Supporting Content-based Searches on Time Series via Approximation[*]

Changzhou Wang, X. Sean Wang
Department of Information and Software Engineering
George Mason University, Fairfax, Virginia
{cwang, xywang}@gmu.edu

Abstract

Fast retrieval of time series in terms of their contents is important in many application domains. This paper studies database techniques supporting fast searches for time series whose contents are similar to what users specify. The content types studied include shapes, trends, cyclic components, autocorrelation functions and partial autocorrelation functions. Due to the complex nature of the similarity searches involving such contents, traditional database techniques usually cannot provide a fast response when the involved data volume is high. This paper hence proposes to answer such content-based queries using appropriate approximation techniques. The paper then introduces two specific approximation methods, one is wavelet based and the other line-fitting based. Finally, the paper reports some experiments conducted on a stock price data set as well as a synthesized random walk data set, and shows that both approximation methods significantly reduce the query processing time without introducing intolerable errors.

1. Introduction

In many application domains, such as economics, medicine and experimental sciences, time series is an important data type and large volumes of time series data have been collected [30, 4]. By analyzing these time series, researchers form theories about the underlying processes and provide explanations of and forecasting for various phenomena. Due to the complex nature of time series analysis methods, however, detailed analysis of each and every series is neither practical nor necessary when the data volume is very large. Instead, the following interactive and iterative mode of operation may be more desirable: Users first retrieve a small number of interesting time series, analyze them in detail, and then use the knowledge obtained from the analysis to form new criteria to retrieve the next set of series for further study. Clearly, a fast response to retrieval requests is required during such an interactive process.

Retrieval criteria often involve contents of time series, i.e., information contained in or derived from the series themselves, in addition to descriptive information (metadata) about series such as the description of the sampling method used. The types of contents that users are interested in obviously depend on the tasks at hand. This paper examines the following types of time series contents: shapes, trends, cyclic components, autocorrelation functions (ACFs) and partial autocorrelation functions (PACFs). The shape of time series is an important subject in the technical analysis of stock market and other fields of studies [12, 21], while the other four types of contents are extensively used in statistical time series analysis [30, 16]. The paper studies similarity queries, i.e., retrieval requests that ask for all the series or subseries which contain similar contents as a given series.

Due to the complex nature of similarity queries involving the above contents, traditional database techniques usually cannot provide a fast response when the data volume is high. On the other hand, in a typical interactive and iterative analysis session, users often begin with a general idea about the contents of the desired series, and subsequently narrow down the scope until they finally identify the target series. In general, at the beginning of an analysis session, the involved data volume is large but users are quite flexible about the query result, i.e., it is not extremely critical to retrieve *exactly* all the time series and subseries that satisfy the query criteria. Only at later stages of the analysis, users become more strict, but the involved data volume usually becomes smaller. Hence, in practice, at early stages of the analysis, when the involved data volume is high, we may trade the accuracy of query results for a fast response.

In this paper, we propose to directly use approximated time series to answer user queries. Approximation usually saves disk storage space and hence reduces disk access time, which often is the bottleneck of the searching process. Of course, query processing using the approximated series may

[*] This work was partially supported by the NSF with the grant number 9875114, as well as by the NASA Earth Science Enterprise WP-ESIP CAN Program (NCC 5-306).

not give a result with 100% precision and recall levels, that is, some series in the result may not meet the query criteria while some series that satisfy the query criteria may not be in the result. However, we find that good approximation methods can significantly reduce disk storage space and access time without introducing intolerable errors.

We present two approximation methods, one is based on the B-spline wavelet transform [14] and the other uses the least square method to fit the time series into consecutive line segments. Both are *general* and *uniform*, namely, a single approximation is used for similarity searches in terms of different content types and on different subseries. Note that these approximation methods might be combined with other database techniques, such as indexing, i.e., it is possible to build indexing structures on the approximated series to further accelerate the searching process. However, this direction is out of scope of this paper.

We then focus on studying the effectiveness and efficiency of the approximation methods. Specifically, we compare the performance of the nearest neighbor search[1] on the approximated time series with the performance of the same nearest neighbor search on the original series. The performance is measured in terms of the precision and recall levels in the result using the approximation, as well as the reduction in query processing time. For each content type, we carefully choose an appropriate similarity measure. The study uses a stock price data set as well as a synthesized random walk data set. The results show that both methods can significantly reduce the query processing time without introducing intolerable errors for nearest neighbor search in terms of all the content types.

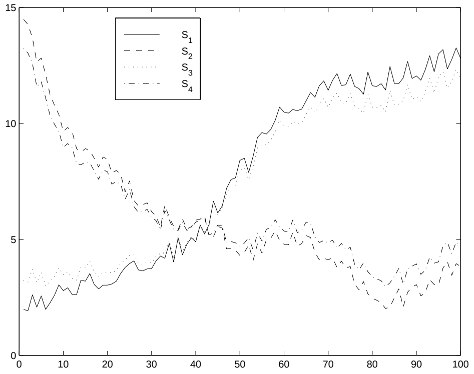
The contribution of the paper is three-fold. Firstly, we introduce statistical contents of time series into the research of similarity queries, and study similarity measures. Secondly, we propose to use approximations to support time series similarity queries and suggest the desired properties for approximation methods. Thirdly, we present two simple, general and uniform approximation methods and demonstrate (via experiments) that both methods support similarity queries efficiently and effectively.

The rest of the paper is organized as follows: The aforementioned content types are discussed in more detail in Section 2. Also in Section 2, similarity measures of time series in terms of these content types are studied. The desired properties of approximation and two specific methods are described in Section 3. The experiment results are reported in Section 4. Our work is related to other researches in Section 5 and concluded in Section 6.

2. Time Series Contents and Similarity Measures

In this section, we discuss several content types which are widely used in analyzing time series, and study appropriate similarity measures of time series in terms of these content types.

In technical analysis of time series, especially for stock market analysis, the study of shapes is very important [21, 12]. The time series is drawn in a two dimensional plane, where one dimension is the time and the other is the value. The shape of a time series is usually taken as the shape of the curve formed by connecting consecutive points on the plane, and the shape similarity of two time series is often measured by the Euclidean distance between the series themselves. Time series of similar shapes are likely to have small Euclidean distance between them, and vice versa. For example, consider the four series in Figure 1. Intuitively, S_1 and S_3 as well as S_2 and S_4 have similar shapes, but the shape of S_1 or S_3 is quite different from that of S_2 or S_4. The Euclidean distances among the four series reflect this intuition. More specifically, for example, the distance between the two similar series S_1 and S_3 is much smaller than the distance between the two dissimilar series S_1 and S_2.



(a) Four time series

Distance	S_2	S_3	S_4
S_1	66.1	6.6	61.3
S_2		61.3	6.8
S_3			54.4

(b) Their Euclidean distances

Figure 1. Time series with similar and dissimilar shapes

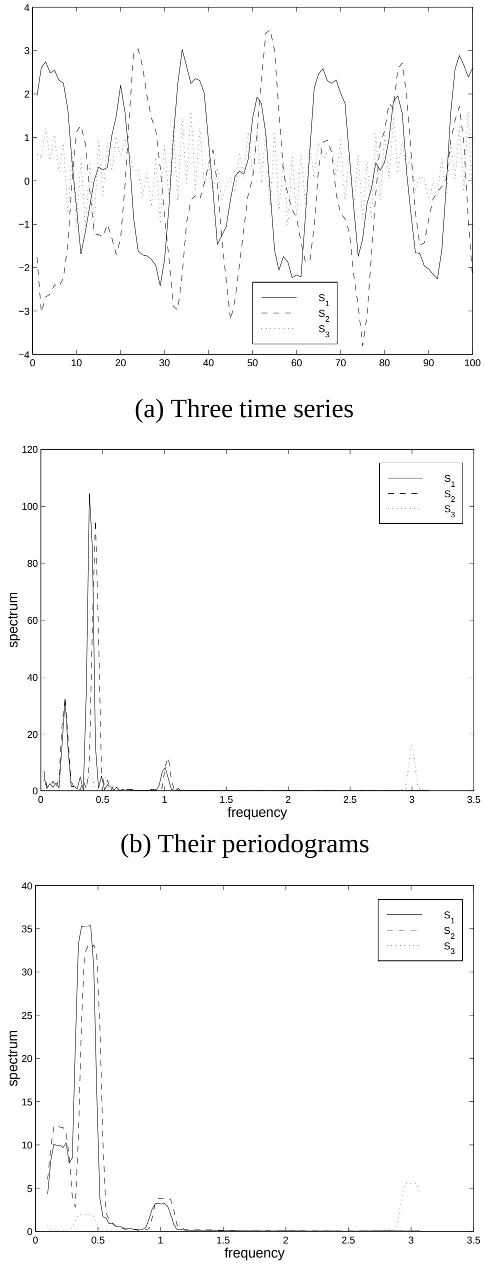
[1] There are two major types of similarity searches, namely near neighbor search and nearest neighbor search. Given a series S, a threshold value τ and a search limit N, a near neighbor search is to find all series whose distances from S are within τ, while the (N-)nearest neighbor search is to find the N series that are closest to S. We use the nearest neighbor search because the threshold used in a near neighbor search is usually data-dependent and results cannot be easily compared.

In statistical time series analysis, a time series is regarded as a realization of a series of (usually correlated) random variables and is often considered as a combination of two independent parts: a deterministic part and a nondeterministic part. The deterministic part usually consists of a linear trend and several cyclic components, while the nondeterministic part is usually characterized by some statistical models, such as the autoregressive ($AR(p)$) model, the moving average ($MA(q)$) model, and the autoregressive integrated moving average ($ARMIA(p, i, q)$) model [16, 30].

The trend of a time series is relatively simple and can be represented by several numbers. The trend similarity measure of time series varies for different types of trends in different applications. For example, if linear trends are considered and only the slope of the trend is of the concern, the two time series are regarded as having similar trends if their trend slope values are close to each other.

In practice, cyclic components of time series can be captured by their periodograms, and usually only the first few significant coefficients of their periodograms are interesting to analysts [16]. However, the similarity measure in terms of cyclic components cannot be as simple as the Euclidean distance between their periodograms (even with non-significant coefficients filtered out). Instead, it is necessary to consider both the magnitudes and the positions of significant coefficients. For example, consider the three time series S_1, S_2 and S_3 shown in Figure 2(a) and their periodograms shown in Figure 2(b).[2] Intuitively, S_1 and S_2 have similar cyclic components, while S_3 has different ones. However, for the Euclidean distances between the periodograms, we have $D_{1,2} = 142.9$, $D_{1,3} = 140.3$, $D_{2,3} = 133.4$, where $D_{i,j}$ denotes the Euclidean distance between the periodogram of S_i and that of S_j for $1 \leq i, j \leq 3$. Obviously, we cannot conclude from these distances that S_1 and S_2 are similar, yet S_1 and S_3 or S_2 and S_3 are dissimilar. Actually, the similar cyclic components of S_1 and S_2 do not have exactly the same frequencies (i.e., x-axis positions in the periodograms), instead, their frequencies are *similar* to each other (i.e., the x-axis positions are close to each other in the periodograms). Since Euclidean distance takes coefficients at different positions as independent values, the similarity between S_1 and S_2 cannot be captured in this way.

In fact, the similarity measure in terms of cyclic components needs to take into account the cross-relationship of coefficients at nearby positions. To this end, we propose to obtain the moving average of the periodograms, and use the Euclidean distance between the moving averages as the cyclic components similarity measure. Intuitively, the moving average will redistribute significant coefficients to their neighbors. Consider two significant coefficients from two periodograms, respectively, and assume that these two coefficients have similar magnitudes and positions. In the moving average of these two periodograms, the neighbors of the two coefficients are likely to overlap and the overlapped positions are likely to have similar magnitudes. Clearly, the



(a) Three time series

(b) Their periodograms

(c) Moving average of periodograms

Figure 2. Time series with similar and dissimilar cyclic components

[2] All periodograms in this study are generated by Matlab 5.0 using the “etfe” function. In Figure 2(b), the x-axis represents 128 equally spaced frequencies between 0 (exclusive) and π.

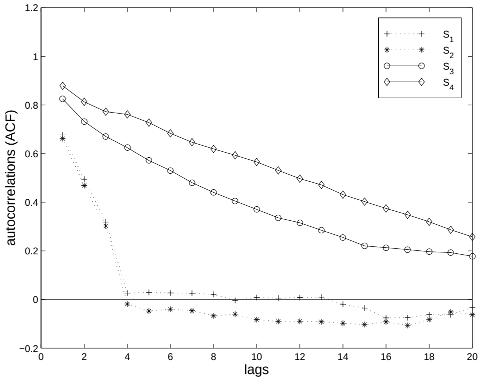
Euclidean distance of the moving averages will be smaller if the two coefficients are closer in positions as well as in magnitudes. In Figure 2(c), we present the moving averages (window size = 7) of the periodograms shown in Figure 2(b). Their Euclidean distances are $D_{1,2} = 42.1$, $D_{1,3} = 88.5$ and $D_{2,3} = 84.6$. From this, we can easily conclude that in terms of cyclic components, S_1 and S_2 are much more similar than S_1 and S_3 (or S_2 and S_3) are.

For the nondeterministic part, there is a large number of models that might be plausible for a given time series. Autocorrelation functions (ACFs) and partial autocorrelation functions (PACFs)[3] are often used in identifying appropriate models (e.g., $ARIMA(p, i, q)$) for the series of interest. For example, if the ACF truncates at lag q (i.e., the autocorrelation values are significant for lags $m = 1, 2, \ldots, q$ and are close to zero thereafter), and the PACF decays exponentially, the series probably fits into a moving average model of order q. On the other hand, if the PACF truncates at lag p and the ACF decays exponentially, the series probably fits into an autoregressive model of order p [16]. Therefore, a useful similarity measure for ACFs or PACFs is the difference in the changing of values as the lag increases, and time series with similar ACFs and/or PACFs likely fit into similar models. The changing of values, or the movements, can be represented in a descriptive language (like the ones introduced in [25, 3]), and their similarity can be defined on the representations of two ACFs or PACFs in the language. This approach deserves a full-length discussion on its own and is out of scope of this paper.

In this study, we use the Euclidean distance between two ACFs (PACFs) as the similarity measure for simplicity. Intuitively, when the Euclidean distance is small, two ACFs (PACFs) will have similar shapes and likely have similar movements (though the reverse is not necessarily true). In Figure 3(a) and (b), we show the ACFs and PACFs up to lag 20 for the following four synthesized time series:

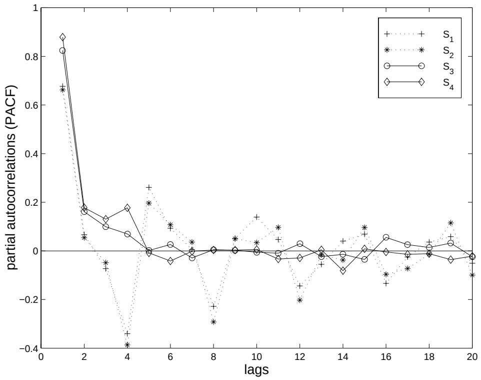
$$\begin{aligned}
S_1 &: x_i = 0.75e_{1,i} + 0.7e_{1,i+1} + 0.65e_{1,i+2} + e_{1,i+3}, \\
S_2 &: x_i = 0.8e_{2,i} + 0.65e_{2,i+1} + 0.7e_{2,i+2} + 0.95e_{2,i+3}, \\
S_3 &: x_i = 0.66x_{i-1} + 0.1x_{i-2} + 0.04x_{i-3} + 0.1x_{i-4} + e_{3,i}, \\
S_4 &: x_i = 0.67x_{i-1} + 0.11x_{i-2} + 0.05x_{i-3} + 0.1x_{i-4} + e_{4,i},
\end{aligned}$$

where $e_{1,i}$, $e_{2,i}$, $e_{3,i}$ and $e_{4,i}$ are values of four independent white noise series. Here, S_1 and S_2 are both generated using the moving average model of degree three, while S_3 and S_4 are both generated using the autoregressive model of degree four[4]. The Euclidean distances between these ACFs and PACFs are also shown in Figure 3. From these distances, we can conclude that the ACFs (PACFs) of S_1 and S_2 as well as those of S_3 and S_4 are similar in shape, hence have similar movements. As a result, we further conclude that S_1 and S_2 (as well as S_3 and S_4) may fit into similar models (in fact, they *are* from similar models).



Distance	S_2	S_3	S_4
S_1	0.28	1.60	2.25
S_2		1.85	2.51
S_3			0.68

(a) ACFs and their Euclidean distances



Distance	S_2	S_3	S_4
S_1	0.22	0.68	0.77
S_2		0.73	0.82
S_3			0.21

(b) PACFs and their Euclidean distances

Figure 3. Similar and dissimilar ACFs and PACFs

3. Approximation

As mentioned in the introduction, it is desirable for the system to quickly respond to retrieval requests that ask for time series or subseries which contain similar contents (in terms of a particular type) as a given series. When the data

[3] Roughly, the autocorrelation of a time series S_t at lag k is the correlation between S_t (the time series itself) and S_{t+k} (the time series delayed by k), while the partial autocorrelation of S_t at lag k is the correlation between S_t and S_{t+k} after their mutual linear dependency on the intervening series $S_{t+1}, \ldots, S_{t+k-1}$ has been removed.

[4] However, the shapes of S_1 and S_2, as well as S_3 and S_4, are quite different due to the randomness of $e_{1,i}$, $e_{2,i}$, $e_{3,i}$ and $e_{4,i}$.

volume is high, scanning each time series is not a feasible solution, and some precomputation and/or indexing techniques are needed. However, the number of values required to represent the shape, as well as the cyclic components (periodogram), the ACF and the PACF, of a time series is comparable to the number of values in the series itself. When the time series is long, the number of required values will be quite large, and hence, traditional indexing structures (such as B-trees or R-trees) cannot be used directly [1].

For simple contents that can be represented by a single value, such as trends, it is possible to precompute them and index the precomputed values to accelerate query processing. However, if users are allowed to issue queries on arbitrary subseries, the number of values needed for each time series will be in the quadratic order of the length of the series. Thus it is not practical to precompute all of them, especially when the involved data volume is high.

We propose a different approach, namely, we generate an approximation for each time series, and store the approximations in the database. When users issue a query, the approximated representations are searched to answer the query (the result consists of pointers to the approximated series found as well as their original counterparts). Usually, the approximations require less disk storage space than the original ones, and the disk access time for query processing can be reduced proportionally to the reduction in storage space. Approximation methods need to be selected carefully because: 1) there is an extra cost introduced in this approach, namely, the storage space used for the approximated series (in addition to the original series) and the processing time used for the approximation computation (although it can be done in an off-line, batch mode); and 2) the result obtained by searching the approximations may not have 100% *precision* or *recall* level, i.e., it is possible that neither all resulting series meet the query criteria nor all desired series are found in the result. To minimize these problems, we suggest the following desiderata for approximation methods:

Effective The query result obtained by searching approximated series should have high precision and recall levels, i.e., most series in the result satisfy the query criteria and most series that satisfy the query criteria are in the result.

Efficient The approximation should be efficient, i.e., the cost (time and space) for computing approximations is small, the storage space used for the approximated representations is small, and query processing time using the approximations is significantly reduced from that using the original series.

Simple The approximation method should be simple to implement, i.e., it is easy to compute the approximated series from the original ones, as well as to derive content information from the approximations.

General The approximation method should be general enough so that different kinds of time series contents can be derived from a single approximated representation.

Uniform The approximation method should be uniform enough so that contents for different subseries can be derived from a single approximated representation.

Clearly, the *effective* property and the *efficient* property are two competing ones. The balance between them should be carefully chosen based on the task at hand and the involved data volume. For example, at early stages of the aforementioned interactive and iterative analysis process, the volume of involved data is high, hence it is important to reduce query processing time significantly by allowing relatively low precision and recall levels as users are usually quite flexible regarding the query result. On the other hand, when the data volume becomes low and/or users become strict about the result, it is necessary to obtain the result with high precision and recall levels. Hence, it may be helpful to provide multiple approximations with different trade-offs between effectiveness and efficiency.

Simplicity is desired for the practical purpose as simple methods are likely to reduce bugs in implementation and are easier to be extended. *Generality and uniformity* are required since the retrieval criteria may involve different kinds of contents and different subseries, even at the same time. For example, the user may ask for time series whose shape in one segment is similar to a given one, and whose trend in another segment is close to a given value. A general and uniform method is desired because it minimizes the extra cost and introduces opportunities for further optimization since only a single approximated representation is used.

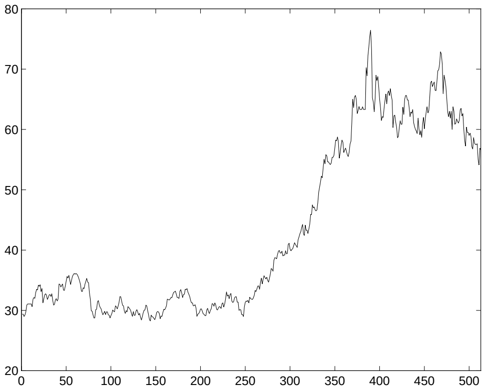


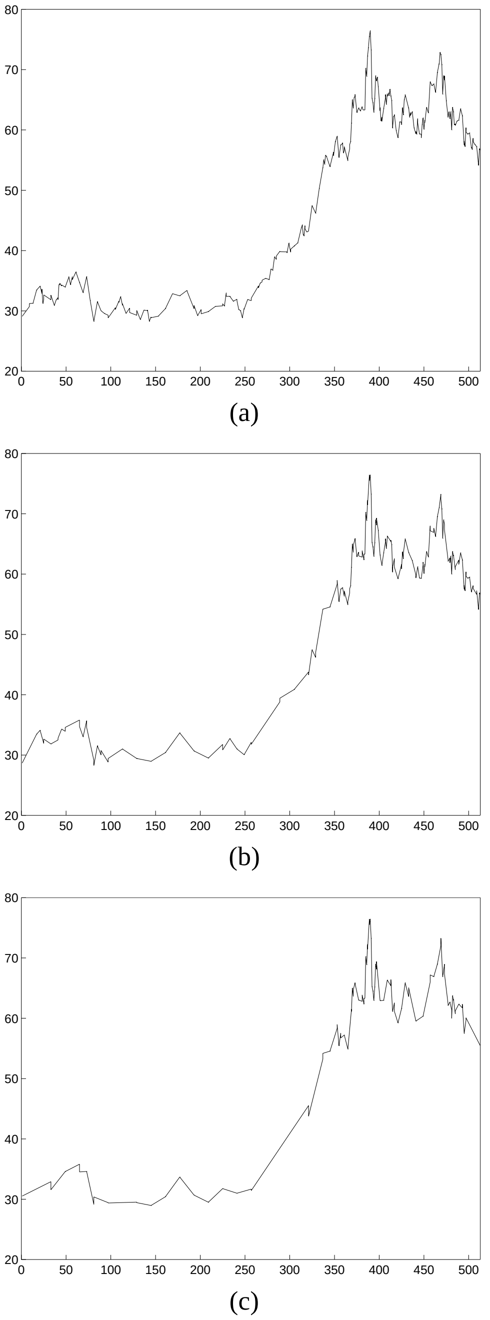
Figure 4. Daily closing prices of INTC

We now show two simple approximation methods, the first one is based on the wavelet transform, and the second based on the line fitting method. An example time series, namely, the daily closing prices of the Intel Corporation (INTC) stock from the middle of December 1993 to the middle of January 1996, is used to demonstrate these two approximation methods. The time series, which has 513 data values, is shown in Figure 4.

Wavelet transforms have been used to provide lossy compression for time series. A wavelet transform decomposes a time series into a linear combination of some given “basis” functions (called wavelets) [17]. The compression is achieved by picking up the wavelets that have the most significant coefficients in the decomposition, while ignoring all other wavelets. The linear combination of the chosen wavelets (with their coefficients) is an approximated representation of the original time series.

The problem of the above wavelet transform approach is that the approximation is not always uniform, i.e., different subseries may be treated differently. Indeed, since wavelets remained in the approximation are chosen from all the wavelets over the whole time series, a particular subseries may be approximated by several wavelets, while another subseries (in the same approximation) may be approximated by only one wavelet. The level of details in different subseries may thus be different. To obtain uniformity, our approach is to refine the above standard compression method as follows: We first use the standard method to obtain an approximation, then compare each subseries of the approximation with the corresponding subseries of the original one, and calculate distance between them. If the distance is greater than a threshold, more wavelets in that region are included into the approximation, i.e., more details are added into that subseries of the approximation.

In Figure 5, we show the approximations of the time series in Figure 4 by using the above method. The wavelet basis used is the *linear B-Spline* wavelet functions [14]. The wavelets in this case are all formed by connected line segments (i.e., the ending point of one line segment coincides with the starting point of the next line segment). Clearly, all the linear combinations of the wavelets, and hence all approximated representations obtained by this wavelet transform, are a series of line segments. Furthermore, some consecutive line segments in the approximations may be connected since many combinations of linear B-Spline wavelets generate connecting line segments. We use the averaged Euclidean distance[5] as our error measure. In Figures 5(a), (b) and (c), the thresholds used are 0.5, 0.8 and 1.1, respectively, and the numbers of line segments chosen by the above method are 208, 120 and 72, respectively.



(a)

(b)

(c)

Figure 5. Approximated representations based on wavelets transform.

In addition to the wavelet-based method described above, we have also considered a method based on line fitting. Intuitively, for each subseries, there is a best fitting line, i.e., the line segment (over the same range as the subseries) that is closest (in terms of averaged Euclidean distance) to the subseries. For a given threshold, we may ob-

[5] The averaged Euclidean distance between $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$ is defined as $\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2/n}$.

tain an approximation of the time series as follows: Starting from the first time point (call it point A) of the series, find the *farthest* point (call it point B) such that (1) when the subseries over $[A, B]$ is considered, the distance of the given time series to its best fitting line is less than the given threshold, and (2) no other point B' nearer to A than B violates condition (1). This process continues with B as the next starting point and so on.

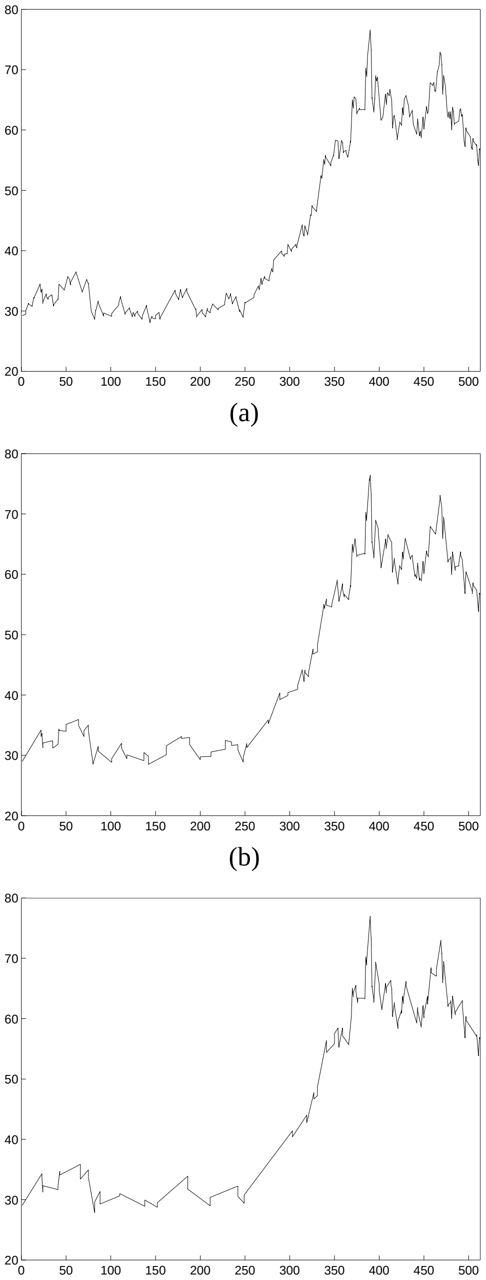
Figure 6 gives approximated representations of the INTC closing prices obtained by using the above method. The thresholds used in Figure 6(a), (b) and (c), are 0.3, 0.5 and 0.7, respectively, and the numbers of line segments are 196, 107, and 75, respectively. The thresholds are chosen so that the numbers of line segments used in the approximations are similar to the corresponding ones shown in Figure 5.

In the above two approximation methods, the line fitting method directly addresses the concern to reduce mean square error, and usually requires a smaller number of lines to meet the same threshold. However, consecutive lines in the approximation are often disconnected and more storage space may be necessary for storing the line segments. On the other hand, the wavelet based method generates more consecutive lines that share same points and hence require less space for storing line segments. However, the wavelet-based approximation requires high precomputation cost when the series are of arbitrary lengths.

4. Experiments

In this section, we present the result of experiments on two data sets showing the efficiency, effectiveness, generality and uniformity of the two approximation methods proposed above. The first data set is the closing prices of 197 North American stocks, each consisting of 513 values, and the temporal range is from late 1993 to early 1996. For each series, the B-spline wavelet approximation method is used to obtain three approximated representations and the thresholds used are 0.5, 0.8 and 1.1, respectively. Also for each series, the line fitting approximation method is used to obtain three additional approximated representations and the thresholds used are 0.3, 0.5 and 0.8, respectively.

The second data set consists of 2000 synthesized random walk series. (Note that *a large family of social and economical series, such as stock movements and exchange rates, have been successfully modeled as random walks* [11, 24].) Each series $S = x_1, x_2, \ldots, x_{513}$ is generated using the following formula: $x_1 = r_1, x_{i+1} = x_i + r_{i+1}$, where $r_1, \ldots, r_{513}$ are independent random variables uniformly distributed within $[-0.5, 0.5]$. Similarly to the stock data set, the B-spline wavelet approximation method is applied to each series with thresholds $0.2, 0.3, 0.4$, respectively, and the line fitting approximation methods is applied to each series with thresholds $0.1, 0.2, 0.3$, respectively.



(a)

(b)

(c)

Figure 6. Approximated representation based on line fitting.

The thresholds are chosen so that the corresponding approximations have the desired number of line segments (which correspond to compression ratios). Hence, there are six sets of approximated series for each data set. The average number of line segments used in each set of approximations is shown in Figure 7.

Stock, Wavelet-based appr.			
Threshold	0.5	0.8	1.1
Avg # of Line	126.2	64.8	38.9
$\frac{\text{Avg \# of Line}}{\text{series length}_{(513)}}$	24.6%	12.6%	7.6%

Stock, Line fitting appr.			
Threshold	0.1	0.2	0.3
Avg # of Line	222.5	79.6	28.7
$\frac{\text{Avg \# of Line}}{\text{series length}_{(513)}}$	43.3%	15.5%	5.6%

Random walk, Wavelet-based appr.			
Threshold	0.5	0.8	1.1
Avg # of Line	126.2	64.8	38.9
$\frac{\text{Avg \# of Line}}{\text{series length}_{(513)}}$	24.6%	12.6%	7.6%

Random walk, Line fitting appr.			
Threshold	0.1	0.2	0.3
Avg # of Line	222.5	79.6	28.7
$\frac{\text{Avg \# of Line}}{\text{series length}_{(513)}}$	43.3%	15.5%	5.6%

Figure 7. Number of lines used in the approximations

For each of these 12 sets of approximations (two data sets, two approximations methods and three thresholds for each approximation), four groups of experiments are performed to test the approximation methods for the nearest neighbor search in terms of shapes, trends, cyclic components and ACFs. The PACF is analogous to the ACF (indeed, the PACF is the "dual function" of the ACF, as the Moving Average (MA) processes and Autocorrelation (AR) processes are considered dual processes), and is not studied in the experiments.

Note that for a given series S and a search limit N, the (N-)nearest neighbor search is to find the N series that are most similar to S. In each group of the experiments, nearest neighbor searches are performed on five different subseries ranges and with five different search limits. Specifically, for search limits $N = 5, 10, 20$, and subseries ranges $R = [1, 33], [1, 65], [1, 129], [1, 257], [1, 513]$ (though subseries are NOT required to start from the first item of the whole series), the nearest neighbor searches are conducted on both the original series and the approximated ones in the following way:

Step 1: For each pair of original series, calculate the similarity measure between their subseries in range R;

Step 2: Denote the original series set as $\{S_1, \dots, S_M\}$, where $M = 197$ for the stock data set and $M = 2000$ for the synthesized data set. For each $i : 1 \le i \le M$, find the N series that are most similar to S_i, denoted $O_i(N)$, using values calculated in Step 1.

Step 3: In the approximated series set, denoted $\{S'_1, \dots, S'_M\}$, for each $i : 1 \le i \le M$, find the N series that are most similar to S'_i, denoted $A_i(N)$, in the way similar to Step 1 and 2.

Step 4: For each $i : 1 \le i \le M$, calculate the number, C_i, of series found in $A_i(N)$ that are also in $O_i(N)$. In other words, $C_i = |\{j | S_j \in O_i(N) \wedge S'_j \in A_i(N)\}|$, and C_i tells the number of correct series found by using the approximation.

The number, C_i, of correctly found series is directly related to the precision and recall levels of the query results on approximated representations. In particular, for the nearest neighbor search, precision and recall levels are always the same:

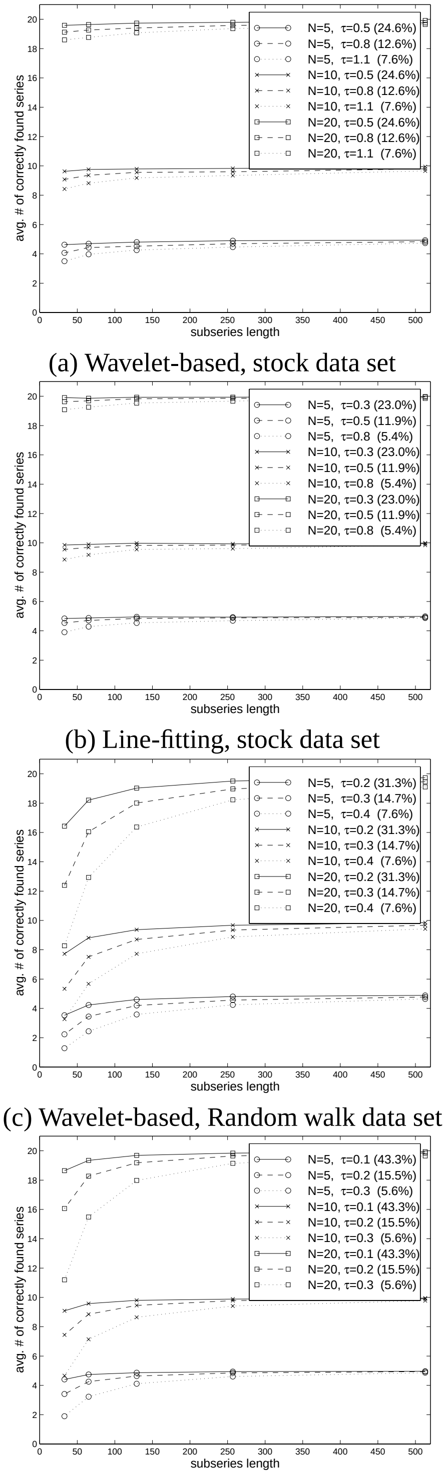
$$\text{Precision} = \frac{\text{Relevant Search Result}}{\text{All Relevant Series}} = \frac{C_i}{N},$$

$$\text{Recall} = \frac{\text{Relevant Search Result}}{\text{All Search Result}} = \frac{C_i}{N}.$$

Therefore, we use the value C_i to indicate the effectiveness of an approximation.

In Figures 8, 9, 10, and 11, we show the average of $C_1, \dots, C_M$ for different search limit N, different subseries range R and different approximations. Specifically, the average numbers of C_i for the nearest neighbor searches in terms of shape are shown in Figure 8. For example, consider the 20-nearest neighbor search on the wavelet-based approximations of stock data set with threshold 1.1 (i.e., the top dotted line in Figure 8(a)), the average numbers of correctly found series are 18.59, 18.76, 19.08, 19.37 and 19.65 for subseries of ranges $[1, 33]$, $[1, 65]$, $[1, 129]$, $[1, 257]$ and $[1, 513]$, respectively. Here, the precision (recall) levels are 92.95%, 93.8%, 95.4%, 96.85% and 98.25%, respectively, while the ratio $\frac{\text{Avg \# of line}}{513} \approx 7.6\%$. Similarly, Figure 9 shows the results from trend searches, Figure 10 shows the results from cyclic components searches, and Figure 11 shows the results from ACF searches. The window size of the moving-averaged periodograms used in Figure 10 is 7.

In general, the precision (recall) levels are reasonably high ($70\% - 95\%$) for different content types and on different subseries, except for short subseries and/or coarse approximations. Specifically, both methods work very well for shape similarity searches, quite well for trend and cyclic components similarity searches, only relatively poor for ACF similarity searches. For subseries of length 65 or less, the precision (recall) level greatly degenerates. Fortunately, in statistical time series analysis, long time series are preferred because trends, cyclic components, ACFs and PACFs



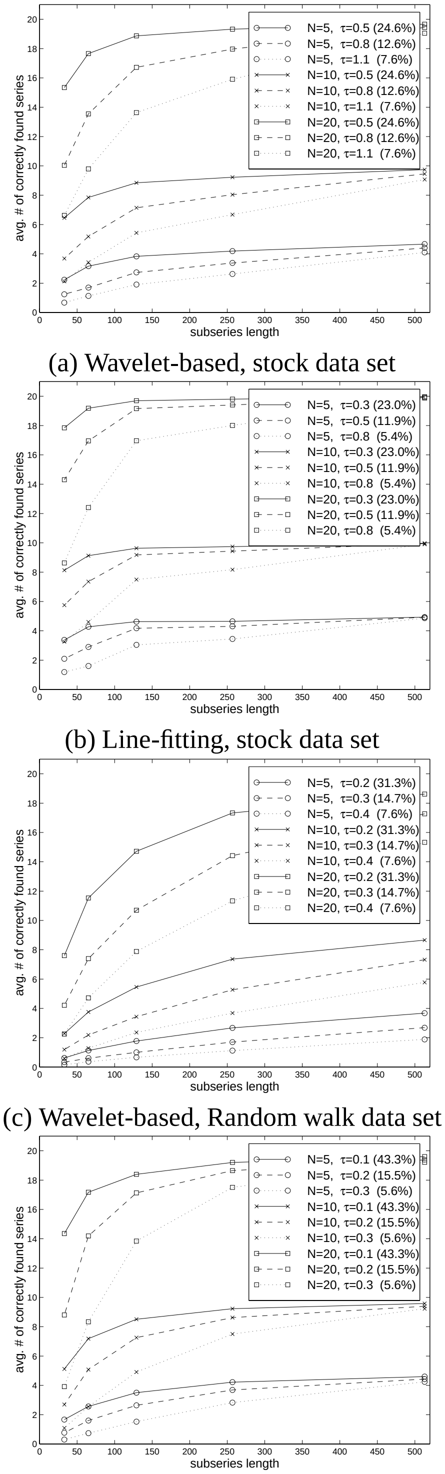
(a) Wavelet-based, stock data set

(b) Line-fitting, stock data set

(c) Wavelet-based, Random walk data set

(d) Line-fitting, Random walk data set

Figure 8. Number of correctly found series in SHAPE similarity searches



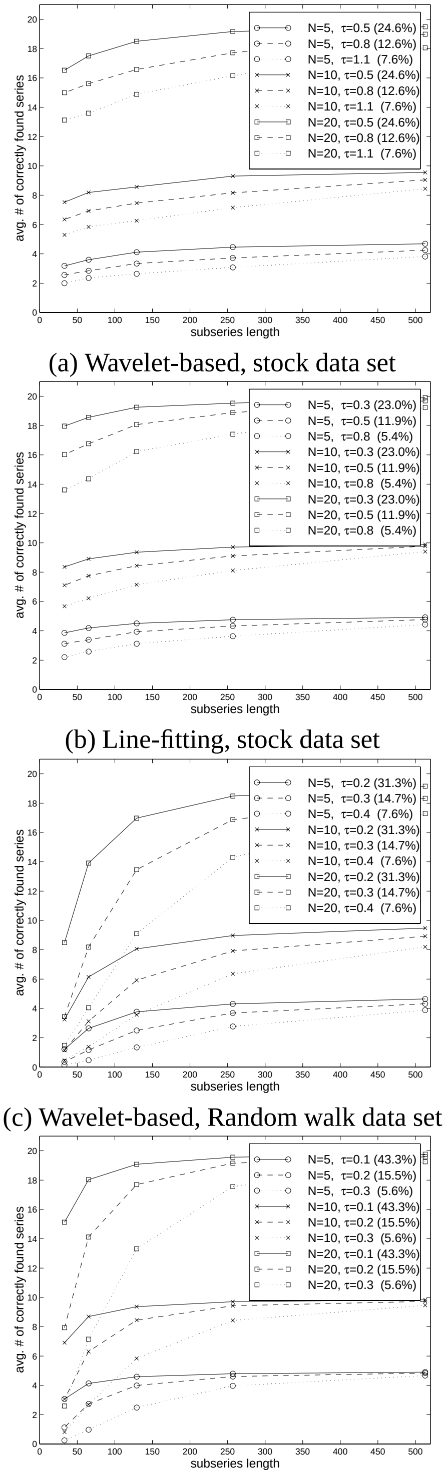
(a) Wavelet-based, stock data set

(b) Line-fitting, stock data set

(c) Wavelet-based, Random walk data set

(d) Line-fitting, Random walk data set

Figure 9. Number of correctly found series in TREND similarity searches



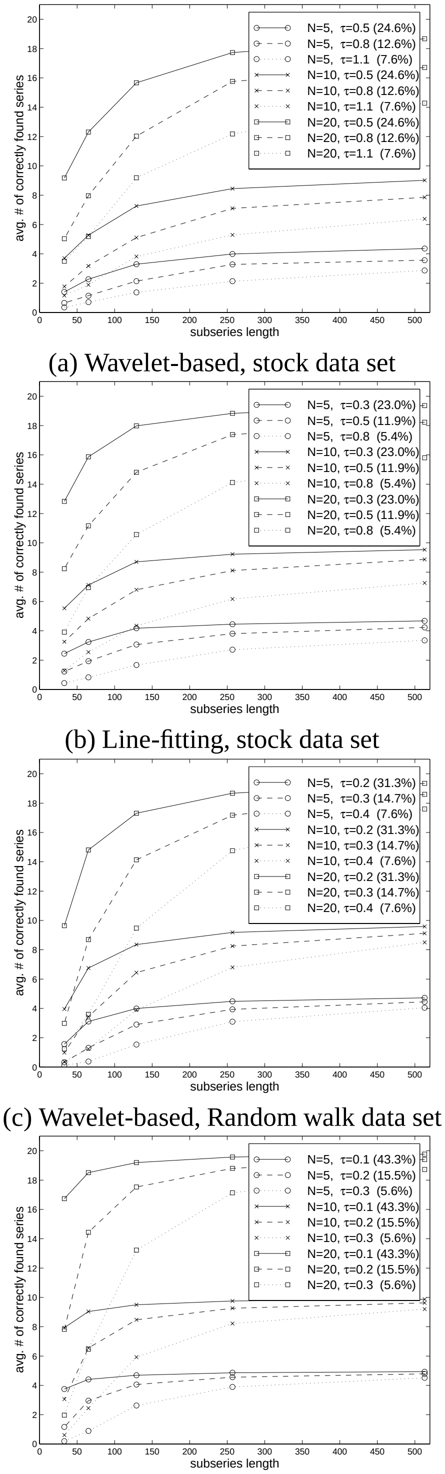
(a) Wavelet-based, stock data set

(b) Line-fitting, stock data set

(c) Wavelet-based, Random walk data set

(d) Line-fitting, Random walk data set

Figure 10. Number of correctly found series in CYCLIC COMPONENTS similarity searches



(a) Wavelet-based, stock data set

(b) Line-fitting, stock data set

(c) Wavelet-based, Random walk data set

(d) Line-fitting, Random walk data set

Figure 11. Number of correctly found series in ACF similarity searches

derived from long time series tend to have small variations and give high confidences [30].

On the other hand, the average compression ratio is usually less than $2 \times \frac{\text{Average number of lines}}{513}$ for wavelet-based approximations (since some pairs of consecutive line segments in the approximation share one end point), and roughly $2 \times \frac{\text{Average number of lines}}{513}$ for line fitting approximations. Hence, as seen in Figure 7, the compression ratios are quite high, and the extra storage space required by the approximation is small. Furthermore, the query processing time will be reduced proportionally to the compression ratio because: 1) the query processing time mainly consists of the disk access time and the computation time involved in comparing the contents from the approximations; 2) the disk access time is proportional to the storage space, and 3) the computation cost can be reduced proportionally to the compression ratio too by taking advantage of the line segment representation of approximations.

The above experiments indicate that we can reduce the query processing time by $25 - 30\%$ to achieve $85 - 90\%$ precision and recall levels (e.g., the dashed lines shown in Figure 8-11 (b) and (d)), and reduce the query processing time to 10% to achieve $60 - 70\%$ precision and recall levels (e.g., the dotted lines shown in Figure 8-11 (b) and (d)).

It is also worth mentioning that the approximation computation algorithm is efficient. The time complexity is $O(nl\log(l))$ for wavelet-based approximation method when fast wavelet transform algorithm exists, and $O(nl^2)$ for line fitting approximation method where n is the number of series and l is the length of series.

5. Related work

Time series data have been studied extensively in various disciplines. The statistical time series analysis approach, widely used in social as well as physical sciences, assumes that time series can be modeled as a combination of different independent components, and studies techniques to identify appropriate models, to estimate model parameters, to evaluate models, and to forecast time series according to the established models [16, 30]. Many commercial tools supporting this process are now available. An example is S-plus [10]. The technical analysis approach, mainly used in stock market analysis, emphasizes on different shape patterns of time series [12, 21]. Both approaches, however, focus on individual or several time series at a time, assuming that the series are given. In this paper, we study techniques to find interesting series from a large set so that they can be analyzed in detail by using these approaches.

Some languages have been proposed to support time series in databases. Unlike temporal logics, they are designed specifically to address the sequential and statistical characteristics of time series. For example, RSQL [33] introduces the order of values into SQL by using the "ROLL" clause in queries, and KSQL [28] introduces the ordered table "arrable" and provides statistical operations on the query results. Their works enhance the expressive power of database query language. In contrast, this paper studies database techniques to support a specific type of queries, namely searching time series whose contents are similar to given ones.

Time series similarity search is an active research area. Papers [20] and [27] propose some general frameworks for similarity queries. Paper [1] first proposes the idea of applying Discrete Fourier Transform to map time series into the frequency domain and building a spatial index on the first few coefficients from the frequency domain. Two series are similar if the distance between their Fourier coefficients is small. The indexing structure is used as a filter to reduce the number of series to be scanned. This idea has been extended in various directions [13, 26, 15]. When the energy of time series does not concentrate on a few specific frequencies, however, the indexing structure will not filter out most of dissimilar series and the performance of this method will not be satisfactory. Paper [2] thus proposes a new method. In this method, all the subseries of a predefined length of all the time series are considered. Two series are "similar" if most of their corresponding subseries (i.e., subseries over the same range) are close in terms of some distance measure. However, this method is not as efficient as those based on the above spatial indexing structures.

In this paper, we also pursue similarity search methods. However, while like the previous researchers, we treat a time series as a geometric curve, and use the Euclidean distance to measure the shape similarity, we also propose to treat the time series as a realization of a stochastic process and measure the similarity according to the statistical time series analysis theory. Thus, queries will be of different types. For example, we show that a direct use of the Euclidean distance measure is not suitable for finding series containing similar cyclic components.

A variety of indexing structures and techniques have been introduced recently for similarity searches of complex objects such as shape similarity for time series [6, 34, 22, 23, 7, 31, 8, 9, 18, 5]. We attack the problem from a different direction and our proposed approximation techniques may be combined with efficient indexing structures to achieve better performance.

Recently, paper [19] applies wavelet transform on a database of 20,000 images and obtained the approximations by choosing a certain fixed number of wavelets that have the largest coefficients. These representations are used as the basis to match a given image against all the images in the database. More recently, [32] uses a similar idea but chooses wavelets of the first few levels. In addition,

[32] proposes an indexing technique designed to prune the search space. In contrast, this paper focuses on time series. Paper [29] does not use wavelets. Instead, it represents a time series by breaking the series into disjoint regions and each region is approximated by a function chosen from a predefined family. In contrast to this paper, query processing on approximated representations of time series is not studied in [29].

There are also other related research on the time series database. Papers [25] and [3] examine the shape or movement of the time series, and study techniques to retrieve time series matching certain shape patterns. Their approaches can be adopted in designing the similarity measure of some time series contents, such as autocorrelation functions and partial autocorrelation functions.

6. Conclusion

This paper discussed some desired properties of approximation and two specific approximation methods for supporting similarity queries of time series based on their content information. The content types studied were shapes, trends, cyclic components, autocorrelation functions and partial autocorrelation functions. The similarity measure for each content type was carefully chosen. Nearest neighbor searches were performed on the approximated series as well as the original series to estimate the effectiveness and efficiency of the approximation methods. Experiments on both a stock price data set and a synthesized random walk data set showed that both approximation methods significantly reduce the query processing time without introducing intolerable errors for similarity searches in terms of different types of contents and on different subseries. Similar results are expected for many other social and financial series as they have been successfully modeled as random walks.

As mentioned in Section 2, the similarity measures for ACFs and PACFs are simplified in this paper. Hence, it is interesting to exploit shape or movement description language to measure their similarity. There are also other kinds of time series contents, such as the Akaike's Information Criterion (AIK), the final prediction error (FPE), etc. Each of them demands a study of its own characteristics and similarity measure.

In addition to similarity searches, there are also other kinds of content-based queries. For example, it is helpful to search time series whose contents match certain pattern specified in some descriptive language. As the approximations often preserve desired characteristics of time series contents, the study of content-based pattern queries using approximation is a promising research direction.

In many applications, the required precision and recall levels may increase gradually as the interactive process goes on. Hence it is helpful to provide multi-level approximations to meet the requirements at different stages. To avoid large extra storage cost, approximations at different levels should be related so that the total storage needed is limited. The wavelet transform provides a good starting point, as finer level approximations can be readily obtained by the combination of wavelet coefficients at the same level and the coarser level approximations.

References

[1] R. Agrawal, C. Faloutsos, and A. Swami. Efficient similarity search in sequence databases. In *The Fourth International Conference on Foundations of Data Organization and Algorithms*, Evanston, Illinois, October 1993.

[2] R. Agrawal, K. I. Lin, H. S. Sawhney, and K. Shim. Fast similarity search in the presence of noise, scaling, and translation in time-series databases. In *Proc. of the 21st VLDB Conference*, Zurich, Switzerland, September 1995.

[3] R. Agrawal, G. Psaila, E. L. Wimmers, and M. Zait. Querying shapes of histories. In *Proc. of the 21st VLDB Conference*, Zurich, Switzerland, September 1995.

[4] G. Asrar and R. Greenstone, editors. *1995 MTPE EOS Reference handbook*. NASA, 1995. `ftp://eospso.gsfc.nasa.gov/docs/handbook95.pdf`.

[5] S. Berchtold, C. Böhm, and H.-P. Kriegel. The pyramid-technique: Towards breaking the curse of dimensionality. In *Proc. ACM SIGMOD*, pages 142–153, 1998.

[6] S. Berchtold, D. A. Keim, and H.-P. Kriegel. The X-tree: An index structure for high-dimensional data. In *Proceedings of the 22th International Conference on Very Large Data Bases*, pages 28–39, Mumbai (Bombay), India, 1996.

[7] T. Bozkaya and Z. M. Özsoyoglu. Distance-based indexing for high-dimensional metric spaces. In *Proc. ACM SIGMOD*, pages 357–368, 1997.

[8] S. Brin. Nearest neighbor search in large metric space. In *Proceedings of the 21th International Conference on Very Large Data Bases*, pages 574–584, Zurich, Switzerland, 1995.

[9] P. Ciaccia, M. Patella, and P. Zezula. M-tree: An efficient access method for similarity search in metric spaces. In *Proceedings of the 23th International Conference on Very Large Data Bases*, pages 426–435, Athens, Greece, 1997.

[10] Data Analysis Products Division, MathSoft Inc., Seattle, Washington. *S-plus 5 for Unix Guide to Statistics*, September 1998.

[11] P. J. Diggle. *Time Series*. Clarendon Press, Oxford, 1990.

[12] R. Edwards and J. Magee. *Technical analysis of stock trends*. American Management Association, 1997.

[13] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos. Fast subsequence matching in time-series databases. In *Proc. ACM SIGMOD Conf.*, pages 419–429, Minneapolis, May 1994.

[14] A. Finkelstein and D. H. Salesin. Multiresolution curves. In *Proc. of SIGGRAPH*, pages 261–268, New York, 1994. ACM.

[15] D. Q. Goldin and P. C. Kanellakis. On similarity queries for time series data: Constraint specification and implementation. In *Proc. of Int'l conference on Principles and Practice of Constraint Programming*, Casis, France, 1995.

[16] J. M. Gottman. *Time-series analysis: A comprehensive introduction for social scientists*. Cambridge University Press, 1981.

[17] A. Graps. An introduction to wavelets. *IEEE Computational Science and Engineering*, 2(2), Summer 1995.

[18] A. Henrich. Adapting a spatial access structure for document representations in vector space. In *Proceeding of the Conference on Information and Knowledge Management*, pages 19–26, Rockville, Maryland, 1996.

[19] C. E. Jacobs, A. Finkelstein, and D. H. Salesin. Fast multiresolution image querying. In *Proc. of SIGGRAPH, Computer Graphics Proc., Annual Conf. Series*, 1995.

[20] H. V. Jagadish, A. O. Mendelzon, and T. Milo. Similarity-based queries. In *Proc. of ACM Symposium on Principles of Database Systems*, 1995.

[21] D. R. Jobman. *The handbook of technical analysis : a comprehensive guide to analytical methods, trading systems and technical indicators*. Cambridge, England : Probus Pub., 1995.

[22] N. Katayama and S. Satoh. The SR-tree: An index structure for high-dimensional nearest neighbor queries. In *Proc. ACM SIGMOD*, pages 369–380, 1997.

[23] K.-I. Lin, H. V. Jagadish, and C. Faloutsos. The TV-tree: An index structure for high-dimensional data. *VLDB Journal*, 3(4):517–542, 1994.

[24] B. Mandelbrot. *Fractal Geometry of Nature*. W.H. Freeman, New York, 1977.

[25] Y. Qu, C. Wang, and X. S. Wang. Supporting fast search in time series for movement patterns in multiple scales. In *CIKM*, pages 251–258, 1998.

[26] D. Rafiei and A. Mendelzon. Similarity-based queries for time series data. In *SIGMOD, Proc. of Annual Conference*, Tucson, Arizona, May 1997.

[27] N. Roussopoulos, S. Kelley, and F. Vincent. Nearest neighbor queries. In *SIGMOD, Proc. of Annual Conference*, San Jose, CA, 1995.

[28] D. Shasha. Time series in finance: the array database approach. http://cs.nyu.edu/shasha/papers/jagtalk.html, August 1998.

[29] H. Shatkay and S. B. Zdonik. Approximate queries and representations for large data sequences. In *Proc. Twelfth Int'l Conf. on Data Engineering*, 1996.

[30] R. H. Shumway. *Applied Statistical Time Series Analysis*. Prentice Hall, 1988.

[31] J. K. Uhlmann. Satisfying general proximity/similarity queries with metric trees. *Information Processing Letters*, 40(4):175–179, 1991.

[32] J. Z. Wang, G. Wiederhold, O. Firschein, and S. X. Wei. Wavelet-based image indexing techniques with partial sketch retrieval capability. In *IEEE Advances in Digital Libraries*, May 1997.

[33] X. Wang and C. Li. Expressing and optimizing order-oriented aggregation and selection. Technical report, George Mason University, March 1996.

[34] D. A. White and R. Jain. Similarity indexing with the SS-tree. In *Proc. International Conference on Data Engineering*, pages 516–523, 1996.