Isomorphism Distance in Multidimensional Time Series and Similarity Search

Guo Wensheng and Ji Lianen

College of Geophysics and Information Engineering, China University of Petroleum-Beijing, Beijing, China

Published online: 1 Feb. 2013

Abstract: Describing the similarity of time series as distance is the basis for most of data mining research. Existing studies on similarity distance is based on the "point distance" without considering the geometric characteristics of time series, or is not a metric distance which doesn’t meet the triangle inequality and can't be directly used in indexing and searching process. A method for time series approximation representation and similar measurement is proposed. Based on the subspace analysis representation, the time series are represented approximately with an isomorphic transformation. The basic concepts and properties of the included isomorphism distance are proposed and proved. This distance overcomes the problem when other non-metric distance is used as the similar measurement, such as the poor robustness and ambiguous concepts. The proposed method is also invariant to translation and rotation. A new pruning method for indexing in large time series databases is also proposed. Experimental results show that the proposed method is effective.

Keywords: Time series, similarity, metric space, data mining

1. Introduction

With the continuous development of the technical level, there are lots of time series data in commerce[1], science[2] and engineering[3]. Such as the sales of commodities in retail, stock price, number of security incidents detected by security facilities deployed in the network, and so on. Analysis of those data will often be relevant to an important issue: how to find the sequence which is similar to the given query from the time series of historical data. For example: In the financial field, people can search the time series similar to the recent stock price changes of one company in the historical time series data, and then predict the future stock price changes according to the historical time series data. Another example is in the field of network security, by looking up the historical time series records similar to recent network traffic and security events, people can identify network security posture and possible attacked events.

In 1993, Agralwal et al. first proposed a total matching algorithm in time series similarity search [4]. Faloutsos et al. who proposed a subsequence matching algorithm [5], promote the application of similarity search. The traditional methods are one-dimensional sequence similarity search, and achieve great success in their respective fields of application [6, 7]. However, with the prevalence and popularity of audio and video equipment and the internet, most of the one-dimensional time series similarity search method does not apply to new data format, so the multi-dimensional similarity search is proposed.

Multidimensional time series, including graphics, images, audio, video and other information, is composed by a set of data vectors change over time. For example: In the financial field, the timing data for Chinese stock index recently points, can be searched not only in their own historical data set, but also can be tried to search for similar subsequence in other countries’ stock index historical data set for policy making [8]. Searching process of timing data in the field of network security, which is formed by the number of hosts controlled by some kinds of Trojan horse in a region, in addition to concentrate in its own Trojan historical data, can also refer to other regions or country to find the similar behavior mode for further analysis and decision-making[9].

Extending similar search to the multidimensional scene can obtain the following two advantages: Firstly, known by the research on the data stream, generally, the recent data is more valuable than long time ago, so recent similar sequence found in multiple dimensions may have...
more value than those obsolete sequence found in one dimension. Secondly, with technology development, a variety of time series data in recent years is gradually increased. Because of this, there may not be sufficient historical data to support applications of similarity search in a single dimension. By searching in multi-dimensional time series data, we can increase the scale of search space to discover more valuable similar historical mode for decision making.

Our contributions are as follows:

We propose a distance function which we call isomorphism distance (ISO). This distance function maintain the geometric characteristics of time series through subspace isomorphic [10, 11]. So it can support local time shifting, and is a metric. We present benchmark results showing that this distance function is natural for time series data.

We propose a new pruning strategy for isomorphism distance, which can be efficiently indexed with a standard B+-tree or other data structure. Given that ISO is a metric distance, we can use the triangle inequality in the pruning process.

We also develop a k-nearest neighbor (k-NN) algorithm that use the isomorphism distance. We give extensive experimental results in Section 5 showing that the algorithm gets the best of pruning power and scalability.

The rest of the paper is organized as follows. In Section 2 we introduce the related works about the distance measuring similarity and its indexing structure. In section 3 we present our isomorphism distance model and prove it’s a metric distance. Because of its metric measure, some indexing and pruning algorithm are analyzed in Section 4. The experimental results are presented in Section 5. We conclude our paper and suggest some possible future directions in Section 6.

2. Related Work

Many researches focus on how to search similar sequence fast and accurate in time series database, especially large database once unable to load in memory. It includes how to represent time series, how to measure the similarity between sequences as well as how to index and search in database. The major role of sequence represented is dimensionality. Let

\[ s_1, \ldots, s_m \]

and

\[ t_1, \ldots, t_n \]

have the same nature of difference in these types of distance.

On this basis, assuming that multidimensional time series lie in a linear manifold in the data space, we propose a new distance function, which is called isomorphism distance. This distance function maintain the geometric characteristics of time series through subspace isomorphic. So it can support local time shifting, and is a metric. To begin with, for any two time series \([s_1, \ldots, s_m]\) and \([t_1, \ldots, t_n]\), consider them as two linear subspaces \(S\) and \(T\) in \(\mathbb{R}^d\). Since discuss similarity issue, we first assume that \(S\) and \(T\) have the same dimensionality. Let \(s_1, \ldots, s_p\) and \(t_1, \ldots, t_q\) be standard
Let $S, T$ denote the so-called isomorphism distance from the end point of vector $s_i$ to subspace $T$. That is, 
\[ d(s_i, T) = \min_{t \in T} ||s_i - t|| \]  
(1)

We then define the subspace isomorphism distance $d(S, T)$ for $p$-dimensional subspaces $S$ and $q$-dimensional subspaces $T$ as 
\[ d(S, T) = \sqrt{\sum_{i=1}^{p} d^2(s_i, T)} \]  
(2)

Since $t_1, t_2, \ldots, t_q$ is a standard orthogonal basis of $T$, it is easy to see that 
\[ d(S, T) = \sqrt{\sum_{i=1}^{p} ||s_i||^2 - \sum_{j=1}^{q} (s_i^T t_j)^2} \]  
(3)

With the above-mentioned Distance, we need prove the following properties:

**Theorem 1.** The isomorphism distance defined above is invariant to the choice of standard orthogonal basis.

**Proof:** Let $S_1, S_2, \ldots, S_p$ and $T_1, T_2, \ldots, T_q$ be two standard orthogonal basis of $S$. Let $t_1, t_2, \ldots, t_q$ be a standard orthogonal basis of $T$. To prove the theorem, it suffices to show that 
\[ \sqrt{p - \sum_{i=1}^{p} \sum_{j=1}^{q} (s_i^T t_j)^2} = \sqrt{p - \sum_{i=1}^{p} \sum_{j=1}^{q} (s_i^T t_j)^2} \]  
(4)

Let $P_j^S$ is the projection of $t_j$ onto subspace $S$. By the Parseval equation and the uniqueness of projection, following equality holds: 
\[ \sum_{i=1}^{p} (s_i^T t_j)^2 = \sum_{i=1}^{p} (P_j^S s_i)^2 \]  
(5)

In fact, we can see that for every $j, j = 1, 2, \ldots, q$, above equality always holds. So it completes the proof.

**Theorem 2.** Non-negativity: $0 \leq d(S, T) \leq \sqrt{\max(p, q)}$

The proof of Theorem 2 is immediate.

**Theorem 3.** Symmetry: $d(S, T) = d(T, S)$

The proof of Theorem 3 is immediate.

**Theorem 4.** Triangle Inequality: 
\[ d(S, T) \leq d(S, \Gamma) + d(T, \Gamma) \]

Let $\Gamma = (\gamma_1, \ldots, \gamma_q)$ be the matrices composed by the orthogonal basis of arbitrary subspaces $S,T,\Gamma$, respectively.

**Lemma 1.** Let $A^H$ denote the conjugate transpose matrix of $A$. The trace of Matrix $A^H A$ and $AA^H$ is equivalent, i.e. 
\[ tr(A^H A) = tr(AA^H) = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} a_{ji} \]  
(6)

The proof of Lemma 1 is immediate according to the properties of trace in [29]. With Lemma 1, we can rewrite the Equation 3 in terms of matrix as follows: 
\[ d(S, T) = \sqrt{\max(p, q) - tr(TT^T SS^T)} \]  
(7)

Then Theorem 4 can be written as a matrix format.

**Lemma 2.** Denote $A_p$ the diagonal matrix, which first $p$ diagonal elements are 1, and rest elements are 0, that is, $A_p = \text{diag}(1, \ldots, 1, 0, \ldots, 0)$.

\[ \max(p, q) = tr(A_p + A_q - A_p A_q) \]

The proof of Lemma 2 is immediate.

**Lemma 3.** If we denote $\tilde{S} = (s_1, \ldots, s_p, \ldots, s_d)$ the orthogonal basis matrix of $\mathbb{R}^d$ extended from $s_1, \ldots, s_p$, then 
\[ (SS^T) = \tilde{S} A_p S^T \]

The proof of Lemma 3 is immediate.
Lemma 4. If we denote:

\[ A = \text{tr}[(A_p - A_r)(A_q - A_r) + (M - A_r)(N - A_r)] \]
\[ B = \text{tr}[(M - A_r)^2 + (A_p - A_r)^2] \]
\[ C = \text{tr}[(N - A_r)^2 + (A_p - A_r)^2] \]

Suppose that matrices \( U, V \) meet the following conditions: \( S = \tilde{\Gamma}U, \tilde{T} = \tilde{\Gamma}V \), then Theorem 4 is equivalent to:

\[ A \leq \sqrt{BC} \]  

(9)

Proof. Let \( M = U \Lambda_p U^T, N = V \Lambda_q V^T \). According to Lemma 1 and the orthogonality of \( \tilde{\Gamma} \), we can obtain that:

\[ \text{tr}(TT^TSS^T) = \text{tr}(\tilde{\Gamma}V \Lambda_q V^T \tilde{\Gamma}U \Lambda_p U^T \tilde{\Gamma}) = \text{tr}( \tilde{\Gamma}V \Lambda_q V^T \Lambda_p \tilde{\Gamma}) = \text{tr}( V \Lambda_q V^T \Lambda_p ) = \text{tr}(NM) \]

and similarly,

\[ \text{tr}(TT^TT^T) = \text{tr}(\tilde{\Gamma}V \Lambda_q V^T \tilde{\Gamma} \Lambda_p \tilde{\Gamma}) = \text{tr}( V \Lambda_q V^T \Lambda_p ) = \text{tr}(NM) \]

\[ \text{tr}(SS^TT^T) = \text{tr}(\tilde{\Gamma}U \Lambda_p U^T \tilde{\Gamma} \Lambda_q \tilde{\Gamma}) = \text{tr}( U \Lambda_p U^T \Lambda_q ) = \text{tr}(MA_r) \]

According to Equation 7 and Lemma 1 we can obtain:

\[ d(S, T) = \sqrt{\text{tr}(A_p + A_q - A_p A_q - NM)} \]
\[ d(S, \Gamma) = \sqrt{\text{tr}(A_p + A_r - A_p A_r - MA_r)} \]
\[ d(T, \Gamma) = \sqrt{\text{tr}(A_q + A_r - A_q A_r - NA_r)} \]

Therefore, Theorem 4 is equivalent to

\[ d^2(S, T) \leq d^2(S, \Gamma) + d^2(T, \Gamma) + 2 \cdot d(S, \Gamma) \cdot d(T, \Gamma) \]

That is

\[ d^2(S, T) - d^2(S, \Gamma) - d^2(T, \Gamma) \leq 2 \cdot d(S, \Gamma) \cdot d(T, \Gamma) \]

Let left side of this inequality as \( LS \):

\[ LS = \text{tr}[(A_p + A_q - A_p A_q - NM) - (A_p + A_r - A_p A_r - MA_r - NA_r) + (MA_r - A_r A_q + N(A_q - A_r)) + (A_r(A_q - A_r) + (S - A_r)(T - A_r)) \]

Let right side of this inequality as \( RS \):

\[ RS = \sqrt{\text{tr}(2A_p + 2A_r - 2A_p A_r - 2MA_r)} \cdot \sqrt{\text{tr}(2A_q + 2A_r - 2A_p A_r - 2NA_r)} \]

From \( A^2 = A, A^2 = A \), we get

\[ \text{tr}(2A_p + 2A_r - 2A_p A_r - 2MA_r) = \text{tr}(A_p^2 + M^2 + A_r^2 - 2A_p A_r - 2MA_r) \]
\[ = \text{tr}((M - A_r)^2 + (A_p - A_r)^2) \]

So

\[ RS = \sqrt{\text{tr}((M - A_r)^2 + (A_p - A_r)^2)} \cdot \sqrt{\text{tr}(2A_q + 2A_r - 2A_p A_r - 2NA_r)} \]

So Lemma 4 completes the proof. Now let’s proof the Theorem 4: Set

\[ a_1 = \text{vec}(M - A_r) = \text{tr}((M - A_r)^2); \]
\[ a_2 = \text{vec}(A_p - A_r) = \text{tr}((A_p - A_r)^2); \]
\[ a_3 = \text{vec}(N - A_r) = \text{tr}((N - A_r)^2); \]
\[ a_4 = \text{vec}(A_q - A_r) = \text{tr}((A_q - A_r)^2); \]

where \( \text{vec}(A) \) indicates the vector that span with all of the matrix \( A \) ‘s column vectors head to tail. According to above lemmas and Cauchy-Schwarz inequality, Theorem 4 is equivalent to

\[
(a_1^2 a_3 + a_2^2 a_4)^2 \leq (||a_1|| ||a_3|| + ||a_2|| ||a_4||)^2 \\
\leq (||a_1||^2 + ||a_2||^2)(||a_3||^2 + ||a_4||^2)
\]

The whole proof is completed.

As mentioned above, the symmetry and non-negativity of isomorphism distance can be seen easily. Particularly, together with matrix analysis techniques, we show the triangular inequality of ISO. Therefore, it is proved to be a metric distance undoubtedly. The measure of the difference between different time series is the basis of many machine learning algorithm. Since it is proved to be distance, isomorphism distance becomes a natural distance measure to characterize the similarity between time series.

Specific solution of isomorphism distance is borrowed from the implementation in [30,31]: Firstly, according to solution idea of LDA algorithm, we can transform the solution process into the following generalized linear equation of eigenvalue and eigenvector problem. Secondly, assumptions to obtain the eigenvalues in ascending order, select the eigenvectors \( s_1, s_2, \ldots, s_p \) corresponding to the first \( p \) (generally \( p < m \)), and then carry out the Gram-schmidt orthogonalization on \( s_1, s_2, \ldots, s_p \) to meet the orthogonality.
4. Indexing and Searching

Recall from Figure 3 that isomorphism distance has the same computational behavior with EDR and DTW. It takes $O(mn)$ time to compute the distance for time series $S$, $T$ of length $m$, $n$ respectively. For large time series databases, it is important for a given query $Q$, we try to minimize the computation of the true distance between $Q$ and $S$ to measure the similarity of them for all sequence $S$ in the database. The topic explore here is indexing and searching for k-NN (the k-Nearest Neighbor) query algorithm. An extension to the range queries is rather straightforward. so we omit its details. Given that isomorphism distance is a metric distance function, one obvious way to prune is to apply the triangle inequality. Metric or not, another common way to prune is to apply the GEMINI framework of Faloutsos et al. - that is, using lower bounds to guarantee no false dismissals. In fact, virtually all approaches to indexing time series under the Euclidean distance do that[13,32,33]. In this section, we can use a new solution to index and search similarity time series with isomorphism distance. The beauty of isomorphism distance is that it can be indexed by a simple B+tree or R-tree.

4.1. Pruning by the Triangle Inequality

The algorithm 1 shows a skeleton of how the Triangle inequality is applied. $S$ is the current time series, while $Q$ is the query time series. The two-dimensional array $matrix$ is used to stored the precomputed pairwise distance between two time series. The array $queue$ is the array of time series with computed true distance to $Q$. It means that if the isomorphism distance $ISO(Q,R_i)$ of time series $\{R_1,\ldots,R_n\}$ has been computed, it will be stored in $queue$. For time series $S$ which is currently being evaluated, the triangle inequality ensures that $ISO(Q,S) \geq ISO(Q,R_i) - ISO(R_i,S)$, for all $1 \leq i \leq n$. Thus, it is necessary that

$$ISO(Q,S) \geq \max_{1 \leq i \leq n} \{ISO(Q,R_i) - ISO(R_i,S)\}$$

If the calculated result $maxPruneDist$ is even worse than the current k-NN distance stored in $result$, $S$ can be skipped entirely. Otherwise, the true distance $ISO(Q,S)$ is computed, and $queue$ array is updated if necessary to reflect the current k-nearest neighbors and distances in stored order.

For large databases, the algorithm 1 makes two assumptions. Firstly, the matrix $matrix$ must be enough small to complete be loaded in memory. This may not be able to meet this condition for large databases. Secondly, the larger the size of $queue$, the more time series can be used for pruning. In the next section, we’ll make a detailed description of how to determine the specific size of $matrix$ and $queue$.

Algorithm 1: TrianglePruning($S,Q,k,queue,matrix$)

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S$, $Q$, $k$, $queue$, $matrix$</td>
<td>$result$</td>
</tr>
</tbody>
</table>

1. $maxPruneDist = 0$;
2. for $i = 1$ to $queue.length$ do
3. if $queue[i].dist - matrix[i][S] > maxPruneDist$ then
4. $maxPruneDist = queue[i].dist - matrix[i][S]$;
5. end
6. end
7. $best = result[k].dist$;
8. if $maxPruneDist < best$ then
9. $dist = ISO(Q,S)$;
10. $insert S and dist into queue$;
11. if $dist < best$ then
12. $insert S and dist into result$ and sort in order to ISO distance;
13. end
14. end

4.2. Multidimensional KNN Search

A number of similarity search speed-up techniques also use indexing structures(e.g., R-trees in [34] or sequential structures in [35]). As isomorphism distance is independent of any underlying indexing approach, it can get efficiency benefit from those data structures. Algorithm 2 shows a skeleton of the algorithm for using the B+-tree index for k-NN search. It first conduct a standard search for $||Q||_{ISO}$ in the tree which is structured in B+-tree. The search result is a leaf node $L$. The first $k$ time series pointed to by $L$ are used to initialize the result array. Next we make a traverse operation in the tree. All the data values bigger than $||Q||_{ISO}$ are visited in ascending order. Similarity, all the data values smaller than $||Q||_{ISO}$ are visited in descending order. If the current computed distance is smaller than the best one stored, the $queue$ will be updated if necessary. Otherwise, the remaining data values can be skipped entirely.

5. Experiments

In this section, we verify the validity of the proposed approach with a comprehensive set of experiments. All experiments were executed on AMD Athlon 64 PC 3600+ (2.09GHz), 1GB memory size, CentOS 6.4 operating system and running JAVA implementations. We used Euclidean distance, DTW, DTW with anticipatory pruning (AP,DTW) [36] as well as isomorphism distance to measure the similarity search’s time cost and effect. If using Euclidean distance, we can take advantage of the multidimensional space index[5,37] to speed up the search. Because the DTW distance does not meet the triangle inequality, it is not possible to use a similar indexing techniques. Here we use sequential scan and sliding window to match the subsequence. The
Algorithm 2: KNNSearch($Q,k$, tree)

Input: $Q$, $k$, tree
Output: result

1. conduct a standard B+-tree search on tree using $||Q||_{ISO}$ and let $L$ be the leaf node which the search ends up with;
2. pick the first $k$ time series as $init_k$ to which are pointed by $L$ and initialize result with those sequences’ isomorphism distance;
3. let $v_1, ..., v_h$ be the data values in all leaf nodes larger than $init_k$, $v_1, ..., v_h$ are sorted in ascending order;
4. initialize queue,matrix;
for $i = 1$ to $h$
5. pick all $l$ time series as array $S$ to which are pointed by $v_i$;
6. TrianglePruning($S, Q, k, queue, matrix$);
end

for $i = 1$ to $j$
7. best = result[$k$].dist;
if ($|w_j| - ||Q||_{ISO} > best$) then
8. pick all $l$ time series as array $S$ to which are pointed by $v_j$;
9. for $j = 1$ to $l$
10. dist = ISO($Q, S[j]$);
11. if dist < best then
12. insert $S[j]$ and dist into result that is sorted in descending order of isomorphism distance;
best = result[$k$].dist
end
end
else
14. break;
end
return result;

Experimental data is an earthquake data sets provided by eamonn(http://www.stat.pitt.edu/stoffer/tsa3/).

5.1. Pruning Power

Given a $k$-NN query $Q$, the pruning power is defined to be the fraction of the time series $S$ in the data set that can be skipped. Follow [34,38], we measure pruning power($P$) because this is an indicator nothing to do with implementation details. To compare the pruning power of those four distance under consideration, we measure $P$ as follow:

$$P = \frac{N_{skipped}}{N_{alt}}$$ (17)

The results shows the pruning power of Euclidean distance, AP_DTW and isomorphism distance on the 16 benchmark data sets for $k =1,5,20$. All metric distance such as Euclidean and isomorphism distance is more efficient at pruning than other DTW-like algorithm. This is due to the DTW-like distances don’t meet the triangle inequality. On average, it was able to prune 1.31 times when $k =1$, 1.95 times when $k =5$ and 1.96 times when $k =20$. Once again, however the most obvious result is the dominance of ISO distance. It wins on most data sets and is able to prune 1.39 times as many items as Euclidean, 2.52 times as many items as DTW and 1.59 times as many items as AP_DTW.

5.2. Database Size

In order to verify the algorithm scalability on massive data sets, we’ll expect the fraction of pruned sequences to increase on larger data sets. The reason is because the larger the data set, the greater the chance there is of a good match being found, and we are able to prune a larger fraction of the data. To demonstrate this effect, we run the same experiment above on increasingly larger time series data set. The results are shown in Figure 5.
5.3. Length of Time Series

Our next study empirically validates the scalability of isomorphism pruning with respect to the length of time series. Figure 6 shows algorithm’s performance gains scale very well with the length of the time series.

5.4. Number of Nearest Neighbors

During the experiment, we also evaluate the number of nearest neighbors which have an influence on pruning. The facts show that, with the increase of parameters k values, the pruning effect of the algorithm is not a significant drop. This result is consistent with the performance of another metric distance, Euclidean distance.

6. Conclusion and Outlook

Distance measure between time series is the basis for further study of the time series data mining tasks. Looking for a good distance measure has a crucial importance for improving the efficiency and accuracy of these data mining tasks. We propose a isomorphism distance measure which can remain the geometric features in high-dimensional space to study the similarity of time series. Our approach is particularly attractive because it is a true metric distance in similarity search. Be compare with Dynamic time warping distance, our approach does not degrade performance, at the same time, the search pruning effect is greatly improved. And compared to the Euclidean distance, our method can better describe the geometric shape of high-dimensional space time series. In the future work, we will attempt to explain the geometric meaning of the time series low-dimensional manifold, research effectively isomorphic transform, compare with other similar distance, and extend it to the multivariate time sequence flow.

Acknowledgement

The author is grateful to the anonymous referee for a careful checking of the details and for helpful comments that improved this paper.

References


Guo Wensheng was born in 1977. He is a lecturer of computer science at the China University of Petroleum, Beijing. He received his Ph.D. degree in University of Science and Technology Beijing in 2007. His research interests are in machine learning, data mining and natural language processing.

Ji Lianen is an associate professor of computer science at the China University of Petroleum, Beijing. His research interests are in virtual reality and scientific visualization, human-computer interaction and software engineering.