Fast Similarity Search of Multi-dimensional Time Series via Segment Rotation

Xudong Gong[1], Yan Xiong[1], Wenchao Huang[1(✉)], Lei Chen[2], Qiwei Lu[1], and Yiqing Hu[1]

[1] University of Science and Technology of China, Hefei, China
lzgxd@mail.ustc.edu.cn, {yxiong,huangwc}@ustc.edu.cn
[2] Hong Kong University of Science and Technology, Hong Kong, China
leichen@cse.ust.hk

Abstract. Multi-dimensional time series is playing an increasingly important role in the "big data" era, one noticeable representative being the pervasive trajectory data. Numerous applications of multi-dimensional time series all require to find similar time series of a given one, and regarding this purpose, Dynamic Time Warping (DTW) is the most widely used distance measure. Due to the high computation overhead of DTW, many lower bounding methods have been proposed to speed up similarity search. However, almost all the existing lower bounds are for general time series, which means they do not take advantage of the unique characteristics of higher dimensional time series. In this paper, we introduce a new lower bound for constrained DTW on multi-dimensional time series to achieve fast similarity search. The key observation is that when the time series is multi-dimensional, it can be rotated around the time axis, which helps to minimize the bounding envelope, thus improve the tightness, and in consequence the pruning power, of the lower bound. The experiment result on real world datasets demonstrates that our proposed method achieves faster similarity search than state-of-the-art techniques based on DTW.

1 Introduction

Multi-dimensional time series are playing an increasingly important role in this "big data" era. For example, with the rapid development of wireless communication and location positioning technologies, we can easily acquire the location of a moving object, e.g. a person, a vehicle, or an animal, at different time. Such movements are generally recorded as a series of triples (x, y, t), where x and y are coordinates and t is the sample time. When talking about the dimensionality of time series, the sample time is often omitted, so such trajectory is regarded as "two dimensional" time series, which is a representative of multi-dimensional time series. Various time series data has enabled many interesting applications, such as finding potential friends according to similar trajectories [12], human activity recognition [20], and climate change prediction [15] etc.

A basic and important operation in various applications of multi-dimensional time series is to find similar time series of a given one, which is a *similarity search*

M. Renz et al. (Eds.): DASFAA 2015, Part I, LNCS 9049, pp. 108–124, 2015.
DOI: 10.1007/978-3-319-18120-2_7

problem. The similarity between two time series is often decided by the distance between them. According to the thorough experiments carried out in [19], among all the proposed distance measures for time series data, DTW may be potentially the best one, and it has achieved great success in highly diverse domains, such as DNA sequence clustering [13], query by humming [26], RFID tag location [18] etc. The straightforward computation of DTW takes quadratic time, which renders it unacceptably slow for applications involving large datasets. In the past years, many techniques have been proposed to prune unqualified candidates by first computing a lower bound, thus reduce the number of required DTW computations [9,11,21,26]. To the best of our knowledge, all the proposed lower bounding techniques are for general time series, which means they don't care the dimensionality of the data, although there are efforts to extend some lower bounding methods to multi-dimensional time series, such as [14,17].

We notice that these general lower bounding techniques can be further improved if we consider some unique characteristics of time series in higher dimensional space. For example, when a time series is in two or more dimensional space, it can be rotated around the time axis without changing its geometrical property, thus the distance between two time series will not be affected. This feature of multi-dimensional time series can be utilized to get a tighter lower bound for candidate pruning.

Inspired by this observation, we introduce a new lower bound called *LB_rotation* to speed up similarity search process. The basic idea is to, for each time series, rotate it by an appropriate angle to reduce the volume of its envelope, because it has been pointed out in [9] that "the envelope is wider when the underlying query sequence is changing rapidly, and narrower when the query sequence plateaus". In such a way, we can improve the tightness of the lower bound, and prune more unqualified candidates to reduce the required DTW computations.

In order to get a satisfactory lower bound, we need to solve several problems:

- **Directly rotate the whole time series may not be a good idea**. As we will show later, the more straight a time series is, the better improvement we can get. Thus we first perform segmentation on the target time series to divide it into several segments as straight as possible, then deal with each segment respectively.
- **Deciding the rotation angle**. This is a key factor affecting the effectiveness of LB_rotation. Rather than directly reducing the volume of the envelope, we aim at reducing the volume of its bounding hypercube, since the envelope is included in the hypercube. For every time series segment, we can find the direction of its major axis by least square fitting. The rotation angle is just the included angle between this direction and the x axis.
- **Computing the lower bound**. After segmentation, the warping range of a point in the candidate time series may intersect with several segments of the query time series, thus we have to compute the distance between the point and its matching point in each segment, and sum up the minimal distance to get final lower bound. We first construct extended envelope for each segment, then locate the corresponding matching point for distance computation.

To demonstrate the superiority of LB_rotation, we compare it with LB_Keogh [9], which is the most widely used lower bound for constrained DTW, and LB_Improved [11], which is recognized as the only lower bound that actually improves LB_Keogh [19], through experiments on real world datasets. The executable and datasets we used are freely available at [1]. We will show that increasing warping constraint has smaller impact on the tightness of LB_rotation, while it may hurt the tightness of LB_Keogh and LB_Improved considerably.

Our major contribution can be summarized as follows:

- We propose a new lower bound *LB_rotation* for constrained DTW based on time series rotation to achieve fast similarity search on multi-dimensional time series. It can shrink the envelope of time series, thus improve the tightness of lower bound, which helps to reduce similarity search time.
- We improve the effectiveness of LB_rotation by dividing the time series into several segments and rotating each segment respectively, rather than directly rotating the whole time series. The experiment result on real world datasets shows that LB_rotation is more effective than existing lower bounds.

The rest of this paper is organized as follows: Section 2 reviews related work, and introduces some necessary extensions. Then we demonstrate the details of LB_rotation in Section 3. Experiment results and discussions are presented in Section 4. Section 5 concludes this paper.

2 Preliminaries

2.1 Related Work

Since retrieval of similar time series plays an important role in many applications, such as time series data mining, a lot of effort has been devoted to solving this problem, and many distance measures have been proposed, such as Euclidean Distance [7], Dynamic Time Warping (DTW) [21], Longest Common Subsequences (LCSS) [5], Edit Distance on Real sequence (EDR) [4], Edit distance with Real Penalty (ERP) [3], etc. Among them, DTW is the most widely used distance measure on time series, because of its effectiveness and robustness.

Due to the high computation complexity of DTW, there are many techniques developed for it to speed up the distance computation. These techniques can be mainly divided into two categories: a) directly speed up DTW computation; b) reduce the number of DTW computations via lower bounding. Lower bounding DTW is a widely used technique, because it can filter a large part of candidate time series using relatively cheap computation. Generally, for any lower bounding algorithm, the nearest neighbor searching process is shown in Algorithm 1. It's clear that the tighter a lower bound is, the higher its pruning power will be, since more candidates will be discarded in Algorithm 1.

The early attempts to lower bound DTW are LB_Yi[21] and LB_Kim[10]. Since they only use the global information of the time series, such as the maximal and minimal values to compute the lower bound, their results are relatively

Algorithm 1. Nearest time series search using lower bounding method

Input: Q ▷ query time series
Input: $\mathcal{C}$ ▷ database of candidate time series
Output: the index of nearest time series regarding Q

```
1: function NearestNeighbor(Q,C)
2:     dist_min = ∞
3:     for i ← 1 to |C|
4:         lb ← lower_bound(Q, C_i)
5:         if lb < dist_min
6:             true_dist ← DTW(Q, C_i)      ▷ C_i is the ith candidate in the database
7:             if true_dist < dist_min
8:                 dist_min ← true_dist
9:                 index ← i
10:    return index
```

loose. Keogh et al. [9] took advantage of the warping constraint to construct an envelope for the query time series, and proposed the first non-trivial lower bound LB_Keogh for DTW, which greatly eliminates the number of required DTW computations.

There are several extensions of LB_Keogh, e.g. [11,16,25,26]. Among them, LB_Improved [11] is recognized as the only improvement that has reproducible result to reduce searching time [19], thus we compare LB_rotation with LB_Keogh and LB_Improved in Section 4. LB_Improved is built upon LB_Keogh, which improves the tightness through a second pass. It is computed as $LB\_Improved(Q,C) = LB\_Keogh(Q,C) + LB\_Keogh(Q,H(C,Q))$, where $H(C,Q)$ is the projection of C on Q [11].

Generally for almost all the non-trivial lower bounding techniques, there are two prerequisites: a) DTW must be compliant to a constraint enforced on the warping path; b) the trajectories should be of the same length. If not otherwise stated, we assume these conditions are already met hereafter. For more details of lower bounding DTW, please refer to [9,11,19].

We note that these lower bounding techniques are all for general time series, without considering the unique characteristics of high dimensional time series. Actually, when it comes to time series in two or more dimensional space, we can rotate the time series to "flatten" them, thus reduce the volume of their bounding envelopes, which will improve the tightness of the lower bound, as we will show in the following of this paper.

2.2 Extending LB_Keogh and LB_Improved to Multi-dimensional Time Series

Now we introduce the extended lower bounds for multi-dimensional time series, and use them for experimental comparison in Section 4.

Originally, LB_Keogh and LB_Improved are proposed to deal with one dimensional (univariate) time series. We start extending them to multi-dimensional time series by introducing multi-dimensional bounding envelopes.

Definition 1. *The bounding envelope of a time series Q of length n in l dimensional space, with respect to the global warping constraint c, is defined as*

$$Env(Q) = (U_1, U_2, \ldots, U_n, L_1, L_2, \ldots, L_n), \tag{1}$$

where $U_i = (u_{i,1}, u_{i,2}, \ldots, u_{i,l})$, $L_i = (l_{i,1}, l_{i,2}, \ldots, l_{i,l})$, and $u_{i,p} = \max\{q_{i-c,p} : q_{i+c,p}\}$, $l_{i,p} = \min\{q_{i-c,p} : q_{i+c,p}\}$, where q_i is the ith point in Q.

For LB_Keogh, we adopt the extension introduced in [14].

Definition 2. *The multi-dimensional extension of LB_Keogh is defined as*

$$LB\_MV(Q,C) = \sqrt{\sum_{i=1}^{n}\sum_{p=1}^{l}\begin{cases}(c_{i,p}-u_{i,p})^2, & \text{if } c_{i,p} > u_{i,p}\\ (c_{i,p}-l_{i,p})^2, & \text{if } c_{i,p} < l_{i,p}\\ 0, & \text{otherwise}\end{cases}} \tag{2}$$

where Q is the query time series, C is the candidate time series, c_i is the ith point in C, u_p and l_p are the maximum and minimum values of dimension p, with respect to Q. n is the length of the time series, and l is the dimensionality of each point in the time series.

The proposition below is proved in [14].

Proposition 1. *For any two sequences Q and C of the same length n, for any global constraint on the warping path of the form $j - c \le i \le j + c$, the following inequality holds: $LB\_MV(Q,C) \le DTW(Q,C)$.*

Following [14], we extend LB_Improved to multi-dimensional time series. We only need to extend the projection function (equation (1) in [11]) as follows.

Definition 3. *The projection of C on Q in multi-dimensional LB_Improved is defined as*

$$H(C,Q)_{i,p} = \begin{cases}u_{i,p} & \text{if } c_{i,p} \ge u_{i,p}\\ l_{i,p} & \text{if } c_{i,p} \le l_{i,p}\\ q_{i,p} & \text{otherwise}\end{cases}, 1 \le p \le l \tag{3}$$

Similarly, we can prove the following proposition.

Proposition 2. *For any two sequences Q and C of the same length n, for any global constraint on the warping path of the form $j - c \le i \le j + c$, the following inequality holds: $LB\_Improved(Q,C) \le DTW(Q,C)$.*

The proof of Proposition 2 is a straightforward extension of Proposition 1, since LB_Improved simply uses LB_Keogh twice; we omit it for brevity.

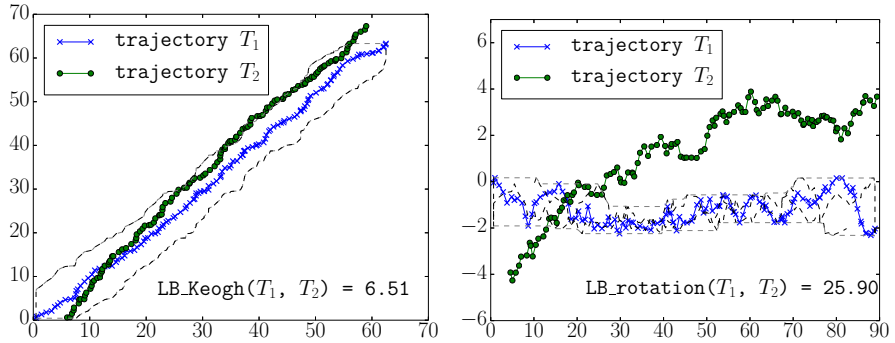
For the succinctness of notations, hereafter we use LB_Keogh and LB_Improved to refer to the multi-dimensional extension of the original version respectively. By convention, we also use *time series* and *trajectory* interchangeably when referring to two or more dimensional time series.

3 LB_rotation

As discussed in Section 1, when it comes to two or more dimensional space, the query time series can be rotated by a certain angle to minimize the volume of its envelope, thus improves the tightness of the lower bound. We will present the details in this section.

3.1 Intuitive Explanation

First we use an example to show the idea. For simplicity, we only plot the projections of the two time series as well as the envelope of the query time series (time series T_1) in the $x - y$ plane. Note that in Figure 1a the four envelopes are partly overlapped.



(a) Original trajectories **(b)** Rotated trajectories

Fig. 1. Two sample trajectories of length $n = 128$, with warping width $c = 0.1n$. The true DTW distance $DTW(T_1, T_2) = 36.84$.

In Figure 1a are the original trajectories, and in Figure 1b are the rotated trajectories. The y axis is scaled with respect to the coordinate range. After rotation, the y axis has a very small span, because those points in time series T_1 almost lie in the same straight line. It's clear that before rotation, time series T_2 is almost wholly inside the envelope of time series T_1, while only a small part of it is contained in the envelope of time series T_1, after rotation. So, if we rotate the trajectories by an appropriate angle, we can reduce the volume of their envelopes, thus get a tighter lower bound.

However, we should note that, by rotating the time series, we can only reduce the area of the envelopes in either the $t - y$ plane or the $t - x$ plane; we cannot achieve area reduction in both planes. This is because the geometrical shape of the time series is rotation-invariant. If it becomes flat in one direction after rotation, it will surely become steep in the perpendicular direction. In the above

example, the area of envelope in the $t-x$ plane actually increases, which will counteract the gain in the $t-y$ plane. Nonetheless, if we can reduce enough area of the envelopes in one direction, the result is still preferable, as we can see in the experiment result in Section 4. To achieve this, we need to divide the time series into several segments that are as straight as possible, because we can see from the example in Figure 1 that straight time series will greatly reduce the volume of envelope after rotation. The details will be introduced later.

3.2 Formal Definition of LB_rotation

Based on above observation, we propose a new lower bound for the constrained DTW, which we call *LB_rotation*. To formally define LB_rotation, we first define time series segmentation, and the distance from a point to the envelope of a segment.

Definition 4. *The segmentation of a time series Q is to divide Q into consecutive and non-overlapping segments $s_i = Q[s_i.start, s_i.end]$ where $\cup_{s_i} = Q \wedge \forall i \neq j : s_i \cap s_j = \emptyset$.*

Definition 5. *The distance from a point q to the envelope $Env(s)$ of a time series segment s is defined as*

$$d(q, Env(s)) = d(q, Env(s)_i) = \sum_{p=1}^{l} \begin{cases} (q_p - u_{i,p})^2 & \text{if } q_p > u_{i,p} \\ (q_p - l_{i,p})^2 & \text{if } q_p < l_{i,p} \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

where i is the index of the matching point in s with respect to p.

How to decide this matching point will be deferred to Algorithm 2.

Definition 6. (LB_rotation). *The lower bound LB_rotation of two time series Q and C of length n is defined as*

$$LB\_rotation(Q, C) = \sum_{i=1}^{n} \min_{s_j \in \mathcal{S}_i} \{d(c_i, Env(s_j))\} \quad (5)$$

where $Env(s_j)$ is the bounding envelope of segment s_j, and $\mathcal{S}_i = \{s_k \mid s_k \in Q \wedge [s_k.start, s_k.end] \cap [i-c, i+c] \neq \emptyset\}$, c is the warping constraint.

For each point $c_i \in C$, we compute the distance from c_i to the segments in Q that overlaps with $Q[i-c, i+c]$ respectively, and sum up the minimal distance regarding each point as the final lower bound. This ensures that no matter which point $q_i \in Q$ is matched by c_i, the contribution of c_i to the lower bound will never exceed $d(c_i, q_i)$.

We can prove the following proposition.

Proposition 3. *For any two sequences Q and C of the same length n, for any global constraint on the warping path of the form $j - c \leq i \leq j + c$, the following inequality holds: $LB\_rotation(Q, C) \leq DTW(Q, C)$.*

Proof. Sketch: $\forall c_i \in C, 1 \leq i \leq n$, it may match the points in the range $Q[i-c, i+c]$, and its contribution to LB_rotation is $d_i = \min_{s_j \in \mathcal{S}_i}\{d(c_i, Env(s_j))\}$. Suppose there are m segments of Q intersecting with this range, and the real matching point q_i belongs to segment s_k, then $d_i \leq d(c_i, Env(s_k))$. Based on Equation (4) we have $d(c_i, Env(s_k)) \leq d(c_i, p_j), \forall p_j \in s_k \wedge j \in [i-c, i+c]$. By transitivity, $d_i \leq d(c_i, q_i)$. Since $DTW(Q,C) \geq \sum_{i=1}^{n} d(c_i, q_i) \geq \sum_{i=1}^{n} \min_{s_j \in \mathcal{S}_i}\{d(c_i, Env(s_j))\} = LB\_rotation(Q,C)$, we can conclude that LB_rotation lower bounds DTW.

3.3 Detailed Steps of LB_rotation

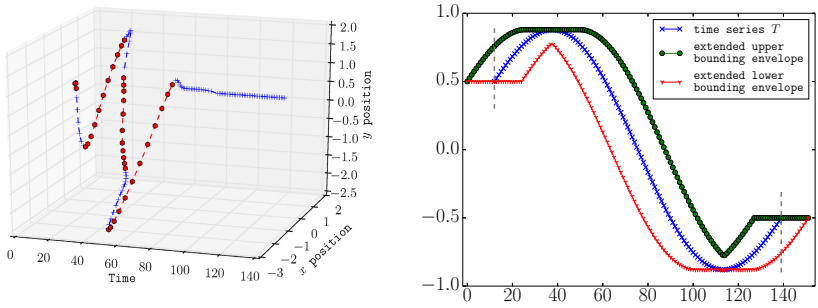
It takes 4 steps to compute LB_rotation:

1. **Time series segmentation**. As noted in Section 3.1, if we want to achieve satisfactory lower bound via time series rotation, we need to apply segmentation on the query time series, then deal with each segment respectively. We want each segment to be as straight as possible, so intuitively we should partition the time series at those "turning points". The classic Douglas-Peucker algorithm [6] is used for the segmentation, since each resulted splitting point is exactly such a turning point.
   We demonstrate the result of segmentation in Figure 2a, where a time series extracted from the *Character Trajectories* dataset is divided into 8 segments. We can see that each segment is almost straight, with different length.
2. **Segment rotation**. After segmentation, we need to find the rotation angle that best reduces the volume of envelopes. For each segment, we use least square fitting to compute the direction of the corresponding major axis, then we rotate each point $p \in s$ around the origin by $-s.\theta$ to get the rotated segment s', where $s.\theta$ is the included angle between the major axis and the x axis. Thus after rotation, the major axis of s' is aligned with the x axis, and the points in the time series segment will have a smaller span around the x axis, which leads to a narrower envelope.
3. **Extended envelope computation**. The next step is to compute the envelope for each rotated segment of query time series Q, which is almost the same as the envelope computation of LB_Keogh. The only difference is that, in the original envelope, each point will cover at least c points of the time series, however, after segmentation, the matching range may only intersect with the beginning or ending $k (1 \leq k \leq c)$ points of a certain segment. Covering extra points will hurt the tightness of LB_rotation.
   To solve this problem, we pad c points at the start and end of segment s respectively. When computing the upper bounding envelope, we fill the first and last c points of the padded s with a value that is smaller than all the values in s (e.g. $-\infty$), while fill with a value that is larger than all the values in s (e.g. $+\infty$) when computing the lower bounding envelope.
   We illustrate the extended envelope in Figure 2b. The original envelopes are between the two dashed vertical lines, while the extended parts lie outside.
4. **Lower bound computation**. Now we have a series of rotated segments of the query time series Q, we will describe for a candidate time series C in the database, how to compute $LB\_rotation(Q,C)$ using these segments.

First, we need to find the corresponding matching point for the points in C, in order to apply Equation (4). Given a point $c_i \in C$, it may match any point in $Q[i-c, i+c]$ with respect to a warping constraint c. Since Q is divided into a series of segments, the points in $Q[i-c, i+c]$ may belong to different segments, thus we should take care of different conditions. Specifically, if a segment s intersects with $Q[i-c, i+c]$, there are four possible situations.

(a) $(s.start \leq i-c) \wedge (s.end \geq i+c)$, i.e. s contains $Q[i-c, i+c]$.
(b) $(s_j.start \geq i-c) \wedge (s_j.end \leq i+c)$, i.e. $Q[i-c, i+c]$ contains s.
(c) $(s_j.start \leq i+c) \wedge (s_j.end > i+c)$, i.e. $Q[i-c, i+c]$ contains the head of s.
(d) $(s_j.start < i-c) \wedge (s_j.end \geq i-c)$, i.e. $Q[i-c, i+c]$ contains the tail of s.

The index of corresponding matching point is computed as in Algorithm 2, which gives the final procedures of LB_rotation.



(a) **(b)**

Fig. 2. (a) The segmentation result of a time series from *Character Trajectories* dataset, with $m = 8$. (b) The extended envelope of a time series T with length $n = 128$, and warping constraint $c = 0.1n$.

3.4 Performance Analysis

We briefly analyze the time complexity of each step in Section 3.3.

1. **Time series segmentation**. For a time series of length n, the Douglas-Peucker algorithm costs on average $O(n \log n)$, and $O(n^2)$ in the worst case.
2. **Segment rotation**. For a time series segment of length k, it costs $O(k)$ to compute the inclination angle of its major axis, and $O(k)$ to rotate each point. So it costs totally $\sum_{i=1}^{m} O(k_i) = O(n)$ in this step.

Algorithm 2. Lower bound computation for LB_rotation

Input: $\{\mathcal{S}_i\}$: each $\mathcal{S}_i$ contains rotated segments of the query time series Q that intersect with $Q[i-c, i+c]$
Input: C: candidate time series in the database
Output: d: the lower bound distance

function LB_ROTATION($\{\mathcal{S}_i\}$, C)
  $d \leftarrow 0$;
  for $c_i \in C$
    $dist_{min} \leftarrow \infty$
    for $s_j \in \mathcal{S}_i$
      if $(s_j.start \leq i-c) \wedge (s_j.end \geq i+c)$
        $index \leftarrow i + c - s_j.start$
      else if $(s_j.start \geq i-c) \wedge (s_j.end \leq i+c)$
        $index \leftarrow c + (s_j.end - s_j.start)/2$
      else if $(s_j.start \leq i+c) \wedge (s_j.end > i+c)$
        $index \leftarrow i + c - s_j.start$
      else if $(s_j.start < i-c) \wedge (s_j.end \geq i-c)$
        $index \leftarrow s_j.end - i$
      $c'_i \leftarrow c_i$ rotated by $-s_j.\theta$ ▷ $s_j.\theta$ is the inclination angle of the major axis of s_j
      $t \leftarrow d(c'_i, Env(s_j)_{index})$ ▷ Equation (4)
      if $dist_{min} > t$
        $dist_{min} \leftarrow t$
    $d \leftarrow d + dist_{min}$
  return d

3. **Extended envelope computation**. For a time series segment of length k, it costs $O(k+2ck/n)$ to compute the extended envelope using the streaming algorithm introduced in [11], so in total $\sum_{i=1}^{m} O(k_i + 2ck_i/n) = O(n+2c)$.
4. **Lower bound computation**. With warping constraint c, and the number of segment m, on average the matching range of each point will cover $\min\{m, \lceil 2cm/n \rceil\}$ segments, thus the time complexity of LB_rotation is asymptotically $O(\min\{m, \lceil 2cm/n \rceil\}n)$.

The first three steps can be precomputed before entering the `for` loop in Algorithm 1 of Algorithm 1, so the cost will be amortized. If there are enough candidate time series, this amortized overhead is negligible, just as what we observed in the experiment. While for the last step, as m is generally fixed, the time complexity increases with c. However, since generally LB_rotation will produce tighter lower bound, it requires fewer expensive DTW computations, thus the overall time needed to perform nearest neighbor search will be reduced.

Because the actual performance of all the lower bounding techniques is data-dependent, we only give a rough analysis here, and compare LB_Keogh, LB_Lemire and LB_rotation through experiments on different datasets.

4 Experiment

4.1 Setup and Datasets

We implemented the algorithms in C++, compiled by g++ 4.9.1. The platform is a ThinkPad X220 running Arch Linux, with 8GB of RAM and a 2.6GHz Intel Core i7 processor.

We use two real world datasets for experiments.

- The *Character Trajectories*[1] dataset from the UCI Machine Learning Repository [2], which contains 2858 trajectories of writing 20 different letters with a single pen-down segment. The length of each trajectory varies between 109 and 205, and we rescaled them to the same length of 128, using Piecewise Aggregate Approximation [8] (for longer trajectories) or linear interpolation (for shorter trajectories).
- The *GeoLife*[2] dataset [22–24] from MSRA, which contains 17,621 trajectories of 182 users in a period of over three years. We extracted those trajectories containing at least 1000 sample points for experiment, and rescaled them to length 256.

The time series in both datasets are all z-normalized [13]. We assume the datasets are already loaded into memory before running following experiments, and the true DTW distance is computed using the standard $O(mn)$ dynamic programming algorithm, subjected to the corresponding warping constraint c.

The compiled executable and preprocessed datasets are freely available at [1], including the `python` script to compute the accuracy of 1 Nearest Neighbor classification on *Character Trajectories* dataset.

4.2 Evaluation Metrics

The effectiveness of a lower bound is usually reflected in the tightness, pruning power and the overall wall clock time. The first two metrics are independent of implementation details, while the last one may vary. Nonetheless, the wall clock time is still an important metric, since although some lower bound may be tighter than others, it actually will cost much more time to compute, which largely nullifies its effectiveness [19].

Following [9], we define the tightness of a lower bound as

$$T = \frac{\text{Lower Bound of DTW Distance}}{\text{True DTW Distance}} \tag{6}$$

and define pruning ratio as

$$P = \frac{\text{Number of Omitted Full DTW Computation}}{\text{Number of Objects}}. \tag{7}$$

Both T and P are in the range $[0, 1]$, and the larger the better.

[1] http://archive.ics.uci.edu/ml/datasets/Character+Trajectories
[2] http://research.microsoft.com/en-us/projects/geolife/

To evaluate the tightness of each lower bounding method, we randomly sampled 100 time series from the dataset, then computed the three lower bounds as well as the true DTW distance for each pair of them (in total 9900 pairs), and recorded the corresponding tightness. The average over 9900 pairs is reported. Note that we have to compute the lower bound between each pair, since these lower bounds are not symmetric, i.e. $LB(T_1, T_2) = d \not\Leftrightarrow LB(T_2, T_1) = d$.

To evaluate the pruning power of each lower bounding method, we randomly sampled 100 time series from the dataset, then for each time series, we performed 1-Nearest Neighbor search on the rest 99 time series, using Algorithm 1, by plugging in each lower bounding method in Algorithm 1, and recorded the corresponding pruning ratio. The average over 100 time series is reported.

To evaluate the efficiency of each lower bounding method, for each dataset, we randomly sampled 1000 time series from it, then performed 1-Nearest Neighbor (1NN) search for 50 randomly sampled time series from the same dataset, using Algorithm 1, by plugging in each lower bounding method in Algorithm 1. In order to rule out the influence of random factors, the 1NN search time for each time series is reported as the average over 10 runs. We repeated above experiments with various parameter combinations.

4.3 The Effect of Segment Number m

First, we inspect how the number of segments will affect the tightness and pruning power of LB_rotation. Since this parameter is only used in LB_rotation, we do not compare with the other two lower bounding methods.

We randomly sample 1000 trajectories from the *GeoLife* dataset, then compute pair-wise lower bound using LB_rotation as well as the true DTW distance, and record the corresponding tightness. The pruning ratio and 1NN search time are gathered through 1NN search for 50 random sampled time series. The averages are reported in Figure 3. For the *Character Trajectories* dataset, we observe similar results, and we only report one of them for brevity.

We can see that the tightness and pruning power increases with m, however, for larger m the 1NN search time becomes longer, because the saved DTW computation cannot break even the time needed to compute LB_rotation. We empirically find that $m = 8$ achieves a good compromise between the pruning ratio and extra computation overhead. In the following, if not otherwise stated, we set $m = 8$ for all the experiments.

4.4 The Effect of Warping Constraint c on Tightness and Pruning Power

In the following, we present the tightness and pruning power of the three lower bounding techniques, with respect to varying warping constraint, on different datasets. The warping constraint c varies from 0 (corresponding to Euclidean distance) to n (corresponding to unconstrained DTW distance), with step size $0.05n$. The results are presented in Figure 4a through Figure 4d.

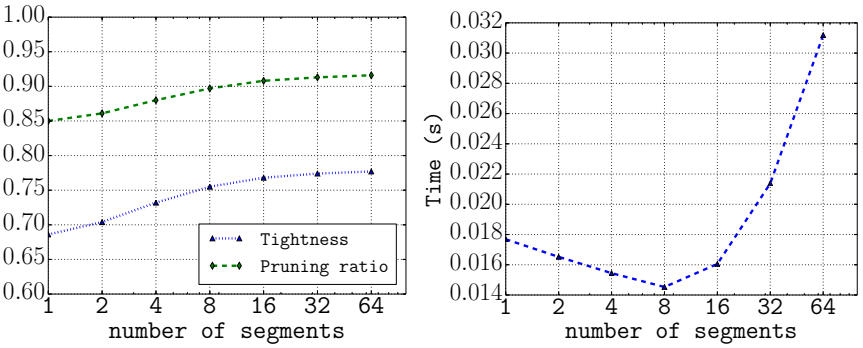


Fig. 3. Tightness, pruning power and 1NN search time vs. number of segments on *GeoLife* dataset. Warping constraint $c = 0.1n$.

First of all, we need to point out that for the *Character Trajectories* dataset, under the optimal constraint $c = 0.4n$, LB_rotation is 2× as tight as LB_Keogh, and 1.3× as tight as LB_Improved. It prunes 30% and 16% more unqualified candidates respectively when compared with LB_Keogh and LB_Improved. This optimal constraint is obtained by testing different warping constraint on the *Character Trajectories* dataset, since the trajectories are labeled, they can be used to test the classification accuracy. We used 1-Nearest Neighbor classification, and validated the result by leave-one-out validation. As for the *GeoLife* dataset, due to the lack of labels, we cannot decide the optimal warping constraint for 1NN classification, however we noticed there are trajectories that are almost identical, while largely shifted along the time axis (about half of the trajectory length), which indicates that large warping constraint should be used to correctly align these trajectories.

From Figure 4a through Figure 4d we can observe that LB_rotation consistently achieves higher tightness and pruning power than LB_Keogh, which demonstrates the effectiveness of time series rotation.

For $c \geq 0.4n$, LB_rotation outperforms LB_Improved in terms of both tightness and pruning ratio. Because as c increases, the volume of the envelope will also increase, since it will cover more data points, and intuitively, the probability of including points with large values is proportional to the covering range of the envelope, thus the envelope will be enlarged. On the other hand, if we rotate each segment respectively, recall Figure 1, even the covering range increases, the volume of the envelope won't increase much. As a result, although larger c will hurt the tightness and pruning ratio of all the three lower bounds, the influence on LB_rotation is obviously smaller.

When c is relatively small ($< 0.4n$), LB_rotation generally achieves comparable or even higher tightness than LB_Improved, although the pruning ratio of the latter is sometimes better. This is because with small warping constraints, LB_Keogh has considerably good tightness, thus it requires only a few second pass computations for LB_Improved, which will not improve the tightness much.

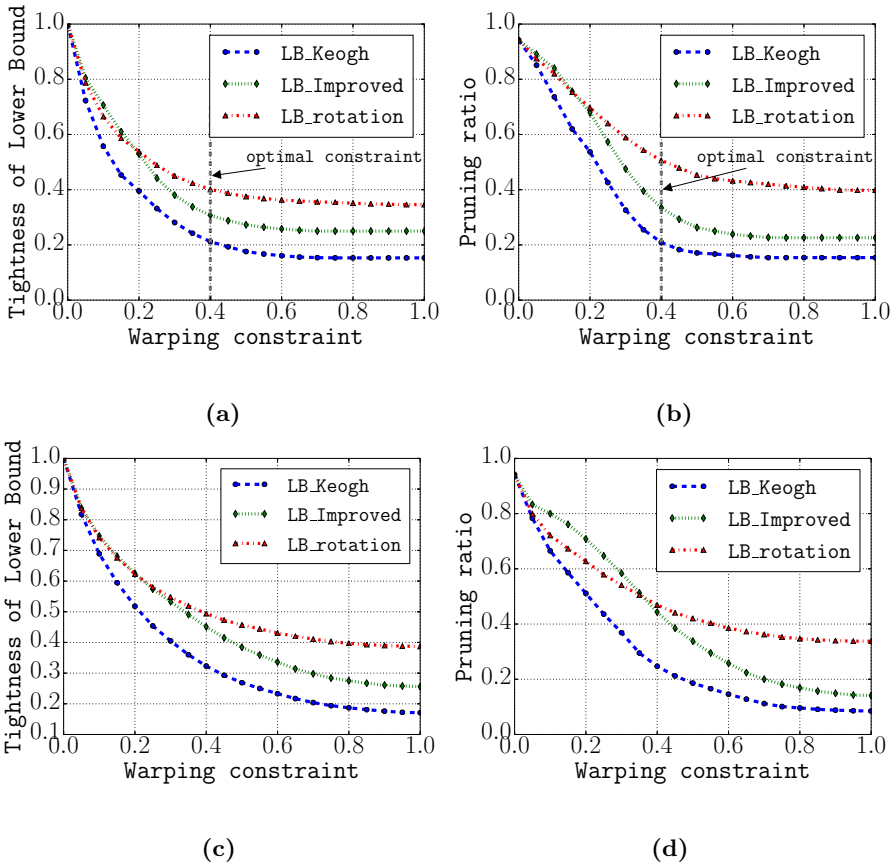


Fig. 4. Tightness and pruning ratio w.r.t. varying warping constraint c on different datasets. (a)-(b): *Character Trajectories* dataset; (c)-(d): *GeoLife* dataset.

However, these second pass computations do help to prune more candidates, so the pruning ratio of LB_Improved will increase.

We also note that even in the extreme situation where $c = n$ (unconstrained DTW), LB_rotation can still achieve pruning ratio around 40%, while both LB_Keogh and LB_Improved have pruning ratio hardly exceeds 20%.

4.5 The Effect of Warping Constraint c on Search Time

In this experiment, we compare the wall clock time for 1NN search on different datasets, with respect to varying warping constraint from 0 to n, increasing at step size $0.05n$.

From Figure 5a and Figure 5b we find that the 1NN search time agrees with tightness and pruning ratio we just depicted in Section 4.4 very well. The result indicates that on all the datasets, LB_rotation will achieve the fastest 1NN

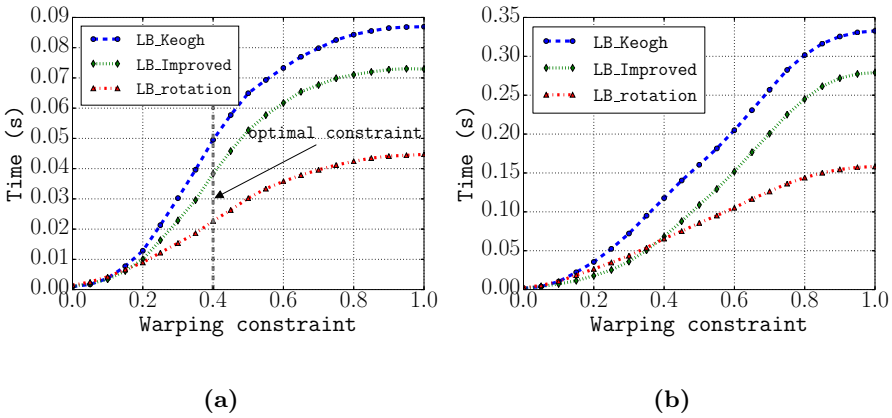


Fig. 5. 1NN search time w.r.t. varying warping constraint c on (a) *Character Trajectories* dataset; (b) *GeoLife* dataset

search once the warping constraint c exceeds $0.4n$. We have shown that warping constraints this large are realistic for some real world applications. Figure 5a shows that for the *Character Trajectories* dataset, under the optimal warping constraint $c = 0.4n$, LB_rotation costs about half the time of LB_Keogh, and about 60% of LB_Improved. Actually this trend starts once c exceeds $0.2n$.

5 Conclusion and Future Work

In this paper, we propose a new lower bounding technique *LB_rotation* for constrained DTW, which is based on the observation that if the time series is in multi-dimensional space, it can be rotated around the time axis to reduce the volume of its bounding envelope, and as a consequence, the tightness and pruning power of the lower bound will increase. Then we notice that if we divide the time series into several segments as straight as possible, then treat them separately, the effectiveness can be further improved, so we use a greedy algorithm to achieve this. We carried out experiments on real world datasets, which demonstrate the superiority of LB_rotation over state-of-the-art lower bounding techniques.

With more and more high dimensional time series being generated nowadays, it is of significant importance to effectively process them. In the future, we intend to further investigate how to utilize the characteristics of multi-dimensional time series to achieve even better result.

Acknowledgments. This work was supported by National Natural Science Foundation of China under Grant No.61202404, No.61170233, No.61232018, No.61272472, No.61272317 and the Fundamental Research Funds for the Cerntral Universities, No. WK0110000041.

References

1. Executable and datasets used in the experiment. https://www.dropbox.com/s/gkmcy9up73y5vmo/data.tar.gz?dl=0
2. Bache, K., Lichman, M.: UCI machine learning repository (2013). http://archive.ics.uci.edu/ml
3. Chen, L., Ng, R.: On the marriage of lp-norms and edit distance. In: Proceedings of the Thirtieth International Conference on Very Large Data Bases - Volume 30, VLDB 2004, pp. 792–803. VLDB Endowment (2004)
4. Chen, L., Özsu, M.T., Oria, V.: Robust and fast similarity search for moving object trajectories. In: Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data, SIGMOD 2005, pp. 491–502. ACM, New York (2005)
5. Das, G., Gunopulos, D., Mannila, H.: Finding similar time series. In: Komorowski, J., Zytkow, J. (eds.) Principles of Data Mining and Knowledge Discovery. LNCS, vol. 1263, pp. 88–100. Springer, Heidelberg (1997)
6. Douglas, D.H., Peucker, T.K.: Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. Cartographica: The International Journal for Geographic Information and Geovisualization **10**(2), 112–122 (1973)
7. Faloutsos, C., Ranganathan, M., Manolopoulos, Y.: Fast subsequence matching in time-series databases. SIGMOD Rec. **23**(2), 419–429 (1994)
8. Keogh, E., Chakrabarti, K., Pazzani, M., Mehrotra, S.: Dimensionality reduction for fast similarity search in large time series databases. Knowledge and Information Systems **3**(3), 263–286 (2001)
9. Keogh, E., Ratanamahatana, C.A.: Exact indexing of dynamic time warping. Knowledge and Information Systems **7**(3), 358–386 (2005)
10. Kim, S.-W., Park, S., Chu, W.: An index-based approach for similarity search supporting time warping in large sequence databases. In: Proceedings of the 17th International Conference on Data Engineering, 2001, pp. 607–614 (2001)
11. Lemire, D.: Faster retrieval with a two-pass dynamic-time-warping lower bound. Pattern Recognition **42**(9), 2169–2180 (2009)
12. Li, Q., Zheng, Y., Xie, X., Chen, Y., Liu, W., Ma, W.-Y.: Mining user similarity based on location history. In: Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, GIS 2008, pp. 34:1–34:10. ACM, New York (2008)
13. Rakthanmanon, T., Campana, B., Mueen, A., Batista, G., Westover, B., Zhu, Q., Zakaria, J., Keogh, E.: Searching and mining trillions of time series subsequences under dynamic time warping. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2012, pp. 262–270. ACM, New York (2012)
14. Rath, T.M., Manmatha, R.: Lower-bounding of dynamic time warping distances for multivariate time series. Technical Report MM-40, Center for Intelligent Information Retrieval, University of Massachusetts Amherst (2002)
15. Sefidmazgi, M.G., Sayemuzzaman, M., Homaifar, A.: Non-stationary time series clustering with application to climate systems. In: Jamshidi, M., Kreinovich, V., Kacprzyk, J. (eds.) Advance Trends in Soft Computing WCSC 2013. STUDFUZZ, vol. 312, pp. 55–63. Springer, Heidelberg (2014)
16. Vlachos, M., Gunopulos, D., Das, G.: Rotation invariant distance measures for trajectories. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2004, pp. 707–712. ACM, New York (2004)

17. Vlachos, M., Hadjieleftheriou, M., Gunopulos, D., Keogh, E.: Indexing multidimensional time-series with support for multiple distance measures. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2003, pp. 216–225. ACM, New York (2003)
18. Wang, J., Katabi, D.: Dude, where's my card?: RFID positioning that works with multipath and non-line of sight. In: Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM, SIGCOMM 2013, pp. 51–62. ACM, New York (2013)
19. Wang, X., Mueen, A., Ding, H., Trajcevski, G., Scheuermann, P., Keogh, E.: Experimental comparison of representation methods and distance measures for time series data. Data Mining and Knowledge Discovery **26**(2), 275–309 (2013)
20. Yang, A.Y., Jafari, R., Sastry, S.S., Bajcsy, R.: Distributed recognition of human actions using wearable motion sensor networks. Journal of Ambient Intelligence and Smart Environments **1**(2), 103–115 (2009)
21. Yi, B.-K., Jagadish, H., Faloutsos, C.: Efficient retrieval of similar time sequences under time warping. In: Proceedings of the 14th International Conference on Data Engineering, 1998, pp. 201–208 (1998)
22. Zheng, Y., Li, Q., Chen, Y., Xie, X., Ma, W.-Y.: Understanding mobility based on GPS data. In: Proceedings of the 10th International Conference on Ubiquitous Computing, UbiComp 2008, pp. 312–321. ACM, New York (2008)
23. Zheng, Y., Xie, X., Ma, W.-Y.: GeoLife: A collaborative social networking service among user, location and trajectory. IEEE Data Eng. Bull. **33**(2), 32–39 (2010)
24. Zheng, Y., Zhang, L., Xie, X., Ma, W.-Y.: Mining interesting locations and travel sequences from GPS trajectories. In: Proceedings of the 18th International Conference on World Wide Web, WWW 2009, pp. 791–800. ACM, New York (2009)
25. Zhou, M., Wong, M.-H.: Boundary-based lower-bound functions for dynamic time warping and their indexing. In: IEEE 23rd International Conference on Data Engineering, ICDE 2007, pp. 1307–1311 (2007)
26. Zhu, Y., Shasha, D.: Warping indexes with envelope transforms for query by humming. In: Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, SIGMOD 2003, pp. 181–192. ACM, New York (2003)