WARP: Time Warping for Periodicity Detection

Mohamed G. Elfeky*
Google Inc.
mgelfeky@google.com

Walid G. Aref Ahmed K. Elmagarmid
Department of Computer Sciences, Purdue University
{aref, ake}@cs.purdue.edu

Abstract

Periodicity mining is used for predicting trends in time series data. Periodicity detection is an essential process in periodicity mining to discover potential periodicity rates. Existing periodicity detection algorithms do not take into account the presence of noise, which is inevitable in almost every real-world time series data. In this paper, we tackle the problem of periodicity detection in the presence of noise. We propose a new periodicity detection algorithm that deals efficiently with all types of noise. Based on time warping, the proposed algorithm warps (extends or shrinks) the time axis at various locations to optimally remove the noise. Experimental results show that the proposed algorithm outperforms the existing periodicity detection algorithms in terms of noise resiliency.

1. Introduction

Time series data captures the evolution of a data value over time. Examples of time series data are meteorological data containing several measurements, e.g., temperature and humidity; stock prices depicted in financial market; power consumption data reported in energy companies; and event logs monitored in computer networks. Periodicity mining is a tool that helps in predicting the behavior of time series data [15]. For example, periodicity mining allows an energy company to analyze power consumption patterns and predict periods of high and low usage so that proper planning may take place.

Noisy data is inevitable in reality, either due to unreliable data sources or during data transmission. In addition, in streaming applications, data elements are subject to be dropped for processing reasons. Therefore, data mining techniques should be aware of such noisy data environments.

Discovering the periodicity rate of time series data, hereafter referred to as *periodicity detection*, has drawn the attention of the data mining research community very recently. Indyk et al. [8] have addressed this problem under the name *relaxed period*, and have developed an $O(n \log^2 n)$ time algorithm, where n is the length of the time series. We refer to the algorithm of [8] as RELAX. Elfeky et al. [7] have proposed an $O(n \log n)$ time algorithm for the same problem under the name *segment periodicity*. As the algorithm of [7] is convolution-based, we refer to it as CONV. In the data streaming context, Papadimitriou et al. [14] have proposed AWSOM that is a linear periodicity detection algorithm using Wavelets. AWSOM discovers only periodicity rates of powers of two due to the use of Wavelets to approximate the data stream. All three algorithms [8, 14, 7] suffer from the poor resilience to noise. To support this claim, we conduct an initial experiment using synthetic data, where different types of noise are considered, mainly replacement, insertion and deletion noise. This experiment studies the behavior of these algorithms: RELAX [8], CONV [7], and AWSOM [14] towards these types of noise as well as different mixtures of them. Results are given in Figure 1, in which we use the symbols "R", "I", and "D" to denote the three types of noise, respectively. Two or more types of noise can be mixed, e.g., "R I D" means that the noise ratio is distributed equally among replacement, insertion, and deletion, while "I D" means that the noise ratio is distributed equally among insertion and deletion only. Figure 1 shows how the three algorithms treat the replacement noise different from the other types of noise, and how their noise resiliency deteriorates to low accuracy levels when the amount of introduced noise increases.

In this paper, we address the problem of periodicity detection in the presence of noise. We propose a new algorithm for periodicity detection that deals efficiently with all types of noise. In Section 2, we introduce the notation used throughout the paper. Then, we spot where the previous periodicity detection algorithms [8, 14, 7] fail to capture the insertion and deletion noise. In Section 3, we present our proposed time warping based algorithm for periodicity detection in the presence of noise, along with its online version that fits the data stream model. In Section 4, we evaluate the performance of the proposed algorithm, and compare it to the previous periodicity detection algorithms. Finally, we briefly discuss the related work in Section 5, and summarize our conclusions in Section 6.

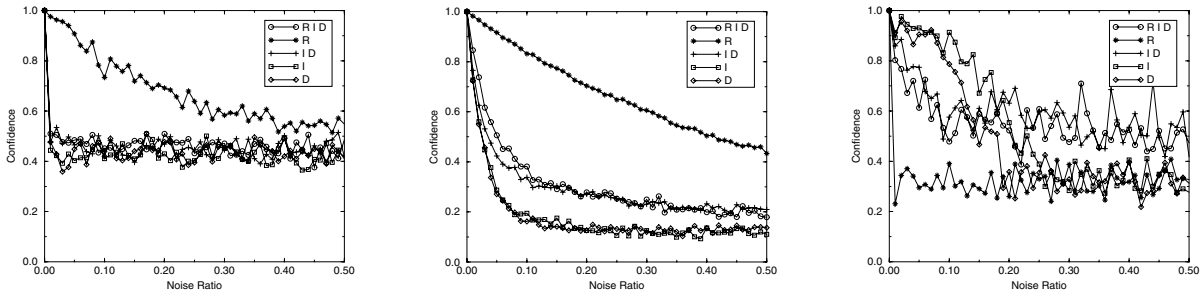
2. Preliminary

2.1. Notation

Assume that a sequence of n timestamped feature values is collected in a time series. For a given feature e, let e_i be the value of the feature at timestamp i. The time series of feature e is represented as $T = e_0, e_1, \ldots, e_{n-1}$. For example, the feature in a time series for power consumption might be the hourly power

*Work done while at Department of Computer Sciences, Purdue University

Proceedings of the Fifth IEEE International Conference on Data Mining (ICDM'05)
1550-4786/05 $20.00 © 2005 IEEE



(a) RELAX (b) CONV (c) AWSOM

Figure 1. Resilience to noise of the periodicity detection algorithms

consumption rate of a certain customer, while the feature in a time series for stock prices might be the final daily stock price of a specific company. If we discretize[1] the time series feature values into nominal discrete levels[2] and denote each level (e.g., high, medium, low, etc.) by a symbol (e.g., a, b, c, etc.), then the set of collected feature values can be denoted as $\Sigma = \{\mathsf{a}, \mathsf{b}, \mathsf{c}, \cdots\}$. Hence, we can view T as a sequence of n symbols drawn from a finite alphabet Σ.

A time series may also be a sequence of n timestamped events drawn from a finite set of nominal event types. An example is the event log in a computer network that monitors the various events. Each event type can be denoted by a symbol (e.g., a, b, c, etc.), and hence we can use the same notation above.

2.2. Hamming Distance versus Time Warping

Segment periodicity [7], and similarly the relaxed period [8], are defined as follows.

A time series T is said to be periodic with a period p if it can be divided into equal length segments, each of length p, that are "almost" similar.

For example, the time series $T = \mathsf{abcabcabc}$ is clearly periodic with a period 3. Likewise, the time series $T = \mathsf{abcabdabc}$ is periodic with a period 3 despite the fact that its second segment is not identical to the other segments. Since the symbols are considered nominal, i.e., no inherent order is assumed, Hamming distance is used as the similarity measure to compare any two segments. Hamming distance compares the symbols position-wise. In other words, the symbol at position i in a segment is compared to the symbol at position i in the other segment. Two segments are considered similar only if there are enough symbols that match position-wise. Hamming distance, however, fails to capture the similarity of two segments if they are out of phase, e.g., when few symbols are added to or deleted from one of the segments due to some sort of noise. For example, although the two segments abcdef and abgcde look similar (only an extra symbol in the third position in the second segment), the Hamming distance between these two segments is 4, which makes them less similar than they should be.

Therefore, we need another similarity metric that takes into account insertion and deletion noise. In string matching problems, this metric is called the edit distance [4]. In time series similarity search, this metric is called *time warping* [2]. Time warping allows an elastic shifting of the time axis to accommodate similar, yet out-of-phase, segments.

Similar to the dynamic computation of edit distance [4], time warping is computed dynamically and hence is commonly referred to as DTW (Dynamic Time Warping). DTW is defined as follows. Let $X = [x_0, x_1, \ldots, x_{n-1}]$ and $Y = [y_0, y_1, \ldots, y_{n-1}]$ be two finite length sequences of symbols[3], each of length n. Let $\widetilde{X}$ be the sequence X after removing the first element x_0, that is $\widetilde{X} = [x_1, x_2, \ldots, x_{n-1}]$. Then, the standard definition of DTW is

$$\mathrm{DTW}\Big(X, Y\Big) = d(x_0, y_0) + \min \begin{cases} \mathrm{DTW}\Big(X, \widetilde{Y}\Big) \\ \mathrm{DTW}\Big(\widetilde{X}, Y\Big) \\ \mathrm{DTW}\Big(\widetilde{X}, \widetilde{Y}\Big) \end{cases}$$

where $d(x_i, y_j)$ is the distance between the two symbols x_i and y_j. Since we deal with nominal symbols, the distance between two symbols is either 0 or 1 according to whether they match or not.

$$d(x_i, y_j) = \begin{cases} 0 & x_i = y_j \\ 1 & x_i \neq y_j \end{cases}$$

The DTW distance is computed as follows. An $n \times n$ matrix is constructed where the cell (i, j) contains the value $d(x_i, y_j)$. In other words, the cell (i, j) corresponds to the alignment of the symbols x_i and y_j. A warping path is a contiguous path $M = m_0, m_1, \ldots, m_{K-1}$ from cell $(0, 0)$ to cell $(n-1, n-1)$ where m_k corresponds to cell (i_k, j_k), i.e., $m_k = d(x_{i_k}, y_{j_k})$.

[1] The problem of discretizing time series into a finite alphabet is orthogonal to our problem and is beyond the scope of this dissertation. See [5, 10] for an exhaustive overview of discretizing and segmentation techniques.

[2] Nominal values are distinguished by name only, and do not have an inherent order or a distance measure.

[3] The general definition of DTW does not assume equal-length sequences. We adapted the general definition to conform to our problem, in which comparisons only take place between equal-length sequences.

There are exponentially many warping paths, each corresponds to a particular alignment between the two sequences. The warping cost of a specific warping path is the total distances to walk through this path: $\sum_{k=0}^{K-1} m_k$. In other words, the warping cost is the distance between the two sequences according to a specific alignment. The DTW distance is the minimum warping cost among all possible warping paths.

$$\text{DTW}\Big(X, Y\Big) = \min\{\textstyle\sum_{k=0}^{K-1} m_k\}$$

Therefore, this minimum warping path can be found using dynamic programming to evaluate the cumulative distance $\gamma(i, j)$ as the distance $d(x_i, y_j)$ found in the current cell and the minimum of the cumulative distances of the adjacent cells.

$$\gamma(i, j) = d(x_i, y_j) + \min\{\gamma(i-1, j-1), \gamma(i-1, j), \gamma(i, j-1)\}$$

The cumulative distance at the last cell $(n-1, n-1)$ corresponds to the sought DTW distance.

$$\text{DTW}\Big(X, Y\Big) = \gamma(n-1, n-1)$$

Clearly, the time and space complexity of computing the DTW distance is $O(n^2)$. Figure 2 gives an example for the DTW matrix constructed for the two segments abcdef and abgcde, where the minimum warping path is circled.

	a	b	c	d	e	f
a	⓪	1	1	1	1	1
b	1	⓪	1	1	1	1
g	1	①	1	1	1	1
c	1	1	⓪	1	1	1
d	1	1	1	⓪	1	1
e	1	1	1	1	⓪	①

Figure 2. An example for the DTW matrix

Note that the Hamming distance can be seen as a special case of time warping where the warping path is constrained to be the diagonal, i.e., $i_k = j_k = k$.

3. Time Warping for Periodicity Detection

The main idea of periodicity detection is that when we shift the time series p positions and compare the original time series to the shifted version, we find both time series very similar if p is a candidate period value. Let $T^{(p)}$ denote the time series T after being shifted p positions. For example, if T = abcabdabc, then shifting T three positions results in $T^{(3)}$ = ∗∗∗abcabd, where the symbol $*$ denotes the "don't care" symbol. Comparing T to $T^{(3)}$ results in four matches out of six possible matches. We argue that such an occurrence of a large number of matches corresponds to a candidate period for the time series in hand. To ascertain this argument, assume that comparing T with $T^{(p)}$ results in $n-p$ matches for a certain index p and a time series T of length n. Then, $e_p = e_0, e_{p+1} = e_1, \ldots, e_{n-1} = e_{n-1-p}$, and also, $e_{2p} = e_p, e_{2p+1} = e_{p+1}, \ldots$, etc., which means that the segment of length p is periodic and p is a *perfect* period for T. If for another index q, comparing T with $T^{(q)}$ results in a number of matches slightly less than $n-q$ due to a few mismatches, then q can be considered a candidate period for T.

In the next sections, we introduce our proposed algorithm that employs time warping for periodicity detection in the presence of noise. Our proposed algorithm is called WARP (WArping foR Periodicity). First, the algorithm is presented for traditional time series, where the whole time series is available (stored) for processing (Section 3.1). Then, the online version of WARP, which fits the data stream model, is presented (Section 3.2).

3.1. The WArping foR Periodicity (WARP) Algorithm

	e_0	e_1	$\cdots$	$\cdots$	e_{n-1}
e_0	0	$d(e_0, e_1)$	$\cdots$	$\cdots$	$d(e_0, e_{n-1})$
e_1	$d(e_1, e_0)$	0	$\cdots$	$\cdots$	$d(e_1, e_{n-1})$
.	.	.		.	.
.	.	.		.	.
.	.	.		.	.
e_{n-1}	$d(e_{n-1}, e_0)$	$d(e_{n-1}, e_1)$	$\cdots$	$\cdots$	0

Figure 3. The DTW matrix of a time series $T = e_0, e_1, \ldots, e_{n-1}$

The main idea of our proposed WARP algorithm is to make use of the DTW matrix described earlier. Figure 3 shows the DTW matrix constructed by comparing a time series $T = e_0, e_1, \ldots, e_{n-1}$ to itself. Clearly, the matrix is symmetric and all the diagonal elements are equal to 0. The minimum warping path is the diagonal, i.e., the DTW distance between a time series and itself is 0, which is trivial. However, this DTW distance is not our concern here. Our concern is rather the DTW distances between T and all its shifted versions. The key observation is that the DTW matrix contains all the comparisons between all the symbols of the time series. Each subdiagonal in the DTW matrix contains the elements that correspond to comparing the original time series T with one of its shifted versions. For example, the first subdiagonal, which starts from cell $(0, 1)$ and ends at cell $(n-2, n-1)$, contains $d(e_0, e_1), d(e_1, e_2), \ldots, d(e_{n-2}, e_{n-1})$ that correspond to comparing T with $T^{(1)}$. Similarly, the second subdiagonal, which starts from cell $(0, 2)$ and ends at cell $(n-3, n-1)$, corresponds to comparing T to $T^{(2)}$. Consequently, the ith subdiagonal, which starts from cell $(0, i)$ and ends at cell $(n-i-1, n-1)$, corresponds to comparing T to $T^{(i)}$. Therefore, the DTW distance between T and $T^{(i)}$ can be computed by finding the minimum warping path starting from cell $(0, i)$.

Formally, consider the DTW matrix shown in Figure 3 of a specific time series T of length n. For each possible period value $p = 1, 2, \ldots, n/2$, we compute the minimum warping path M_p starting from cell $(0, p)$. The warping cost that corresponds to the warping path M_p represents the DTW distance between T and $T^{(p)}$, and can be computed dynamically as the cumulative distance $\gamma(n-p-1, n-1)$.

$$\text{DTW}\Big(T, T^{(p)}\Big) = \gamma(n-p-1, n-1)$$

A low value of DTW$\left(T, T^{(p)}\right)$ indicates a high similarity between T and $T^{(p)}$, and consequently p can be considered a candidate period value for the time series T. The maximum value of DTW$\left(T, T^{(p)}\right)$ is $n-p$, which corresponds to aligning the symbols without any shifts with different symbols at each position. Therefore, the confidence of a period value p is $\frac{n-p-\text{DTW}\left(T,T^{(p)}\right)}{n-p}$, and a candidate period value is the one whose confidence is larger than or equal to the periodicity threshold τ.

	a	b	a	c	d	e
a	0	1	⓪	1	1	1
b	1	0	①	1	1	1
a	0	1	⓪	1	1	1
c	1	1	1	⓪	1	1
d	1	1	1	1	⓪	1
e	1	1	1	1	1	⓪

(a) The Diagonal Problem

	a	b	a	c	d	e
a	∞	1	⓪	1	1	1
b	1	∞	1	①	1	1
a	0	1	∞	1	①	1
c	1	1	1	∞	1	①
d	1	1	1	1	∞	1
e	1	1	1	1	1	∞

(b) Solution in WARP

Figure 4. The DTW matrix of $T =$ abacde

We can spot two problems in this technique. First, the zero values in the diagonal will drag the minimum warping paths towards the diagonal. If a minimum warping path of a specific period value p coincides with the diagonal, this means that the p-positions shift in $T^{(p)}$ has been ignored in aligning T with $T^{(p)}$. For example, Figure 4(a) shows the DTW matrix for $T =$ abacde, where the minimum warping path for $p = 2$ is circled. To solve this problem, our WARP algorithm replaces the zero values in the diagonal by large values (∞). These large values push the minimum warping path away from the diagonal. For example, Figure 4(b) shows the minimum warping path for $p = 2$ of the same DTW matrix of Figure 4(a) after altering the diagonal values to ∞.

The second problem is that if p turns out to be a candidate period value, the pth subdiagonal would contain so many zeros that it might drag the adjacent minimum warping paths, which is similar to the first problem when the diagonal drags the subsequent minimum warping paths. For example, since $p = 4$ is a perfect period for $T =$ abcdabcd, the DTW matrix would suggest that $p = 3$ is also a candidate period. The warping costs of a candidate period value and the adjacent period values take the shape shown in Figure 5. Therefore, the obvious solution would be to consider only the minimum warping cost to indicate a better candidate period value. Consequently, our WARP algorithm considers all local minima warping costs to indicate candidate period values.

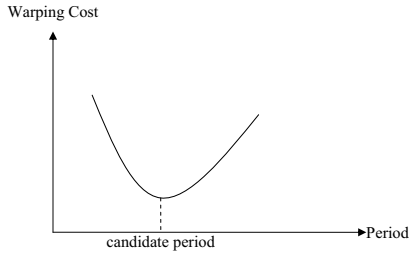


Figure 5. The shape of the warping costs around a candidate period value

3.2. Online WArping foR Periodicity

The key issue to perform the WARP algorithm online is to maintain the DTW matrix over the continuous arrival of data. We use the notion of a single sliding window that slides over the data stream. Let $S = e_0, e_1, \ldots$ be an infinite length data stream. Consider a window of length w, and let $S_{i,w}$ denote the portion of the data stream S that is of length w and starts at position i. Performing the WARP algorithm over the very first window $S_{0,w}$ computes the DTW matrix and discovers all potential periods of values ranging from 1 to $w/2$. Let the window slide $z \leq w$ positions such that the current portion of the data stream is $S_{z,w}$. The WARP algorithm is performed over the current portion, and a new DTW matrix is computed. The two matrices formed from the previous and the current portions should be combined to compute the whole DTW matrix of the entire S of length $z+w$. Figure 6 shows that DTW matrix, where the shaded upper left region reflects the DTW matrix of $S_{0,w}$, and the shaded lower right region reflects the DTW matrix of $S_{z,w}$. The unshaded regions are those whose values cannot be determined directly since sliding the window z positions means losing the first z symbols of the data stream, and hence they cannot be compared to the new arrived symbols.

Since all the values are important in searching for the minimum warping path, our online WARP algorithm tries to estimate the values at those unshaded regions of the DTW matrix of Figure 6. Recall that the distance between two symbols is either 0 or 1 according to whether they match or not, i.e., $u = v \Leftrightarrow d(u, v) = 0$ for any two symbols u and v. Assume that we would like to estimate the value $d(e_i, e_j)$ at cell (i, j) where $0 \leq i < z$ and $w - 1 < j \leq z + w - 1$. In other words, the symbol e_i is one of the lost symbols according to sliding the window, and the symbol e_j is one of the new arrived symbols. Let e_k be one of the symbols that overlap between the previous and current portions of the data stream, i.e., $z \leq k \leq w - 1$. Clearly, as in Figure 6, the cells (i, k) and (k, j) are in the shaded regions, and hence the values $d(e_i, e_k)$ and $d(e_k, e_j)$ are known. Therefore, we can derive $d(e_i, e_j)$ according to the truth table and justifications given in Table 1. In order to resolve the undetermined value of $d(e_i, e_j)$ when $d(e_i, e_k) = d(e_k, e_j) = 1$, we can iterate over all possible

Table 1. Truth table for $d(e_i, e_j)$

$d(e_i, e_k)$	$d(e_k, e_j)$	$d(e_i, e_j)$	Justification
0	0	**0**	$(e_i = e_k \wedge e_k = e_j) \Rightarrow e_i = e_j$
0	1	**1**	$(e_i = e_k \wedge e_k \neq e_j) \Rightarrow e_i \neq e_j$
1	0	**1**	$(e_i \neq e_k \wedge e_k = e_j) \Rightarrow e_i \neq e_j$
1	1	**?**	$(e_i \neq e_k \wedge e_k \neq e_j) \not\Rightarrow$ neither $e_i = e_j$ nor $e_i \neq e_j$

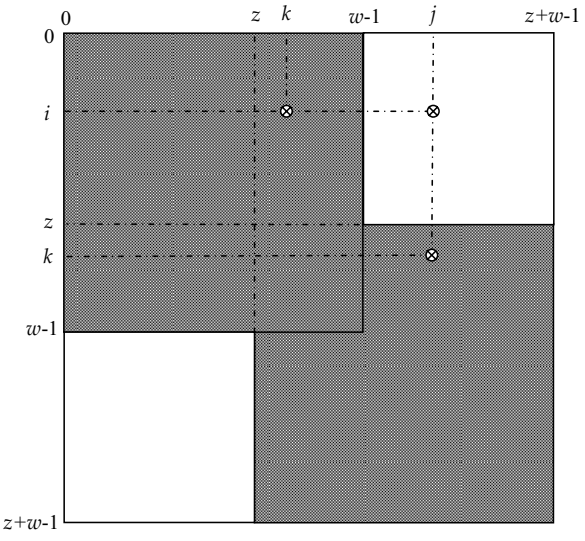


Figure 6. The DTW matrix of $S_{0,z+w}$

values of k ($z \leq k \leq w-1$) until $(e_i, e_k) = 0 \vee d(e_k, e_j) = 0$, where the value $d(e_i, e_j)$ can be determined. If for all values of k, $d(e_i, e_k) = d(e_k, e_j) = 1$, then we can conservatively estimate $d(e_i, e_j)$ to have the value 1.

The value of z is required to be less than or equal to the window length w so that we get an overlap region between the two shaded regions of the DTW matrix. This overlap region is essential to derive the values in the unshaded regions. Moreover, we can observe that the higher the value of z, the more values to be estimated, and so the DTW matrix becomes less accurate. However, the lower the value of z is, the slower the process of advancing the data stream gets. As a compromise, we choose z to be equal to $w/2$. In other words, we slide a window of length w a number of positions equal to $w/2$.

The selection of the window length w has another tradeoff. Whereas a small window length is required for early and real-time output, it slows down the process of advancing the data stream. Moreover, a large window length reduces the number of times the sliding occurs, which consequently improves the accuracy since the estimation occurs less frequently. However, we are bounded by the maximum memory size allocated for the DTW matrix. Clearly, the DTW matrix needs $O(n^2)$ memory, where n is the so-far length of the data stream. Therefore, whenever the memory allocated for keeping the DTW matrix is consumed entirely, the online WARP algorithm throws away the first w rows and w columns of the DTW matrix, leaving only the bottom right $(n-w)^2$ cells. This behavior conforms to the fact that the recent portion of the data stream is of more interest than the past ones.

4. Experimental Study

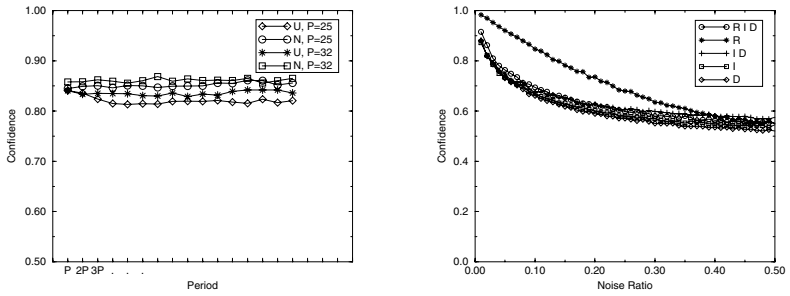
This section contains the results of an experimental study that examines various aspects of both WARP and Online WARP. Experiments are conducted using synthetic data in order to examine the accuracy, resilience to noise and the time performance of the proposed algorithms. The three periodicity detection algorithms: RELAX [8], CONV [7], and AWSOM [14] are compared with our proposed algorithms throughout the experiments.

We generate controlled synthetic time series data by tuning some parameters, namely, data distribution, period, alphabet size, type, and amount of noise. Both uniform and normal data distributions are considered. Types of noise include replacement, insertion, deletion, or any combination of them. Inerrant data is generated by repeating a pattern, of length equal to the period, which is randomly generated from the specified data distribution. The pattern is repeated till it spans the specified time series length. Noise is introduced randomly and uniformly over the whole time series. Replacement noise is introduced by altering the symbol at a randomly selected position in the time series by another symbol. Insertion or deletion noise is introduced by inserting a new symbol or deleting the current symbol at a randomly selected position in the time series. Unless otherwise stated, all the experiments are conducted using noisy data with 10% mixture of noise (replacement, insertion and deletion noises).

4.1. WARP

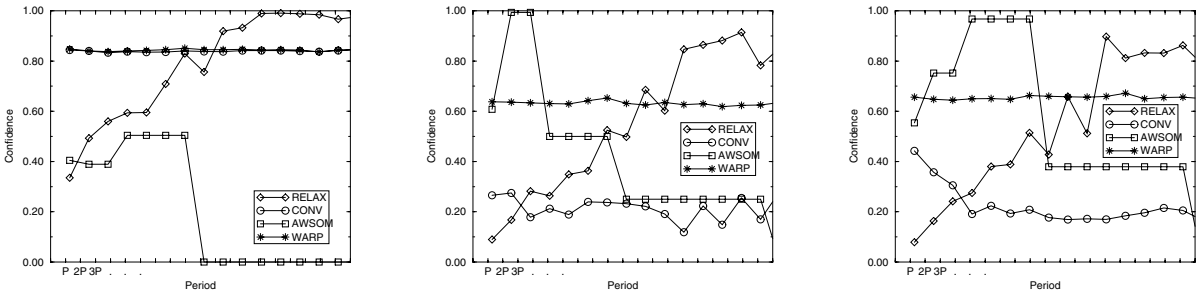
The first set of experiments inspects the accuracy of WARP with respect to the discovered potential periods. The accuracy measure that we use is the ability of the algorithm to detect the periodicities that are artificially embedded into the synthetic data. To discover a period accurately, it is not enough to discover it at any periodicity threshold value. In other words, the periods discovered with a high periodicity threshold value are better candidates than those discovered with a lower periodicity threshold value. Therefore, we define the confidence of a discovered period to be the minimum periodicity threshold value required to detect this period. The accuracy is measured by the average confidence of all the periods that are embedded artificially into the synthetic data. Figure 7(a) gives the accuracy results of WARP for different combinations of the embedded period and the data distribution. Figure 8 compares the accuracy of WARP to that of RELAX, CONV, and AWSOM.

Figure 7(a) shows an unbiased behavior of WARP with respect to the period value, and shows that WARP is able to discover all the embedded periods at 85% confidence level. Figures 8(b) and 8(c) show that WARP is much more accurate than CONV when insertion and deletion noise are considered. In general, Figure 8 shows



(a) Accuracy (b) Resilience to noise

Figure 7. Accuracy and resilience to noise of WARP



(a) Only Replacement Noise (R) (b) Insertion and Deletion Noise (I D) (c) Mixture of Noise (R I D)

Figure 8. Accuracy of WARP versus RELAX, CONV, and AWSOM

that WARP has a uniform accuracy with respect to all the period values, rather than the bias of RELAX towards the longer periods and the bias of AWSOM towards the shorter periods.

Figures 7(b) and 9 give the results of inspecting WARP's resilience to noise and comparing it to RELAX, CONV, and AWSOM. Comparing Figure 7(b) to Figure 1 shows that WARP is much more resilient to all types of noise. Even when insertion and deletion noise are mixed with replacement noise, the accuracy does not deteriorate badly as is the case with the previous periodicity detection algorithms. These results are further supported by the results given in Figure 9, where different mixtures of noise are considered. Figure 9(a) shows that WARP is as resilient to replacement noise as RELAX and CONV, and is more resilient to replacement noise than AWSOM. With respect to insertion and deletion noise, Figures 9(b) and 9(c) show that WARP is as resilient as AWSOM, and is more resilient than RELAX and CONV. Although WARP is as resilient as AWSOM to insertion and deletion noise, Figure 9(c) shows that AWSOM accuracy fluctuates irregularly unlike WARP whose accuracy decreases monotonically with the increase in the noise ratio.

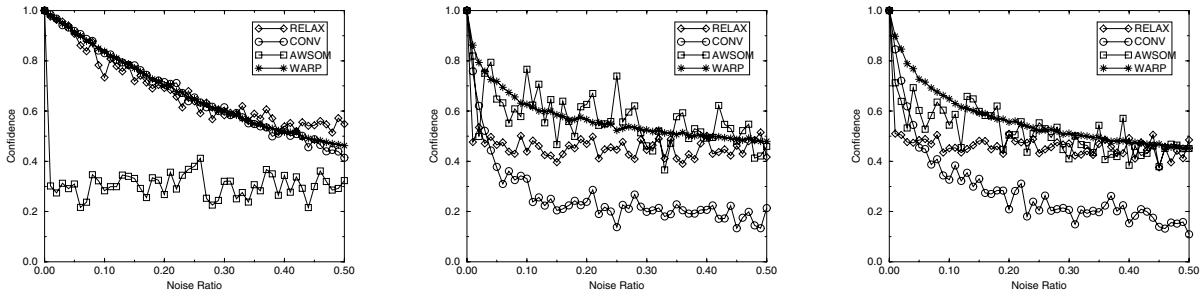
WARP's better accuracy and resilience to noise does not come for free. Figure 10(a) shows that WARP takes more processing time than CONV. This behavior is due to the fact that the time complexity of WARP is $O(n^2)$ while the time complexity of CONV is $O(n \log n)$. Note that the time complexity of RELAX is $O(n \log^2 n)$, which is slower than CONV, and hence we selected CONV for the time performance comparison. Similarly, Figure 10(b) shows that Online WARP takes much more processing time than AWSOM. AWSOM is a Wavelet-based algorithm that takes $O(n)$ time to compute. In conclusion, WARP trades processing time for more accurate periods and more resilience to noise.

4.2. Online WARP

The next set of experiments studies Online WARP in terms of its accuracy and time performance. The two variables that control the behavior of Online WARP are the window length w and the sliding size z.

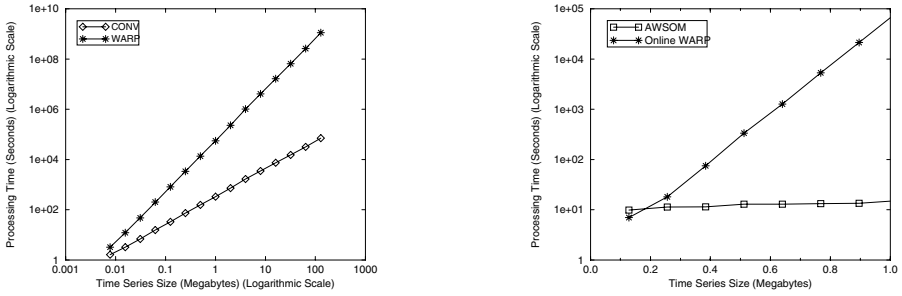
Figure 11(a) gives the accuracy results while varying the window length for three different sliding sizes. Figure 11(a) shows that increasing the window length improves the accuracy up to a certain level when the window length is enough to capture all embedded periods.

Figure 11(b) gives the accuracy results while varying the sliding size for three different window lengths. Figure 11(b) shows that decreasing the sliding size improves the accuracy due to the increase in the overlap region of Figure 6, which helps in better estimation of the values in the unshaded regions. If the sliding size z is made equal to the window length w, there would be no overlap



(a) Only Replacement Noise (R) (b) Insertion and Deletion Noise (I D) (c) Mixture of Noise (R I D)

Figure 9. Resilience to noise of WARP versus RELAX, CONV, and AWSOM



(a) WARP versus CONV (b) Online WARP versus AWSOM

Figure 10. Time Performance of WARP and Online WARP

region. Consequently, all the values to be estimated are set to 1, which deteriorates the accuracy.

To evaluate the time performance of Online WARP, Figure 12 shows the time behavior of Online WARP with respect to varying the window length and the sliding size. Figure 12(a) shows that increasing the window length has a trend of reducing the processing time since the time needed to cover the whole data stream is reduced. Moreover, Figure 12(b) shows that increasing the sliding size reduces the processing time because of two reasons: (i) the number of values to be estimated decreases and therefore less time is needed, (ii) less time is needed to cover the whole data stream. This final result shows that there is a tradeoff for selecting the value of the sliding size and that $z = w/2$ can be considered a fair selection.

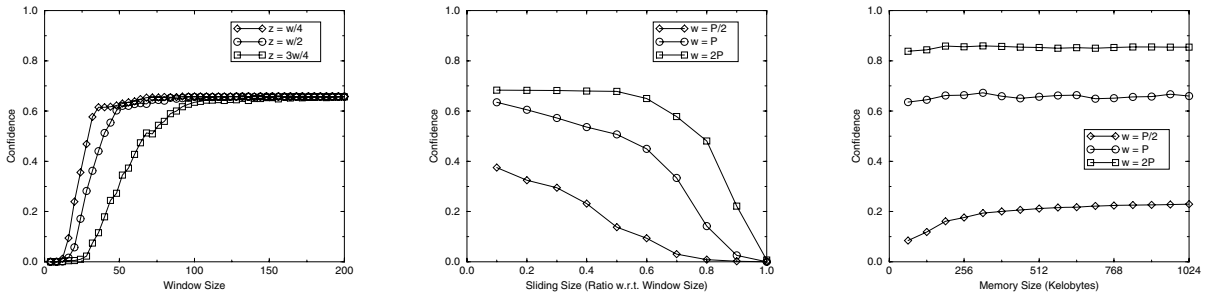
The last experiment studies the accuracy of Online WARP under different allocated memory sizes with three different window lengths. Figure 11(c) gives the results of this experiment. Figure 11(c) shows that varying the memory size does not affect the accuracy as long as the window length is enough to capture the embedded periods. When the window length is so small that it does not capture the embedded periods, the accuracy decreases with the decrease in the allocated memory size. This behavior shows that although the space complexity of Online WARP is high, Online WARP achieves high accuracy levels under small allocated memory as long as the sliding window length is selected to cover the embedded periods.

5. Related Work

In Section 1, we have discussed the significant periodicity detection work in time series data and in data streams. In this section, we cover more work related to periodicity detection, and we cover the use of dynamic time warping in data mining applications.

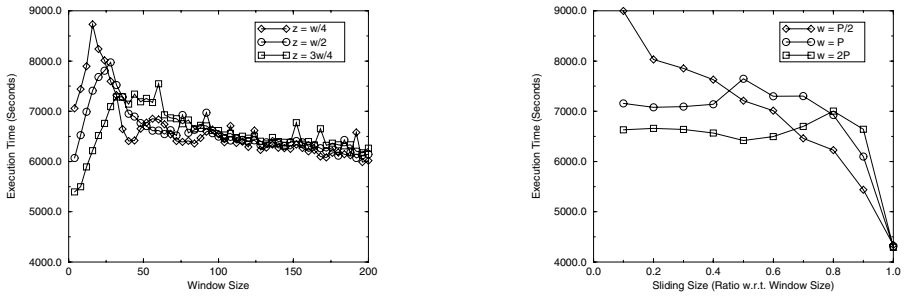
Ma and Hellerstein [13] have developed a linear distance-based algorithm for discovering the potential periods with respect to the symbols of the time series. In other words, the algorithm of [13] discovers the periodicities of the symbols of the time series rather than the periodicity of the entire time series. Elfeky et al. [6] have given a formal definition for this type of periodicity, termed *symbol periodicity*, and have developed a one-pass convolution-based algorithm for symbol periodicity detection.

Berndt and Clifford [2] have introduced the concept of DTW to the data mining community. They have showed how to apply DTW as a time series similarity measure. Agrawal et al. [1] have acknowledged the benefit of DTW over the Euclidean distance as a similarity measure in the presence of noise for time series data. However, they have avoided using DTW for its resistance to indexing and its intensive computation. Since then, much work has focused on indexing DTW [12, 9], and some work has focused on



(a) Window size (b) Sliding size (c) Allocated memory

Figure 11. Accuracy of Online WARP



(a) (b)

Figure 12. Time Performance of Online WARP

reducing its processing time [11, 3].

6. Conclusions

In this paper, we have proposed a time warping algorithm, named WARP, for periodicity detection in the presence of noise. To handle efficiently all types of noise, WARP extends or shrinks the time axis at various locations to optimally remove the noise. Furthermore, we have proposed an online version of WARP that fits the data stream model. An empirical study using synthetic data shows that there is a tradeoff between noise resiliency and time performance. WARP is more noise resilient, yet requires more processing time, than the previous periodicity detection algorithms. Moreover, Online WARP is shown empirically to be reasonably accurate, even under low memory resources.

References

[1] R. Agrawal, K. Lin, H. Sawhney, and K. Shim. Fast similarity search in the presence of noise, scaling, and translation in time series databases. In *VLDB*, 1995.

[2] D. Brendt and J. Clifford. Using dynamic time warping to find patterns in time series. In *AAAI Workshop on Knowledge Discovery in Databases*, 1994.

[3] S. Chu, E. Keogh, D. Hart, and M. Pazzani. Iterative deepening dynamic time warping for time series. In *SDM*, 2002.

[4] M. Crochemore and W. Rytter. *Text Algorithms*. Oxford University Press, 1994.

[5] C. Daw, C. Finney, and E. Tracy. A review of symbolic analysis of experimental data. *Review of Scientific Instruments*, 74(2):915–930, 2003.

[6] M. Elfeky, W. Aref, and A. Elmagarmid. Using convolution to mine obscure periodic patterns in one pass. In *EDBT*, 2004.

[7] M. Elfeky, W. Aref, and A. Elmagarmid. Periodicity detection in time series databases. *IEEE TKDE*, 17(7), July 2005.

[8] P. Indyk, N. Koudas, and S. Muthukrishnan. Identifying representative trends in massive time series data sets using sketches. In *VLDB*, 2000.

[9] E. Keogh. Exact indexing of dynamic time warping. In *VLDB*, 2002.

[10] E. Keogh, S. Chu, D. Hart, and M. Pazzani. Segmenting time series: A survey and novel approach. In M. Last, A. Kandel, and H. Bunke, editors, *Data Mining in Time Series Databases*. World Scientific Publishing, June 2004.

[11] E. Keogh and M. Pazzani. Scaling up dynamic time warping for datamining applications. In *KDD*, 2000.

[12] S. Kim, S. Park, and W. Chu. An index-based approach for similarity search supporting time warping in large sequence databases. In *ICDE*, 2001.

[13] S. Ma and J. Hellerstein. Mining partially periodic event patterns with unknown periods. In *ICDE*, 2001.

[14] S. Papadimitriou, A. Brockwell, and C. Faloutsos. Adaptive, hands-off stream mining. In *VLDB*, 2003.

[15] A. Weigend and N. Gershenfeld. *Time Series Prediction: Forecasting the Future and Understanding the Past*. Addison-Wesley, Reading, Massachusetts, 1994.