Affine and Regional Dynamic Time Warping

by

Tsu-Wei Webber Chen

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Systems Design Engineering

Waterloo, Ontario, Canada, 2014

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Time series are an ubiquitous form of data prevalent in everyday life, and their analysis has gathered immense interest in many domains. Pointwise matches between two time series are of great importance in time series analysis, and dynamic time warping (DTW) has been widely known to provide reasonable matches. There are situations where time series alignment should be invariant to scaling and offset in amplitude or certain regions of a time series should be strongly reflected in the pointwise matches. Two different variants of DTW, affine DTW (ADTW) and regional DTW (RDTW), are proposed to handle scaling and offset in amplitude and regional emphasis respectively. Furthermore, ADTW and RDTW can be combined in two different ways to generate alignments that incorporate advantages from both methods. In global-affine regional DTW (GARDTW), the affine model is applied globally to the entire time series with regional emphasis, whereas in local-affine regional DTW (LARDTW), the affine model is applied locally to each region which are then emphasized. Alignments produced by the proposed methods are evaluated on simulated datasets and their associated difference measures are tested on real datasets. The proposed methods are found to significantly outperform DTW when an evaluated dataset meets the models or preferences of the proposed methods.

Acknowledgements

I would like to thank Dr. Stashuk and Meena with whom I have had many interesting discussions that allowed this thesis to come to fruition. I would also like to thank my father whose support is vital to my pursuit of graduate studies.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

A time series is a sequence of values that are typically arranged in a chronological order in time, and data of such form is abundant in everyday life. In news, daily stock prices, monthly unemployment rate, daily temperature and humidity are reported over time. In scientific studies, much experimental data is gathered with a wide variety of methods over time as well. Naturally, analysis of time series has captured immense interest across many domains. In finance, analysis of stock market prices over time can reveal insights into financial market meltdowns [16]. In biology, gene expression data are analyzed to infer models of dynamic biological systems [3]. In physics, analysis of the area of the solar surface covered by sunspots over time can offer a better understanding of solar-type stars [10]. In geology, analysis of seismic time series is extremely valuable in predicting future earthquake activities [18]. In engineering, vibration measurements over time are analyzed to diagnose faults in rotating machinery [9].

Discovery of a set of matches between points in two time series can be tremendously useful for analysis. If point a in time series s is of high interest, researchers may also be interested in finding the point that best matches point a in another time series t. Points of interest can include hints of financial meltdown from stock market prices, regulatory genes from gene expression data, prominent sunspot activities from the area of the solar surface covered by sunspots, earthquake activities from seismic data, or vibrations indicative of faults from machine maintenance data.

One naive way of matching points in two time series of equal length is to match points vertically. However, two similar time series might be subject to non-linear temporal variations. Figure 1.1 illustrates a set of matches between two similar times series using vertical matches and another more realistic set of matches that account for non-linear temporal

variations. A match between two points is illustrated by connecting the two points with a line. Visually, the matches obtained by considering temporal variations is better than the naive vertical matches, because the local peaks of the two time series tend to be matched together and the local valleys of the two time series tend to be matched together.

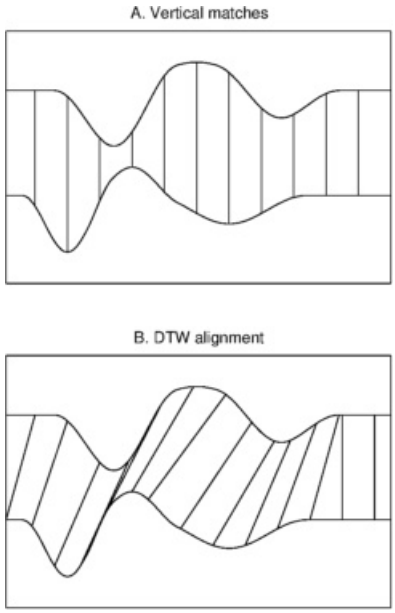

Figure 1.1: Vertical matches and DTW alignments on two time series with non-linear temporal variations.

Dynamic time warping (DTW) is one of the many methods that match points in two time series based on the assumption that non-linear temporal variations might exist. DTW also produced the reasonable matches illustrated in Figure 1.1. This thesis focuses on DTW, because it is a widely known method for matching points in two time series and it has enjoyed much success in many domains [20][6][19][29]. In particular, a comprehensive survey of 8 common times series difference measures applied to 38 different data sets from a wide variety of domains showed supporting evidence that DTW provides the best time series difference measure across many domains [7]. Among the 8 surveyed difference measures, 5 of them possess the ability to overcome temporal variations in different ways. DTW generates matches between points in two time series that account for temporal variations, and the associated time series difference measure can be easily computed based on those matches by summing the distance between a pair of matched points across all matches. As such, it is reasonable to postulate that the matches produced by DTW tend

to be good if the associated difference measure is good.

While DTW has been successfully applied across many domains, Keogh et al. observed that it can produce pathological alignments [15]. Keogh et al. pointed out two disadvantages: 1) a single point in one time series can be matched to a large number of points in another time series and 2) DTW may fail to find obvious alignments in two time series when a feature in one time series is higher or lower than its corresponding feature in another time series [15]. Ultimately, pathological alignments are identified based on context, meaning that the identifier has a particular preference or model in mind. As a result, there has been a large influx of alignment methods based on DTW to realize particular preferences or models for obtaining better alignments under certain contexts. Fu et al. proposed scaled and warped matching (SWM) by modeling one time series to be uniformly scaled in time when aligned with another time series [11]. They find the scaling in time and alignment with DTW simultaneously to address problems similar to query by humming, where significant global scaling in time can exist [11]. Query by humming is a way to retrieve music from user-hummed melodies, and the melodies hummed by different users can have drastically different tempos. Qiao et al. proposed affine-invariant dynamic time warping (AIDTW) by modeling one image as a rotated, shifted and scaled version of another image when aligned [25]. They applied AIDTW to recognize images of rotated handwritten digits, and AIDTW was found to outperform DTW by a large margin [25]. Latecki et al. proposed minimal variance matching (MVM) to allow skipping points to be matched [17]. In DTW, each point in one time series needs to be matched to at least one point in another time series, and this is not desirable when a time series contains points that are detrimental to finding the correct alignment [17]. For example, time series representing the contour of a face include confusing points that reflect the texture of hair, which can be detrimental when used for alignment [17]. Nielsen et al. proposed correlation optimized warping (COW) by modeling the temporal warping to be piecewise linear and optimizing for the associated correlation [21]. COW was designed for chromatographic profiles, where peaks are known to shift linearly in time with different heights [21]. Bahlmann et al. proposed statistical dynamic time warping (SDTW) by incrementing DTW with a probabilistic model [2]. SDTW was shown to work well for online handwriting recognition, where the addition of a probabilistic model in time helps resolve temporal confusions [2].

In this thesis, we propose two alignment methods that increment DTW based on specific models or preferences: affine dynamic time warping (ADTW) and regional dynamic time warping (RDTW). ADTW models one time series as an amplitude-scaled and amplitude-offset version of another time series when aligning them. It tries to find the best scale, offset and alignment simultaneously. Numerous types of time series fall under this affine model. For example, temperature and humidity data can be subject to different scales and offsets

as well as temporal variations depending on the geographic location and environment. The revenue time series for a less well-off and smaller company can also undergo scalings and offsets as well as temporal variations when compared to that of a more successful and larger company. The amount of support over time for an election candidate estimated by different parties can be roughly considered to be affine and varying in time when a fixed sampling population is used. An unintuitive alignment produced by DTW for two temperature time series subject to temporal variations, amplitude scaling and amplitude offset is illustrated in Figure 1.2, where DTW matches a large number of points in s to the peak in t. As we will see later, the proposed ADTW alignment method offers a much more intuitive alignment. Note that ADTW is different from AIDTW proposed by Qiao et al. [25] in that AIDTW assumes scaling, offset and rotation with respect to images. ADTW only models scaling and offset, thereby resulting in a simpler optimization problem. Furthermore, if AIDTW is applied to time series that do not undergo rotations, this additional unnecessary modeling may serve to confuse the output alignment.

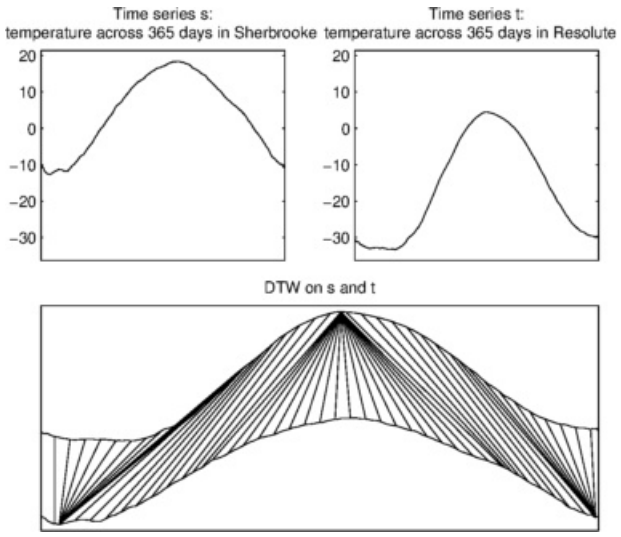

Figure 1.2: DTW alignment of daily temperature time series across a year for Sherbrooke and Resolute, Canada.

Scenarios may arise where there are certain regions in a time series reflective of components of interest, so they should be emphasized to find a more desirable alignment for two time series. Regional dynamic time warping (RDTW) is proposed to accommodate this preference by substituting the pointwise distance in DTW with a regional distance. Many time series contain components of interest that should be focused on, and some examples

are listed below. Time series on stock trading volume might contain bursts of influx or outflow that are reflective of insider trading. Electrocardiography (ECG) is a technique for recording a heart's electrical activities. An ECG time series can be divided into components that are associated with the depolarization of ventricles, which provide insights on the heart's condition. Electromyography (EMG) is a process that records the electrical activity produced by a muscle. In the analysis of an EMG time series, examination of motor unit potentials (MUPs) is helpful for determining muscle abnormalities, where a MUP is comprised of multiple muscle fiber potentials (MFPs). An EMG time series contains multiple MUPs that characterize the muscle, which in turn each contains multiple MFPs that characterize the MUP. In Figure 1.3, each MUP consists of two MFPs, and a match between two points is denoted by a line connecting them. Ideally, the left MFPs should be matched together and the right MFPs should be matched together, as shown in Figure 1.3A. However, Figure 1.3B illustrates that DTW produces an undesirable alignment where a good portion of the left MFP in the top MUP is matched to the right MFP in the bottom MUP. As we will see later, the proposed RDTW alignment method offers a better result by matching the MFPs correctly.

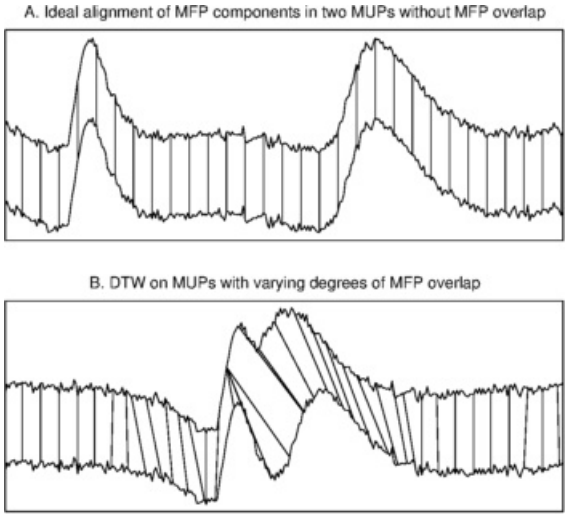

Figure 1.3: DTW alignment of two MUPs with varying degrees of MFP overlap.

ADTW and RDTW can be combined to include both affine modeling and emphasis on components. Two different ways of combining ADTW and RDTW in a global or local manner are proposed. Global-affine regional dynamic time warping (GARDTW) models one time series as a scaled and offsetted version of another time series when aligning

them with regional emphasis. For example, amount of rainfall over time is subject to different scalings, offsets and temporal variations across different locations. In the analysis of this rainfall data, placing an emphasis on sections with short but heavy amounts of rainfall can be useful in predicting such behaviors. The preference for this example is to emphasize on matching components reflective of short but heavy amounts of rainfall within two time series that can undergo scaling, offset and temporal variation. Local-affine regional dynamic time warping (LARDTW) emphasizes on regions when performing an alignment, where the region in one time series is modeled as a scaled and offset version of the respective matched region in another time series. For example, the electrode recording an EMG time series might be subject to movement. As a result, some motor units (MUs) become closer to the electrode surface, and the associated MUPs are increased in scale. Other MUs are now further away from the electrode surface, and the associated MUPs are decreased in scale. The preference for this example is to correctly match the MUPs which can undergo different scalings in amplitude in two EMG time series.

The rest of this thesis is organized as follows. Chapter 2 provides a review for the technical details of DTW. Furthermore, the proposed methods of ADTW, RDTW, GARDTW and LARDTW are formulated and discussed in detail. Chapter 3 covers evaluations of the alignments and difference measures generated by the proposed methods. Finally, Chapter 4 concludes this thesis and points out possible future extensions of this work.

Chapter 2

Methodology for Pairwise Alignment of Time Series

2.1 Dynamic Time Warping Review

Dynamic time warping (DTW) is a method that matches points in two time series that are subject to non-linear temporal variations. It was invented by Sakoe et al. for speech recognition [27]. Let $s = (s_1, s_2, ..., s_n) \in \mathbb{R}^n$ and $t = (t_1, t_2, ..., t_m) \in \mathbb{R}^m$ be two time series of interest. Also, let p represent a sequence of matched points (also called an alignment) between s and t, where

$$p = \{p(1) = (a_1, b_1), p(2) = (a_2, b_2), ..., p(|p|) = (a_{|p|}, b_{|p|})\}$$

and $(a_k, b_k) \in \mathbb{Z}^2_{>0}$ means that point s_{a_k} is matched with point t_{b_k}. In addition, let d be a difference measure between two points. For the remainder of this thesis, d is assumed to be the squared difference unless mentioned otherwise. Then, for a pair of time series s and t, DTW searches for an optimal alignment p^* among all possible alignments p's such that

$$D(s, t, p) = \sum_{k=1}^{|p|} d(s_{a_k}, t_{b_k})$$

is minimized subject to the following constraints:

- Boundary: $p(1) = (1, 1)$ and $p(|p|) = (n, m)$. The start points of s and t are matched, and the end points of s and t are matched.

- Monotonicity: If $p(k) = (a, b)$ and $p(k+1) = (c, d)$, then $c \geq a$ and $d \geq b$ $\forall k$. The sequence of matches can only move forward in time.
- Step Size: If $p(k) = (a, b)$ and $p(k+1) = (c, d)$, then $c - a \leq 1$ and $d - b \leq 1$ $\forall k$. For each time series, the next matched point can only be at most 1 time unit away from the current one.

Note that more than one distinct alignment can exist as solutions to this constrained optimization problem. Also, the reader should keep in mind for the remainder of this thesis that the term 'optimal' only refers to finding values that optimize a function, and the associated solution might not necessarily be the best. For simplicity, the boundary, monotonicity and step size constraints will be jointly referred to as the DTW constraints, and $D(s, t, p)$ subject to the DTW constraints will be denoted as $D(s, t, p)_{\text{constrained}}$.

Figure 2.1 illustrates violation of each DTW constraint under the scenario of aligning time series s with time series t. Figure 2.1 also illustrates how an alignment can be visualized as a path. Each grid contains a path that represents an alignment p between s and t, where filling of location (i, j) in a grid means that point s_i and point t_j are matched and a line connects the current match to the next match. The plots below the grids illustrate how the two time series are aligned, where a match between two points is denoted by connecting the two points with a dotted line. Looking at the grids, any path that starts at position $(1, 1)$ and ends at position $(6, 6)$ by moving east, north, or northeast one step at a time represents a viable alignment that meets the boundary, monotonicity and step size constraints. These constraints are not unreasonable and, as we will see later, enable faster computation for the optimal alignment p^*. The terms 'alignment' and 'path' will be used interchangeably from now on.

One simple way of finding an optimal alignment p^* to this constrained optimization problem is to plug all possible alignments p's that meet the DTW constraints into $D(s, t, p)$ and determine which p yields the minimal $D(s, t, p)$. However, the number of possible alignments between two time series quickly grows out of control as the lengths of the two time series increase. Recall that a path that meets the DTW constraints starts at $(1, 1)$ and ends at (n, m) by going east, north or northeast one step at a time. The number of all such paths is described by the Delannoy number $\text{Del}(n-1, m-1)$, and Table 2.1 lists out the total number of viable alignments for different time series lengths n's where $m = n$. There are as many as $\approx 10^{13}$ viable alignments for two small time series of length $n = 20$, and it can be shown that $\text{Del}(n-1, n-1)$ has a lower bound of 3^{n-1}. Hence, we observe that the number of possible alignments grows exponentially with n, and this fact shows that the simple approach of exhaustively plugging in different p's to find an optimal alignment p^* is not feasible.

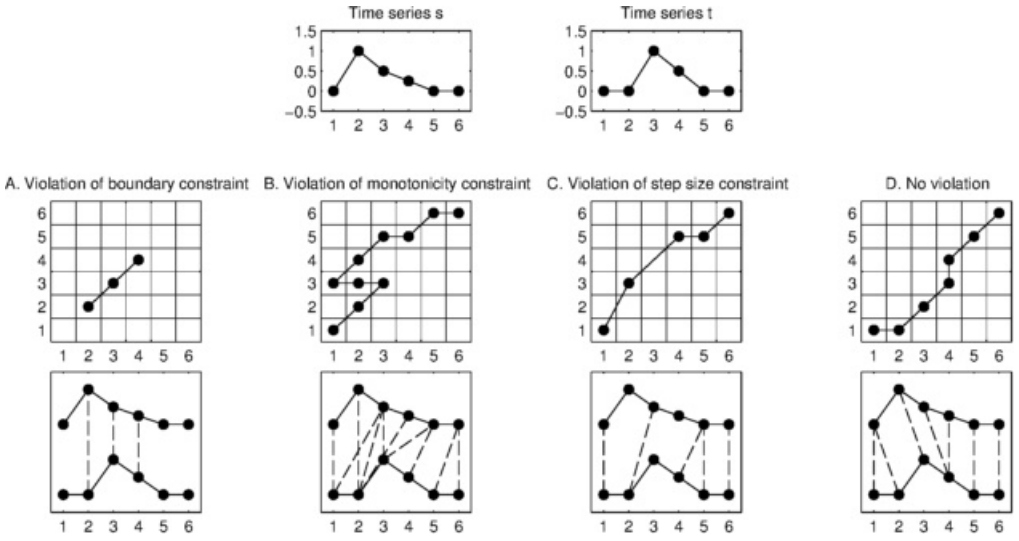

Figure 2.1: Violation of different constraints for DTW and visualization of respective alignments as paths.

Table 2.1: Number of all possible alignments satisfying the boundary, monotonicity and step size constraints for different time series lengths n's.

	Time series length n					
	2	3	4	5	10	20
Number of possible paths	3	13	63	321	1462563	45849429914943

Dynamic programming (faster brute force by storing reusable results in memory) can be effective in reducing the time complexity for finding the optimal path p^* to this constrained optimization problem, because the path p^* has optimal substructures and there are overlapping subproblems. Let $p^*_{(a,b)}$ be the optimal path from $(1,1)$ to (a,b) that minimizes $D(s,t,p_{(a,b)})$ subject to the monotonicity and step size constraints. This path represents the optimal alignment for $(s_1,s_2,...,s_a)$ and $(t_1,t_2,...,t_b)$ subject to the DTW constraints. The optimal alignment $p^* = p^*_{(n,m)}$ has optimal substructures because if the path passes through (i,j) for any $1 < i < n$ and $1 < j < m$, the following subalignment of p^*

$$\{p^*(1) = (1,1), p^*(2) = (a_2,b_2), ..., (i,j)\}$$

is $p^*_{(i,j)}$, the optimal alignment between $(s_1,s_2,...,s_i)$ and $(t_1,t_2,...,t_j)$. Furthermore, overlapping subproblems exist because for any $1 < i < n$ and $1 < j < m$, multiple alignments

Table 2.2: DTW table.

$D(s,t,p^*_{(n,1)})$	$D(s,t,p^*_{(n,2)})$	$\dots$	$D(s,t,p^*_{(n,m)})$
$\vdots$	$\vdots$	$\ddots$	$\vdots$
$D(s,t,p^*_{(2,1)})$	$D(s,t,p^*_{(2,2)})$	$\dots$	$D(s,t,p^*_{(2,m)})$
$D(s,t,p^*_{(1,1)})$	$D(s,t,p^*_{(1,2)})$	$\dots$	$D(s,t,p^*_{(1,m)})$

$p_{(n,m)}$'s can pass through (i,j), and these alignments share the same problem of finding $p^*_{(i,j)}$ when searching for $p^*_{(n,m)}$.

A dynamic programming solution can be formulated using these properties. First, a table of $D(s,t,p^*_{(i,j)})$ values is constructed for all $1 \leq i \leq n$ and $1 \leq j \leq m$. This table is referred to as the DTW table and it is shown in Table 2.2. The optimal alignment p^* can be found after building this table. With the observation that a path from $(1,1)$ to (i,j) needs to pass through either $(i-1,j-1)$, $(i-1,j)$ and $(i,j-1)$, the DTW table can be updated starting from the bottom row from left to right, and then the row above can be filled from left to right as well. The next row above can be computed in the same manner from left to right until the n^{th} row is reached. The update formula is as follows:

$$D(s,t,p^*_{(a,b)}) = d(s_a,t_b) + \underbrace{\min(D(s,t,p^*_{(a-1,b-1)}), D(s,t,p^*_{(a,b-1)}), D(s,t,p^*_{(a-1,b)}))}_{H(s,t,a,b)} \quad (2.1)$$

Note that if $a-1<1$, $D(s,t,p^*_{(a-1,b-1)})$ and $D(s,t,p^*_{(a-1,b)})$ are not considered, and if $b-1<1$, $D(s,t,p^*_{(a-1,b-1)})$ and $D(s,t,p^*_{(a,b-1)})$ are not considered in the update formula.

Now, the question lies in how to obtain the optimal alignment p^* from the DTW table. Recall that $D(s,t,p^*_{(n,m)}) = d(s_n,t_m) + H(s,t,n,m)$. If $H(s,t,n,m) = D(s,t,p^*_{(n-1,m-1)})$, p^* must contain a northeast move from $(n-1,m-1)$ to (n,m) and the remaining task is to align $(s_1,s_2,...,s_{n-1})$ with $(t_1,t_2,...,t_{m-1})$. If $H(s,t,n,m) = D(s,t,p^*_{(n,m-1)})$, p^* must contain a eastbound move from $(n,m-1)$ to (n,m) and a subalignment between $(s_1,s_2,...,s_n)$ and $(t_1,t_2,...,t_{m-1})$ remains to be found. Similarly, if $H(s,t,n,m) = D(s,t,p^*_{(n-1,m)})$, p^* must contain a northbound move from $(n-1,m)$ to (n,m) and a subalignment between $(s_1,s_2,...,s_{n-1})$ and $(t_1,t_2,...,t_m)$ remains to be found. This procedure (also called backtracking) can be applied to the resulting subalignment again and again until the location $(1,1)$ is reached, thereby yielding the optimal alignment p^*. Figure 2.2 illustrates this backtracking process, where $p^* = \{(1,1),(2,2),(3,2),(4,3),(5,4),(6,5),(6,6)\}$.

For simplicity of complexity analysis, let us assume that the two time series to be aligned have the same length such that $n = m$. Since the update formula in Equation 2.1 for filling

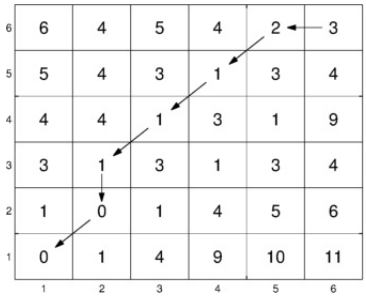

Figure 2.2: Backtracking on DTW table to find the optimal alignment.

each element in the DTW table is $O(1)$, filling the entire table requires a time complexity of $O(n^2)$. The space complexity for constructing the DTW table is $O(n^2)$ as well. The longest path that can be obtained from backtracking is one without any northeast moves, since one northeast move is equivalent to one eastbound move plus one northbound move. As such, the time and space complexity for backtracking is $O(2n-1) = O(n)$, where $2n-1$ is the length of an alignment p that does not have any northeast moves. Hence, the total time and space complexity for finding the optimal alignment p^* is $O(n^2) + O(n) = O(n^2)$.

Certain alignments are unlikely to be the best alignment for aligning two time series. For example, an alignment path p that moves northbound all the way followed by moving eastbound all the way in the context of a DTW table is equivalent to matching point t_1 with points $s_1, s_2, ..., s_n$ and matching point s_n with points $t_1, t_2, ..., t_m$. This aforementioned alignment is unlikely to be the best alignment for two typical time series s and t, even if this alignment is optimal in the sense of minimizing $D(s,t,p)_{\text{constrained}}$. Additional constraints can be placed on the path p to 1) eliminate pathological alignments, and 2) reduce time and space complexity. A well-known path constraint is the Sakoe-Chiba band [28], where s_i can only be matched to points from the set $\{t_{i-w_q}, t_{i-w_q+1}, ..., t_i, ..., t_{i+w_q-1}, t_{i+w_q}\}$ and $w_q \in \mathbb{Z}_{\geq 0}$. This constraint effectively creates the band shown in Figure 2.3, where the grid represents a DTW table and the grey parts are not considered for alignment. The width of the band w_b is given by $w_b = 1 + 2w_q$. Ratanamahatana et al. showed supporting evidence that a small bandwidth w_b achieves good results whereas a high bandwidth can be detrimental [26]. It was recommended that the best bandwidth should be set based on specific problems or datasets.

Note that less than $w_b n$ elements are required to be filled in the DTW table when the Sakoe-Chiba band is imposed. Since the limiting factor in the time and space complexities of the DTW algorithm lies in the construction of the DTW table, the time and space

complexities are both reduced from $O(n^2)$ to $O(w_b n)$ when assuming $n = m$.

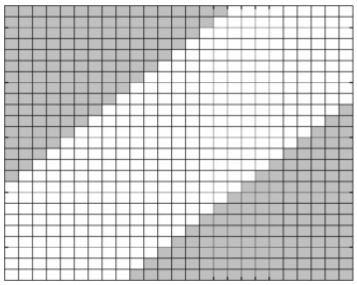

Figure 2.3: Sakoe-Chiba band in a DTW table.

In addition to alignments, DTW also provides a powerful difference measure between two time series s and t in the form of $D(s, t, p^*)$, which has enjoyed much success in many domains [20][6][19][29]. $D(s, t, p^*)$ can be thought of as the sum of squared difference between two time series after temporal variations are accounted for, and it will be referred to as the DTW difference measure for the remainder of this thesis.

2.2 Affine Dynamic Time Warping

Affine dynamic time warping (ADTW) increments DTW to allow arbitrary scaling and offset in amplitude between two time series which are subject to temporal variations. Let s and t be two time series of interest. In ADTW, s is assumed to be a time series that is scaled and offsetted in amplitude with temporal variations. In more formal terms, the goal is to find a path p^*, scaling factor $c^* \in \mathbb{R}$ and offset factor $e^* \in \mathbb{R}$ that minimize

$$D_A(s, t, p, c, e) = \sum_{k=1}^{|p|} d(s_{a_k}, ct_{b_k} + e) = \sum_{k=1}^{|p|} (s_{a_k} - (ct_{b_k} + e))^2$$

subject to the DTW boundary, monotonicity and step size constraints. For brevity, $D_A(s, t, p, c, e)$ subject to the DTW constraints will be referred to as $D_A(s, t, p, c, e)_{\text{constrained}}$. This formulation aims to optimize for a global minimum in $D_A(s, t, p, c, e)_{\text{constrained}}$ with respect to p, c and e simultaneously, which is different from finding the scaling and offset first prior to applying DTW to obtain an alignment.

As a first step to solving this optimization problem, we will show that the optimal scaling and offset (c'_p, e'_p) can be found for a given path p in the context of minimizing

$D_A(s,t,p,c,e)_{\text{constrained}}$. We will begin by showing that $D_A(s,t,p,c,e)$ is convex with respect to c and e when s, t, and p are given. In this context, p has been picked to satisfy the DTW constraints. If a function is convex, a local optimum of the function is necessarily a global minimum. Hence, given that $D_A(s,t,p,c,e)$ is convex with respect to c and e, we can proceed to solve for (c'_p, e'_p) by finding a local optimum in $D_A(s,t,p,c,e)$ using derivatives.

We start by proving that $D_A(s,t,p,c,e)$ is convex with respect to c and e when s, t, and p are given. Let $s_p = [s_{a_1}, s_{a_2}, ..., s_{a_{|p|}}]^T$ and $t_p = [t_{a_1}, t_{a_2}, ..., t_{a_{|p|}}]^T$, where T is the transpose operator. Also, let $\beta = [c,e]^T$ and let $\vec{1} = [1,1,...1]^T$ be a vector of length $|p|$. Furthermore, let $y = s_p$ and $X = [t_p, \vec{1}]$. Then, $D_A(s,t,p,c,e)$ can be rewritten in the form of linear least squares:

$$\begin{aligned} D_A(s,t,p,c,e) &= (y - X\beta)^T(y - X\beta) \\ &= y^T y - 2\beta^T X^T y + \beta^T X^T X\beta \end{aligned}$$

This function is twice differentiable with respect to β and its Hessian (matrix of second-order partial derivatives for a function) with respect to β is

$$\begin{aligned} \frac{\partial^2}{\partial\beta\partial\beta^T} D_A(s,t,p,c,e) &= \frac{\partial}{\partial\beta^T}[\frac{\partial}{\partial\beta} D_A(s,t,p,c,e)] \\ &= \frac{\partial}{\partial\beta^T}[-2X^T y + X^T X\beta] \\ &= X^T X \end{aligned}$$

which is a positive semi-definite matrix. If the Hessian of a function is positive semi-definite, the function is convex. Hence, $D_A(s,t,p,c,e)$ is convex with respect to (c,e) when (s,t,p) are given.

Since we now know that $D_A(s,t,p,c,e)$ has only one global minimum with respect to (c,e), finding any local optimum is equivalent to finding the global minimum. Finding a local optimum of $D_A(s,t,p,c,e)$ can be achieved by taking derivatives of $D_A(s,t,p,c,e)$ with respect to c and e, setting the derivatives to 0, and solving for values of c and e.

Taking the derivative of $D_A(s,t,p,c,e)$ with respect to c, we get

$$\begin{aligned}\frac{\partial}{\partial c}D_A(s,t,p,c,e) &= \frac{\partial}{\partial c}\sum_{k=1}^{|p|}(s_{a_k} - ct_{b_k} - e)^2 \\ &= \sum_{k=1}^{|p|}\frac{\partial}{\partial c}(s_{a_k} - ct_{b_k} - e)^2 \\ &= -2\sum_{k=1}^{|p|}(s_{a_k} - ct_{b_k} - e)t_{b_k}\end{aligned}$$

Taking the derivative of $D_A(s,t,p,c,e)$ with respect to e, we obtain

$$\begin{aligned}\frac{\partial}{\partial e}D_A(s,t,p,c,e) &= \frac{\partial}{\partial e}\sum_{k=1}^{|p|}(s_{a_k} - ct_{b_k} - e)^2 \\ &= \sum_{k=1}^{|p|}\frac{\partial}{\partial e}(s_{a_k} - ct_{b_k} - e)^2 \\ &= -2\sum_{k=1}^{|p|}(s_{a_k} - ct_{b_k} - e)\end{aligned}$$

Setting $\frac{\partial}{\partial e}D_A(s,t,p,c,e'_p) = 0$ and solving for e'_p yields the minimizing e:

$$\begin{aligned}-2\sum_{k=1}^{|p|}(s_{a_k} - c'_p t_{b_k} - e'_p) &= 0 \\ -|p|e'_p + \sum_{k=1}^{|p|}(s_{a_k} - c'_p t_{b_k}) &= 0 \\ e'_p = \frac{1}{|p|}\sum_{k=1}^{|p|}(s_{a_k} - c'_p t_{b_k}) &\qquad (2.2)\end{aligned}$$

Likewise, setting $\frac{\partial}{\partial c}D_A(s,t,p,c'_p,e) = 0$, plugging in Equation 2.2 for e'_p and solving for c'_p yields the minimizing c:

$$-2\sum_{k=1}^{|p|}(s_{a_k} - c'_p t_{b_k} - e'_p)t_{b_k} = 0$$

$$-2\sum_{k=1}^{|p|}(s_{a_k} - c'_p t_{b_k} - \frac{1}{|p|}\sum_{l=1}^{|p|}(s_{a_l} - c'_p t_{b_l}))t_{b_k} = 0$$

$$\sum_{k=1}^{|p|}(s_{a_k} t_{b_k} - c'_p t_{b_k}^2 - \frac{1}{|p|}\sum_{l=1}^{|p|}(s_{a_l} - c'_p t_{b_l})t_{b_k}) = 0$$

$$\sum_{k=1}^{|p|}(s_{a_k} t_{b_k} - c'_p t_{b_k}^2 - t_{b_k}\frac{1}{|p|}\sum_{l=1}^{|p|} s_{a_l} + c'_p t_{b_k}\frac{1}{|p|}\sum_{l=1}^{|p|} t_{b_l}) = 0$$

$$\sum_{k=1}^{|p|} s_{a_k} t_{b_k} - \sum_{k=1}^{|p|} c'_p t_{b_k}^2 - \sum_{k=1}^{|p|} t_{b_k}\frac{1}{|p|}\sum_{l=1}^{|p|} s_{a_l} + \sum_{k=1}^{|p|} c'_p t_{b_k}\frac{1}{|p|}\sum_{l=1}^{|p|} t_{b_l} = 0$$

$$c'_p\sum_{k=1}^{|p|} t_{b_k}^2 - c'_p\frac{1}{|p|}\sum_{k=1}^{|p|} t_{b_k}\sum_{l=1}^{|p|} t_{b_l} = \sum_{k=1}^{|p|} s_{a_k} t_{b_k} - \frac{1}{|p|}(\sum_{l=1}^{|p|} s_{a_l})(\sum_{k=1}^{|p|} t_{b_k})$$

$$c'_p = \frac{\sum_{k=1}^{|p|} s_{a_k} t_{b_k} - \frac{1}{|p|}(\sum_{k=1}^{|p|} s_{a_k})(\sum_{k=1}^{|p|} t_{b_k})}{\sum_{k=1}^{|p|} t_{b_k}^2 - \frac{1}{|p|}(\sum_{k=1}^{|p|} t_{b_k})^2} \tag{2.3}$$

Now, we have obtained formulae (Equation 2.2 and 2.3) for computing (c'_p, e'_p) that minimize $D_A(s, t, p, c, e)$ when (s, t, p) are given and p satisfies the DTW constraints. The original optimization problem of finding (p^*, c^*, e^*) that minimize $D_A(s, t, p, c, e)$ subject to the DTW constraints can thus be rewritten as

$$(p^*, c^* = c'_{p^*}, e^* = e'_{p^*}) = \underset{p}{\operatorname{argmin}}\ D_A(s, t, p, c'_p, e'_p)_{\text{constrained}}$$

As mentioned in the DTW section, the number of possible paths p's grows exponentially as the length of two time series increases. Therefore, it is not feasible to perform this optimization exhaustively. Dynamic programming is also no longer applicable because each path p can have a different pair of scaling and offset factors (c'_p, e'_p), thereby rendering results for existing subpaths not reusable. As a result, instead of finding an optimal solution (p^*, c^*, e^*), we find a suboptimal solution (p^l, c^l, e^l) using Algorithm 1.

Looking Algorithm 1, it may not be immediately clear how to obtain p^l_v, c^l_v or e^l_v. p^l_v can be computed with the standard DTW algorithm with input time series s and $c^l_{v-1}t + e^l_{v-1}$.

Algorithm 1 ADTW

p^l, c^l, e^l
$c_0^l \leftarrow 1$
$e_0^l \leftarrow 0$
$D_{A,\text{prev}} \leftarrow \infty$
$v \leftarrow 1$
while 1 **do**
    $p_v^l \leftarrow \underset{p}{\text{argmin}}\ D_A(s, t, p, c_{v-1}^l, e_{v-1}^l)_{\text{constrained}}$
    $(c_v^l, e_v^l) \leftarrow \underset{c,e}{\text{argmin}}\ D_A(s, t, p_v^l, c, e)_{\text{constrained}}$
    if $D_{A,\text{prev}} - D_A(s, t, p^l, c^l, e^l) < D_{\text{stop}}$ **then**
        $p^l \leftarrow p_v^l$
        $c^l \leftarrow c_v^l$
        $e^l \leftarrow e_v^l$
        break
    $v \leftarrow v + 1$

(c_v^l, e_v^l) can be calculated in closed form with using Equation 2.2 and 2.3. Note that $D_{\text{stop}} \in \mathbb{R}$ is a small value chosen by the user for checking convergence of the algorithm. Algorithm 1 was strongly motivated by the classification expectation-maximization algorithm from Celeux et al.'s work on k-means clustering [5]. How Algorithm 1 finds a suboptimal solution (p^l, c^l, e^l) will be elaborated below. First, we will prove the following important property:

$$D_A(s, t, p_{v+1}^l, c_{v+1}^l, e_{v+1}^l)_{\text{constrained}} \leq D_A(s, t, p_v^l, c_v^l, e_v^l)_{\text{constrained}}\ \forall v \in \mathbb{Z}_{>0}$$

Proof.

$$\begin{aligned}
p_{v+1}^l &= \underset{p}{\text{argmin}}\ D_A(s, t, p, c_v^l, e_v^l)_{\text{constrained}} \\
\text{So, } D_A(s, t, p_{v+1}^l, c_v^l, e_v^l)_{\text{constrained}} &\leq D_A(s, t, p_v^l, c_v^l, e_v^l)_{\text{constrained}} \\
(c_{v+1}^l, e_{v+1}^l) &= \underset{c,e}{\text{argmin}}\ D_A(s, t, p_{v+1}^l, c, e)_{\text{constrained}} \\
\text{So, } D_a(p_{v+1}^l, c_{v+1}^l, e_{v+1}^l)_{\text{constrained}} &\leq D_A(s, t, p_{v+1}^l, c_v^l, e_v^l)_{\text{constrained}} \\
&\leq D_A(s, t, p_v^l, c_v^l, e_v^l)_{\text{constrained}}
\end{aligned}$$

□

Since $D_A(s, t, p^l_{v+1}, c^l_{v+1}, e^l_{v+1})_{\text{constrained}} \leq D_A(s, t, p^l_v, c^l_v, e^l_v)_{\text{constrained}}$, we gain the insight that this algorithm tries to iteratively decrease $D_A(s, t, p, c, e)_{\text{constrained}}$ with respect to p, c and e. Furthermore, it can be proved that the sequence $\{D_A(s, t, p^l_v, c^l_v, e^l_v)_{\text{constrained}}\}^{\infty}_{v=1}$ converges to one value.

Proof. We begin by noting that there is a finite number of possible paths p's. We also know that the scaling and offset factors (c'_p, e'_p) minimize $D_A(s, t, p, c, e)_{\text{constrained}}$ when a path p satisfying the DTW constraints is given. Hence, there is a finite number of (p^l_v, c^l_v, e^l_v) tuples, and this also means that there is a finite number of $D_A(s, t, p^l_v, c^l_v, e^l_v)_{\text{constrained}}$ values. In other words, there is a finite number of distinct values in the sequence

$$\{D_A(s, t, p^l_v, c^l_v, e^l_v)_{\text{constrained}}\}^{\infty}_{v=1}$$

Imposing the additional condition that

$$D_A(s, t, p^l_{v+1}, c^l_{v+1}, e^l_{v+1})_{\text{constrained}} \leq D_A(s, t, p^l_v, c^l_v, e^l_v)_{\text{constrained}}$$

we know that $\{D_A(s, t, p^l_v, c^l_v, e^l_v)_{\text{constrained}}\}^{\infty}_{v=1}$ has to converge to one value. □

It is important to note that the sequence $\{(p^l_v, c^l_v, e^l_v)\}^{\infty}_{v=1}$ may not necessarily converge, because it is possible that $(p^l_{v+1}, c^l_{v+1}, e^l_{v+1}) \neq (p^l_v, c^l_v, e^l_v)$ even though

$$D_A(s, t, p^l_{v+1}, c_{v+1}, e^l_{v+1})_{\text{constrained}} = D_A(s, t, p^l_v, c^l_v, e^l_v)_{\text{constrained}}$$

Therefore, in Algorithm 1, the stopping condition involves comparing the difference between $D_A(s, t, p^l_v, c^l_v, e^l_v)_{\text{constrained}}$ and $D_A(s, t, p^l_{v+1}, c^l_{v+1}, e^l_{v+1})_{\text{constrained}}$ with a small value, which is analogous to checking if $D_A(s, t, p^l_v, c^l_v, e^l_v)_{\text{constrained}}$ has converged.

Capabilities of ADTW are shown in Figure 2.4 and 2.5. They demonstrate that ADTW can offer better alignments for time series that are subject to random scalings and offsets when compared to DTW. In both figures, t is a varied time series based on s. For the alignment plots, s is the bottom time series whereas t is the top time series, and a match between two points denoted by connecting the two points with a line. We start by examining Figure 2.4, where $t = 2s + 1$. If two time series are known to have different scalings and offsets without any temporal variation, the ideal alignment consists of only vertical matches as shown in Figure 2.4A. Visually, DTW offers a worse alignment by matching a large amount of points surrounding $\max(t)$ in t to $\max(s)$ in s. On the other hand, ADTW matches all points vertically, which is identical to the ideal alignment. Furthermore, ADTW correctly estimates the scaling and offset for s in terms of t. In this simple

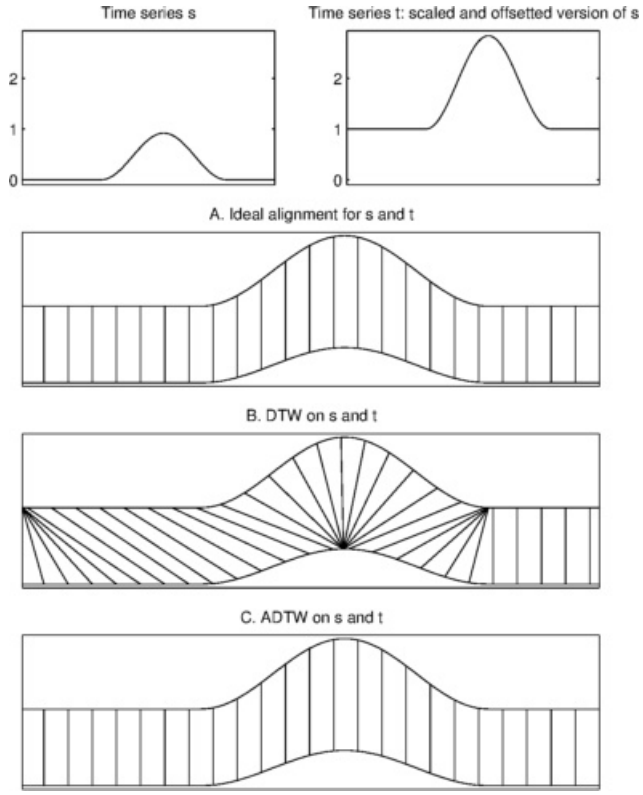

Figure 2.4: DTW and ADTW alignments of same synthetic time series with different scalings and offsets.

example, Algorithm 1 yields the optimal solution even though there is not any guarantee that its solution would be optimal.

For the simplistic example shown in Figure 2.4, it turns out that normalizing s and t prior to performing DTW will output the ideal alignment. If ADTW does not offer any advantage to normalization prior to applying DTW, ADTW would not be preferred because its algorithm outlined in Algorithm 1 is certainly more complex. Again, it is emphasized that ADTW attempts to optimize $D_A(s, t, p, c, e)_{\text{constrained}}$ in terms of (p, c, e) simultaneously, and this goal is quite different from normalization prior to applying DTW. Figure 2.5 demonstrates that ADTW offers better alignments than DTW and the scheme of normalizing prior to applying DTW when two time series are subject to different scalings, offsets and temporal variations. Each time series in Figure 2.5 is composed of two Hamming windows. Time series t is a scaled and offset version of s injected with temporal

variations such that the left Hamming window is shrunk in width and the right Hamming window is expanded in width. Figure 2.5A shows the ideal alignment where the two Hamming windows in s and t are aligned together, and the alignment reflects the increase or decrease in width. As shown in Figure 2.5B and C, the alignment provided by ADTW is much closer to the ideal alignment when compared to DTW or normalization before applying DTW. While the alignment provided by ADTW shows inconsistencies with the ideal alignment in sections that are not associated with any Hamming window, one can argue that these sections are not particularly interesting. Furthermore, the contribution to $D_A(s, t, p, c, e)_{\text{constrained}}$ is 0 across these sections without influence of Hamming windows as long as a point in one section is matched to another point in another section. In fact, both alignments in Figure 2.5A and D would output the same value of $D_A(s, t, p, c^l, e^l)_{\text{constrained}}$ whether p is ideal or $p = p^l$. Since the estimated values for c^l and e^l are equivalent to the imposed scaling and offset values for Figure 2.5, the ADTW solution is optimal in the sense of minimizing $D_A(s, t, p, c, e)_{\text{constrained}}$. This is yet another example where Algorithm 1 offers an optimal solution despite lacking any guarantee for it.

Now, let us revisit the real-world example provided in the introductory chapter for motivating the affine model. Temperature time series across 365 days in two different locations, shown in Figure 2.6, exhibit scaling, offset and temporal variations. The associated alignments using different approaches are also shown in the same figure, where the bottom time series is s and the top time series is t.

Looking at Figure 2.6A, DTW outputs a pathological alignment by matching a large number of points in s with the peak in t. The alignments provided by ADTW and normalization prior to applying DTW in Figure 2.6C and B are far more reasonable. In this context, normalization is equivalent to subtracting the mean followed by dividing by the standard deviation. However, with the normalization approach, a good number of points surrounding the peak in t are matched to the peak in s. Matching only the two peaks together seems to be a more reasonable approach, which is provided by ADTW. Furthermore, the left flat and low segments in s and t are matched together with ADTW, which seems to be more reasonable than the alignment provided by the normalization approach, where a portion of the left flat section in t is matched with the beginning slope in s. Some readers might disagree with the above arguments on which alignment is better because an ideal alignment does not exist here. Nonetheless, most readers would probably agree that ADTW can provide alignments with certain characteristics that are desirable for some tasks. The primary difference between ADTW and the normalization approach is attributed to the fact that normalization prior to applying DTW means that the scale and offset do not account for the alignment at all.

Intuitively, the time complexity for ADTW is higher than that for DTW because every

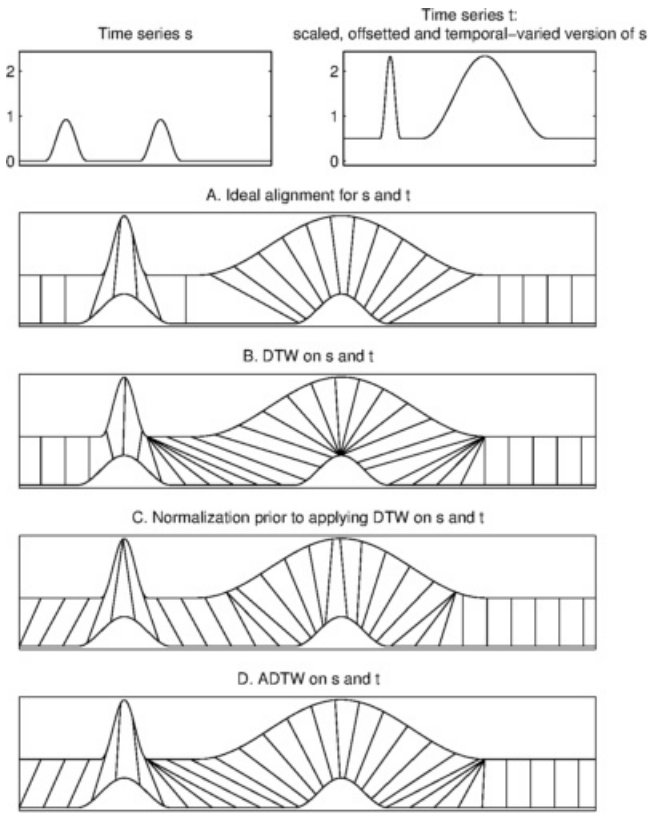

Figure 2.5: DTW and ADTW alignments of the same synthetic time series with different scalings, offsets and temporal variations.

iteration in Algorithm 1 requires a DTW pass, which is $O(w_b n)$ in time for the banded case. In addition, every iteration requires closed-form computations of (c_v^l, e_v^l), which sum along the path p_v^l as shown in Equation 2.3 and 2.2. Again, let us assume $n = m$ for simplicity. There are at most $2n - 1$ elements in a path p, so the time complexity for computing (c_v^l, e_v^l) is $O(n)$. The remaining parts in each iteration in Algorithm 1 can be completed in constant time, so each iteration takes

$$O(w_b n) + O(n) = O(w_b n)$$

Thus, the time complexity for the ADTW algorithm is $O(g w_b n)$, where g is the number of iterations required for convergence. It is difficult to estimate g theoretically. However, in the case of k-means clustering which utilizes the same optimization paradigm, the optimization criterion at each iteration can be shown to converge to a local optimum at a

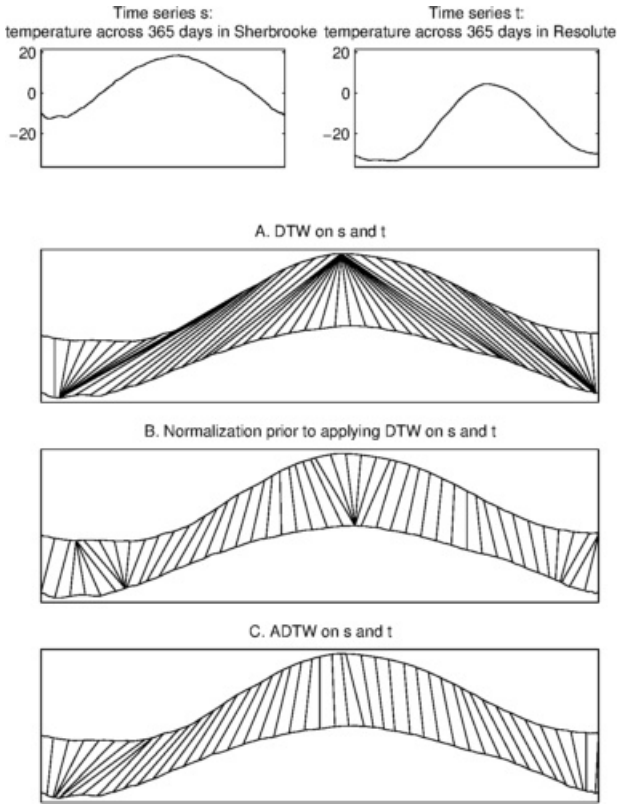

Figure 2.6: DTW and ADTW alignments of the daily temperature time series across a year for Sherbrooke and Resolute, Canada.

linear rate when certain conditions are fulfilled [5]. Therefore, it is not unreasonable to speculate that $D_A(s, t, p_v^l, c_v^l, e_v^l)_{\text{constrained}}$ can converge to a local minimum at a linear rate under certain conditions for ADTW. For the ADTW examples shown in Figure 2.4C, 2.5D and 2.6C, g is 20, 12 and 35 respectively. g is highly dependent on the stopping condition threshold $D_{\text{stop}} \in \mathbb{R}_{>0}$, and g increases as D_{stop} decreases. In the above examples and for the remainder of this thesis, $D_{\text{stop}} = 10^{-5}$ unless mentioned otherwise. The space complexity for ADTW is $O(w_b n)$, which is identical to that for DTW, because computation of (c_v^l, e_v^l) only requires $O(1)$ space when p_v^l is given, and the remaining operations require $O(1)$ space as well. The limiting space complexity still lies in the construction of the DTW table.

So far, examples of ADTW yielding a single solution have been shown. However,

with the optimization paradigm used in Algorithm 1, we know that the solution obtained is not necessarily optimal. In fact, very different solutions can be obtained based on different initializations of the scaling and offset (c_0^l, e_0^l). Figure 2.7 illustrates this effect. In Figure 2.7, an ideal alignment between two time series is one where the two left and identical Hamming windows are matched together. When $(c_0^l, e_0^l) = (1, 0)$, the two identical Hamming windows are aligned to each other, yielding $D_A(s, t, p^l, c^l, e^l)_{\text{constrained}} \approx 8$. On the other hand, when $(c_0^l, e_0^l) = (0, 0)$, the Hamming window in s ends up being aligned to a much smaller Hamming window in t with negative c^l, yielding $D_A(s, t, p^l, c^l, e^l)_{\text{constrained}} \approx 45$. The alignment resulted from $(c_0^l, e_0^l) = (0, 0)$ is clearly suboptimal when compared to that resulted from $(c_0^l, e_0^l) = (1, 0)$.

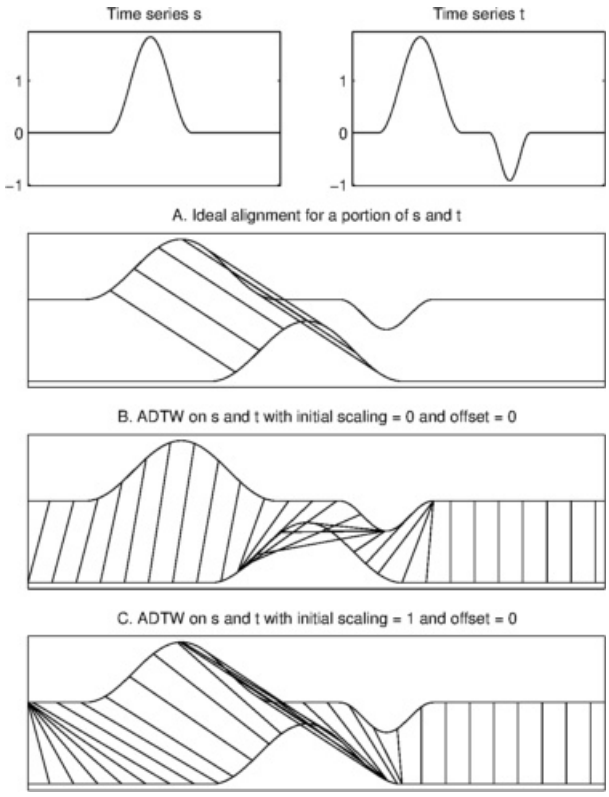

Figure 2.7: ADTW alignments of two synthetic time series with different initializations of scaling and offset.

The example in Figure 2.7 speaks to the importance of a good initialization of (c_0^l, e_0^l) for the ADTW algorithm. Ideally, the ADTW algorithm should be repeated many times with

different initializations, and the solution that offers the minimal $D_A(s,t,p^l,c^l,e^l)_{\text{constrained}}$ should be selected. However, this approach is not computationally feasible because the number of different initializations is infinite given that $(c^l,e^l) \in \mathbb{R}^2$. While there might exist smart ways to arrive at a small finite set of good initializations, these approaches are not explored in this thesis. Practically, one good approach is to initialize the scaling and offset to be reasonable values that are close to the expected true scaling and offset (c^*,e^*). For this thesis, $(c_0^l,e_0^l) = (1,0)$ unless mentioned otherwise. Note that the initialization can also involve setting p_0^l instead of (c_0^l,e_0^l), and Algorithm 1 can be slightly modified to accommodate this change by switching line 7 and line 8. This modification can be advantageous when a good initial alignment is known, but good initial values for scaling and offset (c_0^l,e_0^l) are not known. Furthermore, a good initialization also plays an important role in the computation time. Generally, the closer (c_0^l,e_0^l) is to the resulting (c^l,e^l), the smaller number of iterations required for convergence. Hence, starting with a good initialization for ADTW is highly recommended because it offers better solutions and faster computation time.

Situations may arise where subsets of t are subject to scaling and offset, and each subset can have different scaling and offset. Let $T_{\text{sub},i}$ denote the i^{th} subset of points subject to scaling and offset by (c_i,e_i), and there are n_{sub} subsets. $T_{\text{sub},i}$'s are assumed to be mutually exclusive. Also, let $\vec{c} = \{c_1,c_2,...,c_{n_{\text{sub}}}\}$ and $\vec{e} = \{e_1,e_2,...,e_{n_{\text{sub}}}\}$ denote the list of different scalings and offsets for each subset of points. In this more general case, the optimization criterion minimizes

$$D_{A,\text{sub}}(s,t,p,\vec{c},\vec{e}) = \sum_{\substack{k=1 \\ b_k \notin \cup_{i=1}^{n_{\text{sub}}} T_{\text{sub},i}}}^{|p|} d(s_{a_k},t_{b_k}) + \sum_{i=1}^{n_{\text{sub}}} \sum_{\substack{k=1 \\ b_k \in T_{\text{sub},i}}}^{|p|} d(s_{a_k},c_i t_{b_k}+e_i)$$

subject to the DTW constraints, and this constrained function is also referred to as $D_{A,\text{sub}}(s,t,p,\vec{c},\vec{e})_{\text{constrained}}$. Similar to the original ADTW case, it can be shown that the following equations minimize $D_{A,\text{sub}}(s,t,p,\vec{c},\vec{e})_{\text{constrained}}$ with respect to c_i and e_i when a path p satisfying the DTW constraints is given:

$$c'_{p,i} = \frac{\sum_{\substack{k=1 \\ b_k \in T_{\text{sub},i}}}^{|p|} s_{a_k} t_{b_k} - \frac{1}{|p|}\left(\sum_{\substack{k=1 \\ b_k \in T_{\text{sub},i}}}^{|p|} s_{a_k}\right)\left(\sum_{\substack{k=1 \\ b_k \in T_{\text{sub},i}}}^{|p|} t_{b_k}\right)}{\sum_{\substack{k=1 \\ b_k \in T_{\text{sub},i}}}^{|p|} t_{b_k}^2 - \frac{1}{|p|}\left(\sum_{\substack{k=1 \\ b_k \in T_{\text{sub},i}}}^{|p|} t_{b_k}\right)^2} \tag{2.4}$$

$$e'_{p,i} = \frac{1}{|p|} \sum_{\substack{k=1 \\ b_k \in T_{\text{sub},i}}}^{|p|} (s_{a_k} - c'_{p,i} t_{b_k}) \tag{2.5}$$

The algorithm for this more general version of ADTW is shown in Algorithm 2, which is equivalent to the original ADTW algorithm with a few modifications to accommodate $(\vec{c}, \vec{e})$ instead of (c, e). For clarification, $c^l_{v,i}$ and $e^l_{v,i}$ denote the i^{th} element in the vector $\vec{c^l}_v$ and $\vec{e^l}_v$ at iteration number v. Line 8 and 9 in Algorithm 2 are equivalent to

$$(\vec{c^l}_v, \vec{e^l}_v) \leftarrow \underset{\vec{c},\vec{e}}{\text{argmin}}\ D_A(s, t, p^l_v, \vec{c}, \vec{e})_{\text{constrained}}$$

where $c^l_{v,i}$ and $e^l_{v,i}$ can be computed using Equation 2.4 and 2.5.

Algorithm 2 General ADTW

1: $p^l, \vec{c^l}, \vec{e^l}$
2: $\vec{c^l_0} \leftarrow \vec{1}$
3: $\vec{e^l_0} \leftarrow \vec{0}$
4: $D_{A,\text{sub,prev}} \leftarrow \infty$
5: $v \leftarrow 1$
6: **while** 1 **do**
7: $\quad p^l_v \leftarrow \underset{p}{\text{argmin}}\ D_{A,\text{sub}}(s, t, p, \vec{c^l}_{v-1}, \vec{e^l}_{v-1})_{\text{constrained}}$
8: $\quad$ **for** $i = 1$ `to` n_{sub} **do**
9: $\quad\quad (c^l_{v,i}, e^l_{v,i}) \leftarrow \underset{c^l_{v-1,i}, e^l_{v-1,i}}{\text{argmin}}\ D_A(s, t, p^l_v, \vec{c^l}_{v-1}, \vec{e^l}_{v-1})_{\text{constrained}}$
10: $\quad$ **if** $D_{A,\text{sub,prev}} - D_{A,\text{sub}}(s, t, p, \vec{c^l}_v, \vec{e^l}_v) < D_{\text{stop}}$ **then**
11: $\quad\quad p^l \leftarrow p^l_v$
12: $\quad\quad \vec{c^l} \leftarrow \vec{c^l}_v$
13: $\quad\quad \vec{e^l} \leftarrow \vec{e^l}_v$
14: $\quad\quad$ **break**
15: $\quad v \leftarrow v + 1$

Figure 2.8 illustrates an example aligning a time series with its temporal-varied version with subsets of points scaled and offset differently. For this more complicated example, general ADTW clearly generates a closer alignment to the ideal one when compared to ADTW, since general ADTW models subsets of points having different scalings and offsets. However, a critical catch to applying general ADTW is that we need to know what subsets can be scaled and offset differently.

Now, let us analyze the time and space complexities for general ADTW, and we start off by assuming that the time series lengths are equal ($n = m$) for simplicity. While

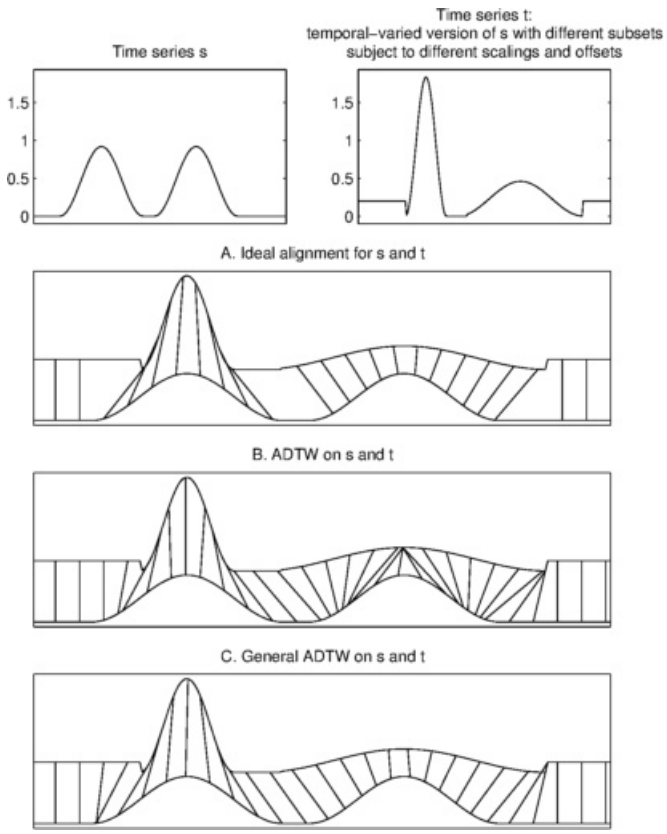

Figure 2.8: ADTW and general ADTW on synthetic time series where subsets of points can be scaled and offset differently.

there seems to be additional operations on line 8 and 9 in Algorithm 2 when compared to Algorithm 1, the time complexity for estimating the scalings and offsets is actually $O(2n-1)$. This observation comes from the model assumption that $T_{\text{sub},i}$'s are mutually exclusive, so line 8 and 9 only iterate through all pairs of matches in p_v^l, which is at most $2n-1$ in length. Line 7 requires applying the DTW algorithm, which is $O(w_b n)$ in time. The remaining operations in each iteration run in constant time, so

$$O(g(2n-1+w_b n)) = O(2gn - g + gw_b n) = O(gw_b n)$$

is the time complexity for general ADTW. As for the space complexity, constructing the DTW table remains to be the limiting factor because other space complexities are either $O(1)$ or $O(n_{\text{sub}})$. We know that $n_{\text{sub}} \leq n$, and in practice $n_{\text{sub}} << n$ because the number of

subsets subject to individual scaling and offset should be much less than the total number of points in a time series. Therefore, the space complexity for general ADTW is $O(w_b n)$. Note that the time and space complexities for general ADTW are identical to those of ADTW.

Revisiting the ADTW case, constraints can also be placed on (c, e) to avoid implausible values for scaling and offset. For example, looking back at Figure 2.7 again, one suboptimal solution has a negative c. In some situations, it is reasonable to hypothesize that $c_{\min} \leq c \leq c_{\max}$, where $c_{\min}$ and $c_{\max}$ are the minimum and maximum scalings that one expects respectively. The same logic applies to the offset e. Going back to the ADTW formulation, the optimization problem is slightly modified to minimize

$$(p^*, c^*) = \underset{p, c_{\min} \leq c \leq c_{\max}, e_{\min} \leq e \leq e_{\max}}{\operatorname{argmin}} D_A(s, t, p, c, e)_{\text{constrained}}$$

Revisiting the ADTW algorithm in Algorithm 1, only the estimation of (c_v^l, e_v^l) on line 8 needs to be changed to accommodate the extra constraints:

$$(c_v^l, e_v^l) = \underset{c_{\min} \leq c \leq c_{\max}, e_{\min} \leq e \leq e_{\max}}{\operatorname{argmin}} D_A(s, t, p_v^l, c, e)_{\text{constrained}}$$

The above constrained minimization problem can be solved using quadratic programming, and Algorithm 2 for general ASDTW can also be modified in a similar way to accommodate a set of constraints $\cup_{i=1}^{n_{\text{sub},i}} \{c_{\min,i} \leq c_i \leq c_{\max,i}, e_{\min,i} \leq e_i \leq e_{\max,i}\}$.

Similar to DTW where $D(s, t, p^*)$ can be interpreted as a difference measure between two time series s and t, $D_A(s, t, p^*, c^*, e^*)$ can also be used as a difference measure for two time series subject to scaling and offset. For ADTW, $D_A(s, t, p^*, c^*, e^*)$ is approximated by $D_A(s, t, p^l, c^l, e^l)$, because finding (p^*, c^*, e^*) is too costly as mentioned previously. If the same type of time series follows the affine model, it is not unreasonable to claim that the ADTW difference measure tends to be better than the DTW difference measure. An example is illustrated in Figure 2.9, where time series u contains a triangular window, and s and t each contains a Hamming window. Time series s is connected to the time series which is more similar to s based on the difference measure. If the discrimination lies in shape and not in differences in scaling and offset, the difference measure between s and t should be lower than the difference measure between s and u, which is exactly what the ADTW measure offers. On the other hand, DTW finds s to be closer to u because it does not discard the difference in amplitude. Hence, this simplistic example supports the notion that the ADTW measure is better than the DTW measure when instances from the same type of time series are known to be scaled and offset differently.

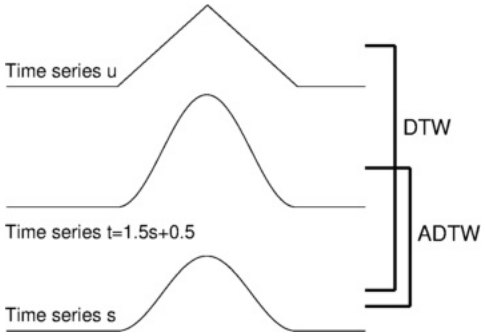

Figure 2.9: ADTW versus DTW difference measure for a simplistic example.

2.3 Regional Dynamic Time Warping

Regional dynamic time warping (RDTW) modifies DTW to place more weight in a region of points potentially representative of a component of interest in a time series. This is accomplished by substituting the pointwise distance measure d with a distance d_r that measures the difference between points in a region. Let $w_r = 1 + 2w_h \in \mathbb{Z}_{>0}$ be the region width to consider, and recall that time series s and t each have length n and m respectively. Then, RDTW has an modified optimization problem of finding an alignment p^* that minimizes

$$D_R(s, t, p, w_h) = \sum_{k=1}^{|p|} d_r(s_{a_k}, t_{b_k}, w_h)$$

subject to the DTW constraints, where

$$d_r(s_a, t_b, w_h) = \frac{1}{w_{a,b}} \sum_{\substack{w=-w_h \\ w:1\leq a+w\leq n \\ w:1\leq b+w\leq m}}^{w_h} d(s_{a+w}, t_{b+w})$$

and

$$w_{a,b} = \sum_{\substack{w=-w_h \\ w:1\leq a+w\leq n \\ w:1\leq b+w\leq m}}^{w_h} 1$$

Note that the $1 \leq a + w \leq n$ and $1 \leq b + w \leq m$ conditions only come into effect for points close to the start and end of the time series, and $w_{a,b} = w_r$ for points that

are away from the boundaries. For brevity, $D_R(s, t, p, w_h)$ subject to DTW constraints is denoted as $D_R(s, t, p, w_h)_{\text{constrained}}$. Since the only difference between RDTW and DTW lies in substituting the distance function d with d_r, this modified optimization problem still exhibits properties of optimal substructures and overlapping subproblems. Hence, dynamic programming can again be utilized in the same manner as DTW, where the update formula for RDTW is modified from the DTW update formula in Equation 2.1 as follows:

$$D_R(s, t, p^*_{(a,b)}, w_h) = d_r(s_a, t_b, w_h) + \min(D_R(s, t, p^*_{(a-1,b-1)}, w_h), D_R(s, t, p^*_{(a,b-1)}, w_h), D_R(s, t, p^*_{(a-1,b)}, w_h)) \tag{2.6}$$

Again, note that if $a - 1 < 1$, $D_R(s, t, p^*_{(a-1,b-1)}, w_h)$ and $D_R(s, t, p^*_{(a-1,b)}, w_h)$ are not considered, and if $b - 1 < 1$, $D_R(s, t, p^*_{(a-1,b-1)}, w_h)$ and $D_R(s, t, p^*_{(a,b-1)}, w_h)$ are not considered in the update formula to accommodate the boundary cases. This RDTW update formula is used to construct a table identical to the DTW table shown in Table 2.2, except each $D(s, t, p^*_{(a,b)})$ is replaced with $D_R(s, t, p^*_{(a,b)}, w_h)$. This table is also referred to as the RDTW table. The same backtracking technique outlined in the DTW section can be used on the RDTW table to obtain the optimal alignment p^*.

By changing the pointwise distance d to the regional distance d_r, RDTW is able to find pointwise alignments that match one region to another region. If this region reflects a typical component within a time series and this component exists in two time series s and t, RDTW encourages pointwise alignments such that these components can be considered as being matched to each another.

The emphasis on a region instead of each individual points in RDTW can bring about situational advantages. Figure 2.10 and 2.11 demonstrate that RDTW can provide better pointwise matches for two time series than DTW when noise or overlap of components is present. For both figures, a match between two points in different time series is denoted by connecting the two points with a solid line. In Figure 2.10, two time series contain the same Hamming window, and white Gaussian noise is added to both time series. The ideal alignment without noise is illustrated in Figure 2.10A. As shown in Figure 2.10B and C, RDTW provides an alignment that is visually closer to the ideal alignment than DTW. Since RDTW accounts for the region surrounding a point for each pointwise match, it has an averaging effect.

In Figure 2.11, each time series is composed of two Hamming windows, where the left Hamming window is subject to random horizontal shifts that can result in an overlap between the two windows. Ideally, the left Hamming windows should be aligned together and the right Hamming windows should be aligned together, as shown in Figure 2.11A.

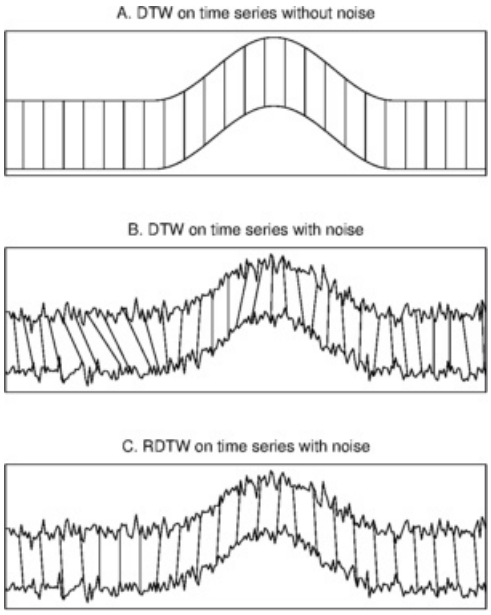

Figure 2.10: DTW and RDTW alignments of two synthetic time series with noise.

Figure 2.11B and C each shows two time series with different degrees of overlap of the Hamming windows. DTW produces an alignment that does not adhere to the morphology by matching a good portion of points in the left Hamming window to points in the right Hamming window. On the other hand, RDTW yields a more reasonable alignment because the left windows are mostly matched together and the right windows are mostly matched together.

Now, let us revisit the motivating real-world example provided in the introductory chapter for emphasizing on components. Recall that a motor unit potential (MUP) is comprised of multiple muscle fiber potentials (MFPs), and analysis of MUPs is useful in finding muscle abnormalities. In Figure 2.12, each MUP contains two MFPs. The ideal alignment is to match the left MFPs together and the right MFPs together, as shown in Figure 2.12A. Looking at Figure 2.12B and C, RDTW offers better alignment because the left MFPs are aligned together and the right MFPs are aligned together, whereas DTW aligns a good portion of the left MFP in the top MUP with the right MFP in the bottom MUP.

For simplicity of complexity analysis, let us assume again that $n = m$ and recall that the time complexity for banded DTW is $O(w_b n)$, where w_b is the width of the band. Similar

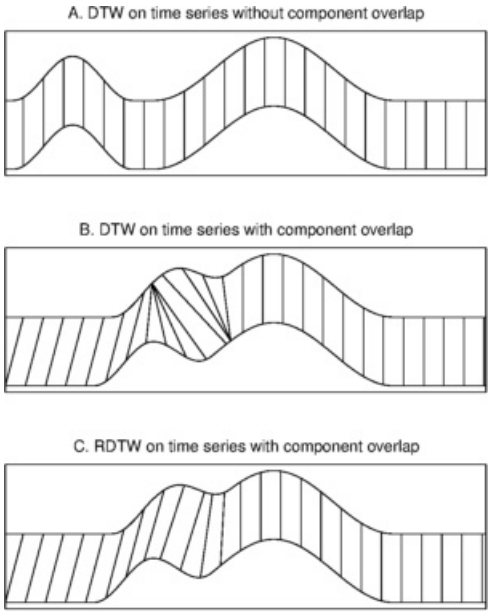

Figure 2.11: DTW and RDTW alignments of two synthetic time series with varying degrees of component overlap.

to the DTW case, the Sakoe-Chiba band can be applied to RDTW as well. At first glance, the time complexity for banded RDTW is $O(w_r w_b n)$, because the RDTW update formula is $O(w_r)$ for computing the regional distance d_r and the RDTW table requires $O(w_b n)$ elements to be filled with the update formula. However, it turns out that most elements in the RDTW table can be updated with a time complexity of $O(1)$ instead of $O(w_r)$ using the following observations:

$$d_r(s_{a-1}, t_{b-1}, w_h) = \frac{1}{w_{a-1,b-1}} \sum_{\substack{w=-w_h \\ w:1\leq a-1+w\leq n \\ w:1\leq b-1+w\leq m}}^{w_h} d(s_{a-1+w}, t_{b-1+w})$$

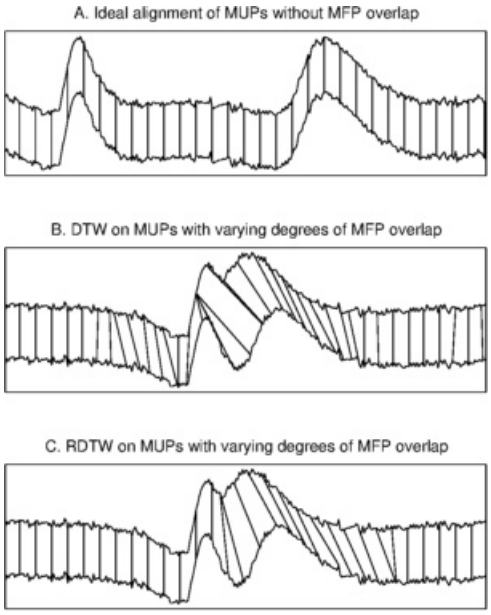

Figure 2.12: DTW and RDTW alignments of two MUPs with varying degrees of MFP overlap.

and

$$\begin{aligned} d_r(s_a, t_b, w_h) &= \frac{1}{w_{a,b}} \sum_{\substack{w=-w_h \\ w:1\leq a+w\leq n \\ w:1\leq b+w\leq m}}^{w_h} d(s_{a+w}, t_{b+w}) \\ &= \frac{1}{w_{a,b}}[-d(s_{a-w_h-1}, t_{b-w_h-1}) + w_{a-1,b-1} d_r(s_{a-1}, t_{b-1}, w_h) + d(s_{a+w_h}, t_{b+w_h})] \end{aligned} \tag{2.7}$$

Note that $d(s_a, t_b) = 0$ if $a < 1 \cup a > n \cup b < 1 \cup b > m$. In other words, $d(s_a, t_b)$ is set to 0 when either s_a or t_b does not exist. If s_{a-1} and t_{b-1} exist, the above observation leads to the following update formula (replacing the one in Equation 2.6):

$$\begin{aligned} D_R(s, t, p^*_{(a,b)}, w_h) =& \frac{1}{w_{a,b}}[-d(s_{a-w_h-1}, t_{b-w_h-1}) + w_{a-1,b-1} d_r(s_{a-1}, t_{b-1}) + d(s_{a+w_h}, t_{b+w_h})] + \\ & \min(D_R(s, t, p^*_{(a-1,b-1)}, w_h), D_R(s, t, p^*_{(a,b-1)}, w_h), D_R(s, t, p^*_{(a-1,b)}, w_h)) \end{aligned}$$

which is $O(1)$ in time when $d_r(s_{a-1}, t_{b-1}, w_h)$ is given. The grey part shown in Figure 2.13 represents elements in the RDTW table that cannot be updated with the above $O(1)$

formula, which only corresponds to the first row and first column of the RDTW table. When the RDTW table is not banded, $n^2 - 2n + 1$ out of n^2 elements can be updated in constant time.

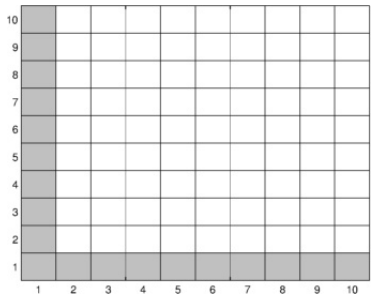

Figure 2.13: Elements in the RDTW Table Which Can Be Updated in Constant Time.

In the banded scenario, the $O(1)$ update formula in Equation 2.6 cannot apply to w_b elements (constrained by the first row and column after considering the band), and the original $O(w_r)$ update formula is required for these elements. Hence,

$$O((w_b)w_r) + O((w_b n - w_b)1) = O(w_b(w_r + n))$$

is the time complexity for constructing a RDTW table. In addition, region width w_r is necessarily less than or equal to n, so the time complexity for constructing a RDTW table can be further approximated as $O(w_b n)$. Similar to DTW, construction of the RDTW table is the limiting factor for finding the optimal alignment, so the total time complexity for RDTW is $O(w_b n)$.

Unbanded RDTW with this $O(1)$ update formula is empirically compared to unbanded RDTW with the original $O(w_r)$ update formula in Figure 2.14. Two random time series of length $n = 1000$ are generated, and the two aforementioned versions of RDTW are applied to the time series across different ratios of region half-width $w_c = \frac{w_h}{n}$, where the region width $w_r = 1 + 2w_h$. As expected, the RDTW runtime increases linearly with w_r when the $O(w_r)$ update formula is used, and it stays constant when the $O(1)$ update formula is used. The two versions have also been empirically confirmed to produce the same alignment.

Originally, the space required for RDTW remains exactly the same as that required for DTW, because only the RDTW table and path need to be stored. However, with the above modification for reducing the RDTW time complexity, additional memory is required to store $d_r(s_{a-1}, t_{b-1})$'s. Note that $d_r(s_{a-1}, t_{b-1}, w_h)$ is only being used by $d_r(s_a, t_b, w_h)$ in the

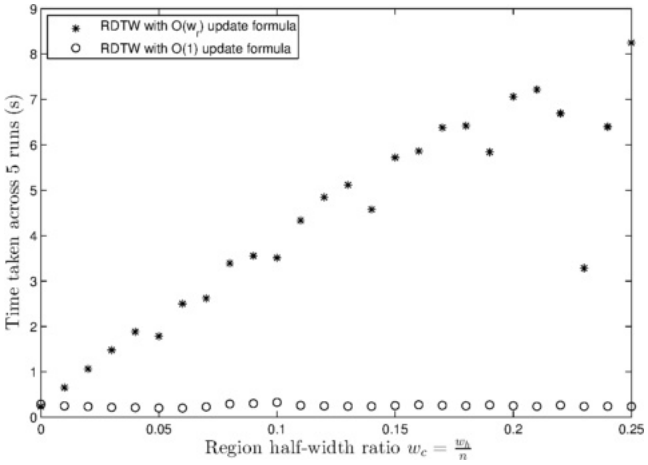

Figure 2.14: Empirical evaluation for time complexity of RDTW using $O(w_r)$ or $O(1)$ update formula with respect to $w_c = \frac{w_h}{n}$.

$O(1)$ update formula, so one strategy is to store an entire row of $d_r(s_{a-1}, t_{b-1}, w_h)$'s for the same a and different b's constrained by the Sakoe-Chiba band, and then the next row of $d_r(s_a, t_b, w_h)$'s can be filled. Thus, the additional space complexity is $O(w_r)$ and the total space complexity is

$$O(w_b n) + O(w_r) = O(w_b n)$$

where $O(w_b n)$ is the limiting space complexity that stores the RDTW table. Surprisingly, banded RDTW turns out to have the same time and space complexities of $O(w_b n)$ as banded DTW.

So far, we have not discussed in detail effects of the region width $w_r = 1 + 2w_h$. This parameter is crucial to RDTW achieving the desired result. For example, in Figure 2.10, w_r is roughly set to be the width of the main Hamming window, and in Figure 2.11, w_r is roughly set to be the width of the left Hamming window. Intuitively, since RDTW finds pointwise matches that minimize the difference between regions of width w_r around each pair of matched points, setting w_r to reflect the width of the component of interest means that RDTW will emphasize on matching regions with the same width. It should be noted that if $w_r = 1$, RDTW is identical to DTW.

Figure 2.15 illustrates effects of different region widths on RDTW alignments on time series with noise or overlapping components. The region width w_r is proportional to $w_c = \frac{w_h}{n}$, so varying w_c is equivalent to varying w_r. In the first column, two identical time series have random noise added to them, so the ideal alignment should consist of purely

vertical matches. In this case, when $w_r = 1$, the matches are far away from being vertical, and the remaining widths seem to be more reasonable, where $\{w_c : w_c \geq 0.3\}$ yield the most vertical matches. In the second column, each time series consists of two Hamming windows that are subject to different degrees of overlap. In this case, when $w_c = 0$, pathological alignments are observed where a good portion of two Hamming windows are aligned to one Hamming window. For $w_c = 0.1$ and $w_c = 0.2$, the alignments seem reasonable because the local peaks representing each Hamming window are aligned correctly. For the remaining alignments with higher w_c's, the peak of the left Hamming window in the top time series is no longer matched to that in the bottom time series, so these alignments are considered to be worse. Furthermore, the examples in Figure 2.15 support the aforementioned intuition that w_r should be set to the width of the component of interest, because the best w_r's are found to be roughly equal to the width of the Hamming windows ($w_c = 0.3$ for the left column and $w_c = 0.1$ or $w_c = 0.2$ for the right column).

From previous examples, we are given the insight that both small and large region widths can lead to bad alignments when using RDTW, so selecting an appropriate region width is instrumental to RDTW's performance. However, selecting a different width can yield a different alignment. One approach is to find the best pair of alignment and region width $(\hat{p}, \hat{w_h})$ that minimizes the RDTW optimization function:

$$(\hat{p}, \hat{w_h}) = \underset{p, w_h : w_h \geq 0}{\operatorname{argmin}} \; D_R(s, t, p, w_h)_{\text{constrained}}$$

Let $p^*_{w_h}$ denote the best alignment for width w_h, and we know that $p^*_{w_h}$ can be found using the RDTW algorithm when w_h is given. Then, this optimization problem can be rewritten as

$$(\hat{p} = p^*_{\hat{w_h}}, \hat{w_h}) = \underset{w_h : w_h \geq 0}{\operatorname{argmin}} \; D_R(s, t, p^*_{w_h}, w_h)_{\text{constrained}} \tag{2.8}$$

The above formulation requires plugging in all possible values of w_h, which is not feasible since there is an infinite number of values to try, but this can be accommodated by only trying out a finite set of selected values. Equation 2.8 is strongly dependent on whether a lower value of $D_R(s, t, p, w_h)$ reflects a better alignment across different p's and w_h's. So far, we have shown that if w_h remains constant, minimizing $D_R(s, t, p, w_h)$ with respect to p can provide better alignments when d is the squared difference function. Does minimizing $D_R(s, t, p, w_h)$ with respect to different p's and w_h's produce better alignments? To answer this question, we first reexamine the noise and overlap examples in Figure 2.15, and we are interested in looking at values for $D_R(s, t, p^*_{w_h}, w_h)$ across the different widths w_h's. If the formulation in Equation 2.8 works well, the visually best alignments should have the same region width as the region width that produces the minimal D_R value. So far, d has been set to be the squared difference. Looking at Table 2.3, the optimal region

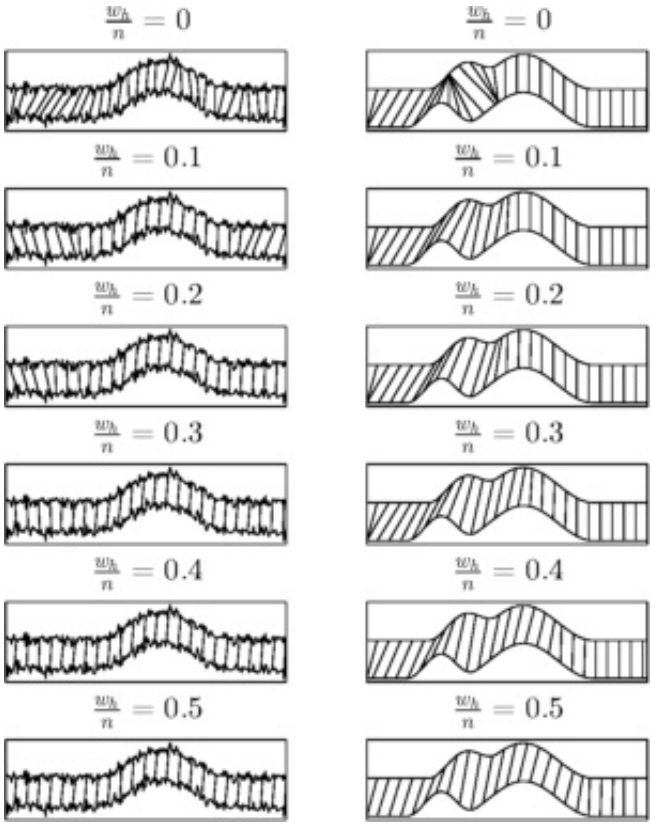

Figure 2.15: Effects of different region widths $w_r \propto \frac{w_h}{n}$ on RDTW alignments.

half-width width ratio w_c in the context of minimizing D_R is found to 0 for both the noise and overlap examples, which corresponds to the worst alignments in Figure 2.15. It is not unreasonable to hypothesize that this result is an artifact of using the squared difference, so the absolute difference is also explored. Empirical evaluation has been done to validate that substituting d with the squared difference or absolute difference both produce the same figure. Looking at Table 2.3 for the absolute difference case, the minimizing w_c's are again 0's, which correspond to the worst alignments. We hence draw the conclusion that minimizing D_R with respect to p and w_h does not necessarily output a better alignment across different w_h's. One practical way of finding the best width is to tune it for the best results, and this approach will be used for future evaluations. In this context, tuning a parameter is equivalent to choosing the value that provides the best result by plugging in different values for the parameter.

Table 2.3: Effects of different region widths w_r and different pointwise distance measures d on D_R.

Region half-width ratio ($w_c = \frac{w_h}{n}$)	Squared difference		Absolute difference	
	Noise	Overlap	Noise	Overlap
0.0	**1.3**	**1.5**	**15.9**	**9.1**
0.1	4.1	4.3	25.5	12.6
0.2	4.3	5.1	26.1	15.9
0.3	4.4	6.9	26.4	21.0
0.4	4.5	8.3	26.6	24.4
0.5	4.5	8.6	26.7	23.9

The previous conclusion that minimizing $D_R(s,t,p,w_h)_{\text{constrained}}$ with respect to both p and w_h does not produce better alignments might seem bewildering at first glance. After all, minimizing $D_R(s,t,p,w_h)_{\text{constrained}}$ with respect to p works well when w_h is fixed. In addition, ADTW also produces good alignments for time series falling under the affine model by trying to minimize $D_A(s,t,p,c,e,)_{\text{constrained}}$ with respect to p, c and e. However, the approach of obtaining the best parameters by optimizing a function is not guaranteed to produce sensible results. In other words, $D_R(s,t,p,w_h)_{\text{constrained}}$ is not a good cost function to minimize with respect to w_h.

In the examples that have been shown thus far, temporal variations other than shifting are not imposed on components. The regional distance d_r computes a distance between two regions of points by assuming vertical matches between the two regions. To deal with general temporal variations between components, it is reasonable to withdraw the assumption of vertical matches and apply DTW to match these regions. To realize this approach, d_r is replaced with $d_{r'}$ to reflect the DTW difference measure:

$$d_r(s_a, t_b, w_h) = \frac{1}{w_{a,b}} D(s'_{a,b,w_h}, t'_{a,b,w_h}, p^*_{s'_{a,b,w_h}, t'_{a,b,w_h}})$$

where $s'_{a,b,w_h} = (s_{a+w} : -w_h \leq w \leq w_h, 1 \leq a+w \leq n, 1 \leq b+w \leq m)$, $t'_{a,b,w_h} = (t_{b+w} : -w_h \leq w \leq w_h, 1 \leq b+w \leq m, 1 \leq a+w \leq n)$, and $p^*_{s'_{a,b,w_h}, t'_{a,b,w_h}}$ is the optimal DTW alignment for the subsequences s'_{a,b,w_h} and t'_{a,b,w_h}. This RDTW-based alignment approach that allows temporal variations within components will be referred to as RDTW'. In Figure 2.16, RDTW' is compared with RDTW based on the previous noise and component overlap examples, and the best region width for each method is utilized. It is found that the matches within each region can be highly pathological, thereby resulting in worse alignments for RDTW'. While additional path constraints can be added to reduce pathological

matches within each region, RDTW remains better for the noise and overlap examples, and RDTW' would require additional complexity in tuning the additional constraints. For the above reasons, RDTW' will not be further explored, and the problem of accounting for general temporal variations within each component is left as an open problem in this thesis.

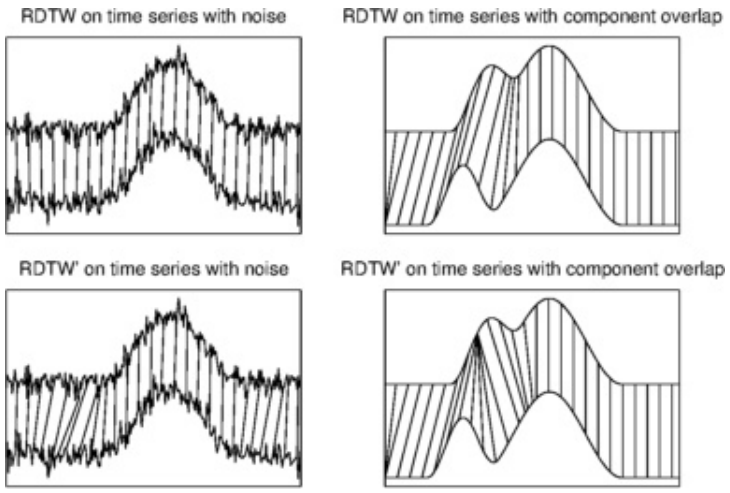

Figure 2.16: RDTW and RDTW' alignments of time series subject to noise or component overlap.

Similar to DTW and ADTW where the optimized function value is interpreted as a difference measure between two time series s and t, $D_R(s, t, p^*, w_h)$ can also be used as a difference measure for two time series with emphasis on components. $D_R(s, t, p^*, w_h)$ will also be referred to as the RDTW difference measure. An example is illustrated in Figure 2.17, where time series u contains a triangular window and a Hamming window, and s and t each consists of two Hamming windows. These time series are subject to varying degrees of component overlap. Time series s is connected to the time series that is more similar to s based on the difference measure. It seems more reasonable to group s and t together and differentiate u from s, which is exactly what the RDTW measure offers. On the other hand, DTW finds s to be closer to u because it lacks component emphasis. This example supports the notion that the RDTW measure can be better than the DTW measure when local components play a crucial role in discrimination.

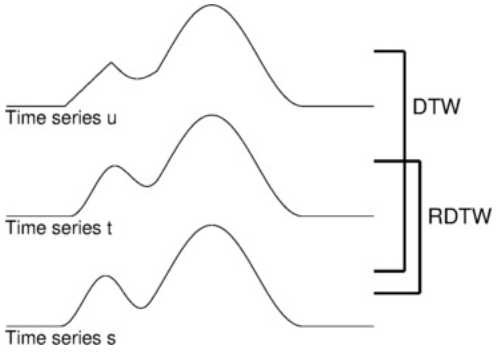

Figure 2.17: RDTW versus DTW difference measure for an example with component overlap.

2.4 Affine Regional Dynamic Time Warping

ADTW and RDTW each has its unique model or preference, where ADTW models time series to be scaled and offset in amplitude when aligned, and RDTW places an emphasis on matching regions. In this section, we show two different ways of combining ADTW and RDTW to realize alignments that incorporate the advantages of both ADTW and RDTW.

2.4.1 Global-Affine Regional Dynamic Time Warping

In global-affine regional dynamic time warping (GARDTW), one time series is modeled as the scaled and offset version of another time series when aligning them with regional emphasis. The affine property is modeled globally for the entire time series. Again, let s and t be two time series of length n and m respectively, and let $w_r = 1 + 2w_h$ denote the region width. Mathematically, the goal is to find a path p^*, scaling factor $c^* \in \mathbb{R}$ and offset factor $e^* \in \mathbb{R}$ that minimize

$$D_G(s, t, p, c, e, w_h) = \sum_{k=1}^{|p|} d_g(s_{a_k}, t_{b_k}, c, e, w_h)$$

subject to the DTW constraints imposed on p, where

$$d_g(s_{a_k}, t_{b_k}, c, e, w_h) = \frac{1}{w_{a_k,b_k}} \sum_{\substack{w=-w_h \\ w:1\le a_k+w\le n \\ w:1\le b_k+w\le m}}^{w_h} d(s_{a_k+w}, ct_{b_k+w} + e)$$

For brevity, $D_G(s, t, p, c, e, w_h)$ constrained by the DTW conditions will be referred to as $D_G(s, t, p, c, e, w_h)_{\text{constrained}}$. Following similar steps to the derivation procedure for c'_p and e'_p in the ADTW chapter, it can be shown that the following c and e minimize $D_G(s, t, p, c, e, w_h)$ when s, t, p and w_h are given and d is the squared difference:

$$c'_p = \frac{\rho - \frac{1}{|p|}\tau\phi}{\gamma - \frac{1}{|p|}\tau^2} \tag{2.9}$$

$$e'_p = \frac{1}{|p|}(\phi - c'_p\tau) \tag{2.10}$$

where

$$\rho = \sum_{k=1}^{|p|} \frac{1}{w_{a_k,b_k}} \sum_{\substack{w=-w_h \\ w:1\leq a_k+w\leq n \\ w:1\leq b_k+w\leq m}}^{w_h} s_{a_k+w} t_{b_k+w}$$

$$\gamma = \sum_{k=1}^{|p|} \frac{1}{w_{a_k,b_k}} \sum_{\substack{w=-w_h \\ w:1\leq a_k+w\leq n \\ w:1\leq b_k+w\leq m}}^{w_h} t^2_{b_k+w}$$

$$\tau = \sum_{k=1}^{|p|} \frac{1}{w_{a_k,b_k}} \sum_{\substack{w=-w_h \\ w:1\leq a_k+w\leq n \\ w:1\leq b_k+w\leq m}}^{w_h} t_{b_k+w}$$

$$\phi = \sum_{k=1}^{|p|} \frac{1}{w_{a_k,b_k}} \sum_{\substack{w=-w_h \\ w:1\leq a_k+w\leq n \\ w:1\leq b_k+w\leq m}}^{w_h} s_{a_k+w}$$

The full derivation for Equation 2.9 and 2.10 is given in Appendix A.

Directly minimizing $D_G(s, t, p, c, e, w_h)_{\text{constrained}}$ with respect to (p, c, e) is not feasible because there are too many paths p's to try even though (c^*, e^*) can be computed when p^* is given. Hence, similar to the ADTW case, we look for an suboptimal solution (p^g, c^g, e^g) using the same iterative strategy outlined in the ADTW chapter. The GARDTW algorithm for finding (p^g, c^g, e^g) is described in Algorithm 3.

Looking at Algorithm 3, it is not clear how to obtain p^g_v and (c^g_v, e^g_v) for each iteration. Line 7 is equivalent to

$$p^g_v \leftarrow D_R(s, c^g_{v-1}t + e^g_{v-1}, p, w_h)_{\text{constrained}}$$

Algorithm 3 GARDTW

p^g, c^g, e^g
$c_0^g \leftarrow 1$
$e_0^g \leftarrow 0$
$D_{G,\text{prev}} \leftarrow \infty$
$v \leftarrow 1$
while 1 **do**
    $p_v^g \leftarrow \underset{p}{\text{argmin}}\ D_G(s, t, p, c_{v-1}^g, e_{v-1}^g, w_h)_{\text{constrained}}$
    $(c_v^g, e_v^g) \leftarrow \underset{c,e}{\text{argmin}}\ D_G(s, t, p_v^g, c, e, w_h)_{\text{constrained}}$
    if $D_{G,\text{prev}} - D_G(s, t, p^g, c^g, e^g) < D_{\text{stop}}$ **then**
        $p^g \leftarrow p_v^g$
        $c^g \leftarrow c_v^g$
        $e^g \leftarrow e_v^g$
        break
    $v \leftarrow v + 1$

so p_v^g can be obtained using RDTW. (c_v^g, e_v^g) can be calculated in closed form with using Equation 2.9 and 2.10. Similar to ADTW, $D_{\text{stop}} \in \mathbb{R}$ is a small value chosen by the user for checking convergence of the algorithm. The iterative optimization strategy utilized in Algorithm 3 has been previously used for ADTW, and the theory behind this strategy has been discussed in detail in the ADTW chapter. $D_G(s, t, p^g, c^g, e^g, w_h)$ will be referred to as the GARDTW difference measure.

Figure 2.18 illustrates the effects of GARDTW compared to DTW, ADTW and RDTW. Time series s and t each consists two Hamming windows, where the Hamming windows in t are the scaled and offsetted versions of the Hamming windows in s, and the components have varying degrees of overlap. Under this model, we prefer to match the respective components together as shown in Figure 2.18A without component overlap. GARDTW offers an alignment that is closest to matching the left components together and matching the right components together. This result is attributed to GARDTW's ability to handle affine properties and component overlap simultaneously. Note that GARDTW assumes that the affine property is applied globally to an entire time series. From this example, we observe that GARDTW can offer advantages of both ASDTW and RDTW, where ADTW better handles different scales and offsets in amplitude for the entire time series, and RDTW better handles scenarios that require an emphasis on regions.

Now, let us examine the time and space complexities for GARDTW. For simplicity,

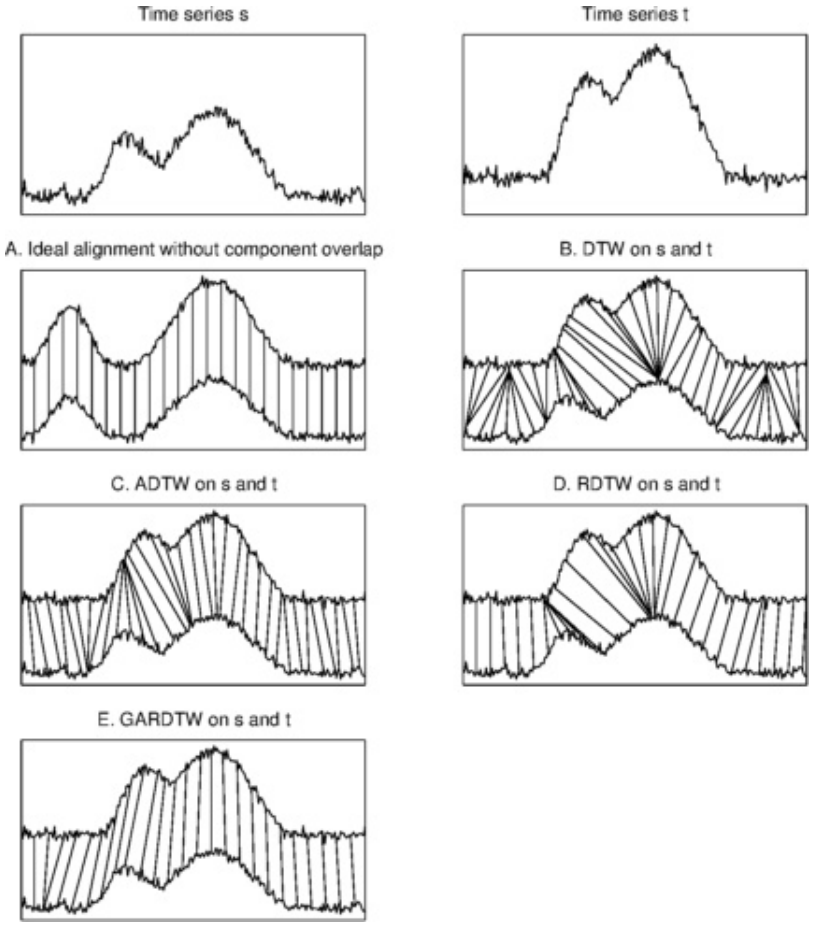

Figure 2.18: DTW, ADTW, RDTW and GARDTW alignments of two time series under the affine model having noise and overlapping components.

let us assume that $n = m$. In each iteration, p_v^g is updated with RDTW, and (c_v^g, e_v^g) is updated with Equation 2.9 and 2.10. RDTW has been shown to be $O(w_b n)$ in time and space. Looking at Equation 2.9 and 2.10, computing ρ, γ, τ and ϕ each requires iterating through each element in a path p, and a region of width w_r around each element is gone over. Recall that a path p satisfying the DTW constraints can at most be of length $2n-1$. Hence, computation of ρ, γ, τ and ϕ takes

$$O((2n-1)w_r) = O(w_r n)$$

in time. When ρ, γ, τ and ϕ are given, Equation 2.9 and 2.10 are $O(1)$ in time, so Equation

2.9 and 2.10 are $O(w_r n)$ in time when ρ, γ, τ and ϕ are not known. Since s, t and p_v^g have been previously stored, Equation 2.9 and 2.10 are $O(1)$ in space. Hence, each iteration of the GARDTW algorithm described in Algorithm 3 is

$$O(w_b n) + O(w_r n) = O((w_b + w_r)n)$$

in time and

$$O(w_b n) + O(1) = O(w_b n)$$

in space. Let λ be the number of iterations required for convergence of the GARDTW algorithm. Then, the GARDTW algorithm is $O(\lambda(w_b + w_r)n)$ in time and $O(w_b n)$ in space. This algorithm combining ADTW and RDTW in a global manner has a higher time complexity than both ADTW and RDTW. Similar to the case for ADTW, a good initialization for (c_0^g, e_0^g) and a reasonable value for D_{stop} can be highly effective in lowering λ. For the remainder of this thesis, $(c_0^g, e_0^g) = (1, 0)$ and $D_{\text{stop}} = 10^{-5}$ unless mentioned otherwise. As an example, $\lambda = 8$ for the GARDTW alignment in Figure 2.18E under the aforementioned setting.

2.4.2 Local-Affine Regional Dynamic Time Warping

Local-affine regional dynamic time warping (LARDTW) is a modified version of RDTW where the region surrounding each point is assumed to be a scaled and offset version of another region surrounding the corresponding matched point. Similar to RDTW, LARDTW emphasizes on regions, but it assumes that a pair of matched regions falls under the affine model. Let $w_r = 1 + 2w_h \in \mathbb{Z}_{>0}$ be the region width to consider, and recall that time series s and t each has length n and m respectively. Then, LARDTW finds an alignment p^* that minimizes

$$D_L(s, t, p, w_h) = \sum_{k=1}^{|p|} d_l(s_{a_k}, t_{b_k}, w_h)$$

subject to the DTW constraints, where

$$d_l(s_a, t_b, w_h) = \frac{1}{w_{a,b}} \min_{c_{a,b}, e_{a,b}} \sum_{\substack{w=-w_h \\ w:1 \le a+w \le n \\ w:1 \le b+w \le m}}^{w_h} d(s_{a+w}, c_{a,b} t_{b+w} + e_{a,b}) \tag{2.11}$$

and

$$w_{a,b} = \sum_{\substack{w=-w_h \\ w:1 \le a+w \le n \\ w:1 \le b+w \le m}}^{w_h} 1$$

If d is the squared difference, the minimizing $(c^*_{a,b}, e^*_{a,b})$ can be obtained using the following equations for each pair of matched points (s_a, t_b):

$$c^*_{a,b} = \frac{\sum_{\substack{w=-w_h \\ w:1\le a+w\le n \\ w:1\le b+w\le m}}^{w_h} s_{a+w}t_{b+w} - \frac{1}{w_{a,b}}(\sum_{\substack{w=-w_h \\ w:1\le a+w\le n \\ w:1\le b+w\le m}}^{w_h} s_{a+w})(\sum_{\substack{w=-w_h \\ w:1\le a+w\le n \\ w:1\le b+w\le m}}^{w_h} t_{b+w})}{\sum_{\substack{w=-w_h \\ w:1\le a+w\le n \\ w:1\le b+w\le m}}^{w_h} t^2_{b+w} - \frac{1}{w_{a,b}}(\sum_{\substack{w=-w_h \\ w:1\le a+w\le n \\ w:1\le b+w\le m}}^{w_h} t_{b+w})^2} \tag{2.12}$$

$$e^*_{a,b} = \frac{1}{w_{a,b}} \sum_{\substack{w=-w_h \\ w:1\le a+w\le n \\ w:1\le b+w\le m}}^{w_h} (s_{a+w} - c^*_{a,b}t_{b+w}) \tag{2.13}$$

For brevity, $D_L(s, t, p, w_h)$ subject to DTW constraints is denoted as $D_L(s, t, p, w_h)_{\text{constrained}}$. Since the only difference between LARDTW and DTW lies in substituting the distance function d with d_l, this modified optimization problem still exhibits properties of optimal substructures and overlapping subproblems. Hence, dynamic programming can again be utilized in the same manner as DTW, where the update formula for LARDTW is modified from the DTW update formula in Equation 2.1 as follows:

$$D_L(s, t, p^*_{(a,b)}, w_h) = d_l(s_a, t_b, w_h) + \min(D_L(s, t, p^*_{(a-1,b-1)}, w_h), D_L(s, t, p^*_{(a,b-1)}, w_h), D_L(s, t, p^*_{(a-1,b)}, w_h)) \tag{2.14}$$

This LARDTW update formula is used to construct a table identical to the DTW table shown in Table 2.2, except each $D(s, t, p^*_{(a,b)})$ element is replaced with $D_L(s, t, p^*_{(a,b)}, w_h)$. This table is also referred to as the LARDTW table. The same backtracking technique outlined in the DTW section can be used on the LARDTW table to construct the optimal alignment p^*. The obtained value of $D_L(s, t, p^*, w_h)$ will be referred to as the LARDTW difference measure.

Figure 2.19 illustrates the effects of LARDTW compared to DTW, ADTW, RDTW and GARDTW. Time series s and t are each comprised of two Hamming windows, where each Hamming window in t is scaled differently. In this case, we prefer to match the respective components together as shown in Figure 2.19A, which corresponds to pure vertical matches. LARDTW generates an alignment that is closest to vertical matches by imposing an affine model on each matched region that can be reflective of the left or right Hamming window. Recall that LARDTW imposes the affine property on the region surrounding each point. On the hand, GARDTW imposes the affine property on the entire time series, so it is not susceptible to different scalings or offsets in different sections of a time series.

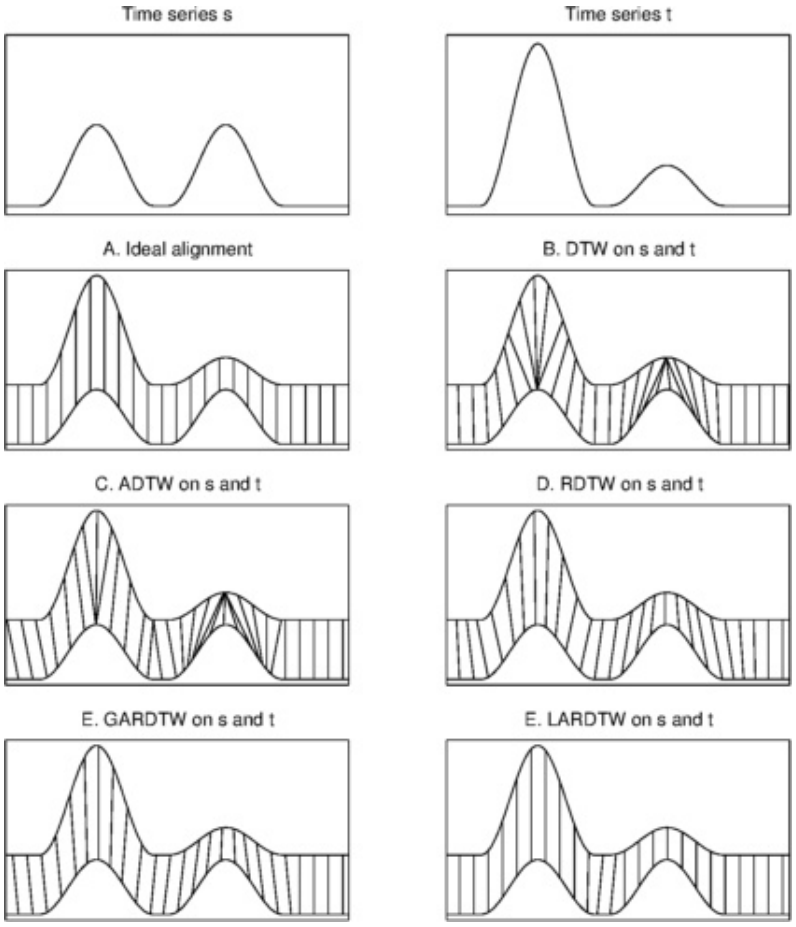

Figure 2.19: DTW, ADTW, RDTW, GARDTW and LARDTW alignments of time series with two components where each component can be scaled differently.

The affine model imposed by $D_L(s, t, p, w_h)_{\text{constrained}}$ may not be robust. For example, consider one region filled with 0's and another region that contains prominent non-zero activities. Looking at Equation 2.11, the minimizing $(c^*_{a,b}, e^*_{a,b})$ is $(0, 0)$, and these parameters yield a value of 0 for d_l. It seems highly unintuitive to be able to match prominent activities to nothing with zero cost. In general, unreasonable values for $(c^*_{a,b}, e^*_{a,b})$ can lead to pathological alignments. These pathological alignments can be avoided by constraining $(c^*_{a,b}, e^*_{a,b})$ to fall under a reasonable range of values such that $c_{\min} \leq c^*_{a,b} \leq c_{\max}$ and

$e_{\min} \leq e^*_{a,b} \leq e_{\max}$. Ideally, d_l should be modified in the following manner:

$$d_l(s_a, t_b, w_h) = \frac{1}{w_{a,b}} \min_{\substack{c_{a,b}, e_{a,b} \\ c_{\min} \leq c_{a,b} \leq c_{\max} \\ e_{\min} \leq e_{a,b} \leq e_{\max}}} \sum_{\substack{w=-w_h \\ w:1 \leq a+w \leq n \\ w:1 \leq b+w \leq m}}^{w_h} d(s_{a+w}, c_{a,b} t_{b+w} + e_{a,b}) \tag{2.15}$$

However, solving for the constrained parameters $(c^*_{a,b}, e^*_{a,b})$ in Equation 2.15 requires active set methods, which generally have higher time complexities by iteratively improving the solution [22]. In this thesis, a much simpler approach is taken. If $c^*_{a,b}$ from Equation 2.12 is less than $c_{\min}$, $c_{\min}$ is assigned to $c^*_{a,b}$. If $c^*_{a,b}$ from Equation 2.12 is larger than $c_{\max}$, $c_{\max}$ is assigned to $c^*_{a,b}$.

Figure 2.20 illustrates the same pair of time series exemplified in Figure 2.19 with the addition of noise, and they are aligned using LARDTW and the constrained version of LARDTW. We observe pathological alignments in Figure 2.20A when LARDTW is unconstrained. The addition of noise means that the two corresponding Hamming windows cannot match perfectly in terms of d_l, which introduces space for matching two distinctly different regions using implausible values of $(c^*_{a,b}, e^*_{a,b})$. On the other hand, when $(c^*_{a,b}, e^*_{a,b})$ is constrained to avoid implausible values, the associated alignment is much more reasonable as shown in Figure 2.20B. In this thesis, $c_{\min} = 0.2$ and $c_{\max} = 5$ unless mentioned otherwise.

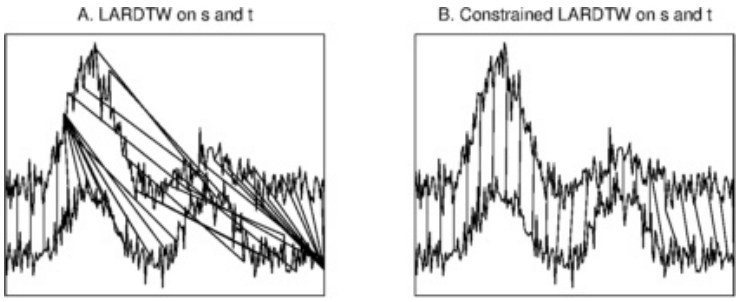

Figure 2.20: LARDTW and its constrained version on noisy time series with two components where each component can be scaled differently.

In the previous examples provided to illustrate LARDTW, components do not vary in width. Figure 2.8 illustrates an example identical to the one from Figure 2.19 with varying component widths. Again, we would like to match the Hamming windows together and the right Hamming windows together, where each Hamming window can have a different

width, scaling and offset. For this more complicated example, LARDTW generate a closer alignment to the ideal one when compared to other methods, since LARDTW allows different regions to have different scalings and offsets. Furthermore, while the region width w_r stays fixed for LARDTW, LARDTW can still offer good alignments for components that can vary in width.

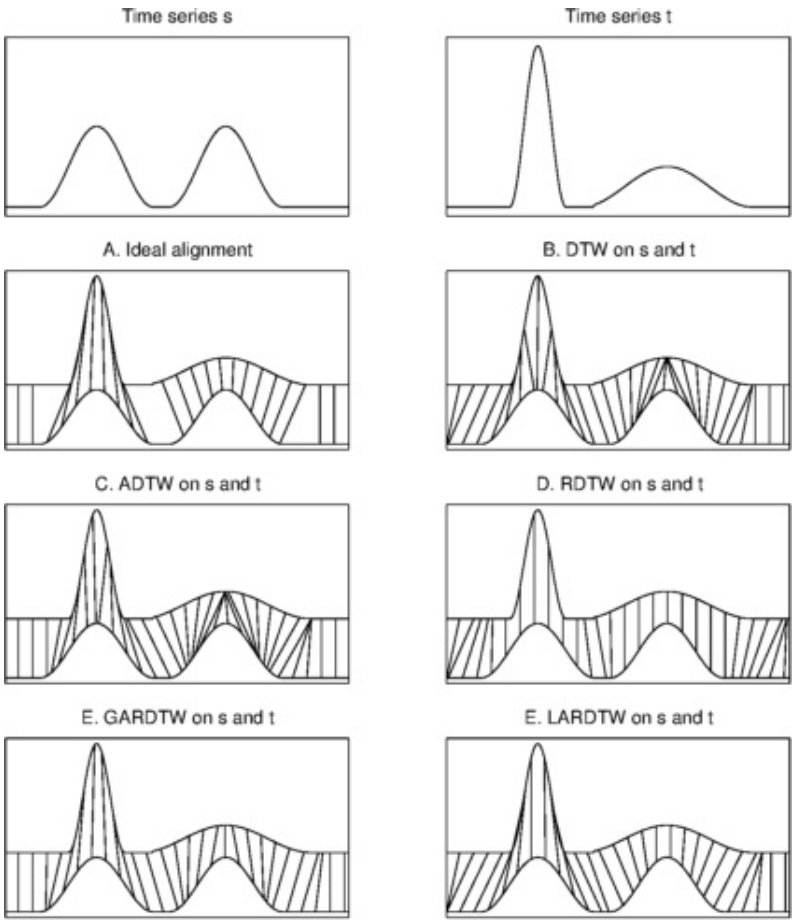

Figure 2.21: DTW, ADTW, RDTW, GARDTW and LARDTW alignments of time series with two components where each component can be scaled differently with varying widths.

Now, let us examine the time and space complexities for LARDTW. For simplicity, let us again assume that $n = m$ and recall that banded DTW is $O(w_b n)$ in time and space. For every region surrounding each point, d_l and $(c^*_{a,b}, e^*_{a,b})$ need to be computed, which are $O(w_r)$ in time and $O(1)$ in space. Hence, it would seem that the time and space

complexities for LARDTW are $O(w_r w_b n)$ and $O(w_b n)$ respectively. However, similar to RDTW, we can update $d_l(s_a, t_b, w_h)$ in constant time using previously computed values if $a > 1$ and $b > 1$. Looking at Equation 2.11, we start by showing that the minimizing $(c^*_{a,b}, e^*_{a,b})$ can be updated from $(c^*_{a-1,b-1}, e^*_{a-1,b-1})$ in constant time. Let

$$\rho_{a,b} = \sum_{\substack{w=-w_h \\ w:1\le a+w\le n \\ w:1\le b+w\le m}}^{w_h} s_{a+w} t_{b+w}$$

$$\gamma_{a,b} = \sum_{\substack{w=-w_h \\ w:1\le a+w\le n \\ w:1\le b+w\le m}}^{w_h} t^2_{b+w}$$

$$\tau_{a,b} = \sum_{\substack{w=-w_h \\ w:1\le a+w\le n \\ w:1\le b+w\le m}}^{w_h} t_{b+w}$$

$$\phi_{a,b} = \sum_{\substack{w=-w_h \\ w:1\le a+w\le n \\ w:1\le b+w\le m}}^{w_h} s_{a+w}$$

$$\eta_{a,b} = \sum_{\substack{w=-w_h \\ w:1\le a+w\le n \\ w:1\le b+w\le m}}^{w_h} s^2_{a+w}$$

Also, set $s_a = 0$ when $a < 1$ or $a > n$, and set $t_b = 0$ when $b < 1$ or $b > m$. Then, the values required for computing $(c^*_{a,b}, e^*_{a,b})$ can be updated in $O(1)$ time from the values required for computing $(c^*_{a-1,b-1}, e^*_{a-1,b-1})$ as follows:

$$\rho_{a,b} = \rho_{a-1,b-1} + s_{a+w_h} t_{b+w_h} - s_{a-w_h-1} t_{b-w_h-1}$$

$$\gamma_{a,b} = \gamma_{a-1,b-1} + t^2_{b+w_h} - t^2_{b-w_h-1}$$

$$\tau_{a,b} = \tau_{a-1,b-1} + t_{b+w_h} - t_{b-w_h-1}$$

$$\phi_{a,b} = \phi_{a-1,b-1} + s_{a+w_h} - s_{a-w_h-1}$$

$$\eta_{a,b} = \eta_{a-1,b-1} + s^2_{a+w_h} - s^2_{a-w_h-1}$$

$$w_{a,b} = \begin{cases} w_{a-1,b-1} + 1 & \text{if } a - w_h - 1 < 1 \text{ or } b - w_h - 1 < 1 \\ w_{a-1,b-1} - 1 & \text{if } a + w_h > n \text{ or } b + w_h > n \\ w_{a-1,b-1} & \text{otherwise} \end{cases}$$

Next, $d_l(s_a, t_b, w_h)$ can be expanded in the following manner:

$$\begin{aligned} d_l(s_a, t_b, w_h) &= \frac{1}{w_{a,b}} \sum_{\substack{w=-w_h \\ w:1\le a+w\le n \\ w:1\le b+w\le m}}^{w_h} d(s_{a+w}, c^*_{a,b} t_{b+w} + e^*_{a,b}) \\ &= \frac{1}{w_{a,b}} \sum_{\substack{w=-w_h \\ w:1\le a+w\le n \\ w:1\le b+w\le m}}^{w_h} (s_{a+w} - (c^*_{a,b} t_{b+w} + e^*_{a,b}))^2 \\ &= \frac{1}{w_{a,b}} \sum_{\substack{w=-w_h \\ w:1\le a+w\le n \\ w:1\le b+w\le m}}^{w_h} (s^2_{a+w} - 2s_{a+w}(c^*_{a,b} t_{b+w} + e^*_{a,b}) + (c^*_{a,b} t_{b+w} + e^*_{a,b})^2) \\ &= \frac{1}{w_{a,b}} [\sum_{\substack{w=-w_h \\ w:1\le a+w\le n \\ w:1\le b+w\le m}}^{w_h} s^2_{a+w} - 2c^*_{a,b} \sum_{\substack{w=-w_h \\ w:1\le a+w\le n \\ w:1\le b+w\le m}}^{w_h} s_{a+w} t_{b+w} - 2e^*_{a,b} \sum_{\substack{w=-w_h \\ w:1\le a+w\le n \\ w:1\le b+w\le m}}^{w_h} s_{a+w} \\ &+ (c^*_{a,b})^2 \sum_{\substack{w=-w_h \\ w:1\le a+w\le n \\ w:1\le b+w\le m}}^{w_h} t^2_{b+w} + 2c^*_{a,b} e^*_{a,b} \sum_{\substack{w=-w_h \\ w:1\le a+w\le n \\ w:1\le b+w\le m}}^{w_h} t_{b+w} + w_{a,b}(e^*_{a,b})^2] \\ &= \frac{1}{w_{a,b}} [\eta_{a,b} - 2c^*_{a,b}\rho_{a,b} - 2e^*_{a,b}\phi_{a,b} + (c^*_{a,b})^2 \gamma_{a,b} + 2c^*_{a,b} e^*_{a,b} \tau_{a,b} + w_{a,b}(e^*_{a,b})^2] \end{aligned}$$

where $(w_{a,b}, c^*_{a,b}, e^*_{a,b}, \gamma_{a,b}, \rho_{a,b}, \tau_{a,b}, \phi_{a,b}, \eta_{a,b})$ can all be updated from their $(a-1, b-1)$ counterparts in $O(1)$ time. Hence, $d_l(s_a, t_b, w_h)$ can be updated in $O(1)$ time when $a \neq 1$ and $b \neq 1$. When considering the Sakoe-Chiba band, there are w_b distinct (a, b) pairs for which $a = 1$ or $b = 1$, and updating each associated d_l element in the LARDTW table requires $O(w_r)$ time. The remaining $w_b n - w_b$ elements can be updated in $O(1)$ time. As a result, construction of the LARDTW table requires a time complexity of

$$O((w_b)(w_r) + (w_b n - w_b)(1)) = O(w_b(w_r + n)) = O(w_b n)$$

because $w_r \le n$. Thus, LARDTW offers the same time and space complexities of $O(w_b n)$ as DTW.

Chapter 3

Evaluation of Pairwise Alignment Methods

3.1 Alignment Evaluation

The proposed methods (ADTW, RDTW, GARDTW and LARDTW) have been demonstrated to generate better alignments for specific examples in the previous sections. Toy examples tend to be unconvincing, and the purpose of this section is to offer more comprehensive evaluations of the alignments generated by DTW and its proposed variants. DTW and its variants that have been discussed thus far are methods for global alignment. They output an alignment path p between two time series that show how all points in one time series are mapped to points in another time series. Component-based alignment, on the other hand, shows how subsets of points in one time series are mapped to subsets of points in another time series, where each subset is conceptualized as a component. While DTW and its variants offer global alignments, only subsets of these alignments pertaining to the different components might prove to be interesting, and this perspective brings about a different evaluation method. Both types of alignment will be further elaborated in the next subsections.

3.1.1 Global Alignment

A direct approach to evaluating an alignment method involves comparing the true alignment with the alignment generated by the alignment method in question. Unfortunately,

the author is unable to find well-established time series datasets that include true alignments between pairs of time series. Therefore, simulation is necessary to realize direct evaluation.

Recall that DTW offers global alignments that account for non-linear temporal variations between two time series. We will start by modeling these non-linear temporal variations. Given a time series $s = (s_1, ..., s_n)$, we want to create a temporal variant of s, denoted $g = (g_1, ..., g_m)$. This can be achieved by defining a warping function $w : \{1, ..., m\} \mapsto \{1, ..., n\}$ and setting $g_i = s_{w(i)}$. Furthermore, w is constrained to be monotonic so that g will not be a distorted version of s that is beyond recognition. The warping function w can also be thought of as a sequence $w = (w(1), ..., w(m))$, and this sequence is modeled as a random process in the following manner:

$$w(i+1) = \begin{cases} w(i)+1 & \text{with probability } P_{\text{match}} \\ w(i)+2 & \text{with probability } P_{\text{delete}} \\ w(i) & \text{with probability } P_{\text{insert}} \end{cases}$$

where $P_{\text{match}} + P_{\text{delete}} + P_{\text{insert}} = 1$, $w(1) = 1$, and this sequence ends when $w(i+1) > n$ with $m = i$. With this definition, m can be considered as a random variable. Note that if $w(i+1) = w(i) + 2$, we are effectively skipping the assignment of $s_{w(i)+1}$ to g, so the corresponding probability is named as P_{delete}. Likewise, if $w(i+1) = w(i)$, we are effectively inserting the same point $s_{w(i)}$ into g, so the associated probability is denoted as P_{insert}.

The true alignment $p^{\text{true}} = \{(a_1^{\text{true}}, b_1^{\text{true}}), ..., (a_m^{\text{true}}, b_m^{\text{true}})\}$ between t and g can be constructed with $(a_j^{\text{true}}, b_j^{\text{true}}) = (w(j), j)$. Note that the model thus far generates a distorted time series g of length m that does not necessarily have the same length as t. For certain types of time series such as recorded temperature across 365 days, it makes more sense to simulate a distorted version h that has the same length as t. h can be obtained from g by interpolating g to be a time series of length n. Similarly, the true alignment for t and h can be obtained by interpolating the sequence w to be of length n and rounding each resultant real value to an integer. For the remainder of this thesis, $P_{\text{match}} = 0.6$ and $P_{\text{delete}} = P_{\text{insert}} = 0.2$ unless mentioned otherwise. Figure 3.1 illustrates a real-world time series s and its simulated time-distorted version h based on the above model. Scrutinization of the two time series reveals many local differences due to temporal variations.

Additional affine properties can be imposed on top of the simulated temporal variations. Instead of creating a time-distorted version h of a time series s, the simulated time series u is distorted based on $\hat{c}s + \hat{e}$, $(\hat{c}, \hat{e}) \in \mathbb{R}^2$. The scaling $\hat{c}$ is assumed to be uniformly distributed from $\hat{c}_{\min}$ to $\hat{c}_{\max}$, and they are set to 0.25 and 2 respectively in this thesis. The offset $\hat{e}$ is modeled to come from a normal distribution with zero mean and standard

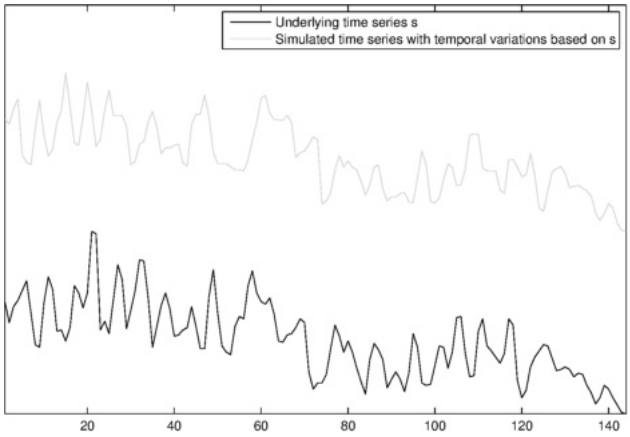

Figure 3.1: Illustration of a real-world time series and its simulated time-distorted version.

deviation σ. In this thesis, σ is set to be proportional to the standard deviation of the examined time series data.

Now that we are able to obtain two time-varying time series under the affine model and the corresponding true alignment, the next question lies in how to evaluate a generated alignment given the true alignment. One simple approach is to count the number of correct matches. However, the weakness behind this approach is that if a generated match between two points is incorrect, this generated match can be very close to the true match or very far away from the true match. Merely counting the number of correct matches cannot discriminate between the matches that are very close to the true matches and the matches that are very far away from the true matches. As an example, if all matches are shifted by one pertaining to one time series with $p^{\text{example}} = \{(a_1^{\text{true}}, b_1^{\text{true}} + 1), ..., (a_{m-1}^{\text{true}}, b_m^{\text{true}} + 1)\}$, the number of correct matches is 0 even though all matches are just off by one. This observation justifies an alignment measure M_{absolute} that accounts for the quality of a match with the absolute difference. This alignment measure is taken from Keogh et al.'s work [15] with a slight modification, and it is defined as follows:

$$M_{\text{absolute}}(p^{\text{true}}, p) = \frac{2}{n(n+1)} \sum_{i=1}^{n} \sum_{b_j^{\text{true}}: a_j^{\text{true}} = i} \min_{b_l : a_l = i} |b_j^{\text{true}} - b_l| \tag{3.1}$$

where $\frac{2}{n(n+1)}$ acts as a normalization factor. Note that better alignments corresponds to ones with lower M_{absolute} values.

Time series from three different real-world datasets are simulated to possess different

temporal variations, scalings and offsets for alignment evaluation. These three time series datasets are taken from a textbook on functional data analysis [12]. The datasets include number of deaths across different ages in 1 year, position of lower lip when saying a certain word, and temperature across 365 days. They will be referred to as the hazard dataset, lip dataset and temperature dataset respectively. All aforementioned datasets are easily subject to temporal variations, scalings and offsets. For evaluation, ten time series are taken from each dataset. For each time series, 10 time-distorted and affine versions are created with the aforementioned model. Each distorted version is aligned with the original time series using a given method and the respective evaluation measure M_{absolute} is obtained. The mean and standard deviation of each evaluation measure are recorded across all datasets. The compared alignment methods are DTW, ADTW, RDTW, GARDTW and LARDTW, and parameters need to be tuned for DTW and its variants. The Sakoe-Chiba bandwidth $w_b = 1 + 2w_q$ is utilized for DTW, ADTW, RDTW, GARDTW and LARDTW, and $\frac{w_q}{n}$ is tuned across $\{0, 0.05, ...0.45, 0.5\}$ based on M_{absolute}. Similarly, the region width $w_r = 1 + 2w_h$ is utilized for RDTW, GARDTW and LARDTW, and $\frac{w_h}{n}$ is tuned across $\{0.05, 0.1, ...0.45, 0.5\}$ based on M_{absolute}. For RDTW, GARDTW and LARDTW, both parameters ($\frac{w_q}{n}$ and $\frac{w_h}{n}$) are tuned across all combinations from 0.05 to 0.5.

Evaluation of DTW, ADTW, RDTW, GARDTW and LARDTW based on real time series with simulated temporal variations and affine properties is shown in Table 3.1, where the mean and standard deviation of M_{absolute} are recorded for each method and dataset. The best results for each dataset is highlighted in bold. Unsurprisingly, ADTW offers the best results by imposing an affine model on the entire time series. GARDTW consistently offers the second best results across the different datasets because it also applies the affine property globally. GARDTW is reasoned to offer worse alignments than ADTW due to its emphasis on regions instead of individual points, but temporal variations are simulated based on each point. The exact same evaluation conducted for Table 3.1 is done in Table 3.2, with the only difference being the addition of noise. Additive white Gaussian noise is injected into each time series prior to applying an alignment method. The standard deviation of the noise is set to be 10 percent of the standard deviation of the time series. Looking at Table 3.2, we observe that GARDTW consistently produces the best alignment scores. While ADTW also models the entire time series to be scaled and offset, GARDTW offers the advantage of regional emphasis based on RDTW. Instead of taking only the point value itself into consideration for matching a point, the region surrounding the point is accounted for, thereby reducing the adverse effects of noise. Based on simulation of temporal variations, scalings and offsets on real time series, we conclude that ADTW offers the best alignment for such models and GARDTW offers the best alignment when noise is added to these models.

Table 3.1: M_{absolute} of different alignment methods on real datasets with simulated temporal variations, scalings and offsets.

Alignment method	Dataset		
	Hazard	Lip	Temperature
DTW	0.004 ± 0.012	0.026 ± 0.030	0.021 ± 0.009
ADTW	$\mathbf{0.000 \pm 0.001}$	$\mathbf{0.009 \pm 0.011}$	$\mathbf{0.003 \pm 0.006}$
RDTW	0.004 ± 0.012	0.021 ± 0.024	0.021 ± 0.009
GARDTW	0.001 ± 0.003	0.011 ± 0.012	0.005 ± 0.001
LARDTW	0.001 ± 0.003	0.017 ± 0.018	0.007 ± 0.003

Table 3.2: M_{absolute} of different alignment methods on real datasets with simulated temporal variations, scalings, offsets and noise.

Alignment method	Dataset with noise		
	Hazard	Lip	Temperature
DTW	0.004 ± 0.013	0.029 ± 0.034	0.020 ± 0.010
ADTW	$\mathbf{0.002 \pm 0.006}$	0.018 ± 0.022	0.020 ± 0.015
RDTW	0.004 ± 0.013	0.022 ± 0.025	0.020 ± 0.010
GARDTW	$\mathbf{0.002 \pm 0.006}$	$\mathbf{0.014 \pm 0.016}$	$\mathbf{0.010 \pm 0.005}$
LARDTW	0.002 ± 0.006	0.018 ± 0.020	0.014 ± 0.006

3.1.2 Component-Based Alignment

In global alignment, the alignment of every point in a time series is considered for evaluation. However, there might exist subsets in a time series where the alignment is not important at all. For example, we do not care much about the alignment of a period populated with only noise. On the other hand, we would like to focus on the alignment of subsections that contain interesting activities. In component-based alignment, we assume that a time series is the superposition of interesting components at different locations, and only the alignment of points associated with these components is considered for evaluation.

Let us start by describing how a component-based time series is generated with varying component locations, widths and amplitudes. Let s and t be two simulated component-based time series of length n that we are aligning. Let n_c be the number of components within a time series. Then,

$$s = \sum_{j=1}^{n_c} a_j^{(s)} \phi(i_j^{(s)}, w_j^{(s)}, z_j)$$

and

$$t = \sum_{j=1}^{n_c} a_j^{(t)} \phi(i_j^{(t)}, w_j^{(t)}, z_j)$$

where $\phi(i, w, z)$ denotes a component of type z centered at location $i \in \mathbb{Z}$ with width $w \in \mathbb{Z}$, and $a_j \in \mathbb{R}$ denotes the scaling factor for the associated component. In this thesis, different component types are associated with different windows commonly used for spectral analysis. In particular, $z \in \{1, 2, 3, 4\}$, where $z = 1$ denotes a Parzen window, $z = 2$ denotes a rectangular window, $z = 3$ denotes a triangular window and $z = 4$ denotes a flat top weighted window. These windows are used for simulating different types of components and they are illustrated in Figure 3.2. Note that parts of a component function ϕ can be truncated if its region exceeds the boundaries of a time series, and each component is forced to have a finite width. The component locations $i_j^{(s)}$ and $i_j^{(t)}$ are each generated from a discrete uniform distribution spanning the entire time series with an imposed constraint. Recall that one of the DTW constraints (monotonicity condition) requires a sequence of matches to not proceed backward in time. As such, alignments similar to the one in Figure 3.3 are impossible using DTW or its proposed variants in this thesis. This motonicity condition is imposed on the simulated components such that the chronological order of components in s will not be changed in t.

To induce variations in component width, $w_j^{(s)}$ and $w_j^{(t)}$ are each generated from a discrete uniform distribution. In addition, $a_j^{(s)}$ and $a_j^{(t)}$ are each generated from a folded

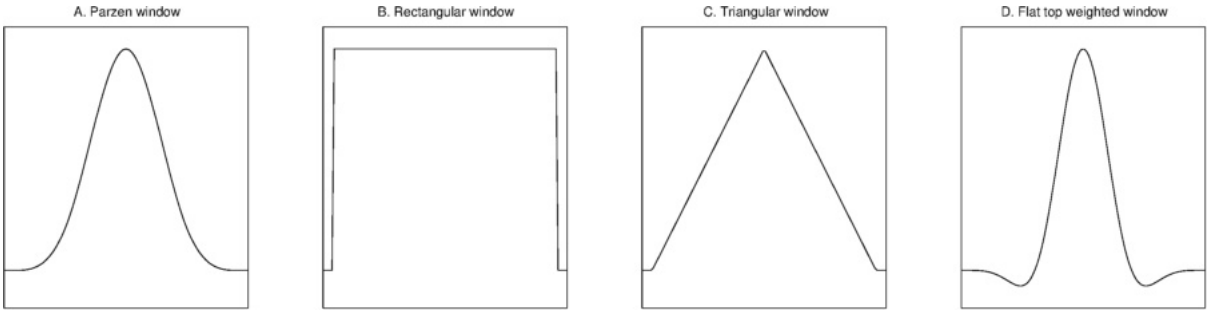

Figure 3.2: Simulated component types.

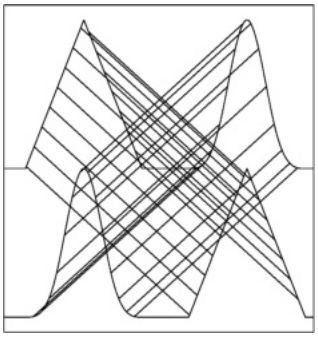

Figure 3.3: Alignment prevented by the DTW monotonicity condition.

normal distribution to simulate variations in component amplitude. Furthermore, the component type z_j is generated from a discrete uniform distribution to enable variation in component types.

To generate a true alignment from simulation, we only consider points associated with components. The true alignment $p^{\text{true}} = \{(a_1^{\text{true}}, b_1^{\text{true}}), ..., (a_n^{\text{true}}, b_n^{\text{true}})\}$ between s and t can be broken down into two parts: non-overlapping and overlapping. For sections that do not have any overlap of components, the alignment is relatively straightforward because we know exactly how to match one component in s to the corresponding component in t during simulation of component amplitude, location and width. However, when component overlap exists, one point can be matched to points coming from different components. One approach is to match the overlapped point to points belonging to the component that made the greatest contribution in amplitude for that overlapped point. However, this approach can omit intuitive matches where the interesting part of one component

with lower amplitude is overlapped with an uninteresting part of another component with higher amplitude. This observation motivates another approach where an overlapped point is determined to come from a component whose center is closest to the overlapped point. Looking at the different component types shown in Figure 3.2, all components are most interesting at the center, thus motivating the aforementioned approach. Two simulated component-based time series with two components and their alignment are shown in Figure 3.4. Again, a line connects two points from two time series if those points are matched.

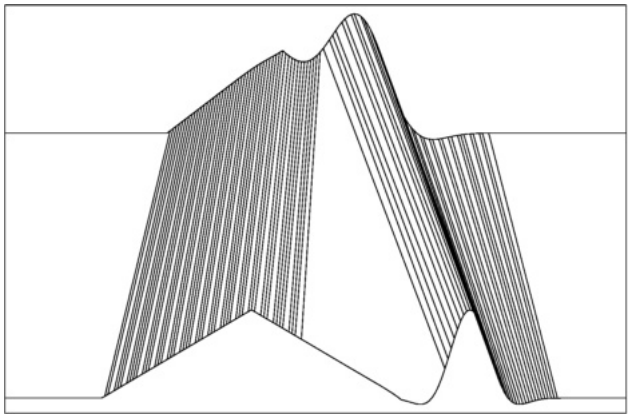

Figure 3.4: Two simulated component-based time series and their true alignment.

The utilized component-based evaluation measure $M_{\text{component}}$ is a slight variation of the absolute difference alignment measure used for global alignment from Equation 3.1:

$$M_{\text{component}}(p^{\text{true}}, p) = \frac{1}{\alpha} \sum_{\substack{i=1 \\ i:i \text{ belongs to a component}}}^{n} \sum_{b_j^{\text{true}}:a_j^{\text{true}}=i} \min_{b_l:a_l=i} |b_j^{\text{true}} - b_l| \tag{3.2}$$

where

$$\alpha = (\sum_{j=1}^{n_c} w_j^{(s)})(\sum_{j=1}^{n_c} w_j^{(t)})$$

acts as a normalization factor.

Evaluation of different alignment methods on component-based time series with different simulation settings is presented in Table 3.3. The number of simulated components per time series n_c is set to 4. The previous discussed simulation of $(w_j^{(s)}, w_j^{(t)}$ each generated from a discrete uniform distribution and $(a_j^{(s)}, a_j^{(t)})$ each generated from a folded normal

Table 3.3: $M_{\text{component}}$ of different alignment methods on component-based time series with different simulation settings with respect to width and amplitude.

Simulation setting	Alignment method				
	DTW	ADTW	RDTW	GARDTW	LARDTW
Fix width and amplitude	0.081	0.097	0.072	0.073	**0.060**
Vary width and fix amplitude	0.078	0.092	0.077	0.079	**0.075**
Fix width and vary amplitude	0.274	0.276	0.202	0.200	**0.159**
Fix width and vary amplitude *	0.070	0.074	0.058	**0.053**	0.059
Vary width and amplitude	0.235	0.242	0.188	0.187	**0.139**
Vary width and amplitude *	0.074	0.077	0.063	0.053	**0.049**

distribution is equivalent to varying both component width and amplitude. The above case of varying both component width and amplitude will be the default setting on which modifications will be made to obtain different settings. To fix both component width and amplitude of two time series to be matched, $w_j^{(s)} = w_j^{(t)} = w$ and $a_j^{(s)} = a_j^{(t)} = a$ for all j, where a and w are each generated from a uniform distribution. To vary component width and fix component amplitude, $a_j^{(s)} = a_j^{(t)} = a$ for all j where a is generated from a uniform distribution. To fix component width and vary component amplitude, $w_j^{(s)} = w_j^{(t)} = w$ for all j, where w is generated from a uniform distribution. The starred version of fixing width and amplitude refers to the case where $w_j^{(s)} = w_j^{(t)} = w$, $a_j^{(s)} = a^{(s)}$ and $a_j^{(t)} = ca^{(s)}$ for all j, where w, $a^{(s)}$ and c are each generated from a uniform distribution. Likewise, for the starred version of varying component width and amplitude, $a_j^{(s)} = a^{(s)}$ and $a_j^{(t)} = ca^{(s)}$ for all j, where $a^{(s)}$ and c are each generated from a uniform distribution.

Looking at Table 3.3, we can observe that LARDTW offers the best result across most simulation settings. On the other hand, DTW and ADTW consistently yields the worse results, because they do not possess any concept of a component. At first glance, it might seem surprising that LARDTW performs better than RDTW when the component width and amplitude are fixed, because LARDTW supposedly accounts for amplitude variation. However, one good approximated way of modeling a region where two components are overlapped is to impose a scaling and offset on the component with more discernible shape. When the component width is varied, LARDTW performs better because it can match parts of a component better than RDTW and GARDTW. Again, RDTW and GARDTW suffer from only emphasizing on a region with a fixed region width. While LARDTW also has a fixed region width, it promotes flexibility of matching regions by imposing the affine model on each region. When the amplitude is varied for each component independently, it is unsurprising that LARDTW always offer the best result because it is the only method

that models different amplitudes for each component. The starred version of varying component amplitude follows the global affine model where components in t are assumed to be scaled in the same manner from s. When the component width is fixed for the global affine model, it is not surprising that GARDTW offers the best result. When the component width is varied under the global affine model, GARDTW becomes the second best method after LARDTW. This observation is again attributed to LARDTW's ability to handle varying component widths. In conclusion, through component-based simulation, RDTW, GARDTW and LARDTW are found to consistently outperform DTW and ADTW because they emphasize on regions. Furthermore, LARDTW outperforms other methods most of the time due to its high flexibility of matching regions under the affine model, thereby focusing more on component shapes.

3.2 Difference Measure Evaluation

The focus of this thesis thus far is on alignment of two time series using the proposed DTW variants. In the previous section, alignments from these methods are evaluated by comparing them with true alignments that are generated through simulation. However, in the ideal scenario, we would want true alignments of two real-world time series, because the models that generate these simulated signals and the associated true alignments tend to be not as comprehensive. The author only explores direct alignment evaluation through simulation because he was unable to find a public dataset that contains true alignments for a pair of real-world time series. One indirect way of evaluating alignments that bypasses simulation of true alignments is to evaluate the difference measure accompanying each alignment method discussed in this thesis. Intuitively, if the generated alignment p is good, the associated difference measure tends to be good at differentiating a time series from another that is very different. Ding et al.'s work provides supporting evidence for the previous statement by showing that the DTW difference measure reliably outperforms a naive difference measure that assumes strictly vertical matches between two time series across a large number of datasets [7]. It is important to note that while better alignments tend to be associated with better difference measures, there exists cases where worse alignments can yield better difference measures.

One popular way of evaluating a dissimilarity measure involves applying it on top of a nearest-neighbor (NN) classifier to obtain the associated classification error rate. In this thesis, the 1-NN classifier is selected to avoid additional parameterization. We will apply the difference measures associated with the proposed DTW variants on a subset of the UCR time series database [14], which offers a large number of datasets from a wide variety

Table 3.4: Subset of UCR time series datasets used for difference measure evaluation.

Dataset	Train cases	Test cases	Length	Classes
Coffee	28	28	286	2
Beef	30	30	470	5
ECG	100	100	96	2
ECGFiveDays	23	861	136	2
TwoLeadECG	23	1139	82	2
Adiac	390	391	176	37
OSULeaf	200	242	427	6
MedicalImages	381	760	99	10
Lightning7	70	73	319	7
SyntheticControl	300	300	60	6

of domains. The ten datasets used for evaluation in this thesis are listed in Table 3.4. Each UCR time series dataset is broken down into a training set and a testing set. Each time series in the training set and testing set has a class associated with it. The goal is to develop a model using the training set to predict the classes of the time series in the testing set. The time series in the Coffee and Beef datasets are spectroscopic data on coffee and beef products [4][1]. The ECG, ECGFiveDays and TwoLeadECG datasets each contain abnormal and normal ECG data [14][23]. The Adiac, OSULeaf and MedicalImages datasets contain one-dimensional time series that are converted from images [14]. For example, for the Adiac dataset, a time series is constructed by recording the curvature along the main contour of an image of a diatom [13]. The Lightning7 dataset consists of power densities from different types of lightning strikes [8]. The SyntheticControl dataset include control charts (graphs of how processes change over time) that reflect patterns such as normal, increasing trend and decreasing trend [24].

DTW, ADTW, RDTW, GARDTW and LARDTW are parameterized by user-selected values (the bandwidth $w_b = 1 + 2w_q$ and region width $w_r = 1 + 2w_h$) that can lead to drastically different results. Recall that n denotes the length of a time series. Similar to the evaluation procedure outlined in the previous section, $\frac{w_q}{n}$ and $\frac{w_h}{n}$ are automatically tuned for each dataset by choosing $\frac{w_q}{n}$ from the set of $\{0, 0.05, ..., 0.5\}$ and $\frac{w_h}{n}$ from the set of $\{0.05, 0.1, ..., 0.5\}$ that yield the lowest error rate. Table 3.5 shows the error rates on 10 UCR time series data sets with respect to the DTW, ADTW, RDTW, GARDTW and LARDTW difference measures based on 1-NN classification. The lowest error rate is bolded for each dataset.

Looking at Table 3.5, we start by observing that ADTW performs significantly better

Table 3.5: Time series classification error rates of different difference measures on 10 UCR datasets

Dataset	DTW	ADTW	RDTW	GARDTW	LARDTW
Coffee	0.179	**0.000**	0.179	**0.000**	**0.000**
Beef	0.467	0.333	0.467	0.333	**0.233**
ECG200	0.110	0.120	0.090	0.090	**0.060**
ECGFiveDays	0.187	0.203	0.008	0.008	**0.000**
TwoLeadECG	0.096	0.098	0.035	0.037	**0.003**
Adiac	0.389	0.389	0.338	0.340	**0.325**
OSULeaf	0.397	0.409	0.306	0.293	**0.169**
MedicalImages	0.250	**0.218**	0.272	0.257	0.305
Lightning7	**0.233**	**0.233**	0.247	0.247	0.260
SyntheticControl	**0.007**	0.157	0.033	0.330	0.090

than DTW for the Coffee and Beef datasets. The Coffee and Beef dataset both contains spectroscopic data on food, where differences in the food container and food samples contribute to variability in the spectral baseline and overall spectral intensity [4][1]. The aforementioned effect fits the affine model well, so ADTW outperforms DTW by a significant margin. RDTW is observed to perform better than DTW on the ECG200, ECGFiveDays, TwoLeadECG, Adiac and OSULeaf datasets. In particular, RDTW performs better than DTW across all ECG-related datasets, because ECG data is component-based, where each component can reflect the P, Q, R, S or T wave. The Adiac and OSULeaf datasets consist of one-dimensional time series extracted from images along their main contours [13], and it is reasonable for the resulting time series to contain discriminative components. Unsurprisingly, RDTW is shown to outperform DTW when the time series is known to contain components that can benefit from regional emphasis. GARDTW is observed to usually possess an error rate close to the better one among the ADTW and RDTW error rates. While there are datasets where GARDTW outperforms DTW, ADTW and RDTW, the gain is small when compared to the lowest error rate among the DTW, ADTW and RDTW error rates. LARDTW is observed to outperform other methods for 7 out of 10 datasets. LARDTW provides lower error rates for all ECG-related datasets, because ECG data consists of components from multiple sources subject to amplitude variation. By applying the affine model on each region and emphasizing on regions, LARDTW is able to outperform the remaining methods for the datasets that fit this model. We can also observe that DTW does not necessarily always provide a worse error rate. For the Lightning7 and SyntheticControl datasets, DTW offers the best result, and this observation is attributed to the

lack of affine modeling and preference for component emphasis within these two datasets. For example, the SyntheticControl dataset contains control charts that should be classified into patterns such as increasing trend, decreasing trend, upward shift and downward shift. Such patterns are applied globally to the time series, so there is no benefit from regional emphasis. Furthermore, ADTW models the upward or downward shift of an entire time series to be inconsequential to alignment when these properties are in fact an important discrimination factor, so ADTW performs far worse than DTW on the SyntheticControl dataset.

Figure 3.5 offers pairwise visual comparisons of experimental error rates from Table 3.5 of ADTW vs DTW, RDTW vs DTW, GARDTW vs DTW and LARDTW vs DTW. The plotted points come from experimental results shown in Table 3.5. The line serves as a reference where the error rates for both compared dissimilarity measures are equal. Points below the line indicate that first difference measure has smaller error rates and points above the line indicate that the later difference measure has higher error rates. From Figure 3.5, we see that GARDTW and LARDTW outperforms DTW on 7 datasets. On the other hand, ADTW outperforms DTW on 3 datasets and RDTW outperforms DTW on 5 datasets. The above observations support the notion that GARDTW and LARDTW are able to combine the advantages of ADTW and RDTW. Furthermore, the improvement from LARDTW tends to be more pronounced when compared to GARDTW, and this observation is attributed to LARDTW's high flexibility by modeling each matched region to be affine as opposed to the entire time series.

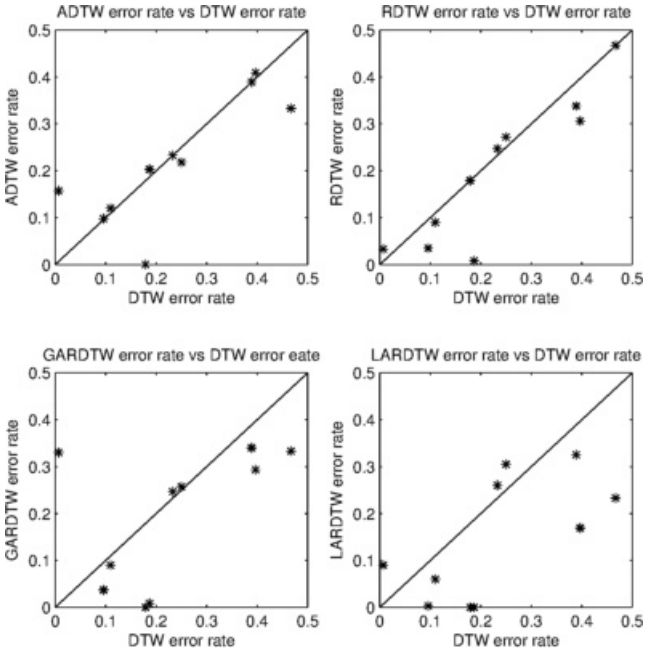

Figure 3.5: Visualization of pairwise dissimilarity measure error rates of proposed methods vs DTW.

Chapter 4

Conclusion and Future Work

In this thesis, two variants of DTW, ADTW and RDTW, are proposed to improve alignments under certain imposed models or preferences. ADTW increments DTW to allow arbitrary scaling and offset in amplitude between two time series which are subject to temporal variations. ADTW was shown to be different from normalization prior to applying DTW because ADTW tries to optimize for the scaling, offset and alignment simultaneously. A more general version of ADTW where subsets of time series are subject to the same scaling and offset was also explored, but this approach is not readily applicable since these subsets are usually not known. ADTW and its more general version are analyzed to be both $O(gw_bn)$ in time and $O(w_bn)$ in space, where g is the number of iterations for the ADTW algorithm to converge, w_b is the imposed Sakoe-Chiba bandwidth and n is the time series length.

RDTW modifies DTW to place more weight on a region of points potentially representative of a component of interest in a time series. RDTW introduces an additional parameter which is the region width w_r, and an appropriate value for this parameter is found to be crucial in generating good alignments. An approach of finding a good region width and alignment simultaneously is proposed, but it did not generate good results. Thus, it is recommended that w_r should be tuned according to the task or criterion at hand. RDTW is analyzed to be $O(w_bn)$ in both time and space, so its complexities are identical to those of DTW.

ADTW and RDTW each has its unique model or preference, where ADTW models time series to be scaled and offset in amplitude when aligned, and RDTW places an emphasis on matching regions. We showed two different ways of combining ADTW and RDTW to realize alignments that incorporate the advantages of both ADTW and RDTW. In

GARDTW, one time series is modeled as the scaled and offset version of another time series when aligning them with regional emphasis. In LARDTW, the region surrounding each point is assumed to be a scaled and offset version of another region surrounding the corresponding matched point. In other words, the affine model is applied globally to the entire time series in LARDTW, whereas the affine model is applied locally to each region in GARDTW. GARDTW is analyzed to be $O(\lambda(w_b + w_r)n)$ in time and $O(w_b n)$ in space, where λ is the number of iterations required for the GARDTW algorithm to converge. LARDTW is analyzed to be $O(w_b n)$ in both time and space.

The proposed DTW variants were evaluated with respect to their generated alignments and difference measures. Alignment-based evaluation is based on simulated time series and their true pairwise alignments, because the author cannot find public well-established datasets that contain real time series and associated true pairwise alignments. Time series were simulated in two different ways to represent different underlying models. In global alignment, artificial temporal variations, scalings and offsets are imposed on a real time series to generate another time series and their true alignment. ADTW was found to outperform all other methods for this global alignment model. The aforementioned result is expected because ADTW handles temporal variations, scalings and offsets between time series simultaneously. On the other hand, when a non-trivial amount of noise is added to each time series, GARDTW was found to outperform all other methods. This result is again expected because GARDTW applies the affine model on the entire time series and its emphasis on regions has an averaging effect in terms of alignment. In component-based alignment, a time series is assumed to be a superposition of different components and its alignment to another time series is created accordingly. Different simulation settings were applied where the component width and amplitude can vary for a pair of matched time series. For this component-based simulation, RDTW always outperformed DTW, although LARDTW achieved the best results. It is not surprising that RDTW was found to be better than DTW in component-based simulation, because RDTW's regional emphasis focuses on matching components and not individual points. LARDTW's excellent performance in component-based simulation is attributed to its high flexibility of matching regions by imposing the affine model on each region, which effectively handles component overlap. The different measures associated with each proposed method were evaluated on 10 real datasets by applying the difference measures on a 1-NN classifier for time series classification. Similar to the simulated cases, a proposed method was found to outperform DTW if the evaluated dataset falls under the model or preference of the proposed method. In particular, when components are involved in a time series, LARDTW consistently outperformed RDTW and GARDTW. This result is attributed to the nature of component-based datasets, where components can vary in width and amplitude.

To conclude, ADTW, RDTW, GARDTW and LARDTW are alignment methods whose models include the affine assumption and a preference for regional emphasis. If they are applied to problems whose underlying models are similar to the models behind the methods, the proposed DTW variants are expected to provide significant performance gains as demonstrated in this thesis with respect to certain simulated models and real datasets.

For future work, a variant of RDTW where the region width can be variable is recommended to be explored. While LARDTW was shown to outperform DTW and other proposed methods when the component width varies, it does not model changes in region width directly, and a direct approach may better solve this problem. Furthermore, the runtime of GARDTW is high, and there might be room for improving its time complexity to make this method more practical.

Appendix A

Derivation of GARDTW Scaling and Offset Equations

We want to find a scaling factor $c'_p \in \mathbb{R}$ and offset factor $e'_p \in \mathbb{R}$ that minimize

$$D_G(s,t,p,c,e,w_h) = \sum_{k=1}^{|p|} \frac{1}{w_{a_k,b_k}} \sum_{\substack{w=-w_h \\ w:1\leq a_k+w\leq n \\ w:1\leq b_k+w\leq m}}^{w_h} d(s_{a_k+w}, ct_{b_k+w}+e)$$

when s, t, p and w_h are given, and d is the squared difference.

We start by proving that $D_G(s,t,p,c,e,w_h s)$ is convex with respect to c and e when s, t, p and w_h are fixed. Let us try to express $D_G(s,t,p,c,e,w_h)$ with vectors by defining

$$s_p^{(g)} = \begin{pmatrix} \frac{1}{w_{a_1,b_1}} s_{a_1} \\ \frac{1}{w_{a_1,b_1}} s_{a_1+1} \\ \vdots \\ \frac{1}{w_{a_1,b_1}} s_{a_1+w_h} \\ \frac{1}{w_{a_2,b_2}} s_{a_2} \\ \frac{1}{w_{a_2,b_2}} s_{a_2+1} \\ \vdots \\ \frac{1}{w_{a_2,b_2}} s_{a_2+w_h} \\ \vdots \\ \vdots \\ \frac{1}{w_{a_{|p|},b_{|p|}}} s_{a_{|p|}} \end{pmatrix}, t_p^{(g)} = \begin{pmatrix} \frac{1}{w_{a_1,b_1}} t_{b_1} \\ \frac{1}{w_{a_1,b_1}} t_{b_1+1} \\ \vdots \\ \frac{1}{w_{a_1,b_1}} t_{b_1+w_h} \\ \frac{1}{w_{a_2,b_2}} t_{b_2} \\ \frac{1}{w_{a_2,b_2}} t_{b_2+1} \\ \vdots \\ \frac{1}{w_{a_2,b_2}} t_{b_2+w_h} \\ \vdots \\ \vdots \\ \frac{1}{w_{a_{|p|},b_{|p|}}} t_{b_{|p|}} \end{pmatrix}$$

Let $\beta = [c, e]^T$ and let $\vec{1} = [1, 1, ...1]^T$ be a vector with the same length as $t^{(g)}$. Also, let $y = s_p^{(g)}$ and $X = [t_p^{(g)}, \vec{1}]$. Then, $D_G(s, t, p, c, e, w_h)$ can be rewritten in the form of linear least squares:

$$D_G(s, t, p, c, e, w_h) = (y - X\beta)^T (y - X\beta)$$

This function is twice differentiable with respect to β and its Hessian with respect to β is $X^T X$, which is a positive semi-definite matrix. If the Hessian of a function is positive semi-definite, the function is convex. Thus, $D_G(s, t, p, c, e, w_h)$ is convex with respect to (c, e) when (s, t, p, w_h) are given.

Since we now know that $D_G(s, t, p, c, e, w_h)$ has only one global minimum with respect to (c, e), finding any local optimum is equivalent to finding the global minimum. Finding a local optimum of $D_A(s, t, p, c, e)$ can be achieved by taking derivatives of $D_G(s, t, p, c, e, w_h)$ with respect to c and e, setting the derivatives to 0, and solving for values of c and e. Taking

the derivative of $D_R(s,t,p,c,e,w_h)$ with respect to c, we get

$$\begin{aligned}\frac{\partial}{\partial c}D_G(s,t,p,c,e,w_h) &= \frac{\partial}{\partial c}\sum_{k=1}^{|p|}\frac{1}{w_{a_k,b_k}}\sum_{\substack{w=-w_h\\ w:1\le a_k+w\le n\\ w:1\le b_k+w\le m}}^{w_h} d(s_{a_k+w}, ct_{b_k+w}+e)\\ &= \sum_{k=1}^{|p|}\frac{1}{w_{a_k,b_k}}\sum_{\substack{w=-w_h\\ w:1\le a_k+w\le n\\ w:1\le b_k+w\le m}}^{w_h}\frac{\partial}{\partial c}(s_{a_k+w}-ct_{b_k+w}-e)^2\\ &= -2\sum_{k=1}^{|p|}\frac{1}{w_{a_k,b_k}}\sum_{\substack{w=-w_h\\ w:1\le a_k+w\le n\\ w:1\le b_k+w\le m}}^{w_h}(s_{a_k+w}-ct_{b_k+w}-e)t_{b_k+w}\end{aligned}$$

Taking the derivative of $D_R(s,t,p,c,e,w_h)$ with respect to e, we obtain

$$\begin{aligned}\frac{\partial}{\partial e}D_G(s,t,p,c,e,w_h) &= \sum_{k=1}^{|p|}\frac{1}{w_{a_k,b_k}}\sum_{\substack{w=-w_h\\ w:1\le a_k+w\le n\\ w:1\le b_k+w\le m}}^{w_h}\frac{\partial}{\partial e}(s_{a_k+w}-ct_{b_k+w}-e)^2\\ &= -2\sum_{k=1}^{|p|}\frac{1}{w_{a_k,b_k}}\sum_{\substack{w=-w_h\\ w:1\le a_k+w\le n\\ w:1\le b_k+w\le m}}^{w_h}(s_{a_k+w}-ct_{b_k+w}-e)\end{aligned}$$

Setting $\frac{\partial}{\partial e}D_G(s,t,p,c,e'_p,w_h)=0$ and solving for e'_p yields the minimizing e:

$$-2\sum_{k=1}^{|p|}\frac{1}{w_{a_k,b_k}}\sum_{\substack{w=-w_h\\ w:1\le a_k+w\le n\\ w:1\le b_k+w\le m}}^{w_h}(s_{a_k+w}-ct_{b_k+w}-e'_p)=0$$

$$-e'_p[\sum_{k=1}^{|p|}\frac{1}{w_{a_k,b_k}}\sum_{\substack{w=-w_h\\ w:1\le a_k+w\le n\\ w:1\le b_k+w\le m}}^{w_h}1]+[\sum_{k=1}^{|p|}\frac{1}{w_{a_k,b_k}}\sum_{\substack{w=-w_h\\ w:1\le a_k+w\le n\\ w:1\le b_k+w\le m}}^{w_h}(s_{a_k+w}-ct_{b_k+w})]=0$$

$$e'_p = \frac{\sum_{k=1}^{|p|}\frac{1}{w_{a_k,b_k}}\sum_{\substack{w=-w_h\\ w:1\le a_k+w\le n\\ w:1\le b_k+w\le m}}^{w_h}(s_{a_k+w}-ct_{b_k+w})}{\sum_{k=1}^{|p|}\frac{1}{w_{a_k,b_k}}\sum_{\substack{w=-w_h\\ w:1\le a_k+w\le n\\ w:1\le b_k+w\le m}}^{w_h}1} \tag{A.1}$$

Likewise, setting $\frac{\partial}{\partial c} D_G(s,t,p,c'_p,e,w_h) = 0$, plugging in Equation A.1 for e'_p and solving for c'_p yields the minimizing c:

$$-2\sum_{k=1}^{|p|}\frac{1}{w_{a_k,b_k}}\sum_{\substack{w=-w_h\\ w:1\le a_k+w\le n\\ w:1\le b_k+w\le m}}^{w_h}(s_{a_k+w}-c'_p t_{b_k+w}-e'_p)t_{b_k+w}=0$$

$$\sum_{k=1}^{|p|}\frac{1}{w_{a_k,b_k}}\sum_{\substack{w=-w_h\\ w:1\le a_k+w\le n\\ w:1\le b_k+w\le m}}^{w_h}(s_{a_k+w}-c'_p t_{b_k+w}-\frac{\sum_{l=1}^{|p|}\frac{1}{w_{a_l,b_l}}\sum_{\substack{v=-w_h\\ v:1\le a_l+v\le n\\ v:1\le b_l+v\le m}}^{w_h}(s_{a_l+v}-c'_p t_{b_l+v})}{\sum_{l=1}^{|p|}\frac{1}{w_{a_l,b_l}}\sum_{\substack{v=-w_h\\ v:1\le a_l+v\le n\\ v:1\le b_l+v\le m}}^{w_h}1})t_{b_k+w}=0$$

$$\sum_{k=1}^{|p|}\frac{1}{w_{a_k,b_k}}\sum_{\substack{w=-w_h\\ w:1\le a_k+w\le n\\ w:1\le b_k+w\le m}}^{w_h}s_{a_k+w}t_{b_k+w}-c'_p\sum_{k=1}^{|p|}\frac{1}{w_{a_k,b_k}}\sum_{\substack{w=-w_h\\ w:1\le a_k+w\le n\\ w:1\le b_k+w\le m}}^{w_h}t^2_{b_k+w}-$$
$$\sum_{k=1}^{|p|}\frac{1}{w_{a_k,b_k}}\sum_{\substack{w=-w_h\\ w:1\le a_k+w\le n\\ w:1\le b_k+w\le m}}^{w_h}\frac{\sum_{l=1}^{|p|}\frac{1}{w_{a_l,b_l}}\sum_{\substack{v=-w_h\\ v:1\le a_l+v\le n\\ v:1\le b_l+v\le m}}^{w_h}(s_{a_l+v}-c'_p t_{b_l+v})}{\sum_{l=1}^{|p|}\frac{1}{w_{a_l,b_l}}\sum_{\substack{v=-w_h\\ v:1\le a_l+v\le n\\ v:1\le b_l+v\le m}}^{w_h}1}t_{b_k+w}=0$$

$$\sum_{k=1}^{|p|}\frac{1}{w_{a_k,b_k}}\sum_{\substack{w=-w_h\\ w:1\le a_k+w\le n\\ w:1\le b_k+w\le m}}^{w_h}s_{a_k+w}t_{b_k+w}-c'_p\sum_{k=1}^{|p|}\frac{1}{w_{a_k,b_k}}\sum_{\substack{w=-w_h\\ w:1\le a_k+w\le n\\ w:1\le b_k+w\le m}}^{w_h}t^2_{b_k+w}-$$
$$(\sum_{k=1}^{|p|}\frac{1}{w_{a_k,b_k}}\sum_{\substack{w=-w_h\\ w:1\le a_k+w\le n\\ w:1\le b_k+w\le m}}^{w_h}t_{b_k+w})(\frac{\sum_{l=1}^{|p|}\frac{1}{w_{a_l,b_l}}\sum_{\substack{v=-w_h\\ v:1\le a_l+v\le n\\ v:1\le b_l+v\le m}}^{w_h}s_{a_l+v}}{\sum_{l=1}^{|p|}\frac{1}{w_{a_l,b_l}}\sum_{\substack{v=-w_h\\ v:1\le a_l+v\le n\\ v:1\le b_l+v\le m}}^{w_h}1})+$$
$$c'_p(\sum_{k=1}^{|p|}\frac{1}{w_{a_k,b_k}}\sum_{\substack{w=-w_h\\ w:1\le a_k+w\le n\\ w:1\le b_k+w\le m}}^{w_h}t_{b_k+w})(\frac{\sum_{l=1}^{|p|}\frac{1}{w_{a_l,b_l}}\sum_{\substack{v=-w_h\\ v:1\le a_l+v\le n\\ v:1\le b_l+v\le m}}^{w_h}t_{b_l+v}}{\sum_{l=1}^{|p|}\frac{1}{w_{a_l,b_l}}\sum_{\substack{v=-w_h\\ v:1\le a_l+v\le n\\ v:1\le b_l+v\le m}}^{w_h}1})=0$$

Let

$$\rho = \sum_{k=1}^{|p|} \frac{1}{w_{a_k,b_k}} \sum_{\substack{w=-w_h \\ w:1\le a_k+w\le n \\ w:1\le b_k+w\le m}}^{w_h} s_{a_k+w} t_{b_k+w}$$

$$\gamma = \sum_{k=1}^{|p|} \frac{1}{w_{a_k,b_k}} \sum_{\substack{w=-w_h \\ w:1\le a_k+w\le n \\ w:1\le b_k+w\le m}}^{w_h} t_{b_k+w}^2$$

$$\tau = \sum_{k=1}^{|p|} \frac{1}{w_{a_k,b_k}} \sum_{\substack{w=-w_h \\ w:1\le a_k+w\le n \\ w:1\le b_k+w\le m}}^{w_h} t_{b_k+w}$$

$$\phi = \sum_{k=1}^{|p|} \frac{1}{w_{a_k,b_k}} \sum_{\substack{w=-w_h \\ w:1\le a_k+w\le n \\ w:1\le b_k+w\le m}}^{w_h} s_{a_k+w}$$

Then,

$$c'_p = \frac{\rho - \frac{1}{|p|}\tau\phi}{\gamma - \frac{1}{|p|}\tau^2} \tag{A.2}$$

and e'_p can be re-expressed as

$$e'_p = \frac{1}{|p|}(\phi - c'_p\tau) \tag{A.3}$$

which are identical to Equation 2.9 and 2.10.

References

[1] Osama Al-Jowder, EK Kemsley, and Reginald H Wilson. Detection of adulteration in cooked meat products by mid-infrared spectroscopy. *Journal of agricultural and food chemistry*, 50(6):1325–1329, 2002.

[2] Claus Bahlmann and Hans Burkhardt. The writer independent online handwriting recognition system frog on hand and cluster generative statistical dynamic time warping. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(3):299–310, 2004.

[3] Ziv Bar-Joseph, Anthony Gitter, and Itamar Simon. Studying and modelling dynamic biological processes using time-series gene expression data. *Nature Reviews Genetics*, 13(8):552–564, 2012.

[4] Romain Briandet, E Katherine Kemsley, and Reginald H Wilson. Discrimination of arabica and robusta in instant coffee by fourier transform infrared spectroscopy and chemometrics. *Journal of agricultural and food chemistry*, 44(1):170–174, 1996.

[5] Gilles Celeux and Gérard Govaert. A classification em algorithm for clustering and two stochastic versions. *Computational statistics & Data analysis*, 14(3):315–332, 1992.

[6] Andrea Corradini. Dynamic time warping for off-line recognition of a small gesture vocabulary. In *Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, 2001. Proceedings. IEEE ICCV Workshop on*, pages 82–89. IEEE, 2001.

[7] Hui Ding, Goce Trajcevski, Peter Scheuermann, Xiaoyue Wang, and Eamonn Keogh. Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment*, 1(2):1542–1552, 2008.

[8] Damian R Eads, Daniel Hill, Sean Davis, Simon J Perkins, Junshui Ma, Reid B Porter, and James P Theiler. Genetic algorithms and support vector machines for time series classification. In *International Symposium on Optical Science and Technology*, pages 74–85. International Society for Optics and Photonics, 2002.

[9] C Emmanouilidis, J MacIntyre, and C Cox. Neurofuzzy computing aided machine fault diagnosis. In *Proc. of JCIS'98, The Fourth Joint Conference on Information Sciences. Research Triangle Park, North Carolina, USA*, 1998.

[10] Francesco Feminella and Marisa Storini. Large-scale dynamical phenomena during solar activity cycles. *Astronomy and Astrophysics*, 322:311–319, 1997.

[11] Ada Wai-Chee Fu, Eamonn Keogh, Leo Yung Lau, Chotirat Ann Ratanamahatana, and Raymond Chi-Wing Wong. Scaling and time warping in time series querying. *The VLDB JournalThe International Journal on Very Large Data Bases*, 17(4):899–921, 2008.

[12] Spencer Graves, Giles Hooker, and J Ramsay. Functional data analysis with r and matlab, 2009.

[13] Andrei C Jalba, Michael HF Wilkinson, Jos BTM Roerdink, Micha M Bayer, and Stephen Juggins. Automatic diatom identification using contour analysis by morphological curvature scale spaces. *Machine Vision and Applications*, 16(4):217–228, 2005.

[14] Eamonn Keogh, Xiaopeng Xi, Li Wei, and Chotirat Ann Ratanamahatana. The ucr time series classification/clustering homepage. *URL= http://www. cs. ucr. edu/˜ eamonn/time_series_data*, 2006.

[15] Eamonn J Keogh and Michael J Pazzani. Derivative dynamic time warping. In *SDM*, volume 1, pages 5–7. SIAM, 2001.

[16] Young Shin Kim, Svetlozar T Rachev, Michele Leonardo Bianchi, Ivan Mitov, and Frank J Fabozzi. Time series analysis for financial market meltdowns. *Journal of Banking & Finance*, 35(8):1879–1891, 2011.

[17] Longin Jan Latecki, Vasilis Megalooikonomou, Qiang Wang, Rolf Lakaemper, Chotirat Ann Ratanamahatana, and Eamonn Keogh. Elastic partial matching of time series. In *Knowledge Discovery in Databases: PKDD 2005*, pages 577–584. Springer, 2005.

[18] A Morales-Esteban, Francisco Martínez-Álvarez, A Troncoso, JL Justo, and Cristina Rubio-Escudero. Pattern recognition to forecast seismic time series. *Expert Systems with Applications*, 37(12):8333–8342, 2010.

[19] Meinard Müller. Dtw-based motion comparison and retrieval. *Information Retrieval for Music and Motion*, pages 211–226, 2007.

[20] Cory Myers, Lawrence Rabiner, and Aaron E Rosenberg. Performance tradeoffs in dynamic time warping algorithms for isolated word recognition. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 28(6):623–635, 1980.

[21] Niels-Peter Vest Nielsen, Jens Michael Carstensen, and Jørn Smedsgaard. Aligning of single and multiple wavelength chromatographic profiles for chemometric data analysis using correlation optimised warping. *Journal of Chromatography A*, 805(1):17–35, 1998.

[22] Jorge Nocedal and Stephen J Wright. Numerical optimization 2nd. 2006.

[23] Robert T Olszewski. Generalized feature extraction for structural pattern recognition in time-series data. Technical report, DTIC Document, 2001.

[24] DT Pham and AB Chan. Control chart pattern recognition using a new type of self-organizing neural network. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 212(2):115–127, 1998.

[25] Yu Qiao and Makoto Yasuhara. Affine invariant dynamic time warping and its application to online rotated handwriting recognition. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 2, pages 905–908. IEEE, 2006.

[26] Chotirat Ann Ratanamahatana and Eamonn Keogh. Everything you know about dynamic time warping is wrong. In *Third Workshop on Mining Temporal and Sequential Data*, pages 22–25, 2004.

[27] Hiroaki Sakoe and Seibi Chiba. A dynamic programming approach to continuous speech recognition. In *Proceedings of the seventh international congress on acoustics*, volume 3, pages 65–69, 1971.

[28] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 26(1):43–49, 1978.

[29] Jérôme Vial, Hicham Noçairi, Patrick Sassiat, Sreedhar Mallipatu, Guillaume Cognon, Didier Thiébaut, Béatrice Teillet, and Douglas N Rutledge. Combination of dynamic time warping and multivariate analysis for the comparison of comprehensive two-dimensional gas chromatograms: application to plant extracts. *Journal of Chromatography A*, 1216(14):2866–2872, 2009.