Using Multi-Scale Histograms to Answer Pattern Existence and Shape Match Queries over Time Series Data

Lei Chen and M. Tamer Özsu
University of Waterloo
School of Computer Science
{l6chen,tozsu}@uwaterloo.ca

Vincent Oria
New Jersey Institute of Technology
Dept. of Computer Science
oria@njit.edu

Abstract

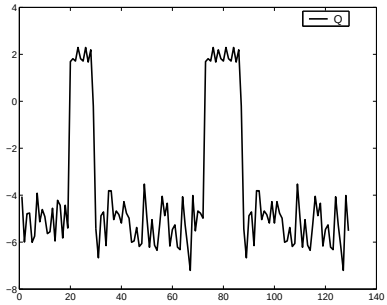
Similarity-based querying of time series data can be categorized as pattern existence queries and shape match queries. Pattern existence queries find the time series data with certain patterns while shape match queries look for the time series data that have similar movement shapes. Existing proposals address one of these or the other. In this paper, we propose multi-scale time series histograms that can be used to answer both types of queries, thus offering users more flexibility. Multiple histogram levels allow querying at various precision levels. Most importantly, the distances of time series histograms at lower scale are lower bounds of the distances at higher scale, which guarantees that no false dismissals will be introduced when a multi-step filtering process is used in answering shape match queries. We further propose to use averages of time series histograms to reduce the dimensionality and avoid computing the distances of full time series histograms. The experimental results show that multi-scale histograms can effectively find the patterns in time series data and answer shape match queries, even when the data contain noise, time shifting and scaling, or amplitude shifting and scaling.

1 Introduction

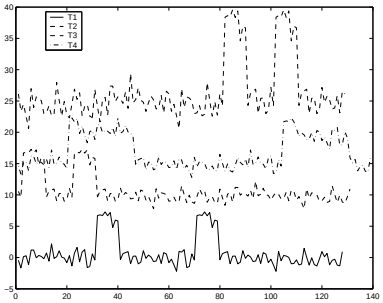
Similarity-based time series data retrieval has been studied in the database and knowledge discovery communities for several years, due to its wide use in various applications, such as financial data analysis [26], content-based video retrieval [17], and musical retrieval [28]. Typically, two types of queries on time series data are studied: pattern existence queries [2, 18, 19], and shape match queries [1, 6, 10].

In pattern existence queries, users are interested in the general pattern of time series data and ignore the specific details. For example,"Give me all the temperature data of patients in last 24 hours that have *two peaks*".

The example query is used to detect "goalpost fever", one of the symptoms of Hodgkin's disease, in the temperature time series data that contain peaks exactly twice within 24 hours [19]. Figure 1(a) shows an example time series with *two peaks*. Using this as query data, a *two peaks* existence query should retrieve various time series that contain *two peaks* as shown in Figure 1(b). Therefore, for pattern existence queries, the important thing is the existence of the specified pattern in time series data, regardless of where the pattern appears and how it appears.



(a) An example query time series with two peaks

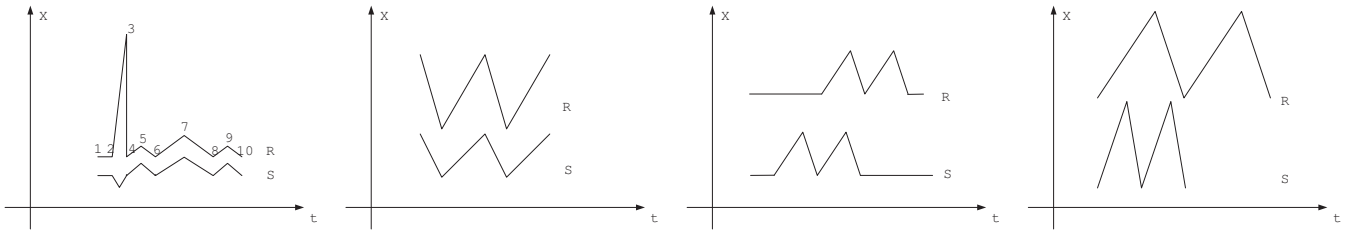


(b) Various time series data containing two peaks

Figure 1. A pattern existence query on two peaks

The retrieval techniques for pattern existence queries should be invariant to the following:

- **Noise.** In Figure 2(a), two similar time series R and S are shown. Point 3, which is likely a noise data point, will introduce a large difference when Euclidean distance is computed between two time series, possibly



(a) Two similar time series with noise (b) Two similar time series with amplitude scaling and shifting (c) Two similar time series with time shifting (d) Two similar time series with time scaling

Figure 2. The different factors that may affect the similarity between two time series

causing them to be treated as dissimilar.

- **Amplitude scaling and shifting.** In Figure 2(b), the two time series R and S are similar in terms of the pattern that they contain (they both have the "two bottom" pattern), but their amplitudes are different (in fact, $R = aS + b$ where a and b are scaling factor and shifting factor, respectively).
- **Time shifting.** In Figure 2(c), time series R and S record the same event (e.g. temperature data) but from different starting times; shifting R to the left on the time axis can align the shape with S. R and S are considered dissimilar if they are simply compared by the respective positions of the data points. However, it can be argued that R and S are similar because they contain the same pattern ("two peaks").
- **Time scaling.** In Figure 2(d), time series R and S have different sampling rates. However, both R and S contain "two peaks".

Several approximation approaches have been proposed to transform time series data to character strings over a discrete alphabet and apply string matching techniques to find the patterns [2, 18, 19]. However, the transformation process is sensitive to noise. Furthermore, because of the quantization of the value space for transforming time series data into strings, data points located near the boundaries of two quantized subspaces may be assigned to different alphabets. As a consequence, they are falsely considered to be different by string comparison techniques.

For shape match queries, users are interested in retrieving the time series that have similar movement shapes to the query data, e.g. "Give me all stock data of last month that is similar to IBM's stock data of last month". As shown in Figure 3, we are looking for the stock data within the boundaries that are described by two dashed curves. Most of the previous work focused on solving these types of queries, by applying distance functions such as Euclidean distance [1, 6, 10], DTW [3, 27] and LCSS [4, 22] to compute the distance between two data sequences.

There exist applications that require answers from both types of queries. An example is an interactive analysis of time series data. Users may be initially interested in retrieving all the time series data that have some specific pattern that can be quickly answered by pattern existence queries. They can then apply shape match queries to these results to retrieve those that are similar to a time series that is of interest. However, there are no techniques that have been developed to answer both pattern existence queries and shape match queries. Of course, two different techniques used to answer pattern existence queries and shape match queries can be applied to these applications together; however, different types of representation (e.g. symbolic representation and raw representation), distance functions (e.g. string matching and Euclidean distance), and indexing structures (e.g. suffix tree and R*-tree) will require storing redundant information and cause difficulties in improving the retrieval efficiency.

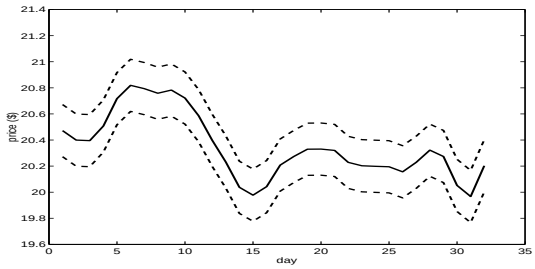


Figure 3. A query example on stock time series data

In this paper, based on the observation that data distributions can capture patterns of time series data, we propose a multi-scale histogram-based representation to approximate time series data that is invariant to noise, amplitude shifting and scaling, and time shifting and scaling. The histogram representation can answer both pattern existence queries and shape match queries, while multiple scales offer users flexibility to search time series data with different precision levels (scales). The cumulative histogram distance [20], which is closer to perceptual similarity than L_1-norm, L_2-norm, or weighted Euclidean distance, is used to measure the similarity between two time series histograms. We prove that distances of lower scale time series histograms are lower bounds of higher scale distances, which guarantees that using multi-step filtering process in answering shape match queries will not introduce false dismissals. In order to reduce the computation cost in computing the dis-

tances of full time series histograms, we use the distances between averages of time series histograms as the first step of the filtering process, since the distances between averages are lower bounds of distances of time series histograms at scale 1. We investigate two different approaches in constructing histograms: *equal size bin* and *equal area bin*. The experimental results show that our histogram-based representation, together with the cumulative histogram distance measure, can effectively capture the patterns of time series data as well as the movement shapes of time series data. Finally, We compare our representation with another multi-scale representation, namely wavelets, and conclude that wavelets are not suitable to answer pattern existence queries and are not robust to time shifting when used to answer shape match queries.

The rest of the paper is arranged as follows: Section 2 presents, as background, concepts of time series histograms and two variations of them. We present multi-scale time series histograms in Section 3. In Section 4, we present experimental results using multi-scale time series histograms on finding patterns and answering shape match queries, followed, in Section 5, by a comparison of our representation with wavelet. Related work are addressed in Section 6. We conclude in Section 7 and indicate some further work.

2 Time Series Histograms

A *time series* R is defined as a sequence of pairs, each of which shows the value (r_i) that is sampled at a specific time denoted by a timestamp (t_i): $R = [(r_1, t_1), \ldots, (r_N, t_N)]$ where N, the number of data points in R, is defined as the *length* of R. We refer to this sequence as the *raw representation* of the time series data.

Given a set of time series data $\mathcal{D} = \{R_1, R_2, \ldots, R_L\}$, each time series R_i is normalized into its *normal form* using its mean (μ) and variance (σ) [8]:

$$Norm(R) = [(t_1, \frac{r_1 - \mu}{\sigma}), \ldots, (t_N, \frac{r_N - \mu}{\sigma})] \quad (1)$$

The similarity measures computed from time series normal form are invariant to amplitude scaling and shifting.

Time series histograms are developed in the following way. Given the maximum ($max_{\mathcal{D}}$) and minimum ($min_{\mathcal{D}}$) values of normalized time series data, the range $[min_{\mathcal{D}}, max_{\mathcal{D}}]$ is divided into τ disjoint equal size subregions, called *histogram bins*. Given a time series R, its histogram H_R can be computed by counting the number of data points h_i ($1 \leq i \leq \tau$) that are located in each histogram bin i: $H_R = [h_1, \ldots, h_\tau]$.

We normalize the time series histogram by dividing the value of each histogram bin by the total number of data points in the time series. Since a time series histogram is computed from the normal form of a time series, the distance that is computed from two time series histograms are invariant to amplitude scaling and shifting. Furthermore, because time series histograms ignore the temporal information, they are also robust to time shifting and scaling. For example, in Figures 2(c) and 2(d), the histogram of normalized R are similar to that of S. Moreover, since time series histograms show the whole distribution of the data, and noise only makes up a very small portion, comparisons based on histograms can remove the disturbance caused by noise. Therefore, time series histograms are ideal representations for answering pattern existence queries.

L_1-norm or L_2-norm [21] can be used to measure the similarity between two histograms. However, these do not take the similarity between time series histogram bins into consideration, which may lead to poor comparison results. Consider three histograms H_R, H_S and H_T representing three time series of equal length. Assume that H_R and H_S have the same value on consecutive bins and H_T has the same value in a bin which is quite far away from the bins of H_R and H_S. L_1 and L_2-norm distances between any two of these three histograms are equal. However, for answering shape match queries, H_R is closer to H_S than it is to H_T. Even for pattern existence queries, data points which are located near the boundary of two histogram bins should be treated differently compared to those points that are far apart, which is not considered by L_1-norm and L_2-norm.

A weighted Euclidean distance can be used to compute the distance between two histograms [9]. Given two time series R and S, the *weighted Euclidean distance* (WED) between their time series histograms H_R and H_S is:$WED(H_R, H_S) = Z^T A Z$ where $Z = (H_R - H_S)$, Z^T is the transpose of Z, and $A = [a_{ij}]$ is a similarity matrix whose element $a_{ij} = 1 - |j-i|/\tau$ (where τ is the number of bins) denotes similarity between two time series histogram bins i and j. As a_{ij} gets larger, bins i and j become more similar.

Compared to L_1-norm and L_2-norm, WED underestimates distances because it tends to estimate the similarity of data distribution without a pronounced mode [20].

Cumulative histogram distances [20] overcome the shortcomings of L_1-norm, L_2-norm and WED, the similarity that is measured by cumulative histogram distance is closer to the perceptual similarity of histograms. Therefore, we use this distance function to measure the similarity between two time series histograms. Given a time series R and its time series histogram $H_R = [h_1, h_2, \ldots, h_\tau]$, where τ is the number of bins, the *cumulative histogram* of R is:$\hat{H}_R = [\hat{h}_1, \hat{h}_2, \ldots, \hat{h}_\tau]$ where $\hat{h}_i = \sum_{j \leq i} h_j$. Given two cumulative histograms $\hat{H}_R$ and $\hat{H}_S$ of two time series R and S, respectively, the cumulative histogram distance (CHD) is defined as:

$$CHD(\hat{H}_R, \hat{H}_S) = \sqrt{\hat{Z}^T \hat{Z}} \quad (2)$$

where $\hat{Z} = (\hat{H}_R - \hat{H}_S)$.

The algorithm for answering pattern existence queries using CHD is then simple. For each $\hat{H}_i$ of R_i, if $CHD(\hat{H}_i, \hat{H}) \leq \epsilon$, then R_i is in the result set (ϵ is a matching threshold). Later, we experimentally compare the effectiveness of WED and CHD in answering pattern existence queries. In this algorithm, a matching threshold (ϵ) has to be set to determine whether the examined time series contains a pattern similar to that of the query time series. In our experiments, we find a suitable threshold based on the histogram of the query time series.

So far, we defined a histogram with equal size bins. However, values of time series data normally are not uniformly distributed, leaving a lot of bins empty. In such cases, computing the distances between two time series histograms that contain many zeros is not helpful to differentiate the corresponding time series data.

It has been claimed [14] that distributions of most time series data follow normal distribution, which we have also verified on the data sets that we use in our experiments. Consequently, instead of segmenting the value space into τ equal size sub-regions, we segment the value space into sub-regions (called sub-spaces) that have the same area size under the normal distribution curve of that data. The boundary of each subspace can be computed as follows. Assuming that $h_{i,l}$ is the lower bound of subspace i and $h_{i,u}$ is the upper bound: $\int_{min_{\mathcal{D}}}^{max_{\mathcal{D}}} p(x)dx = \sum \int_{h_{i,l}}^{h_{i,u}} p(x)dx$, where $p(x)$ is the normal distribution function, $1 \leq i \leq \tau$, $h_{1,l} = min_{\mathcal{D}}$, and $h_{\tau,u} = max_{\mathcal{D}}$. Even though we use the normal distribution function to create equal area bin histogram bins, the idea can be easily extended to other data distributions.

With equal area bin segmentation, we assign equal probability to each histogram bin into which data points of time series fall. For example, for the "cameramouse" data that we used in our experiment, the average filling ratio of 16 equal area bin histograms is about 98% (the filling ratio is defined as the number of non-empty bins to the total number of bins). However, it is only 40% for the 16 bin equal size bin histograms. In our experiments, we compare the effectiveness of equal size bin histograms and equal area bin histograms in terms of classification accuracy.

3 Multi-scale Histograms

The time series histograms, as defined in the previous section, give a global view of the data distribution of time series data. However, they do not consider the order of values in the sequence.

A multi-scale representation of a time series histogram is designed for better discrimination of time series data based on their order details to facilitate shape match queries. Given a time series R of length N, it can be equally divided into two segments, and each segment can be recursively divided into two, and so on. For each segment, its time series histogram can be computed and all these histograms form the multi-scale time series histograms of R. The number of levels (scales) is controlled by a single parameter, δ, that is the *precision level* (i.e. *scale*) of the histogram. For $\delta = 1$, the histogram covers the entire time series, as defined in the previous section. If further precision is required, one can set $\delta > 1$, which would segment the time series data into $2^{(\delta-1)}$ equal length subsequences and histograms are computed for each segment.

With multi-scale time series histograms, the shape match queries can be answered at several precision levels. It has to be guaranteed that similar time series at a higher scale will be also identified as similar at a lower scale. In order to guarantee this property, the cumulative histogram distance must be formulated in such a way that the distance at the lower scale is the lower bound of the distance at the higher scale. The average of the cumulative histogram distances is used as the result of comparison at scale δ, which, as proven below, satisfies this property.

Definition 3.1 *Given two time series data R and S, let their $\delta \geq 1$ scale histograms be $H^i_{R,\delta}$ and $H^i_{S,\delta}$, respectively. where $1 \leq i \leq 2^{\delta-1}$. The CHD at scale δ is:*

$$CHD_\delta = \frac{\sum_{i=1}^{2^{\delta-1}} CHD(\hat{H}^i_{R,\delta}, \hat{H}^i_{S,\delta})}{2^{\delta-1}} \tag{3}$$

where $\hat{H}^i_{R,\delta}$ ($\hat{H}^i_{S,\delta}$) denotes the cumulative histograms of i^{th} segment of R (S) at scale δ.

Theorem 3.1 *In a δ-level multi-scale time series histogram, if CHD_l denotes the cumulative histogram distance between two time series histograms at scale l, then*

$$CHD_{l-1} \leq CHD_l \tag{4}$$

where $2 \leq l \leq \delta$.

We only prove the base case, which is $CHD_1 \leq CHD_2$; the general case can be proven by induction.

Proof. Given two time series data R and S, their cumulative histograms at scale i ($i \geq 1$) are denoted as: $\hat{H}_{R,i}$ and $\hat{H}_{S,i}$, respectively. Then: $CHD_1 = \sqrt{(\hat{H}_{R,1} - \hat{H}_{S,1})^T(\hat{H}_{R,1} - \hat{H}_{S,1})}$ and $CHD_2 = \frac{1}{2}\sum_{i=1}^{2} \sqrt{(\hat{H}^i_{R,2} - \hat{H}^i_{S,2})^T(\hat{H}^i_{R,2} - \hat{H}^i_{S,2})}$. Define $\|X\| = \sqrt{X^T X}$ and $\|X - Y\| = \sqrt{(X-Y)^T(X-Y)}$, where X and Y are τ dimensional vectors. Then, $CHD_1 = \|\hat{H}_{R,1} - \hat{H}_{S,1}\|$ and $CHD_2 = \frac{1}{2}\sum_{i=1}^{2} \|\hat{H}^i_{S,2} - \hat{H}^i_{R,2}\|$.

$$\begin{aligned} \|X+Y\| &= \sqrt{\sum_{i=1}^{n}(x_i + y_i)^2} \\ &= \sqrt{\|X\|^2 + \|Y\|^2 + 2\sum_{i=1}^{n}(x_i y_i)} \end{aligned}$$

Using Cauchy's inequality, which is

$$(\sum_{i=1}^{n}(x_i y_i))^2 \leq \sum_{i=1}^{n}(x_i)^2 \sum_{i=1}^{n}(y_i)^2 = \|X\|^2\|Y\|^2$$

we get

$$\|X+Y\| \leq \|X\| + \|Y\| \quad (5)$$

It is known that $\hat{H}_{R,1} = \frac{1}{2}(\hat{H}^1_{R,2} + \hat{H}^2_{R,2})$ and $\hat{H}_{S,1} = \frac{1}{2}(\hat{H}^1_{S,2} + \hat{H}^2_{S,2})$. Thus:

$$\begin{aligned}\|\hat{H}_{R,1} - \hat{H}_{S,1}\| &= \|\frac{1}{2}\sum_{i=1}^{2}(\hat{H}^i_{R,2}) - \frac{1}{2}\sum_{i=1}^{2}(\hat{H}^i_{S,2})\| \\ &\leq \frac{1}{2}\sum_{i=1}^{2}\|\hat{H}^i_{R,2} - \hat{H}^i_{S,2}\| \text{(from 5)}\end{aligned}$$

Therefore, $CHD_1 \leq CHD_2$. □

As we stated earlier, time series histograms at higher scales have better discrimination power; however, the computation of CHD at higher scales is more expensive than those at lower scales. Fortunately, with the lower bound property of CHD at each scale (Theorem 1), when we need to answer a shape match query at high scale l, instead of directly computing the CHD at scale l,we can start computing the CHD for each candidate time series at scale 1, and then scale 2 and so on [13]. This multi-step filtering strategy will not introduce false dismissals.

For both pattern existence queries and shape match queries, directly comparing τ-dimensional time series histograms is computationally expensive, even with the help of multidimensional access methods such as the R-tree. Since τ is higher than 12-16 dimensions, the performance of using an indexing structure will be worse than that of sequential scan [25]. Therefore, we use the averages of time series cumulative histograms to avoid comparisons on full cumulative histograms.

Definition 3.2 *Given a time series R and its cumulative histogram at scale level 1, $\hat{H}_{R,1} = [\hat{h}_{R,1}, \hat{h}_{R,2}, \ldots, \hat{h}_{R,\tau}]$, where τ is the number of histogram bins, the average of cumulative histogram is:*

$$\hat{H}^{avg}_{R,1} = \frac{\sum_{i=1}^{\tau}\hat{h}_{R,i}}{\tau} \quad (6)$$

Definition 3.3 *Given two time series R and S, let $\hat{H}_{R,1}$ and $\hat{H}_{S,1}$ be their cumulative histograms at scale level 1. The distance between averages of cumulative histograms is:*

$$ACHD = \sqrt{\tau(\hat{H}^{avg}_{R,1} - \hat{H}^{avg}_{S,1})^2} \quad (7)$$

The averages of time series cumulative histograms are one dimensional data, which means that a simple B+-tree can be used to improve the retrieval efficiency. Moreover, based on the definition of ACHD, the time series data retrieved by comparing ACHD are guaranteed to include all the time series data that should be retrieved by comparing cumulative histograms of time series data at scale 1. This is stated in the following theorem.

```
Procedure shape match queries{
  /* An example time series Q, its δ levels cumulative histograms
     a matching threshold ε, precision level δ*/
  (1) for each average of cumulative histograms Ĥ_i^avg of R_i{
  (2)   compute ACHD between Ĥ^avg and Ĥ_i^avg
  (3)   if (ACHD ≤ ε) { /* need to check */
  (4)     insert the time series id i into the result list resultlist_0
        } /* end-if, line 3 */
    } /* end-for, line 1 */
  (5) j = 1
  (6) do {
  (7)   for each i in resultlist_{j-1} {
  (8)     if (CHD_j(Ĥ_i, Ĥ) ≤ ε){
  (9)       insert the time series id i into the result list resultlist_j
          } /* end-if, line 8 */
        } /* end-for, line 7 */
  (10) j = j + 1
  (11) }while (j == δ + 1) or (resultlist_{j-1} is empty)
  (12) if (resultlist_{j-1} is empty)
  (13) return NULL
  (14) else return the result list resultlist_δ
}
```

Figure 4. The algorithm for answering shape match queries with multi-scale filtering

Theorem 3.2 *For any two τ-dimensional time series cumulative histograms $\hat{H}_{R,1}$ and $\hat{H}_{S,1}$ at scale 1, $ACHD \leq CHD_1$.*

Proof. Given two histograms of scale 1 $\hat{H}_{R,1}$ and $\hat{H}_{S,1}$,$CHD_1^2 = \sum_{i=1}^{\tau}(\hat{h}_{R,i} - \hat{h}_{S,i})^2$ and $ACHD^2 = \frac{(\sum_{i=1}^{\tau}\hat{h}_{R,i} - \sum_{i=1}^{\tau}\hat{h}_{S,i})^2}{\tau}$

Define $X = \hat{H}_{R,1} - \hat{H}_{S,1}$, where $x_i = \hat{h}_{R,i} - \hat{h}_{S,i}$ and $1 \leq i \leq \tau$. Therefore, the only thing that needs to be proven is: $\sum_{i=1}^{\tau} x_i^2 \geq \frac{(\sum_{i=1}^{\tau} x_i)^2}{\tau}$. According to Arithmetic-Geometric Mean inequality:

$$\begin{aligned}\frac{(\sum_{i=1}^{\tau} x_i)^2}{\tau} &\leq \frac{(\sum_{i=1}^{\tau}(x_i)^2 + \sum_{i=1}^{\tau-1}\sum_{i=1}^{\tau}(x_i)^2)}{\tau} \\ &= \sum_{i=1}^{\tau}(x_i)^2 \quad \square\end{aligned}$$

Therefore, instead of directly computing the cumulative histogram distances, we can first compute the distance between the averages of two time series cumulative histograms to prune false alarms from the database. Averages can be considered as the first filter when we use multi-step filtering to answer an shape match query at a higher scale l. The algorithm for multi-step filtering is given in Figure 4. We also show experimentally the pruning power of averages of cumulative histograms.

4 Experiments and Discussion

In this section, we present the results of experiments that we have conducted to evaluate the efficacy and robustness

of the histogram-based similarity measure and the matching algorithm. All programs are written in C and experiments are run on a Sun-Blade-1000 workstation under Solaris 2.8 with 1GB of memory.

4.1 Efficacy and Robustness of Similarity Measures

Experiment 1. This experiment is designed to test how well the cumulative histogram distances perform in finding patterns in time series data. We first compare our approach with Shatkay's algorithm [19] on labelled time series data sets: Cylinder-Bell-Funnel (CBF). In Shatkay's algorithm, first, a best fitting line algorithm is used to detect the movement slope of segments of time series data. Then, the slope of each line segment is mapped to a symbol according to the predefined mapping tables and consecutive symbols are connected together to form a string. Finally, a string matching algorithm is used to find the matches between query regular expression and the converted strings. Another related work [18] requires users to specify, in addition to the pattern, the "unit length" (the length of time series data) in which the specified pattern may appear. It is quite difficult for users to know the "unit length" beforehand if they only want to check for the existence of a pattern. Therefore, we do not consider this one. The reason for using CBF data set is that it contains very simple distinct patterns, allowing a direct comparison of the techniques. The CBF data set has three distinct classes of times series: cylinder, bell and funnel. We generate 1000 CBF data sets with 100 examples for each class. Since the general shape of bell and funnel are treated as similar (both of them contain a *peak*), only cylinder and bell data sets are used to test existence queries.

In order to test robustness of the similarity measures, we added random Gaussian noise and time warping to both data sets using a modified version of the program in [23]. The modification includes the addition of non-interpolated noise and large time warping ($20-30\%$ of the series length) since pattern existence queries only involve the existence of a given pattern.

We use the first scale time series histograms to search the patterns in the time series data. We generated another "pure" CBF data set of size 150 as query data, where each class contains 50 examples. The histograms are also computed from the query data set. We use the well-known precision and recall to measure our retrieval results. Precision measures the proportion of correctly found time series, while recall measures the proportion of correct time series that are detected. In this experiment, we need to determine the matching threshold. We tested several values and found that half the cumulative histogram distance between the query histogram and an empty histogram gives the best results. Besides using cumulative histogram distance, we also run the program with weighted Euclidean distance to compare their effectiveness.

We run the query 50 times and average the results, as shown in Table 1, where histogram experiments are parameterized by the number of bins. In these experiments, results are reported as WED/CHD values. Even though Shatkay's approach achieves relatively high precision, its recall is very low; this is the effect of noise. Our histogram-based approach (both WED and CHD) can achieve relatively high precision and recall, which confirms that our approach is suitable for answering pattern existence queries. From Table 1, we also find that dividing the value space into too many sub-regions (histogram bins) does not improve the results significantly; we can achieve reasonably good results around 16 bins. As we expect, the results also show that CHD performs better than WED, this is because WED overestimates neighborhood similarity.

	Cylinder		Bell	
	precision	recall	precision	recall
Shatkay	81	44	75	31
his(8)	72/80	88/91	63/ 70	71/ 74
his(16)	72/81	90/92	78/ 84	80/ 85
his(32)	75/81	88/90	79/ 85	85/ 87
his(64)	72/80	88/90	79/ 83	85/ 88

Table 1. Comparison on pattern existences queries

Experiment 2. This experiment is designed to check the effectiveness and robustness of multi-scale histograms in answering shape match queries. According to a recent survey on time series data [11], the efficacy of a similarity measure for shape match queries can be evaluated by classification results on labelled data sets. We compare the classification error rates in results produced by CHD to those of DTW and LCSS by evaluating them on the Control-Chart (CC) data set. The comparison is done against DTW and LCSS because these two can also handle time shifting or noise disturbances. The classification error rate is defined as a ratio of the number of misclassified time series to the total number of the time series. There are six different classes of control charts. Each class contains 100 examples. We later tested our techniques using more complicated data sets. We added non-interpolated noise and time warping ($10-20\%$ of the series length) to test the robustness of cumulative histogram distance in answering shape match queries. For simplicity, we carried out a simple classification using 1-Nearest Neighbor with three similarity measures and checked the classification results using the "leave one out" verification mechanism.[1]

We repeat the experiment on all the time series of CC

[1]The "leave one out" verification mechanism takes one sample data from a class and finds a nearest neighbor of the data in the entire data set by applying a predefined distance metric. If the found nearest neighbor belongs to the same class as the sample data, it is a hit, otherwise, it is a miss.

data set. The error rates using DTW and LCSS are 0.12 and 0.16, respectively. It has been claimed that LCSS is more accurate than DTW [22]. However, our experiments could not replicate this result. The possible reason for this discrepancy is the difference in the choice of the matching threshold value. We report comparable results using the cumulative histogram distances in Table 2. We run the experiment with different number of bins (τ) and scales (δ) using CHD on time series histograms with equal size bin.

Comparing the error rates of DTW (0.12), LCSS (0.16) and those of Table 2, we observe that CHD performs better than the other two similarity measures in answering shape match queries. From Table 2, we find an interesting fact that very high scales (i.e., high values of δ) may lead to worse classification results. This is because the higher the scale, the more temporal details will be involved in computing CHD, and this causes time series histograms at that scale to be more sensitive to time shifting and scaling. Another fact demonstrated in Table 2 is that higher number of bins may not lead to more accurate classification results. This is because as the number of bins increases, more detailed information will be captured in time series histogram, including noise. These characteristics of multi-scale time series histograms exactly fit our needs, since they suggest that we only need to check the first few scale histograms with small number of histogram bins. However, the improvements over DTW or LCSS as reported in Table 2 are modest, especially for the first few scale histograms.

	number of bins τ			
scale δ	8	16	32	64
1	0.62	0.59	0.56	0.53
2	0.22	0.16	0.12	0.13
3	**0.14**	**0.10**	**0.11**	**0.11**
4	0.20	0.13	0.14	0.15
5	0.24	0.19	0.20	0.20

Table 2. Error rates with equal size bin histograms

To better understand these results, we investigated the filling ratio of equal size bin histograms. We found that nearly 50% of the bins are empty! Consequently, CHD between two time series histograms is not able to distinguish them properly, since most of the bins are identical! Therefore, we run the same experiment with equal area bin histograms. Table 3 reports the results.

	number of bins τ			
scale δ	8	16	32	64
1	0.15	0.11	0.12	0.13
2	0.10	0.08	0.07	0.06
3	**0.07**	**0.05**	**0.06**	**0.06**
4	0.10	0.08	0.08	0.09
5	0.18	0.15	0.17	0.17

Table 3. Error rates with equal area bin histograms

The results of Tables 2 and 3 demonstrate that the improvement when equal area bin is used is nearly 2 times for CC data! The filling ratio of time series histogram is around 80%. Table 3 also shows that using only the first few scales (e.g. $\delta = 3$), provides reasonably high accuracy.

Experiment 3. In this experiment, we test the matching accuracy of multi-scale time histograms versus DTW and LCSS on classifying time series data with more complicated shapes. Classifying these types of time series requires matching on temporal details of the data. We evaluate CHD, DTW and LCSS by two labelled trajectory data sets that are generated from the Australian Sign Language (ASL)[2] data and the "cameramouse" [7] data. Only x positions are used for both data sets. We first select a "seed" time series from each of the two data sets and then create two additional data sets by adding interpolated Gaussian noise and small time warping (5-10% of the time series length) to these seeds [23]. The ASL data set from UCI data consists of samples of Australian Sign Language signs. Various parameters were recorded as a signer was signing one of 95 words in ASL. We extracted one recording from each of 10 words[3]. The "cameramouse" data set contains 15 trajectories of 5 words (3 for each word) obtained by tracking the finger tip as people write various words. We use all the trajectories of each word as the seeds. The data set that is generated from seeds of "cameramouse" contains 5 classes and 30 examples of each class, while the one from ASL seeds contains 10 classes and each class has 10 examples. We use the same classification and verification algorithms as in the second experiment. Based on the observation of the second experiment, we use time series histogram with equal area bin. The error rate of using DTW for ASL and "cameramouse" was 0.11 and 0.08, respectively. Comparable values for LCSS were 0.29 and 0.30. Table 4 reports the error rates of CHD.

Table 4 shows that CHD again achieves better classification result. For both data sets, CHD can achieve 0 error rate. A surprising observation is that even at lower scales (e.g. $\delta = 2$), CHD can achieve better results than DTW and LCSS. Through the investigation of the filling ratios of both histograms, we find that the number of empty bins only accounts for less than 5% of the total number of bins. The standard deviations of both data sets are also very high with respect to their mean values, which indicates that data points are widely spread in their value space. Compared with results of the second experiment, we conclude that CHD on time series data whose data points are widely distributed in their value space (higher histogram filling ratio) can achieve better classification accuracy at lower scales.

[2] http://kdd.ics.uci.edu

[3] "seed" time series of ASL data: "Norway", "cold", "crazy", "eat", "forget", "happy", "innocent", "later", "lose" and 'spend"[23].

	ASL DATA				Cameramouse DATA			
	number of bins τ				number of bins τ			
scale	8	16	32	64	8	16	32	64
1	0.12	0.11	0.12	0.12	0.13	0.13	0.07	0.07
2	**0.06**	**0.05**	**0.04**	**0.04**	**0.04**	**0.03**	**0.02**	**0.02**
3	0.06	0.06	0.04	0.04	0.03	0.02	0.02	0.02
4	0.03	0.04	0.04	0	0.04	0.01	0	0.01
5	0.06	0.06	0.04	0.02	0.04	0.04	0.02	0.04

Table 4. Error rates using CHD on time series histograms with equal area bin

4.2 Efficiency of Multi-Step Filtering

Experiment 5. Theorem 1 in Section 3 established that multi-step filtering using multi-scale time series histograms will not introduce false alarms. However, the question remains as to how many histogram comparisons are saved using multi-step filtering? We use a real stock data set that contains 193 company stocks' daily closing price from late 1993 to early 1996, each consisting of 513 values [24]. We use each time series as a "seed" and create a new data set by adding interpolated Gaussian noise and small time warping (2-5% of the time series length) to each seed. The new data set contains 1930 time series (each seed is used to create 10 new time series). We randomly select a time series and conduct a range query at precision level 4. We compute the number of comparisons that are needed for searching using only level 4, using level 1 and then jumping to 4, and using all 4 levels on different range thresholds. We run the experiments 100 times and the average results are reported in Table 5. Step-by-step filtering is clearly the best strategy

	level 1 only	level 1 and level 4	all 4 levels
$\epsilon = 0.5$	11580	13498	17190
$\epsilon = 0.2$	11580	12760	7590
$\epsilon = 0.1$	11580	6838	3668
$\epsilon = 0.05$	11580	2356	2072

Table 5. Comparisons on different filtering approaches

to reduce the total number of comparisons for small thresholds. For large thresholds, directly computing the distance at a higher level is a better choice in terms of the number of comparisons. However, large thresholds are not very useful for finding the desirable results, since a larger portion of the data in the database will be returned.

Experiment 6. The number of comparisons needed for computing the histogram distance only relates to the CPU computation cost. However, the I/O cost for sequentially reading in the data files becomes a bottleneck as the database size grows. If we can filter out the false alarms without reading in the data files, a significant speed-up will be achieved. For our time series histograms, we store their averages separately from histograms. The distances between the average cumulative histogram of query data and those of stored data are first computed to remove the possible false alarms. Therefore, the pruning power of average cumulative histograms is quite important. The pruning power (P) is defined as the fraction of the database that must be examined before we can guarantee that the nearest match to 1-Nearest Neighbor query is found [10]. Figure 5 shows the pruning power of CHD with 16 bins (based on the previous experimental results, CHD can already achieve reasonable good results when the histogram bin size is 16). Because "cameramouse" and ASL data sets are small, we did not include them in this experiment. Figure 5 demonstrates that using the distance of average time series histograms, we can remove nearly 40% of the false alarms, which is helpful when the database size becomes large.

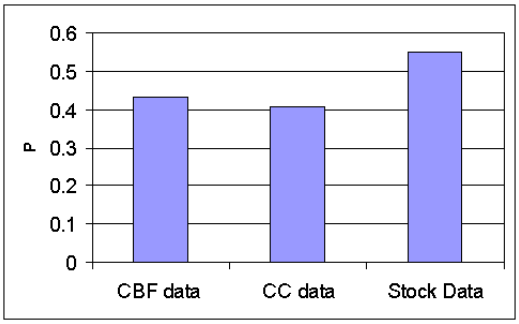


Figure 5. The punning power of averages of histograms

5 Comparison to Wavelets

In this section, we compare our representation with another multi-scale representation, wavelets. Wavelets have been widely used as a multi-resolution representation for image retrieval [16]; they have also been used as a dimensionality reduction technique in shape match queries in which Euclidean distance is used as the distance function [12]. Different from Fourier transform, wavelets can transform time series data into a multi-scale representation, where lower frequency bands are represented in lower scales and higher frequency bands are represented in higher scales. In this experiment, we used Euclidean distance to measure the similarity between two Haar wavelet coefficients (we used Haar wavelet, since it is the most popular wavelet transformation and has been used for similarity-based time series retrieval [12]).The same CBF data set used in the first experiment is used to measure the efficacy using different number of wavelet coefficients to answer pattern

existence queries. Table 6 reports the results.

	Cylinder		Bell	
	precision	recall	precision	recall
8	49	97	48	96
16	45	61	46	67
32	40	44	37	40
64	31	35	29	33

Table 6. Using wavelet for pattern existences queries

Compared to results that obtained by time series histograms (81/90 and 85/87), wavelets perform significantly worse in terms of accuracy. In fact, the wavelet transform transfers the time series data into time-frequency domain, therefore, it is difficult, using wavelets, to answer pattern existence queries, since frequency appearance order is encoded in the wavelet coefficients. Even using only the first few coefficients (8), which have very lower time resolution (scale), we could get high recall but low precision as shown in Table 6. The high recall was achieved by returning nearly all the data in the database as results and the lower precision is due to most of the important frequency bands are not used to answer queries. However, if we use more (32), both precision and recall drop. This is because the high frequency bands have higher time resolution, making them sensitive to time shifting.

For shape match queries, we compared the efficacy of wavelets and multi-scale histograms on CC data set. We did not use the ASL and Cameramouse data sets, because Euclidean distance requires the two compared wavelet coefficients to have the same length, thus the original time series must have the same length as well. Furthermore, since the wavelet transform requires the length of the sequences be a power of 2, we pad the sequences with 0 to make the length 64. We carried out the same classification test using different number of wavelet coefficients and the results are shown in Table 7.

	8	16	32	64
error rate	0.43	0.39	0.32	0.21

Table 7. Error rates using wavelet coefficients

Again the results are worse than the results achieved by using multi-scale histograms (0.06). The lower scale wavelets can not capture the information about higher frequency band, thus, the error rate with lower scale is higher than that of higher scale. The higher scale wavelets contain too much detail about the appearance time of frequency bands, which causes the wavelets to be sensitive to time shifting. In fact, with Euclidean distance, using all wavelet coefficients is the same as using raw representation of time series [12]. Thus, based on these two experiments and analysis, we conclude that wavelet representation is not robust to time shifting, making it unsuitable to answer pattern existence queries and introducing difficulties in answering shape match queries when data contain time shifting.

6 Related Work

The earliest proposal for similarity-based time series data retrieval was by Agrawal et. al. [1], which used Euclidean distance to measure the similarity between time series data and applied Discrete Fourier Transform (DFT) to reduce the dimensionality for fast retrieval. Later, this work was extended to design new similarity measures, other than Euclidean distance, for measuring the similarity between two time series data, such as Dynamic Time Warping (DTW) [3], Longest Common Subsequences (LCSS) [22], Edit distance with Real Penalty (ERP) [5] etc. All these techniques can be put into the category of shape match retrieval. A few approaches [2, 19, 18] have been proposed for finding movement patterns in time series data. The moving direction (the slope between two values) of a user specified interval [18], consecutive values [2], or a segment (obtained by a segmentation algorithm) [19] was represented as a distinct alphabet. Thus, these approaches converted the time series data into strings and could apply string-matching techniques to find the patterns. Lin et al. [14] have proposed a symbolic representation for time series data, where data points at neighborhood subspaces are treated as same and the distances between them are assigned value 0. However, this method overestimates the neighborhood similarity.

Compared to all the previous work on similarity-based time series retrieval, the multi-scale time histogram has the following advantages:

- Multi-scale time series histograms can be used to answer shape match and pattern existence queries.
- The cumulative histogram distance reduces the boundary effects introduced by value space quantization and overcomes the shortcomings of overestimating (such as L_1-norm and L_2-norm) or underestimating (such as weighted Euclidean distance) the distance.
- Multi-scale time histograms are invariant to time shifting and scaling, amplitude shifting and scaling. Moreover, they can reduce the noise disturbance.
- Multi-scale time histograms offer users flexibility to query the time series data on different accuracy level. Furthermore, lower scale histograms can be used to prune the false alarms from the database before we querying at higher scale histograms.

Recently, in spatial database domain, multi-scale histograms have been proposed to summarize rectangle objects for window queries [15]. In these approaches, the multi-scales refer to resolutions on value space; however, the multi-scales in our work refer to resolutions on time space.

7 Conclusions and Future Work

In this paper, we propose a novel representation of time series data using multi-scale time series histograms. This

representation is based on the intuition that the distribution of time series data can be an important cue for comparing similarity between time series. Multi-scale time series histograms are capable of answering both pattern existence queries and shape match queries. Moreover, they are invariant to time shifting and scaling, and amplitude shifting and scaling. A robust similarity measure, cumulative histogram distance, is used to measure the similarity between two time series histograms. Our experiments indicate that multi-scale time series histograms outperform string representations in finding patterns and outperform DTW and LCSS in answering shape match queries when the time series data contain noise or time shifting and scaling. The experiments also show that equal area bin histogram is more suitable for time series data comparison and distances of averages of histograms can effectively prune the false alarms from the database before computing the distance between two full histograms.

In future work, we will investigate the possibility of automatically setting up the scale value for users. We also plan to extend multi-scale histograms to subsequence matching.

References

[1] R. Agrawal, C. Faloutsos, and A. N. Swami. Efficient similarity search in sequence databases. In *Proc. 4th Int. Conf. of Foundations of Data Organization and Algorithms*, pages 69–84, 1993.

[2] R. Agrawal, G. Psaila, E. L. Wimmers, and M. Zaït. Querying shapes of histories. In *Proc. 21th Int. Conf. on Very Large Data Bases*, pages 502–514, 1995.

[3] D. J. Berndt and J. Clifford. Finding patterns in time series: A dynamic programming approach. In *Advances in Knowledge Discovery and Data Mining*, pages 229–248, 1996.

[4] T. Bozkaya and M. Ozsoyoglu. Distance-based indexing for high-dimensional metric spaces. pages 357–368, 1997.

[5] L. Chen and R. Ng. On the marriage of edit distance and Lp norms. In *Proc. 30th Int. Conf. on Very Large Data Bases*, pages 792–803, 2004.

[6] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos. Fast subsequence matching in time-series databases. In *Proc. ACM SIGMOD Int. Conf. on Management of Data*, pages 419–429, 1994.

[7] J. Gips, M. Betke, and P. Fleming. The camera mouse: Preliminary invertigation of automated visaul tracking for computer access. In *In Proc. Conf. on Rehabilitation Engineering and Assistive Technology Society of North America*, pages 98–100, 2000.

[8] D. Goldin and P. Kanellakis. On similarity queries for time series data: Constraint specification and implementation. In *Proc. of the Int. Conf. on Principles and Pratice of Constraint Programming*, pages 23–35, 1995.

[9] J. Hafner, H. Sawhney, W. Equitz, M. Flickner, and W. Niblack. Efficient color histogram indexing for quadratic form distance functions. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 17(7):729–739, 1995.

[10] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra. Locally adaptive dimensionality reduction for indexing large time series databases. In *Proc. ACM SIGMOD Int. Conf. on Management of Data*, pages 151–162, 2001.

[11] E. Keogh and S. Kasetty. On the need for time series data mining benchmarks: a survey and empirical demonstration. In *Proc. 8th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pages 102–111, 2002.

[12] K.P.Chan and A.-C. Fu. Efficient time series matching by wavelets. In *Proc. 15th Int. Conf. on Data Engineering*, pages 126–133, 1999.

[13] K. Leung and R. T. Ng. Multistage similarity matching for sub-image queries of arbitrary size. In *Proc. of 4th Working Conf. on Visual Database Systems*, pages 243–264, 1998.

[14] J. Lin, E. Keogh, S. Lonardi, and B. Liu. A symbolic representation of time series, with implications for streaming algorithms. In *Proc. 8th Int. Workshop on Research Issues in Data Mining and Knowledge Discovery*, pages 2–11, 2003.

[15] X. Lin, Q. Liu, Y. Yuan, and X. Zhou. Multiscale histograms: Summarizing topological relations in large spatial datasets. In *Proc. 29th Int. Conf. on Very Large Data Bases*, pages 814–825, 2003.

[16] A. Natsev, R. Rastogi, and K. Shim. Walrus: a similarity retrieval algorithm for image databases. In *Proc. ACM SIGMOD Int. Conf. on Management of Data*, pages 395–406, 1999.

[17] S. Park and W. Chu. Similarity-based subsequence search in image sequence databases. *Int'l. J. of Image and Graphics*, 3(1):31–53, 2003.

[18] Y. Qu, C. Wang, , L. Gao, and X. S. Wang. Supporting movement pattern queries in user-specified scales. *IEEE Trans. Knowledge and Data Eng.*, 15(1):26–42, 2003.

[19] H. Shatkay and S. Zdonik. Approximate queries and representations for large data sequences. In *Proc. 12th Int. Conf. on Data Engineering*, pages 536–545, 1996.

[20] M. Stricker and M. Orengo. Similarity of color images. In *Proc. 7th Int. Symp. on Storage and Retrieval for Image and Video Databases*, pages 381–392, 1995.

[21] M. J. Swain and D. H. Ballard. Color indexing. *Int. Journal of Computer Vision*, 7(1):11–32, 1991.

[22] M. Vlachos, G. Kollios, and D. Gunopulos. Discovering similar multidimensional trajectories. In *Proc. 18th Int. Conf. on Data Engineering*, pages 673 – 684, 2002.

[23] M. Vlachos, J. Lin, E. Keogh, and D. Gunopulos. A wavelet-based anytime algorithm for k-means clustering of time series. In *Proc. Workshop on Clustering High Dimensionality Data and Its Applications*, pages 23–30, 2003.

[24] C. Wang and X. Wang. Supporting content-based searches on time series via approximation. In *Proc. 12th Int. Conf. on Scientific and Statistical Database Management*, pages 69–81, 2000.

[25] R. Weber, H.-J. Schek, and S. Blott. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In *Proc. 24th Int. Conf. on Very Large Data Bases*, pages 194–205, 1998.

[26] H. Wu, B. Salzberg, and D. Zhang. Online event-driven subsequence matching over financial data streams. In *Proc. ACM SIGMOD Int. Conf. on Management of Data*, pages 23–34, 2004.

[27] B.-K. Yi, H. Jagadish, and C. Faloutsos. Efficient retrieval of similar time sequences under time warping. In *Proc. 14th Int. Conf. on Data Engineering*, pages 23–27, 1998.

[28] Y. Zhu and D. Shasha. Warping indexes with envelope transforms for query by humming. In *Proc. ACM SIGMOD Int. Conf. on Management of Data*, pages 181–192, 2003.