we provide an efficient algorithm to compute MVM.

6 Acknowledgments

This work was supported in part by NSF under Grant No. IIS-0237921, and by NIH under Grant No. R01MH68066-01A1. We would like to thank the two models that appear in Figure 3, Sandeep Gupta and Paul Dilorenzo. Finally we acknowledge the helpful comments of the reviewers.

References

1. Chiu, B., Keogh, E., Lonardi, S.: Probabilistic discovery of time series motifs. In: Proc. ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, Washington (2003)
2. Keogh, E., Lonardi, S., Ratanamahatana, C.: Towards parameter-free data mining. In: Proc. ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, Seattle (2004)
3. Höppner, F.: Discovery of temporal patterns. learning rules about the qualitative behavior of time series. In: Proc. European Conf. on Principles and Practice of Knowledge Discovery in Databases, Freiburg (2001) 192–203
4. Rafiei, D.: On similarity-based queries for time series data. In: Proc. Int. Conf. on Data Engineering, Sydney (1999) 410–417
5. Aach, J., Church, G.: Aligning gene expression time series with time warping algorithms. Bioinformatics **17** (2001) 495–508
6. Megalooikonomou, V., Wang, Q., Li, G., Faloutsos, C.: A multiresolution symbolic representation of time series. In: Proc. IEEE Int. Conf. on Data Engineering (ICDE05), Tokyo (2005) 668–679
7. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. IEEE Trans. on Acoustics, Speech, and Signal Processing **26** (1978) 43–49
8. Berndt, D., Clifford, J.: Using dynamic time warping to find patterns in time series. In: Proc. AAAI-94 W. on Knowledge Discovery and Databases. (1994) 229–248
9. Kollios, G., Vlachos, M., Gunopoulos, D.: Discovering similar multidimensional trajectories. In: Proc. Int. Conf. on Data Engineering, San Jose (2002) 673–684
10. Chu, S., Keogh, E., Hart, D., Pazzani, M.: Iterative deepening dynamic time warping for time series. In: Proc. SIAM Int. Conf. on Data Mining. (2002)
11. Yi, B., Jagadish, K., Faloutsos, C.: Efficient retrieval of similar time sequences under time warping. In: Proc. Int. Conf. on Data Engineering. (1998) 23–27
12. Ratanamahatana, C.A., Keogh, E.: Everything you know about dynamic time warping is wrong. In: W. on Mining Temporal and Sequential Data, Seattle (2004)
13. Das, G., Gunopoulos, D., Mannila, H.: Finding similar timie series. In: Proc. 1st PKDD Symposium. (1997) 88–100
14. Vlachos, M., Hadjieleftheriou, M., Gunopoulos, D., Keogh, E.: Indexing multidimensional time-series with support for multiple distance measures. In: Proc. of ACM SIGKDD, Washington (2003) 216–225