Interactive Exploration of Time Series Powered by Time Warped Distances

by

Rodica Neamtu

A Dissertation

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Doctor of Philosophy

in

Computer Science

by

April 12, 2017

APPROVED:

Prof. Elke A. Rundensteiner
Worcester Polytechnic Institute
Advisor

Prof. Gabor Sarkozy
Worcester Polytechnic Institute
Advisor

Prof. George Heinemann
Worcester Polytechnic Institute
Committee Member

Prof. Samuel Madden
Massachusetts Institute of Technology
External Committee Member

Prof. Craig Wills
Worcester Polytechnic Institute
Chair of CS Department
Abstract

Domains such as astronomy, finance, e-commerce and genome sequencing are collecting a staggering amount of data, a significant percentage of which is in the form of time series. To make sense of it, scientists need to interactively explore these time series by formulating hypotheses and progressively refining them. The goal of this dissertation is to introduce new and efficient mechanisms to explore big time-series collections. To this extent we devise techniques to support a rich class of operations for time series exploration and analysis, and focus on their scalability to large datasets. Finding similar trends and patterns among time series data is critical for many applications ranging from financial planning to policy making. The detection of these relationships, especially as sequences of different lengths must be compared, is prohibitively expensive - leading to slow responsiveness and high storage needs. In addition, specific application domains have their own interpretations of similarity reflected in the use of various parameters including similarity thresholds and distances. Having to compute similarity distances based on varying parameters compounds the problem of efficient exploration of large collections of time series.

Domain-specific distance measures preferred by analysts for exploring similarities among time series tend to be “point-to-point” distances. The point-wise nature limits their ability to perform meaningful comparisons between sequences of different lengths and with mis-alignments. Analysts thus instead require “elastic” alignment tools such as Dynamic Time Warping (DTW) which enable this kind of flexible comparisons. However, these existing expressive alignment tools do not incorporate the most suitable distances for specific applications. To address this shortcoming, this dissertation introduces the first conceptual unified framework called Generalized Dynamic Time Warping (GDTW) that supports alignment (warping) of a large array of domain-specific distances. While the classic DTW and its prior extensions incorporate the Euclidean Distance, this is the first work to generalize the ubiquitous DTW distance and “extend” its warping capabilities to a rich diversity of popular point-to-point distances. The resulting repository of robust alignment tools enabled by our GDTW methodology holds promise as a valuable resource for the community at large. It also opens the venue for new research towards improving the state-of-the-art ensemble classifiers.

This dissertation focuses on the detection of relationships between time series
and tackles the challenge of the inherent high cardinality of data and the complexity of the process of mining it. The need for parameter tuning and recommendations for similarity distances and thresholds is also addressed.

Within this scope, we focus on three research areas, namely interactive exploration of time series similarity, generalized similarity models, and flexible interpretations of similarity using diverse distances.

We first propose a novel paradigm called Online Exploration of Time Series (ONEX) that employs a powerful one-time preprocessing step to compress the raw data into the compact ONEX base encoding critical similarity relationships among time series. The ONEX framework takes advantage of the computationally inexpensive Euclidean Distance for the construction of the ONEX base, while supporting powerful time-warping using DTW to facilitate the comparison of sequences of different lengths and flexible alignments within a few seconds of response time. ONEX overcomes the prohibitive computational costs associated with the complex Dynamic Time Warping distance by using it over the surprisingly compact ONEX base instead of the raw data.

We introduce a general model for exploring time series similarity. In this work, we design a universal alignment tool, called Generalized Dynamic Time Warping (GDTW), which is flexible enough to allow the use of many different point-to-point distances in computing the warping path, yet powerful enough to enable time warping. The challenge of defining such generalized distance is compounded by the need to devise efficient strategies for exploring time series datasets. Our work fundamentally changes the classic DTW, while keeping and extending its main purpose, which is robustness to local misalignments in time. We propose a three-step methodology that successfully enables a large number of well-known point-wise distances to perform warping and do so efficiently. Its merit stems from (1) the ability to consistently of researchers to expand the repository of warped distances (2) the broad range of problems that these newly warped distances can solve, including but not limited to classification, clustering, best match retrieval, addressing over-warping, etc.

We generalize the work for interactive exploration of time series previously described by the ONEX paradigm by incorporating multiple distances into the similarity model, leading to a general framework for knowledge discovery powered by multiple distances or in short GENEX. GENEX provides deep insights into time se-
eries datasets by revealing similarity results based on using different combinations of distances. The work of this dissertation ties together the previous two research efforts by combining specific point-wise distances with their “time-warped” counterparts to create efficient mechanisms for exploring time series in specific application domains. Our framework facilitates general exploration of time series based on diverse distances and reveals new actionable insights that would be missed by the use of only one single distance. It also enables researchers to incorporate new distances in a uniform manner and minimal mathematical and programming effort.

We conduct experimental studies on real datasets to evaluate both the effectiveness and efficiency of the proposed approach. The experimental evaluation includes studies based on several traditional tasks including best match retrieval and motif discovery, over the 85 datasets from the UCR time-series collection\(^1\), the largest collection of public time series datasets. Case studies on application specific datasets are also conducted for empirical evaluation of our newly warped distances. Such example is the MIT-BIH Arrhythmia Database [58], [31] which contains 48 half-hour excerpts of two-channel ambulatory ECG recordings, obtained from 47 subjects studied by the BIH Arrhythmia Laboratory between 1975 and 1979.

\(^1\) www.cs.ucr.edu/~eamonn/time_series_data
Acknowledgements

I express my deepest gratitude to my co-advisors: Prof. Elke Rundensteiner and Prof. Gabor Sarkozy.

Prof. Elke Rundensteiner has been first instrumental in my choice of pursuing a PhD at WPI and then she became a great mentor and role model. I thank her for the many hours spent discussing everything from my research goals and problems to my dreams and hopes. Our lively meetings made research fun and productive at the same time. I thank her for standing by me and supporting all the choices and decisions I made in the past three years.

I thank Prof. Gabor Sarkozy for reviving my interest in math, and for exploring with me an area that is that at the confluence of our research interests: Big Data and math. I thank him for great conversations, support and the calm attitude he brings to our meetings. I thank him for choosing to work with me and for making our collaboration meaningful, fun and hopefully long lasting.

Special thanks to Prof. Sam Madden from MIT for his invaluable comments and suggestions, for the time spent meeting with me and for feedback on my research. I thank him for his generous support and guidance in both my research and my career.

Thanks to Prof. George Heineman for always believing in me and helping me advance on this academic path.

Special thanks to Dr. Eamonn Keogh from UC Riverside for his invaluable and always prompt feedback, detailed suggestions on my papers, and for sharing datasets and code. His work is my source of inspiration for studying time series.

Thanks to Prof. Dan Dougherty for being the one person where I always go for friendly advice and support. He is a role model with his kindness, patience, expertise and most of all, for understanding exactly who I am and who I strive to become.

Thanks to Prof. David Brown who helped me sharpen my analytical and critical research skills, and guided me along the way.

Thanks to Prof. Craig Wills who always encouraged me to pursue my calling to teach and mentor students and who made it possible for me to continue my work at WPI.

Thanks to my friend and research partner Ramoza Ahsan, together we managed to get
through tough coursework and research. I hope our research collaboration and friendship will continue.

Thanks to former undergraduate students Michael Andrews and Samantha Swartz for help with the implementation. Thanks to undergraduate students Charlie Lovering and Cuong Nguyen who are helping us create interesting visual analytics for our system.

I thank all DSRG members for their useful feedback.

I thank the Computer Science department at WPI for financial support as a TA and RA all throughout my education here.

Last but not least, I am grateful to my family whose unconditional love and support allowed me to focus on working hard to turn this life long dream into reality. It is to my husband Tom Wall and my children Alex and Andrew Neamtu that I dedicate this most important professional achievement of my life.
# Contents

## 1 Introduction
- 1.1 Motivation .............................................. 13
- 1.2 Challenges and State-of-the-Art ......................... 18
- 1.3 Proposed Solutions ...................................... 24
- 1.4 Dissertation Organization ................................. 30

## 2 Preliminaries
- 2.1 Point-wise Similarity Distances ......................... 31
- 2.2 Dynamic Time Warping .................................. 33
- 2.3 Other Similarity Concepts ................................. 36

## 3 Interactive Exploration of Time Series
- 3.1 Theoretical Foundation of the ONEX Framework ....... 39
- 3.2 The ONEX Base ........................................... 45
- 3.3 ONEX Online Query Processor .......................... 51
- 3.4 Experimental Evaluation of ONEX ....................... 59
- 3.5 Evaluation of Query Processing Time and Accuracy .... 60
- 3.6 Evaluation of Preprocessing Performance ................ 65
- 3.7 ONEX Visual Analytics .................................. 66

## 4 Generalized Model for Time Warping
- 4.1 Generalized DTW Distance ............................... 72
- 4.2 Fundamentals of the DTW Warping Path ................. 72
- 4.3 Devising Efficient Strategies for Computing the GDTW Warping Path .... 75
- 4.4 Proposed GDTW Methodology ............................ 76
- 4.5 Practical Requirements for Newly Designed Distances .. 77
- 4.6 GDTW Case Study ........................................ 78
- 4.7 Warping $L_p$ Norm Distances ............................ 78
- 4.8 Warping the Sorensen Distance ......................... 81
- 4.9 Warping the Cosine Distance ............................. 81
List of Figures

1  The ONEX approach ................................................. 24
2  (a) Define warping path, (b) Apply dynamic programming to find minimum weight path. ........................................ 33
3  Example of a matrix for computing DTW using dynamic programming . 36
4  ONEX base intuition .................................................. 42
5  Similarity parameter space .......................................... 49
6  Response time for similarity queries .............................. 61
7  Response time for similarity queries varying the number of time series in the dataset ........................................... 61
8  Offline construction time varying ST ............................. 62
9  Number of representatives .......................................... 62
10 Time response for seasonal similarity queries .................. 64
11 Visual example of seasonal similarity ............................. 65
12 Tradeoff accuracy vs time varying ST for ItalyPower, ECG, Wafer, Face datasets in clockwise order .......................... 66
13 ONEX visual analytics framework ................................ 67
14 Matched misaligned sequences from ECG dataset .......... 68
15 Similarity View with Overview Pane, Query Selection Pane, Query Preview Pane, and Similarity Results Pane showed respectively in counter-clockwise order. ............................................. 68
16 Seasonal view displaying Patterns in the power usage dataset .... 70
17 Visualizing MATTERS tech employment .......................... 70
18 Seasonal View displaying patterns in the power usage dataset .... 71
19 General warping path ................................................. 73
20 Computing the warping path with classic DTW .................. 75
21 Computing the warping path with $GDTW_{MD}$ ...................... 80
22 Computing the warping path with $GDTW_{Mink}$ ................. 80
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>23</td>
<td>A trivariate plot comparing $GDTW_{ED}$, $GDTW_{MD}$, and $GDTW_{Mink}$. For points close to the center of the figure, all three methods produce similar error rates. For points away from the center, at least one method is performing poorly.</td>
</tr>
<tr>
<td>24</td>
<td>Best matches in ECG retrieved with point-wise distances and their warped counterparts</td>
</tr>
<tr>
<td>25</td>
<td>Average linkage hierarchical clustering</td>
</tr>
<tr>
<td>26</td>
<td>Warping for a pair of sequences in ECG</td>
</tr>
<tr>
<td>27</td>
<td>Warping for distorted sequences</td>
</tr>
<tr>
<td>28</td>
<td>Heart rate time series</td>
</tr>
<tr>
<td>29</td>
<td>Case study best match sequences</td>
</tr>
<tr>
<td>30</td>
<td>Detailed results of the 1NN classifier over 85 datasets</td>
</tr>
<tr>
<td>31</td>
<td>Error-rate Binary Plot $GDTW_{ED}$ vs $GDTW_{MD}$</td>
</tr>
<tr>
<td>32</td>
<td>Error-rate Binary Plot $GDTW_{ED}$ vs $GDTW_{Mink}$</td>
</tr>
<tr>
<td>33</td>
<td>Error-rate Binary Plot $GDTW_{MD}$ vs $GDTW_{Mink}$</td>
</tr>
<tr>
<td>34</td>
<td>Warping for a pair of sequences in ItalyPower</td>
</tr>
<tr>
<td>35</td>
<td>GENEX overview</td>
</tr>
<tr>
<td>36</td>
<td>Class diagram of the general distance framework</td>
</tr>
<tr>
<td>37</td>
<td>Response time for similarity queries using $GDTW_{ED}$</td>
</tr>
<tr>
<td>38</td>
<td>Response time for similarity queries using $GDTW_{MD}$</td>
</tr>
<tr>
<td>39</td>
<td>Response time for similarity queries using $GDTW_{Mink}$</td>
</tr>
<tr>
<td>40</td>
<td>Tradeoff accuracy vs time varying ST for ItalyPower using $GDTW_{ED}$</td>
</tr>
<tr>
<td>41</td>
<td>Tradeoff accuracy vs time varying ST for ItalyPower using $GDTW_{MD}$</td>
</tr>
<tr>
<td>42</td>
<td>Tradeoff accuracy vs time varying ST for ItalyPower using $GDTW_{Mink}$</td>
</tr>
<tr>
<td>43</td>
<td>Tradeoff accuracy vs time varying ST for ECG using $GDTW_{ED}$</td>
</tr>
<tr>
<td>44</td>
<td>Tradeoff accuracy vs time varying ST for ECG using $GDTW_{MD}$</td>
</tr>
<tr>
<td>45</td>
<td>Tradeoff accuracy vs time varying ST for ECG using $GDTW_{Mink}$</td>
</tr>
<tr>
<td>46</td>
<td>Tradeoff accuracy vs time varying ST for Face using $GDTW_{ED}$</td>
</tr>
<tr>
<td>47</td>
<td>Tradeoff accuracy vs time varying ST for Face using $GDTW_{MD}$</td>
</tr>
<tr>
<td>48</td>
<td>Tradeoff accuracy vs time varying ST for Face using $GDTW_{Mink}$</td>
</tr>
<tr>
<td>Page</td>
<td>Section Title</td>
</tr>
<tr>
<td>------</td>
<td>---------------</td>
</tr>
<tr>
<td>49</td>
<td>Tradeoff accuracy vs time varying ST for Symbols using $GDTW_{ED}$</td>
</tr>
<tr>
<td>50</td>
<td>Tradeoff accuracy vs time varying ST for Symbols using $GDTW_{MD}$</td>
</tr>
<tr>
<td>51</td>
<td>Tradeoff accuracy vs time varying ST for Symbols using $GDTW_{Mink}$</td>
</tr>
<tr>
<td>52</td>
<td>Response times for motif discovery using ED, MD, Mink</td>
</tr>
<tr>
<td>53</td>
<td>Seasonal similarity using ED in Face dataset</td>
</tr>
<tr>
<td>54</td>
<td>Motif discovery using MD in Face dataset</td>
</tr>
<tr>
<td>55</td>
<td>Motif discovery using Mink in Face dataset</td>
</tr>
<tr>
<td>56</td>
<td>Detailed results for preprocessed 85 datasets using $GDTW_{ED}$, $GDTW_{MD}$, $GDTW_{Mink}$</td>
</tr>
<tr>
<td>57</td>
<td>Size of GENEX base in MB for 85 datasets using ED, MD, Mink</td>
</tr>
<tr>
<td>58</td>
<td>Number of representatives varying ST using ED</td>
</tr>
<tr>
<td>59</td>
<td>Number of representatives varying ST using MD</td>
</tr>
<tr>
<td>60</td>
<td>Number of representatives varying ST using Mink</td>
</tr>
<tr>
<td>61</td>
<td>Construction times for GENEX base for 85 datasets using ED, MD, Mink</td>
</tr>
<tr>
<td>62</td>
<td>Offline construction time varying ST using ED</td>
</tr>
<tr>
<td>63</td>
<td>Offline construction time varying ST using MD</td>
</tr>
<tr>
<td>64</td>
<td>Offline construction time varying ST using Mink</td>
</tr>
<tr>
<td>65</td>
<td>Similar ECG sequences for arrhythmia study record 107</td>
</tr>
</tbody>
</table>
List of Tables

1. List of notations for ONEX ...................................................... 36
2. List of clauses for ONEX operations ........................................ 51
3. Datasets statistics ................................................................. 59
4. Time response similarity solution same length as query ............... 62
5. Accuracy for similarity solution of same length as query ............. 63
6. Accuracy for similarity solution for any length ............................. 63
7. Number of representatives, total number of subsequences and size in MB for specific datasets ............................................................ 65
8. Average warpings for sequences of the same length ....................... 90
9. Average warpings for sequences of different lengths .................... 91
10. Best match comparison ED vs DTW in ItalyPower ....................... 97
11. Best match comparison MD vs $GDTW_{MD}$ in ItalyPower ............... 98
12. Best match comparison Mink vs $GDTW_{Mink}$ in ItalyPower ............. 98
13. Best match comparison ED vs classic DTW in ECG ....................... 99
14. Best match comparison MD vs $GDTW_{MD}$ in ECG ....................... 99
15. Best match comparison Mink vs $GDTW_{Mink}$ in ECG ..................... 99
16. Popular point-wise similarity distances ..................................... 104
17. List of notations for GENEX ....................................................... 104
18. List of clauses for GENEX ....................................................... 122
19. Percentages of scenarios for returning the same motif ................... 133
1 Introduction

1.1 Motivation

In applications ranging from finance, business, and medicine to meteorology [33],[68], time series data is prevalent presented as stock fluctuations, ECG, rainfall amounts, etc. In these applications the latent value of data is unlocked by detecting similarities between time series of different lengths and alignments. This way the analysts “transform” raw data into quality insights. To accomplish that, analysts need interactive systems that can answer complex questions in real-time and that are flexible enough to work in diverse domains.

Motivating interactive exploratory systems that can perform rich classes of operations. Let’s consider a real life example revealing challenges involved in finding and leveraging data similarities for decision making. In 2013 in Massachusetts, organizations set out to repeal the Sales and Use Tax on computer and software services, perceived as potentially having a negative impact on the economic health of the state. Data-driven evidence was analyzed to show similarities between tax rates over time and fluctuations of social and economic factors, all modeled as time series, obtained from a large spectrum of public governmental websites. Many difficulties were encountered, chief among them being the time to find and interpret the similarities between economic indicators represented as time series.

(1) The presence of data from different domains reported over specific intervals required the comparison of time series of different lengths and alignments, as the impact of a tax change might take different durations to become apparent. Such time-aware comparisons must be performed using robust distances like Dynamic Time Warping (DTW) [10], Longest Common Subsequence (LCSS)[35] and Edit Distance with real penalty (ERP) [18]. The “power” of these distances, beneficial in terms of accuracy, is shadowed by their computational complexity and thus slow time responsiveness especially for large datasets2.

(2) During this process analysts used specific indicators, like the economic growth

2https://followthedata.wordpress.com/2014/06/24/data-size-estimates/
rate, to evaluate the potential impact of introducing the new tax. For example, they “designed” a sample growth rate time line indicative of a positive impact of the tax. Then they searched for matches for this sample economic growth rate among all states. In such scenario, it is possible that the sequence may or may not exist in the dataset. Perfect matches, if found, reflect identical impacts that such tax had in those specific states. If only close matches are found, they indicate slightly different impacts in specific states. We conclude that analysts need to have the ability to explore large time series data by using sample sequences that might or might not be present in these datasets.

(3) Analysts had to answer complex questions that were not necessarily based on finding the best match sequence. For example, they searched for recurring similarity patterns in the income growth rate of a state over a few years. Analysts also searched for similar growth rates and other economic indicators in different states over specific time lengths, indicating similar “shorter-term” impacts.

(4) Data from diverse domains requires the use of different parameter settings such as similarity thresholds, leading to repeated and redundant computations for each parameter setting. For example, the most suitable thresholds for studying similarity of demographic data are different than the ones used for growth rates. Analysts don’t always know what similarity threshold leads to the most useful results, so they have to try different values for these thresholds. It is also important for analysts to understand how the similarity “panorama” changes, i.e. what is the best match for a sample when different values for the threshold are chosen.

So far, we showed that getting insights from large time series data is a complex problem. Analysts need efficient exploratory systems with different levels of temporal granularity – so they can examine sequences of different lengths – powered by time-warping distances. This exploration should empower analysts to discover diverse aspects of similarity, such as finding best matches or recurring patterns, and interpret them in the context of using appropriate parameter settings. To accomplish that, we need powerful systems that support rich interactive analytics on large time series data sets, enabling analysts to get near instantaneous time responses to their exploratory requests.
The above motivating example shows that in an era when time series are produced at an unprecedented rate in most application domains, it is crucial to be able to find trends and patterns among them. Some of the tasks that analysts had to perform revealed the need for using robust time warping distances that enable the comparisons of sequences with different alignments and lengths. We explore this need further in a more detailed motivating example for using time warping distances.

**Motivating Time Warping.** In most real-life scenarios there is a need for comparing sequences of different lengths and alignments. For example, an analyst might want to find stocks that had a similar growth with that of the Apple stock between the years 2005 and 2007 but spread over a different time period. Or a doctor might wonder if a certain shape found on the ECG of a patient recorded over one week is similar to any previously recorded ECGs for other patients. These situations require the use of distances with warping capabilities, also known as “elastic distances”. Dynamic Time Warping (DTW), introduced by Berndt and Clifford [10], is such a distance that allows sequences to be stretched or compressed along the time axis, namely an element of one sequence can be matched to one or more elements of another sequence. DTW is popular for diverse applications due to its rich warping power and its efficiency in computation thanks to extensive performance optimizations [45], [65]. Applications for DTW include bioinformatics for applying DTW to RNA expression data [1], medicine for ECG pattern matching [12], image retrieval for aligning biometric data [29], and in chemical engineering, for synchronizing batch processes in polymerization [32], to just name a few.

Despite its great popularity, DTW has been proven not to always be the most appropriate distance for exploring time series similarity because it can produce pathological results through none-intuitive alignments [46]. This stems from the fact that DTW uses the Euclidean distance as a base distance [10]. This limits its utility for many applications that require customized distances beyond the Euclidean. In fact, many applications prefer other distances than Euclidean, but they end up settling for the use of DTW mostly due to its warping abilities.

We now examine the need for using domain-specific distances in exploring time
series similarity.

Motivating Diverse Domain-Specific Distances. Application domains require customized interpretations of similarity. These semantics can be expressed through the use of domain-specific distances. We now give examples of such distances. In chemistry, applications for compound classification select the most relevant set of chemical descriptors using the Minkowski distance [40]. In motion detection applications where d-dimensional trajectories must be indexed [56], the Chebyshev polynomials are preferred. For image retrieval where the desirable distance measure should reflect human perception, common distance measures include Manhattan [75], [76] and Mahalanobis distances [73]. For network analysis including shortest network time distances across the metropolitan area [5], Cartesian distances like Manhattan are popular.

Unfortunately, all these distances suffer from the major limitation of being “point-to-point distances”. That is, they can’t effectively be used for the comparison of sequences of different lengths and sequences that are not aligned in time. Application developers thus face the major dilemma of having to choose between the most appropriate distance measure for their domain versus supporting flexible sequence matches.

We conclude that there is a clear need for a general strategy that can enable analysts to use the distance that is best suited for their specific domain, yet without being limited to only comparing sequences of the same length and sequences without any local shifting.

Motivating diverse distances across domains. Most existing systems [39], [4], [27], [65] implement only one specific distance to explore similarity, which sometimes leads to a fairly narrow domain of applications. In reality, complex applications, such as economic indexes, combine information from diverse domain applications. For example, the MATTERS³ economic indicator uses information from a variety of public data sources to calculate the rankings of states. Websites like the Tax Policy Center⁴, the Census Bureau⁵, the National Science Foundation⁶ and the Bureau of Economic Analysis⁷ refer to distinct application domains and their reports mostly contain time series

³http://matters.mhtc.org/
⁴www.taxpolicycenter.org/
⁵www.census.gov
⁶www.nsf.gov/
⁷www.bea.gov/
data. Attempting to find similar trends between the time series originating from these sites has to tackle the obvious challenge of seamlessly working with distinct domain-specific distances. Distances that are well suited for a domain like education or demographics might not be the best for taxes or finances.

In conclusion, the problem of time series exploration is a complex problem, whose solution aims to offer analysts interactive exploratory tools that can: (1) perform best match retrieval, pattern discovery and other classes of similarity searches, (2) enable parameter tuning and recommendations, (3) use domain-specific time warping distances that can compare sequences of different lengths and alignments, (4) provide analysts with the ability to explore datasets using diverse distances and discover new insights that would otherwise be missed, and (5) enable researchers to incorporate new distances in a consistent manner and without significant mathematical and programming skills.
1.2 Challenges and State-of-the-Art

1.2.1 State-of-the-Art

The research community has been working relentlessly to address the challenges related to exploring time series similarity in the past two decades. Because it is such a complex problem, often the research community ends up providing solutions that focus on a specific aspect, rather than try to solve all the issues within one solution. As the Related Work shows (Sec. 6.3), many existing solutions either provide fast response time to a specific class of operations, such as exact fast retrieval of the best match [65], or find repeating patterns [10]. Others aim to modify existing warping distances [46] to produce superior alignments or to include domain-specific distances [69]. We examine here the state-of-the-art and its limitations first in the area of interactive exploration of time series similarity, then in the use of diverse similarity distances.

**State-of-the-art in time series similarity exploration.** Applications are becoming increasingly intolerant to slow response times and are looking for more flexible interpretations of similarity in the quest to answer complex questions. Existing state-of-the-art systems are providing much shorter response times, but for a specific class of operations rather than across many classes.

Lag in responsiveness is known to risk losing an analyst’s attention during the exploration process. Often systems have to make a choice between providing a fast response time or providing a large class of solutions. Many systems exploring similarity focus on providing an exact or a highly accurate solution [4], [45], [70] at the expense of a timely response. This could be detrimental for applications like medicine and the stock market that are depending on immediate answers to act on. Others [6] use preprocessing steps to improve the timely responsiveness. Some solutions are extremely fast [65], but they can only perform one operation, namely the retrieval of the best match for a given sample sequence.

As shown above, often systems have to compromise between the choice of similarity distance and time responsiveness. Many applications rely on the use of simple-to-compute distances like the Euclidean Distance [27], [42], [83] to achieve faster response
Other distances have been exploited in image retrieval, including Manhattan distance [75], [76], Euclidean distance [80], and Mahalanobis distance [73]. Because they are point-wise distances, they are inexpensive to compute and provide fast response times. As a consequence, though, they cannot handle sequences that are not aligned or have different lengths. The use of time-warping distances like DTW is overshadowed by their computational complexity [15] leading to slow responsiveness and poor scaling as data grows. Analysts thus have to make a difficult choice between the ability to perform powerful comparisons and the resulting time delays and storage demands.

Similarity exploration systems have to also address the difficulty of supporting varying similarity parameter values. Our motivating example speaks to the need for flexible interpretations of similarity, as similarity between taxes or financial time lines may be interpreted differently than for demographics data. Analysts also need the valuable experience of understanding the way similarity between time lines changes when using different parameter values. Thus, a truly interactive system has to provide the ability to explore a dataset using varying similarity thresholds. Unfortunately, most existing systems [45], [65], [70] do not support such flexible similarity insights.

Along with supporting varying parameter values, there is an increasing need for providing parameter recommendations. An analyst may not know the precise parameter settings that would result in the most interesting panorama being extracted. It might take a large number of successive trial-and-error interactions using different parameter values to get insights from the data. Existing exploratory systems [6], [27], [41] tend to be “black boxes” that provide no suggestions of parameter settings. A system that makes recommendations for parameter tuning would be extremely helpful for analysts in finding similarities using fewer trial-and-error attempts.

In summary, although much progress has been made in exploring time series, limitations remain to be addressed including the trade-off between the time responsiveness and the use of complex distance measures to support the comparison of unaligned and different length sequences, compounded by the need for a general solution suitable for diverse application domains.

State-of-the-art in elastic similarity distances for specific application domains.
1.2 Challenges and State-of-the-Art

**State-of-the-art in Dynamic Time Warping.** To accommodate the large array of application domains, a variety of distance functions have been defined. Elastic distances like Dynamic Time Warping [43], [10], the Longest Common Subsequence (LCSS) [27], and Edit Distance with Real Penalty (EDR) [18] enable flexible comparisons between time series of different lengths and alignments. Their use is often overshadowed by the complexity of their computation and the inherently slow time response associated with them.

DTW in particular is popular for time series data analysis due to its tolerance for local shifting and handling of variable lengths time series. Countless modifications of the classic DTW have been proposed to either (1) further optimize its performance by indexing, caching, and other optimization strategies [27], [83], [43], [79], [65] or (2) to improve the quality of alignments between time series [46], [34], [37]. Among them, [69] replaces the Euclidean base distance utilized by the classic DTW with similar sum-based distances to improve the warping quality. However, even this state-of-art method [69] is rather limited, as it only incorporates simple metrics based on sums as base distances, such as Euclidean or Manhattan. Unfortunately, none of the state-of-art DTW warping methods succeeds in supporting any of the widely popular distances, like Cosine [15], Pearson [15], Jaccard [15], and so on, which are based on combinations of operations including fractions, products, min, and max.

The answers to complex questions often involve exploring data in diverse domains. Since different domains favor specific distances, exploratory systems could benefit from flexibility in using more than one distance or combinations of similarity distances. Although there are many systems implementing similarity distances specific to certain applications, most systems only implement one similarity distance. By only accommodating one specific distance measure, these systems can not provide the ability to compare side by side the results of exploring time series using different distance measures. Analysts could benefit from the ability to perform such side-by-side comparisons and understand the changes in the results produced by the use of different distance measures. This would enhance their ability to choose the most appropriate similarity distance for specific applications.
1.2.2 Research Challenges Addressed in this Dissertation

Here are some of the hurdles encountered in exploring time series:

1. High data cardinality leading to decreased responsiveness. Time series datasets like financial records or collections of ECG data from hospital patients tend to be huge. Worse yet, the need for accommodating different temporal granularities requires us to consider time series of different lengths. For a dataset containing $N$ time series, each of length $n$, the total number of subsequences to consider is $Nn(n-1)/2$. For example, a benchmark dataset like StarLightCurves from the UCR Time Series Data Mining Archive \(^8\) with 9236 time series, each of length 1024, is composed of $4.83e^9$ subsequences. Real-world datasets tend to be orders of magnitude larger than this\(^9\). Performing similarity comparisons for all these subsequences is clearly impractical.

Many state-of-the-art techniques try to address this lack of instantaneous responsiveness. Lag in responsiveness risks losing an analyst’s attention during the exploration process. The state-of-the-art is faced with the trade-off between accuracy and time response, especially in exploring very large datasets. Some systems provide an exact or a highly accurate solution \(^4\), \(^45\), \(^70\) at the expense of a timely response. This could be detrimental for certain applications in medicine and finances that are depending on immediate answers to act on. Others \(^6\) use a preprocessing step to improve the timely responsiveness of serving subsequent requests. The preprocessing step encodes similarity relationships to be used for further exploration.

2. Comparing sequences with different lengths and time alignments. Further, to compare sequences of different lengths and alignments, complex distances such as DTW \(^10\) must be used. To precompute, store and explore the many pairwise similarity comparisons between sequences using DTW is very expensive even for datasets that are not as large as the above example. Systems resort to either accepting increased time responses or settling for faster-to-compute distances at the expense of a decreased accuracy.

In other words, the state-of-the-art has been wrestling with the compromise be-

\(^8\) www.cs.ucr.edu/~eamonn/time_series_data/
\(^9\) https://followthedata.wordpress.com/2014/06/24/data-size-estimates/
1.2 Challenges and State-of-the-Art

tween the choice of similarity distance and the time responsiveness. Many applications rely on the use of fast-to-compute distances like the Euclidean Distance [27], [42], [83] to achieve a faster response time. As a consequence they cannot handle sequences that are not aligned or have different lengths. Yet the use of time-warping distances like DTW is overshadowed by their computational complexity [19], leading to slow responsiveness and poor scaling as data grows. Applications thus have to make a difficult choice between the ability to perform powerful comparisons and the resulting time delays and storage demands.

3. Offering rich classes of exploratory queries. As shown in our motivating example, getting insights involves not just finding the best match for a sequence, but also having the ability to discover recurring similarity patterns. This requirement compounds the previous challenges. For this reason, most state-of-the-art systems suffer from a lack of flexibility in exploring similarity. Instead they focus only on finding the best match for a sequence [65], [27], [43]. In reality, analysts would benefit from having the opportunity to answer complex questions related to different facets of similarity, including finding patterns and repetitions using the same system. This would help them better understand the datasets.

4. Supporting guided similarity exploration and the use of varying similarity thresholds. Analysts in different application domains interpret similarity differently. Thus, they may select different settings for parameters like similarity thresholds. Keeping the result set for each possible similarity parameter is inefficient, resulting in large storage and repeated computation. Many state-of-the-art systems can not overcome the difficulty of supporting varying similarity parameters. Understanding the way similarity between time lines changes when using different parameter values is critical for analysts. Unfortunately, most existing systems [45], [65], [70] do not offer a dedicated support for flexible similarity insights.

Most state-of-the-art systems also suffer from the lack of parameter recommendations. An analyst may not know the precise parameter settings that would result in the most interesting panorama being extracted. It might take a large number of successive trial-and-error interactions using different parameter values to get insights from the
data. Existing exploratory systems [6], [27], [41] tend to be “black boxes” that provide no suggestions of parameter settings. A system that makes recommendations for parameter tuning would help analysts in finding similarities using fewer trial-and-error attempts.

5. **Defining versatile elastic similarity distances for diverse application domains**

Analysts have to compromise their choice of similarity distance because most of the distances preferred by specific domains are point-to-point distances, therefor they can not be used for comparing sequences with different lengths or time alignments. The most popular elastic distance, DTW has been proved to produce unnecessary warpings in different application domains, mostly due to the fact that it incorporates the Euclidean distance as base distance [46]. Such shortcoming can only be avoided by “warping” more general distances.

6. **Supporting similarity exploration using multi-distances.** Most systems use a specific distance measure for similarity. Yet, the similarity results change based on the distance measure used. A comparative study of similarity based on using different distances would give analysts better insights into datasets. Having the ability to incorporate new distances with ease would benefit researchers and analysts who can now use a plethora of distances for knowledge discovery.

*In summary, it is difficult to build interactive systems for efficient exploration of time series. Such systems must rely on combining different strategies like using preprocessing steps, compromising between the choice of similarity distance and time responsiveness, reducing data cardinality, as well as efficient indexing. Even the most promising solutions focus on a specific class of queries, still leaving room for improvement in the area of answering richer classes of queries.*
1.3 Proposed Solutions

This dissertation tackles the following three topics related to exploring time series similarity.

1. **Topic 1.** Interactive exploration of time series (ONEX)
2. **Topic 2.** Generalized Dynamic Time Warping Framework (GDTW)
3. **Topic 3.** Knowledge discovery powered by multiple distances (GENEX)

1.3.1 **Topic 1: Interactive Exploration of Time Series.**

In the first task we create a new paradigm for exploring time series. To address the inefficiencies in state-of-the-art techniques in exploring similarity we devise efficient ways to explore time series similarity through rich classes of operations and very fast response times. ONEX offers an innovative knowledge-centric panorama of time series data by using an interactive similarity exploration model. As shown in Figure 1, our approach includes a one-time preprocessing step for the construction of a rich, compact ONEX base using an inexpensive to compute distance yet supporting subsequent powerful time-warping online interactive analytics. The preprocessing step is crucial for achieving interactivity, as it allows us to compute similarity relationships ahead of time and subsequently leverage the results.

ONEX offers a viable answer to the trade-off between the need for real time responsiveness and the high computational complexity due to the use of non-metric dis-
tances for comparing sequences with different lengths and alignments. Because it is prohibitively expensive to use DTW, we solve this problem by employing two distances. While we use the computationally inexpensive Euclidean Distance to construct compact similarity groups for specific lengths, we support the exploration of these groups using a powerful time-warping method DTW [10]. This unique combination yields very accurate results at much reduced response time rates, as DTW is successfully applied over the compact ONEX base instead of the raw data. ONEX offers rich interactive analytics to explore time series.

We complete the following tasks:

1. We precompute offline critical similarity relationships between subsequences and preserve them in the form of “similarity groups” and their representatives (Sec. 3.1.1). The resulting ONEX base plays a key role in achieving interactive responsiveness.

2. We formally prove a triangle inequality between ED and DTW that builds a conceptual bridge between the offline construction of the ONEX base and its online exploration. Our solution offers a viable answer to the trade-off dilemma between the use of complex time warped distances and timely responsiveness. (Sec. 3.1.2).

3. The ONEX base enables analysts to get insights into datasets by exploring similarity and its recurring patterns. ONEX also provides analysts with parameter tuning guidance to give a better intuition of the major changes in similarity. (Sec. 3.3.1)

4. We devise and implement query processing strategies to efficiently construct the ONEX base and perform traditional and new classes of operations. (Sec. 3.2, 3.3)

5. In depth experimental evaluation of the accuracy and time response is conducted, comparing ONEX with the fastest known method for best match retrieval [65], as well as the popular Piece Aggregate Approximation for dimensionality reduction [45] and the standard DTW brute force method. Our ONEX system proves to be faster and more accurate than the fastest-known-so-far state-of-the-art method.
6. We develop complementary visual analytics to help analysts better understand similarity in specific datasets. Our web interface helps analysts better understand the data distribution as well as compare sequences using diverse visual representations.

1.3.2 Topic 2: Generalized Dynamic Time Warping Framework

In this task we identify the limitations of DTW and propose a new framework for warping point-wise distances to enable them to compare subsequences of different lengths and alignments. The DTW as introduced by Berndt and Clifford [10] is a popular method for dealing with alignment shifting and locally out-of-phase sequences as well as comparing sequences of different lengths. The research community has been relentlessly working to address issues of scalability and dealing with “singularities” [46] related to the use of DTW.

Prior generalizations of the DTW algorithm [69] are based on the use of a more generic function for determining the weight of the path by replacing the Euclidean Distance with more general distances. Many base distances can be considered here and incorporated in the algorithm, but the computation of the DTW still follows the same methodology as the classic algorithm.

Our proposed approach is more general than any of the previously proposed methods [21], [37], [48], because it fundamentally changes the way the warping path is computed. As shown in our introduction, different application domains favor the use of specific distances. These distances are better suited for getting insights about domain-specific data, but most of them are limited by their inability to compare sequences with different lengths and alignments. Thus our aim is to introduce a general framework that allows time warping, while incorporating the versatility of many distances. The intuition behind our generalization of DTW is to define the warping path to be a specific distance and then use this as a basis to devise efficient strategies for computing it.

We complete the following tasks:

1. We introduce GDTW, the first conceptual framework that overcomes the open problem mentioned above and allows a diversity of point-wise distances to be
transformed into elastic distances by extending warping properties using a uniform methodology. (Sec. 4.1 and 4.2).

2. We devise a step-by-step methodology that empowers analysts to “warp” desired distances of their choice with minimal mathematical and programming effort (Sec. 4.3).

3. We validate our GDTW methodology by applying it to popular point-wise distances with diverse mathematical characteristics [15], especially distances that could not work under the classic DTW algorithm. Through our case study, we develop a repository of warped distances that are now ready to be used by the community at large (Sec. 4.6).

4. Our extensive experimental study comparing our newly warped distances on 85 benchmark datasets demonstrates that one of our newly introduced warped distances produces more reliable classification results compared to the state-of-the-art DTW distance (Sec. 4.13 and 4.15).

5. Furthermore, our real world use case using the Arrhythmia Dataset guided by a domain expert shows the utility of our distances for better interpreting ECG similarity in the medical domain. (Sec. 4.14).

1.3.3 Topic 3: Efficient Knowledge Discovery Powered by Multiple Distances

In this task we address the need for exploring time series datasets using diverse distances. As our Related Work section shows, there are many different similarity distances. Choosing the most appropriate distance for specific application domains is hard because most systems only implement one specific distance, thus not enabling the analysts to compare side-by-side the results produced by other distances. Our goal is to create a more flexible similarity model, one that incorporates a large array of distances and thus empowers analysts to choose the similarity distance that best fits the dataset they are interested in at the time.

We design a framework for efficiently exploring time series datasets using multiple
distances. While researchers and analysts advocate specific similarity distances, a major contribution of our work is the ability to support various distances within the same system and compare the results. We extend our previous work for online exploration of time series (ONEX) with GENEX (for general exploration of time series using multiple distances). GENEX provides deep insights into time series datasets by exploring them in the rich context of multiple distances and using rich classes of operations, all with real time responsiveness. This work binds together the two previous tasks by enabling the exploration of time series using specific combinations of “point-to-point” distances and their time-warped counterparts. Having the ability to use multiple distances and get answers rapidly empowers analysts to learn what distances work best to solve specific data mining tasks as well as understand the changes in similarity due to the use of diverse similarity distances. This invaluable experience comes with many challenges including: (1) integrating multiple distances in the same system (2) guaranteeing interactivity through short response times (3) enabling analysts to compare the results retrieved by specific distances. This versatility is unique to our approach.

In order to accomplish this, we complete the following subtasks:

1. We create a data discovery tool that enables analysts to get better insights into time series datasets based on a large array of distances. GENEX provides analysts with the opportunity to seamlessly work with multiple distances on the same data set, see recommendations or output from those matches and then decide what distance is best suited for exploring each dataset. Thus GENEX facilitates multi-distance driven decision making. In addition, our GENEX framework enables the exploration of complex time series collections aggregating data from heterogeneous domains using multiple distances, therefore providing insights that might be missed by exploring this data using only one single distance. (Sec. 5.1)

2. Our generalized data exploration is based on applying diverse time-warping distances over compact similarity groups constructed using simple-to-compute point-wise distances instead of the raw data. Our theoretical foundation rests on proving a generalized triangle inequality between pairs of point-wise distances and their warped counterparts. This allows us to explore compact simi-
larity groups constructed with inexpensive point-to-point distances using elastic
distances only when absolutely needed, while guaranteeing accurate results and
very fast response times. (Sec. 5.2.1)

3. Our compact GENEX bases leverage similarity relationships between sequences
computed using easy-to-compute pairwise distances and preserved in the form
of representatives. These bases play a major role in exploring sequences with fast
time responses and high accuracy. (Sec. 5.2.2)

4. Our modular framework can be extended to incorporate new similarity distances
with guaranteed results. Our interfaces based on template patterns insure that
adding new distances can be done with some mathematical and little program-
ing effort. (Sec. 5.4.3)

5. Our extensive experimental evaluation over the 85 datasets in the UCR archive
shows how GENEX can assist analysts in finding answers to complex questions
through rich classes of exploratory operations including similarity searches, best
match retrieval, motif discovery, finding correlated sequences. Experiments also
evaluate the size and construction times for the GENEX bases, as well as offer
guidance through trade-off experimental results for specific datasets. (Sec. 5.5)
1.4 Dissertation Organization

All research tasks have been accomplished, resulting in the implementation of ONEX, GDTW and GENEX systems, and the following research papers:


The rest of this dissertation is organized as follows:

- Chapter 2 first provides the preliminaries of this dissertation.

- Chapter 3 discusses the proposed mechanism for online exploration of time series based on establishing a triangle inequality between Euclidean Distance and Dynamic Time Warping (Topic 1).

- Chapter 4 proposes a new framework for warping point-wise distances, namely Generalized Dynamic Time Warping. (Topic 2).

- Chapter 5 introduces the framework for knowledge discovery using multiple distances based on generalizing the inequality triangle for point-wise distances and their warped counterparts (Topic 3).

- Chapter 6 discusses related work.

- Chapter 7 contains the Conclusion and sketches avenues for future work.
2 Preliminaries

Here we introduce key terms and concepts relevant to our work.

Definition 1 Time Series: A time series \( X = (x_1, x_2, ..., x_n) \) is an ordered set of \( n \) real values. Here \( x_i \) represents the value of the time series \( X \) at the time \( t_i \). A dataset \( D = \{X_1, X_2, ..., X_N\} \) is a collection of \( N \) such time series.

2.1 Point-wise Similarity Distances

There are many distances that can be used to measure time series similarity. From the scientific and mathematical point of view, distance is defined as a quantitative degree of how far apart two objects are. The smaller the distance between two time series, the more similar they are. Generally, similarity measures are indicative of the degree of similarity (i.e. positive correlation) by considering two time series to be similar when the value of the measure is very high [15]. The similarity measures can be expressed in terms of distances, so for the remaining of this proposal we will not make the distinction between the two categories and will refer to them as “distances” or “similarity distances”.

We now give examples of such distances used for exploring similarity. They include the \( L_p \) norms [4], namely the Euclidean Distance, the Manhattan Distance, the Minkowski Distance and the Chebyshev Distance. We also define more complex distances [15] like Sorrensen, Cosine and the Pearson correlation coefficient.

Definition 2 Euclidean Distance. Given two time series \( X = (x_1, ..., x_n) \) and \( Y = (y_1, ..., y_n) \), their Euclidean distance \( ED \) is defined as

\[
ED(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}
\] (1)

Definition 3 Manhattan Distance. Given two time series \( X = (x_1, ..., x_n) \) and \( Y = (y_1, ..., y_n) \), their Manhattan distance \( MD \) is defined as

\[
MD(X,Y) = \sum_{i=1}^{n}|x_i - y_i|
\] (2)
Definition 4 Minkowski Distance. Given two time series $X = (x_1, ..., x_n)$ and $Y = (y_1, ..., y_n)$, their Minkowski distance $M_{in}$ is defined as

$$M_{in}(X, Y) = \left( \sum_{i=1}^{n} (|x_i - y_i|^p)^{1/p} \right)^{1/p}$$

For $p=1$, we get the Manhattan Distance.
For $p=2$, we get back the Euclidean Distance.
For $p \to \infty$, we get

$$M_{in}(X, Y) = \max_{i=1}^{n} |x_i - y_i|$$

Definition 5 Chebyshev Distance. Given two time series $X = (x_1, ..., x_n)$ and $Y = (y_1, ..., y_n)$, their Chebyshev distance $CD$ is defined as

$$CD(X, Y) = \max_{i=1}^{n} |x_i - y_i|$$

This is the same case as the Minkowski Distance for $p \to \infty$.

The following distance is widely used in ecology [53] and known as the Sorensen distance.

Definition 6 Sorensen Distance. Given two time series $X = (x_1, ..., x_n)$ and $Y = (y_1, ..., y_n)$, their Sorensen distance $d_{sor}$ is defined as

$$d_{sor} = SD(X, Y) = \frac{\sum_{i=1}^{n} |x_i - y_i|}{\sum_{i=1}^{n} |x_i + y_i|}$$

Definition 7 Cosine Distance. Given two time series $X = (x_1, ..., x_n)$ and $Y = (y_1, ..., y_n)$, their Cosine distance $d_{cos}$ is defined as

$$d_{cos}(X, Y) = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \sqrt{\sum_{i=1}^{n} y_i^2}}$$

The Pearson coefficient used for measuring correlation is defined below.

Definition 8 Pearson Coefficient. Given two time series $X = (x_1, ..., x_n)$ and $Y = (y_1, ..., y_n)$,
their Pearson coefficient $d_p$ is defined as

$$d_p(X, Y) = \sum_{i=1}^{n} \frac{(x_i - \mu_x)(y_i - \mu_y)}{\sigma_x \sigma_y}$$

(8)

where $\mu_x$, $\mu_y$ and $\sigma_x$, $\sigma_y$ are the mean and respective standard deviations of sequences $X$ and $Y$.

2.2 Dynamic Time Warping

Suppose we have two time series $X = (x_1, x_2, ..., x_n)$ and $Y = (y_1, y_2, ..., y_m)$. We assume here $m \leq n$, but except for the notations, the case of $n \leq m$ works similarly. To align these sequences using DTW, an $n \times m$ matrix $M(X, Y)$ is constructed, where the $(i, j)^{th}$ element of the matrix is the Euclidean Distance between $x_i$ and $y_j$, i.e., $w_{i,j} = ED(x_i, y_j)$. Then a warping path $P$ is a set of elements that forms a path in the matrix $M$ from the position $(1, 1)$ to $(n, m)$. The $t^{th}$ element of $P$ denoted as $p_t = (i_t, j_t)$ refers to the indices $i_t, j_t$ of $(x_{i_t}, y_{j_t})$ of this matrix element in the path. Thus a path $P$ is $P = (p_1, p_2, ..., p_t, ..., p_T)$, where $n \leq T \leq 2n - 1$, $p_1 = (1, 1)$ and $p_T = (n, m)$.

**Definition 9 Warping Path Weight:** Given two time series $X = (x_1, ..., x_n)$ and $Y = (y_1, ..., y_m)$, the weight of the warping path $P$ is defined as:

$$w(P) = \sqrt{\sum_{t=1}^{T} w_{i_t,j_t}^2}$$

(9)

The DTW distance then is defined to be the weight of the path with the minimum weight ($\min_P(w(P))$).

![Warping Path](image1.png)

![Dyn. prog. principle](image2.png)

Figure 2: (a) Define warping path, (b) Apply dynamic programming to find minimum weight path.
2.2 Dynamic Time Warping

A warping path, depicted in Figure 2a, is subject to the following constraints:

1. **Boundary condition.** \( p_1 = (1, 1) \) and \( p_T = (n, m) \) or the path has to start and end on the opposite corners of the matrix.

2. **Continuity condition.** The steps in the warping path are restricted to adjacent cells, including diagonally adjacent cells. We express this using the simplified notations [46]: for \( p_i = (u, v) \) we have \( p_{i-1} = (u', v') \), where \( u - u' \leq 1 \) and \( v - v' \leq 1 \).

3. **Monotonicity condition.** The elements on the path have to be monotonically spaced or \( u - u' \geq 0 \) and \( v - v' \geq 0 \) and \( (u, v) \neq (u', v') \).

There is an exponential number of warping paths satisfying these conditions. Thus finding the minimum weight warping path is prohibitively expensive. The warping path can be efficiently calculated using dynamic programming [69]. Conceptually, given the matrix \( M \) containing pairwise Euclidean distances of all elements in \( X \) and \( Y \), we construct a dynamic programming matrix \( \Gamma \) by filling in the values using the following recursive expression.

\[
\gamma(i, j) = ED^2(x_i, y_j) + \min(\gamma(i - 1, j - 1), \gamma(i - 1, j), \gamma(i, j - 1))
\]

This expression determines the current distance \( \gamma(i, j) \) in the cell \((i, j)\) of \( \Gamma \) as the sum of the square of the distance currently found in the cell in the same position in the original matrix \( M \) and the minimum of the cumulative distances found in the adjacent cells (diagonal, left and down) in the dynamic programming matrix \( \Gamma \). Then \( DTW^2(X, Y) = \gamma(n, m) \). This principle is illustrated in Figure 2b, while further details on DTW can be found in [48] and [65].

Instead of attempting to solve the entire problem all at once, solutions to sub-problems are computed and used to repeatedly find solutions to a slightly larger problem until the solution is found for the entire time series. A two-dimensional \( m \times n \) cost matrix \( \gamma \) is constructed where the value at \( \gamma(i, j) \) is the minimum distance warping path that can be constructed from the two time series \( \mathbf{X} = x_1, ..., x_i \) and \( \mathbf{Y} = y_1, ..., y_j \). The value at \( \gamma(m, n) \) will contain the minimum-distance warping path between time series \( X \) and \( Y \). To find the minimum-distance warping path, every cell of the cost matrix must
be filled. If the minimum warping distances are already known for all slightly smaller portions of that time series that are a single data point away from lengths i and j, then the value at $\gamma(i, j)$ is the minimum distance of all possible warping paths for the time series that are one data point smaller than i and j, plus the distance between the two points $x_i$ and $y_j$. Since the warping path must either be incremented by one or stay the same along the i and j axes, the distances of the optimal warping paths one data point smaller than lengths i and j are contained in the matrix at $\gamma(i, j)$. After the entire matrix is filled, a warping path must be found from $\gamma(1, 1)$ to $\gamma(m, n)$. The warping path is actually calculated in reverse order starting at $\gamma(m, n)$. A greedy search is performed that evaluates cells to the left, down, and diagonally to the bottom-left. Whichever of these three adjacent cells has the smallest value is added to the beginning of the warping path found so far, and the search continues from that cell. The search stops when $\gamma(1, 1)$ is reached.

The expression below gives the current distance $\gamma(i, j)$ as the sum of the square of the distance currently found in the cell in the same position in the original matrix and the minimum of the cumulative distances found in the adjacent cells (diagonal, left and down) in the dynamic programming matrix, as indicated in Figure 2b:

$$\gamma(i, j) = ED^2(x_i, y_j) + \min(\gamma(i - 1, j - 1), \gamma(i - 1, j), \gamma(i, j - 1))$$

and then $DTW(X, Y) = \gamma(n, m)$.

The time and space complexity of the DTW are easy to determine. Each cell in the mxn cost matrix is filled exactly once, and each cell is filled in constant time. This yields both a time and space complexity of mxn, which is $O(n^2)$ if $m = n$. The quadratic space complexity is particularly prohibitive. A linear space-complexity implementation of the DTW algorithm is possible by only keeping the current and previous columns in memory as the cost matrix is filled from left to right (see Fig. 3). By only retaining two columns at any one time, the optimal warping distance between the two time series can be determined. Fig. 3 shows an example of a matrix for computing DTW using dynamic programming.
2.3 Other Similarity Concepts

**Definition 10** The subsequence of a time series $X_p$, denoted $(X_p)_i^j$, is a time series of length $i$ starting at position $j$ where $1 \leq i \leq n$ and $0 \leq j \leq n - 1$.

**Definition 11** Similar Time Series. Two time series $X$ and $Y$ are said to be similar if the chosen distance $d$ between them is within a user specified similarity threshold $ST$.

$$d(X, Y) \leq ST.$$  

In Table 17 we provide a list of the notations used throughout the dissertation.

**Definition 12** We define the normalized distance $\overline{d}$ between two sequences of the same length $n$, $X = (x_1, x_2, ..., x_n)$ and $Y = (y_1, y_2, ..., y_n)$ as

$$\overline{d}(X, Y) = \frac{d(X, Y)}{f(n)},$$

where $d(X, Y)$ is the standard point-to-point distance, e.g. $d \in \{ED, MD, Mink, CD\}$ and $f(n)$ is specific for each distance. This function is usually dependent of the length of the time
series and can vary for specific distances.

For example, we define the following normalized distances used in this dissertation.

**Definition 13** We define the normalized Euclidean distance \( \overline{ED} \) between two time series \( X = (x_1, x_2, ..., x_n) \) and \( Y = (y_1, y_2, ..., y_n) \) as

\[
\overline{ED}(X, Y) = \frac{ED(X, Y)}{\sqrt{n}},
\]

where \( ED(X, Y) \) is the standard Euclidean distance.

Here the chosen \( f(n) \) function is the square root instead of the length \( n \) because the mathematical expression of the ED is based on square root.

**Definition 14** We define the normalized Manhattan distance \( \overline{MD} \) between two time series \( X = (x_1, x_2, ..., x_n) \) and \( Y = (y_1, y_2, ..., y_n) \) as

\[
\overline{MD}(X, Y) = \frac{MD(X, Y)}{n},
\]

where \( MD(X, Y) \) is the standard Manhattan distance.

Some distances like Minkowski don’t need to be normalized because they are based on the max difference between pairs of coordinates, so the length of sequences has no impact.

**Definition 15** We define the normalized Minkowski Distance between two time series \( X = (x_1, x_2, ..., x_n) \) and \( Y = (y_1, y_2, ..., y_n) \) as

\[
\overline{M_{\text{ink}}}(X, Y) = M_{\text{ink}}(X, Y),
\]

where \( M_{\text{ink}}(X, Y) \) is the standard Minkowski distance.

**Definition 16** We define the normalized Dynamic Time Warping distance \( \overline{DTW} \) between two time series \( X = (x_1, x_2, ..., x_n) \) and \( Y = (y_1, y_2, ..., y_m) \) as

\[
\overline{DTW}(X, Y) = \frac{DTW(X, Y)}{2n},
\]
where again $DTW(X, Y)$ is the standard DTW distance. For simplicity of notations we consider here $m \leq n$, but the two cases are interchangeable. We divide by $2n$ in the normalization because the warping path may have length up to $2n$.

We only define here the normalized distances that will be used in this proposal, but normalized distances can be defined for any specific distance according to Def. 12 by choosing the appropriate function $f(n)$. 
3 Interactive Exploration of Time Series

In this chapter, I describe the solution for interactive exploration of time series similarity. This research effort resulted in the following publications:


3.1 Theoretical Foundation of the ONEX Framework

In this section, we first establish a formal foundation for the ONEX framework. As core, we prove a triangle inequality between ED and DTW, which is conceptually similar to the well-known triangle inequality for ED. This allows us to build a compact space of representative sequences, called the ONEX base, based on the ED distance. We then process DTW-based queries against this compact space instead of raw time series – while still guaranteeing the quality of the retrieval process.

3.1.1 ONEX Similarity Groups

Our ONEX base compactly encodes similarity relationships between all subsequences of the time series in a dataset D. As foundation, we group subsequences of the same length that are similar according to Def. 29 using the ubiquitous and inexpensive ED distance into what we call “ONEX similarity groups” and then summarize these groups by a carefully chosen representative, as further described below.

**Definition 17** Given a group S of time series $X_p$ of equal length i, then the representative $R^i_k$ of S is defined as the point-wise average of the sequences in the set S [25]. That is, $R^i_k = \text{avg}(X_p^i)$, for all $(X_p^i) \in S$.

We now introduce the notion of ONEX similarity groups that encode similarity relationships between subsequences by imposing several key requirements that, as we...
prove in Sec. 3.2, assure that these groups instead of the raw data can be safely explored through their representatives.

**Definition 18** Given the set $T$ of all possible subsequences $(X_p)_{ij}$ of the time series of dataset $D$, assume these subsequences $(X_p)_{ij} \in T$ are grouped into similarity groups with their respective representatives $R_{ik}$, such that all subsequences $(X_p)_{ij} \in T$ are in exactly one group $G_{ik}$. These similarity groups are defined to be **ONEX groups**, denoted by $G_{ik}$, if the following three properties hold:

1. all subsequences $(X_p)_{ij}$ in a group $G_{ik}$ must have the same length $i$,
2. $ED$ between any subsequence $(X_p)_{ij}$ in $G_{ik}$ and the representative $R_{ik}$ of this group $G_{ik}$ is smaller than half of the similarity threshold $ST$ used by the system, that is $ED((X_p)_{ij}, R_{ik})) \leq ST/2, \forall i, j \in [1, n]\forall p \in [1, N]$.
3. $ED$ between the subsequence $(X_p)_{ij}$ and the representative $R_{ik}$ of its group $G_{ik}$ is the smallest compared to $ED$ of $(X_p)_{ij}$ and all other representatives $R_{il}$ of the same length $i$ defined over $D$, or $ED((X_p)_{ij}, R_{ik})) \leq ED((X_p)_{ij}, R_{il})) (\forall i, j \in [1, n]) (\forall p \in [1, N]) (\forall l \in [1, g])$, where $g$ denotes the number of representatives of length $i$ and $N$ is the number of time series in the dataset $D$.

The key requirements for placing sequences into the same ONEX similarity group are two-fold. First, $ED$ of the sequences to the representative of the group must be the smallest compared to the $ED$ to any other representative, and second, it is also smaller than $ST/2$.

**Lemma 1** For any two subsequences $X$ and $Y$ belonging to the same group $G_{ik}$, with $G_{ik}$ defined in Def. 18, the $ED(X,Y)$ defined in Def. 13 is within the threshold $ST$ holds, that is, $ED(X,Y) \leq ST$, for all $X, Y \in G_{ik}$.

**Proof.** Let $X = (x_1...x_j)$ and $Y = (y_1...y_j)$ be two time series in the same similarity group $G_{ik}$ and $R = (r_1...r_j)$ be the representative of that group. According to Def. 18, we have: $ED(X, R) \leq ST/2$ and $ED(Y, R) \leq ST/2$

Using Def. 13, this means:
\[ \sqrt{\sum_{k=i}^{j} (x_k - r_k)^2} \leq ST/2 \]  \hspace{1cm} (10)

\[ \sqrt{\sum_{k=i}^{j} (y_k - r_k)^2} \leq ST/2, \]  \hspace{1cm} (11)

We want to prove that: \( ED(X,Y) \leq ST \) which means:

\[ \sqrt{\sum_{k=i}^{j} (x_k - y_k)^2} \leq ST. \]  \hspace{1cm} (12)

Squaring the two equations above, we get:

\[ ED^2(X, R) = \sum_{k=i}^{j} (x_k - r_k)^2 \leq \frac{ST^2}{4}. \]  \hspace{1cm} (13)

\[ ED^2(Y, R) = \sum_{k=i}^{j} (y_k - r_k)^2 \leq \frac{ST^2}{4}. \]  \hspace{1cm} (14)

We have:

\[ x_k - y_k = x_k - r_k + r_k - y_k = (x_k - r_k) + (r_k - y_k). \]

Using the Cauchy-Schwarz inequality \[74\], we get:

\[ (x_k - y_k)^2 \leq 2(x_k - r_k)^2 + 2(r_k - y_k)^2. \]

Then using this equation, Def. 13 and the above squared equations, we get:

\[ \sum_{k=i}^{j} (x_k - y_k)^2 \leq 2 \sum_{k=i}^{j} (x_k - r_k)^2 + 2 \sum_{k=i}^{j} (y_k - r_k)^2 \]

\[ \leq 2(ST^2)/4 + 2(ST^2)/4 = ST^2. \]  \hspace{1cm} (15)

Extracting the square root we proved that (11) is true.

As suggested in Fig. 4, we “represent” each group constructed over a data set D by only one single sequence, namely, the group’s representative. We collect the representatives for all groups over G into a collection, called the Representative Space (R-Space).
3.1 Theoretical Foundation of the ONEX Framework

![Figure 4: ONEX base intuition](image)

**Definition 19** Given a data set \( D \) and a collection of mutually exclusive groups \( \{ G^i_k \} \) covering all sequences of all lengths \( i \in \mathcal{L} \), then the set of representatives, with \( \{ R^i_k \} \) the representative of \( \{ G^i_k \} \), along with their associated sequences \( (X_p)_k^i \) in \( \{ G^i_k \} \), is called the representative space.

Lastly, we compute the pairwise Inter-Representative Distances between all pairs of representatives.

**Definition 20** Given the collection of representatives over \( D \), i.e., the R-Space, then the Inter-Representative Distance \( Dc \) between two representatives \( R^i_k \) and \( R^i_l \) of the same length \( i \) is defined by \( Dc_{kl} = ED(R^i_k, R^i_l) \).

This distance plays a central role in dealing with variable similarity thresholds, as shown in Sec. 3.2.3.

### 3.1.2 Time-Warped Solution Retrieval Based on ED-DTW Triangle Inequality

The cornerstone of our ONEX time-warped retrieval framework is this unique conceptual solution based on proving a triangle inequality between ED and DTW. We prove that the similarity between a sample sequence \( seq \) provided by the user and the representative \( rep \) of an ONEX similarity group as defined in Def. 18 “extends” to all the subsequences in that group. This empowers ONEX to perform time warped comparisons of the sample sequence over the compacted R - Space instead of the entire dataset. More specifically, if \( DTW \) between the \( seq \) and the representative \( R^i_k \) is smaller than
then we can guarantee that all sequences in that group $G^i_k$ are similar to this sequence $seq$ and that $DTW$ between $seq$ and any of these sequences is within the similarity threshold $ST$.

**Lemma 2** Given $Y' = (y_1', \ldots, y_n')$ an arbitrary sequence of length $n$ in any group as per Def. 18), with the representative of the group $Y = (y_1, \ldots, y_n)$ and a sample sequence $X = (x_1, \ldots, x_m)$, then the following is true: If $ED(Y, Y') \leq ST/2$ and $DTW(X, Y) \leq ST/2$ then we have $DTW(X, Y') \leq ST$.

**Proof. (Case: subsequences of same length).** We know from Def. 13, Def. 18 and assumptions of Lemma 1:

$$ED(Y, Y') = \sqrt{\sum_{i=1}^{n} (y_i - y_i')^2} \leq \sqrt{n \frac{ST}{2}}.$$ 

Squaring this we get:

$$ED^2(Y, Y') = \sum_{i=1}^{n} (y_i - y_i')^2 \leq n \frac{ST^2}{4}. \quad (16)$$

We define matrices $M(X, Y)$ and $M(X, Y')$ as in Section 3. Given the assumptions related to this case we know that there is a warping path $P$ in $M(X, Y)$ from $(1, 1)$ to $(n, n)$ with the DTW weight at most $2n \frac{ST}{2} = nST$. We now have to show that there is a warping path from $(1, 1)$ to $(n, n)$ in $M(X, Y')$ with weight at most $2nST$. In fact we will show that the same warping path $P$ from $M(X, Y)$ will be good. Let $P = (p_1, p_2, \ldots, p_t, \ldots, p_T)$, where $n \leq T \leq 2n - 1$, $p_1 = (1, 1)$, $p_T = (n, n)$, $p_t = (i_t, j_t)$.

Then from the assumptions of Lemma 1, we have

$$\sqrt{\sum_{t=1}^{T} (x_{i_t} - y_{i_t})^2} \leq 2n \frac{ST}{T} = nST.$$

Squaring this, we get:

$$\sum_{t=1}^{T} (x_{i_t} - y_{i_t})^2 \leq n^2 ST^2. \quad (17)$$

We want to prove that

$$\sqrt{\sum_{t=1}^{T} (x_{i_t} - y_{i_t}')^2} \leq 2nST.$$
This is the same as:

$$\sum_{t=1}^{T} (x_{it} - y'_{it})^2 \leq 4n^2 ST^2. \quad (18)$$

We have:

$$x_i - y'_i = x_i - y_i + y_i - y'_i = (x_i - y_i) + (y_i - y'_i)$$

Using the Cauchy-Schwarz inequality [74], we get:

$$(x_i - y'_i)^2 \leq 2(x_i - y_i)^2 + 2(y_i - y'_i)^2.$$  

Then using this and the above expressions, we get:

$$\sum_{t=1}^{T} (x_{it} - y'_{it})^2 \leq 2 \sum_{t=1}^{T} (x_{it} - y_{it})^2 + 2 \sum_{t=1}^{T} (y_{it} - y'_it)^2 \leq 2n^2 ST^2 + 2 \sum_{t=1}^{T} (y_{it} - y'_it)^2. \quad (19)$$

In order to estimate the second term, note that it is $\sum_{i=1}^{n} (y_i - y'_i)^2$ with some of the terms repeated. The total number of repetitions is at most $n$, since the length of the warping path is at most $2n$. Each fixed term is repeated at most $n - 1$ times. Thus from Equations (17) and (18), we have:

$$\sum_{t=1}^{T} (x_{it} - y'_{it})^2 \leq 2n^2 ST^2 + 2n \sum_{i=1}^{n} (y_i - y'_i)^2 \leq 2n^2 ST^2 + \frac{2nnST^2}{4}. \quad (20)$$

Thus Eqn (19) will be true, if we have:

$$\frac{nnST^2}{2} \leq 2n^2 ST^2.$$  

This in turn is true if $n^2 \leq 4n^2$, which is always true.

**Proof sketch (Case: subsequences of different lengths)** Let $Y$ and $Y'$ be subsequences of length $n$ where $Y$ is the representative of the group, $Y'$ an arbitrary sequence in the group and $X$ a query sequence of length $m$, with $m \leq n$. Without loss of generality we consider here the case of $m \leq n$ but the proof is very similar for $n \leq m$. In the DTW defined in Def. 16 we divide by $2n$ because the warping path may have length
up to $m + n \leq 2n$. Then the matrix $M(X, Y)$ is an $m \times n$ matrix and the warping path connects $(1, 1)$ to $(m, n)$. Other than this, the proof for sequences of different lengths and the proof for sequences of the same length are the same.

**Lemma 2** guarantees that our solution is always within $ST$ of the exact solution.

**Discussion: DTW Clustering in ONEX framework.** We now show that using DTW for ONEX similarity group formation would require us to guarantee the triangle inequality for DTW, which is still an open research problem [18], [55] due to the non-metric nature of DTW. This, along with the efficiency of ED for clustering compared to DTW, confirms our ED-DTW design choice of the framework.

Let’s assume that some clustering method were to exist that would create clusters as in our Def. 18, i.e. it is guaranteed to place all subsequences in similarity groups for which the following conditions hold:

1. $DTW(X, Y) \leq ST$, where $X$ and $Y$ are any sequences in the same group and $ST$ is the given similarity threshold.
2. $DTW(X, R) \leq ST/2$, where $R$ is the representative of a group and $X$ is any sequence in that group.

In other words, our ONEX framework would have to work now with an “adjusted” Lemma 2, where ED would be replaced by DTW. For this important foundation of our ONEX methodology to still be applicable, we would have to prove the following:

If $DTW(Y, Y') \leq ST/2$ and $DTW(X, Y) \leq ST/2$ then we have $DTW(X, Y') \leq ST$,

which unfortunately corresponds to the triangle inequality for DTW. In conclusion, using a clustering methodology based on DTW would impede our framework’s functionality, rendering its core formal foundation unproven.

### 3.2 The ONEX Base

#### 3.2.1 Strategies for ONEX Base Construction

The algorithm for building the ONEX Base, namely, for finding the groups for specific length subsequences and then computing their representatives is illustrated in Algo. 3. In order to make sure we can capture diverse temporal granularities as
described in our motivating example, as first step, we decompose the existing time series in the dataset D into subsequences of all possible lengths. We then randomize the order of subsequences of each length using the well-known RANDOMIZE-IN-PLACE method [22] to remove data-related bias (Lines 1-5). In lines 7-10, we construct the first group by randomly selecting a subsequence and designating it as the representative of this first group. Choosing the first representative randomly ensures that the groups are not biased by the order in which subsequences are supplied [9]. In lines 12-20, a new randomly chosen subsequence of the same length is compared with previous representatives. Among all representatives for which their $ED$ is smaller than $ST/2$, the one group with the minimum distance $ED$ is chosen. Then the subsequence is placed into this group. Otherwise it will be placed in a new group and designated as the representative of the new group. We repeat this until all subsequences of each specific length are placed into a similarity group. Lastly, the representative for each group is determined. The final result is a panorama of all groups $G^i_k$ and their representatives $R^i_k$ for all possible lengths.

**Algorithm 1:** Construction of Similarity Groups and Representatives for Subsequences of Equal Length

| Input: Similarity threshold $ST$, TimeSeries $\{X\}$, Length $L$ |
| Output: Representatives $\{R\}$, Groups $\{G\}$ |
| begin |
| $\{G\} = \emptyset$, $\{R\} = \emptyset$ |
| Randomized_IN_PLACE($X$); |
| $\{X\} =$ all subsequences of length $L$ |
| minSM=0, mink=0 |
| for $X_p \in X$ do |
| if ($G = \emptyset$) then |
| $G \leftarrow G_1$ |
| $G_k \leftarrow X_p$ |
| $R \leftarrow X_p$ |
| else |
| for $k = 1$ to Representative.count do |
| minSM=closest Representative distance |
| mink=Representative index |
| if (minSM $\leq (\sqrt{L} \times ST/2)$) then |
| $G_{mink} \leftarrow X_p$ |
| update $R_{mink}$ |
| else |
| $G \leftarrow G_{K+1}$ //new Group |
| $R_{k+1} \leftarrow X_p$ |

The complexity of R-Space construction is $O(nl^2g)$ where $l$ is the number of distinct lengths that each time series is decomposed into, $g$ the total number of groups and
the number of time series in the dataset. Typically, \( n \) is much larger than \( l \). Certainly
\( l \) does not tend to go towards infinity since our initial time series are of a fixed length,
and we furthermore divide them into smaller numbers. This is in contrast to \( n \) which
tends to grow with the size of the dataset. Thus, we can treat \( l \) as a constant in regards
to \( n \). The complexity now can be summarized as \( O(n g) \). Regarding \( g \), let’s examine
the following probabilistic argument. Suppose we have \( k \) groups at some point and a
new time series \( X \) comes in. There are \( (k+1) \) events that can happen, \( X \) either belongs
to the one of the \( k \) groups or \( X \) starts a new group. Let’s make the assumption that
these events are all equally likely. Then the probability that \( X \) starts a new group is
\( \frac{1}{k+1} \). How many expected trials do we need until we get a new group? This is
a geometric distribution with expected value \( \frac{1}{p} \) so \( \frac{1}{1/(k+1)} = k+1 \) [67]. Thus
\[ \sum_{k=1}^{g} (k+1) = n; \] the expected number of groups is \( \sqrt{n} \). Then the complexity becomes
\( O(n^{3/2}) \), which is much better than \( O(n^3) \).

3.2.2 Discussion Of ONEX Base

ONEX Base Construction. While above we have introduced a viable and robust solu-
tion for ONEX base construction, the question arises if alternate clustering methods
based on ED could equally be utilized in the context of our framework. The key ob-
servation here is that any clustering solution we employ must observe our core group
requirements, namely, they must produce clusters with a maximum diameter equal to
\( ST \) and have centers (our representatives) whose ED to any sequence in the group is less
than \( ST/2 \). While the well-known \texttt{kmeans} algorithm appears to be a contender on first
sight, we note that there are key differences: (1) \texttt{K-means} must know the fixed number
of clusters in advance, while we give a lower bound to the number of groups, but we
expect it to grow. (2) \texttt{K-means} is batch oriented, while Algorithm 1, after a small setup
batch mode, is online. In a sense, Algorithm 1 is similar with online nearest neighbor
clustering or the on-line k-center problem\(^{10} \).

ONEX Base Maintenance under Updates. Next, we sketch strategies for accommodat-
ing insertions and deletions of time series. If a time series is deleted, the groups can be
updated to remove any sequences with that specific time series id. These groups don’t

\(^{10}\)http://cseweb.ucsd.edu/~dasgupta/291geom/streaming.pdf
need to be re-constructed. However, the representatives (and their envelopes) have to be re-computed according to the sequences that remain in the group. We keep the locations of all objects in an index, thus if we delete an object, we can find it in constant time and set it to NULL. In short, deletions, so long as they are relatively rare (say less than 10% of all objects), have almost no cost, and almost no effect on speed or accuracy. If a time series is inserted into the dataset, some groups have to be updated. When possible we use strategies that avoid re-construction of the groups. One possibility is to use the original ONEX ED based methodology to place the subsequences of the new time series into groups. The groups into which the new subsequences are placed are updated to include the new sequence, and their representatives are recomputed. Another option is to use each subsequence of the inserted time series as a query sample and to find its best match representative using the ONEX approach (See Section 5.5.2). Then we insert the subsequence into this group and recompute the representative.

3.2.3 Handling Multiple Similarity Parameters

As described above, our R-Space is constructed using a specific user-supplied ST. However, analysts might want to use different thresholds $ST'$ to customize their notion of similarity based on the application. This means they should be able to consider as similar sequences whose distance to each other is within a specific value. We thus extend the ONEX base to incorporate different values for ST, as desired by analysts. More precisely, during the construction of groups we identify similarity threshold values corresponding to specific lengths for which the groups of that length change significantly and incorporate them into a Similarity Parameter Space (SP-Space).

**Definition 21** The SP-Space is a conceptual space with length $L$ and similarity threshold $ST$ as its dimensions.

We build an SP-Space for each specific length based on establishing relevant values for similarity thresholds. These could be $ST_{half}$ and $ST_{final}$ indicating when a half and respectively all precomputed groups of subsequences of a specific length merge – resulting in major differences in the result set. Alternatively, the ranges of relative similarity thresholds might be “customized” by imposing a certain granularity of the
ranges, i.e., by splitting the overall range in 10 equal sized ranges. In general, two groups merge for a new similarity threshold $ST'$, if $ST' \geq ST + Dc$ where $Dc$ is defined in Def. 20. In Fig. 5 we showcase three specific lengths and the thresholds for which half and respectively all the groups of that length merge. For example, for length $i$ we have $ST_{\text{half}} = 0.5$ and $ST_{\text{final}} = 0.78$. This means that for this specific length half of the groups merge if the analyst selects a new threshold $ST' = 0.5$ and all groups merge if $ST' = 0.78$ or higher.

These “local” critical thresholds specific to each length can be combined to find the global similarity threshold values for which half and respectively all the groups across all lengths merge. We compute these “global” critical thresholds $ST_{\text{half}}$ and $ST_{\text{final}}$ by selecting the maximum of the local values of each length as depicted with dashed lines in Fig. 5. Choosing the maximum of all “local” $ST_{\text{final}}$ values to be the “global” $ST_{\text{final}}$ value ensures that all the groups of that length merge for a threshold $ST' \geq ST_{\text{final}}$.

We use the local and global threshold values to offer guidance in exploring similarity as shown later in Sec. 3.3.1. We now introduce the concept of similarity degree for our system:
3.2 The ONEX Base

S=Strict, where \( ST \leq ST_{half} \)
M=Medium, where \( ST \in [ST_{half}, ST_{final}] \)
L=Loose, where \( ST \geq ST_{final} \).

For the example illustrated in Fig. 5, if an analyst asks for a recommendation for similarity thresholds ranges for “Strict” similarity, the recommended values are in the range \([0, 0.6]\). Any value chosen by the analyst in this interval returns results with “strict similarity”. Lower values chosen in this interval lead to “stricter” similarity translated in smaller distances between the similar sequences.

3.2.4 Storage and Indexing of ONEX Base

ONEX uses in-memory structures that achieve indexing in linear or constant time. We first construct a **Global Time Index** (GTI) as an array indexed by length for quick retrieval of the set of groups for each specific length. Each entry in GTI is composed of:

- a vector \( V_i(k) \) containing the identifiers \( k \) for the groups \( G^i_k \) of a specific length \( i \). They are used to retrieve the groups associated with that length.
- a two-dimensional array \( D_{i}(k, j) \) containing the pairwise Inter-Representative distances \( D_c \) between groups \( G^i_k \) and \( G^i_j \) of length \( i \). They help refine the results for different user-provided similarity thresholds. (Sec. 5.5.2)
- an array \( S_i(k, sum_k) \) containing the representatives’ identifiers and their associated sums of the pairwise Inter-Representative distances \( D_c \) for length \( i \). We sort this array based on the sum of the distances to optimize the online search of representatives based on a given sample sequence. (Sec. 3.3.3).
- similarity threshold values \( ST_{half} \) and \( ST_{final} \) for length each \( i \) (Sec. 3.2.3) for local or length based parameter recommendations.

For each group \( G^i_k \), we maintain a **Local Sequence Index** (LSI) with the following elements:

- a two-dimensional array \( ED_k(m, ED_m) \) containing the subsequence identifiers \( m \) in the group and their \( ED \) to the representative of the group. Each subsequence \( m \) in the group is a subsequence \( (X_p)^i_j \) further identified by the time series identifier \( p \).
and its starting position \( j \). We sort this array based on the \( ED \) to the representative to optimize the online retrieval of sequences inside a “selected” group.

- vector containing the representative \( R^i_k \) of the group \( G^i_k \).
- array containing the envelopes around each representative \( R^i_k \) using \( LB(Keogh) \) [65], which is a well-known lower bound for pruning unpromising candidates when performing time warped comparisons with a sample sequence.

### 3.3 ONEX Online Query Processor

#### 3.3.1 ONEX Queries Classes

**User-driven** analytics in ONEX allow analysts to control the exploration by providing a target sample sequence \( \text{seq} \). This user-supplied sample sequence may or may not exist in the dataset as we show in our use cases below.

\[
\begin{align*}
\text{Q: OUTPUT } X[k] \\
\text{FROM D} \\
\text{WHERE Sim} \leq \text{ST, } [\text{seq} = q \mid \text{NULL}] \\
\text{MATCH} = \text{Exact}(L) \mid \text{Any}
\end{align*}
\]

The above general ONEX query syntax uses notations described in Table 18. The **data-driven** analytics provide insights into the dataset without the user giving a sample sequence as a target match.

**Class I: Similarity queries** fall into the user-driven query class. They return time series most similar to a user-supplied sample sequence \( \text{seq} \). If “Match=Any” then sequences

<table>
<thead>
<tr>
<th>Clause</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MATCH</td>
<td>Exact(( L )) refers to a specific length ( L ). Any refers to any length.</td>
</tr>
<tr>
<td>( X_p )</td>
<td>Subsequence of the time series ( p )</td>
</tr>
<tr>
<td>( seq )</td>
<td>NULL means no sample sequence is given. ( X_p, q ) – samples sequences provided by user</td>
</tr>
<tr>
<td>( Sim )</td>
<td>Similarity Distance</td>
</tr>
<tr>
<td>( ST )</td>
<td>Similarity Threshold</td>
</tr>
<tr>
<td>( L )</td>
<td>Length</td>
</tr>
</tbody>
</table>
of every possible length are searched, otherwise only sequences of length l indicated by the sample are considered.

\[
\text{Q1: OUTPUT } X_k \\
\text{FROM D} \\
\text{WHERE Sim <= min| ST, seq = q} \\
\text{MATCH = Exact(L)|Any}
\]

**Use Case:** A financial analyst may want to retrieve the stock similar to the stock fluctuations of the Apple Stock for a specific time period. If the length is not specified, a similar stock of any length is retrieved. This illustrates the case when the sample sequence is a sequence present in the dataset. Alternatively, an analyst can “design” the desired stock fluctuation of interest and search for the best match for that sample sequence in the dataset. Such sequence is likely not to exist in the dataset, but rather the closest match sequence will be retrieved. Alternatively, referring back to our motivating example, similar economic indicators, like economic growth, can be found over specific time intervals or any time intervals.

**Class II: Seasonal similarity** queries allow the analyst to gain insights into the dataset by identifying similarity patterns. In the user-driven class, for example, queries like Q2, given a sample time series, return the “recurring” similarity by retrieving all similar subsequences of a specific length that belong to this sample time series. In the data-driven class, the analyst explores the dataset without a sample query, by just providing a specific length. The result consists of groups of similar sequences of the specified length.

\[
\text{Q2: OUTPUT SeasonalSim } \{X_p\} \\
\text{FROM D} \\
\text{WHERE seq=\ X_p\ | NULL} \\
\text{MATCH = Exact(L)}
\]

**Use Case:** For the user-driven class, in the Stock Market application, an analyst can find all 30 days long subsequences of the Apple stock having similar prices. In the data-driven class an analyst can retrieve all the stocks whose prices were similar to
each other over any 30 days periods.

Class III: Similarity threshold recommendations help the user better understand the dataset by transforming the intuition of “loose, medium or strict similarity” (introduced in Sec. 3.2.3) into actual parameter values. These terms are connected to the interpretations of similarity based on user and application domains.

<table>
<thead>
<tr>
<th>Q3 OUTPUT ST</th>
</tr>
</thead>
<tbody>
<tr>
<td>FROM D</td>
</tr>
<tr>
<td>WHERE simDegree=NULL</td>
</tr>
<tr>
<td>MATCH = Exact(L)</td>
</tr>
</tbody>
</table>

Use case: Sometimes a user submits mining requests using different similarity thresholds, yet ends up receiving the same or very similar results. In such situations Q3 saves time and effort by allowing the user to make use of similarity thresholds that will cause a difference in the output. For example, if a user is not sure what value to use for a similarity threshold, then when prompted for the similarity strength they are interested in, they can enter “S” for strict similarity. The system will return a range of similarity thresholds for which the subsequences returned have very small similarity distances. We later explain in Sec. 3.3.2 how varying similarity thresholds can be managed without re-constructing the entire knowledge base.

3.3.2 Query Processing over ONEX Base

Based upon our solid formal foundation (Sec. 3.1), our query processor (Algo. 2) now applies time-warped strategies on the compact ED-based ONEX base using indexing strategies introduced in Sec. 3.2.4.

ONEX handles similarity queries by first exploring the R-Space using a three-step process. In Lines 4-7, we use GTI (Sec. 3.2.4) to retrieve groups of specific lengths. Then the processor uses LSI (Sec. 3.2.4) to find the representative with the minimum DTW to the sample sequence, the so-called the best matching representative. Third, we use
the LSI to find the best match sequence inside the selected group.

**Processing costs** for this class include: Cost(getGroups), Cost(compareRep) and Cost(getKSim). The complexity of Cost(getGroups) is constant for each specific length. The Cost(compareRep) is $O(g)$ where $g$ denotes the number of groups. The cost for finding the best matching representative is Cost(getKSim) which is $O(m)$ where $m$ is the number of subsequences in the best matching group. Thus the total complexity is $O(g) + O(m)$.

For **seasonal similarity queries** we distinguish two scenarios. In the first scenario (Lines 19-21), the analyst provides a sample time series $seq$ and a specific length. ONEX explores the R-Space first using GTI (Sec. 3.2.4) to retrieve the groups of that specific length. After the groups are retrieved, they are explored using LSI (Sec. 3.2.4) based on the identifier of the sample time series. ONEX returns only the sequences in each group having the same sequence identifier as the sample $seq$. In the second scenario (Lines 22-24), if no sample sequence is provided, ONEX retrieves the groups of similar sequences of the length specified by the analyst. We retrieve the groups using GTI and then we find the sequences in each group using LSI.

**The processing costs** for this class include: Cost(getGroups) and Cost(getSubsequences). As shown in Sec. 3.2.4, we maintain the sequence identifiers for each group. So we can retrieve all subsequences in $O(1)$. The complexity of Cost(getGroups) is constant for each specific length. The cost for finding sequences Cost(getSubsequences) is $O(n)$ where $n$ is the number of subsequences in the group. The complexity for scenario two is $O(ng)$, where $n$ is the number of time series in each group and $g$ is the number of groups. The complexity for scenario one, seasonal similarity using a sample time series, is $O(g)$. In both scenarios there is no additional storage overhead because all needed information is already contained in the index structures.

The complexity of the **similarity threshold recommendations** is highly dependent on the dataset, as the recommendations are based on the computations of the local and global similarity thresholds. We reduce this computation by re-using the pre-computed results for a specific $ST$ and adapt them to newly provided $ST$: thresholds. For the construction of similarity groups (Sec. 3.1.1), we use a specific threshold value $ST$. In
Algorithm 2: Online Time Series Exploration

2A: Similarity Queries
Input: Query $Q, k, L, \text{match}$
Output: Sequences $\{X\}$

begin

$\{X\} \leftarrow \emptyset$

switch $\text{match}$ do

case exact do

$\{G_{i}^{k}\} = \text{GTI.getgroups(}L\}$

$\text{minGp} = \text{LSI.compareRep}(Q, G_{i}^{k}.\text{Representatives})$;

$X \leftarrow \text{LSI.getKSim(}\text{minGp})$;

case Any do

for each $L$ do

$\{G_{i}^{k}\} = \text{GTI.getgroups(}L\})$

$\text{minGp} = \text{LSI.compareRep}(Q, G_{i}^{k}.\text{Representatives})$;

$\{X\} \leftarrow \text{LSI.getKSim(}\text{minGp})$;

end

2B Seasonal Similarity Queries
Input: $\text{queryType}, \text{TimeSeries}(X_{p}, X_{q}), \text{Length} L$
Output: Subsequences $X$

begin

$\{G_{i}^{k}\} = \text{GTI.getgroups(}L\}$

for each $G \in G_{i}^{k}$ do

switch $\text{queryType}$ do

case Single do

if $X_{p} \in G$ then

$\{X\} \leftarrow \text{LSI.getSubsequences}(X_{p})$

case NULL do

for each $X_{p} \in G \& X_{q} \in G$ do

$\{X\} \leftarrow \text{LSI.getSubsequences}(X_{p}, X_{q})$

end

2C Varying Similarity thresholds
Input: Similarity threshold $ST^{'}$, Inter − Representativedistance $Dc$
Output: Groups $\{G^{'}\}$

begin

$G^{'} \leftarrow \emptyset$

if $ST^{'} == ST$ then

$G = G^{'}$ \hspace{1em} //return precomputed computed groups

else if $ST^{'} < ST$ then

split groups $G$

else

if $ST^{'} < Dc$ then

merge pair of groups with this condition

else if $ST^{'} \geq Dc$ then

merge groups with this condition

else if $ST^{'} < ST$ then

$G^{'} = G^{'}$ \hspace{1em} //return precomputed computed groups

end
practice the optimal threshold varies from dataset to dataset. Such threshold offers the
best trade-off between the accuracy of the results and the time response. We discuss
later in our experimental evaluation how to empirically find these thresholds and we
use them for our experiments. However, analysts might be interested in using a differ-
ent domain-specific $ST'$ in their queries. In such situations, the R-Space does not have
to be re-constructed from scratch. Instead, we provide an efficient strategy to “refine”
the similarity groups based on pre-computed information. This strategy leverages the
indices. We distinguish the following scenarios:

1. $ST' = ST$. In this case, the ST matches the one provided by the analyst. So the
precomputed groups are used “as is” to find the best match.

2. $ST' < ST$. In this case, the precomputed groups contain similar sequences, but
they must be refined. The intuition behind the refinement is that sequences that are
similar for the threshold ST remain similar for a smaller threshold ST'. Groups are split
in “smaller” similarity groups so no possible answer is missed. Assuming without loss
of generality $ST/ST' = k$ where $k \in N$, each precomputed group is now split into
$k$ groups. We use the same methodology for constructing these smaller groups as we
originally used to construct the groups for specific lengths (Sec. 3.1.1).

3. $ST' > ST$. This is the more complex case, as some pairs of groups may merge,
depending on the value of the Inter-Representative Distances ($Dc$) defined in Sec. 3.1.1.

3.1. $ST' < Dc$. In this case, all precomputed pairs of groups for which this condi-
tion is true are used to find the best match sequence. The intuition is that groups whose
representatives are farther from each other by more than the given similarity threshold
$ST'$ can’t merge. The rest of the groups will be processed as follows:

3.2. $ST' \geq Dc$. we distinguish two scenarios:

3.2a. $ST' - ST \geq Dc$. The pairs of precomputed groups for which this condition
is true are merged and used to find the best match seq. The intuition is that a higher
similarity threshold allows more sequences to be similar. The remaining groups are re-
turned without any action. If the case that the same group could be merged with more
than one group arises, we randomly choose a pair of qualifying groups and perform the
merge. We then compute the Inter-Representative distance between the newly formed
group and the remaining groups to determine if any additional mergers are needed. This process repeats in a cascading manner as long as the above condition holds.

3.2b. $ST' - ST < Dc$. In this case the precomputed groups are used “as is” to find the best match sequence.

Discussion of range queries. ONEX could be expanded to include range queries. For example, adding a time index to LSI for each group would allow quick retrieval of all sequences in the group associated with a specific time interval. Another option could be to use time-interval clustering instead of length-based clustering. This would enable ONEX to directly support range-queries. Our Lemma 2 holds for time intervals. The proof is very similar with the proof for length-based clustering.

3.3.3 Optimizations for Processing Queries.

To efficiently retrieve the best match sequence to a given sample sequence $seq$, we adopt optimizations for retrieving the best group representative and the best-match sequence within this group. The optimizations include the early abandoning of DTW [65] and stop the calculation if the lower bound exceeds the best-so-far distance, and pruning of the representatives. We combine these with the re-ordering of the early abandoning. Finally, we take advantage of the cascading lower bounds to prune unpromising candidates, a method also popular with DTW [26]. In addition, we develop the following optimizing strategies specific to ONEX:

• For a given sample sequence of length $L$, we start the search for the best match representative with the ones of the same length as the query. If we don’t find the “best match” representative for this length (the one with $\overline{DTW}$ to $seq$ within $ST/2$), we continue by searching the representatives in decreasing order of their length until we reach the smallest length, followed by the search in increasing order of the length.

• Our strategy to find the “best match” representative of a specific length is optimized as well. We use the ordered array $S_i(k, sum_k)$ in GTI (Sec. 3.2.4) containing the sums of pairwise Inter-Representative distances. We find the “median representative” or the one whose sum is “in the middle” of this list and start our search
with this representative. We follow up by alternately checking the closest representatives to its left and then right in the index, until we reach the representative with the min and respectively the max sum.

- The search inside the particular group with the best match representative could be optimized as follows: using the ordered array $ED_k(p, ED_p)$ containing the $ED$ of the sequences in the group to the representatives (Sec. 3.2.4), we search for the “best match” sequence to $seq$. This is the sequence in the group whose $ED$ to the representative has the closest value to the $DTW$ between the $seq$ and the representative. We follow up if needed by checking the neighboring sequences with smaller and respectively larger distances alternatively until we find the best match sequence.
Table 3: Datasets statistics

<table>
<thead>
<tr>
<th></th>
<th>Italy</th>
<th>Power</th>
<th>ECG</th>
<th>Face</th>
<th>Wafer</th>
<th>Symbols</th>
<th>Two Patterns</th>
<th>Star Light Curves</th>
</tr>
</thead>
<tbody>
<tr>
<td>No.</td>
<td>67</td>
<td>200</td>
<td>560</td>
<td>1000</td>
<td>995</td>
<td>4000</td>
<td>9236</td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>24</td>
<td>96</td>
<td>131</td>
<td>152</td>
<td>398</td>
<td>129</td>
<td>1024</td>
<td></td>
</tr>
</tbody>
</table>

3.4 Experimental Evaluation of ONEX

3.4.1 Experimental Setup

Experiments are conducted on a Windows machine with 3.35 GHz Intel Core i5 processor and 64GB GB of RAM. **Alternative State-of-the-Art Techniques.** We compare ONEX with three methods: the **Standard DTW** which is an exact brute-force method computing all pairwise distances and guaranteeing the best match, **PAA** (Piecewise Aggregate Approximation) [45] which finds an approximate solution by reducing the dimensionality of the data using an average approximation, and **Trillion**[65] which uses Lower Bounds and early abandoning steps to retrieve similar sequences. Trillion exhibits the fastest known-to-date response time. Their experiments show faster times than EBSM [6] for similarity queries [65] and many other competitors. For this and other reasons detailed in the Related Work section, we do not compare with EBSM, but rather with Trillion. The ONEX system and alternate methods PAA [45] and the standard DTW are implemented in C++ using Qt Creator with Clang 64-bit compiler. Trillion code is downloaded from the UCRsuite\(^\text{11}\).

**Datasets:** We select our datasets from the UCR time-series collection\(^\text{12}\), the largest collection of public datasets. Statistics of our datasets can be found in our Tech Report. We display their statistics (No. is the number of time series and L is the length) in Table 3. To make meaningful comparisons between time series, they need to be normalized [65]. We normalize each sequence based on the maximum (max) and minimum (min) values in each dataset. For any sequence \( X = (x_1...x_n) \), we compute the normalized values for each point \( x_i \) as \( \frac{x_i - \min}{\max - \min} \).

**Performance Statistics.** Our experimental evaluation is geared towards illustrating that ONEX is an interactive system offering results within less than one second for

\(^{11}\) [www.cs.ucr.edu/~eamonn/UCRsuite.html](http://www.cs.ucr.edu/~eamonn/UCRsuite.html)

\(^{12}\) [www.cs.ucr.edu/~eamonn/time_series_data](http://www.cs.ucr.edu/~eamonn/time_series_data)
3.5 Evaluation of Query Processing Time and Accuracy

3.5.1 Similarity Queries

This class retrieves the sequence closest to a given query sample $seq$. Standard DTW compares the query sample with all sequences in the dataset. PAA compares the query sample with the sequences in the dimensionality-reduced dataset. Trillion computes the lower bound of the query sample first and then uses the early abandoning optimization to find the best match of the same length as the query.

In our experiment we vary the length of the query sequence to cover a wide range of lengths. We compute the average response time for each system using 20 queries of different lengths chosen to cover a wide range from the smallest to the largest length. Further, our aim is to experiment both with query sequences that are present and sequences that might not be present in the dataset. As our use cases motivate, it is important to find the best match for a query, regardless if the sequence is present or not in the dataset. In both cases we look to find the best match, namely the solution with the closest DTW to the query sequence. First we randomly select 10 subsequences of different lengths from each dataset and “promote” them to become query sequences. This is our query “in the dataset” part of the experiment. Then we adopt the methodology proposed in [28], where a random subsequence is chosen from each dataset to act as the query and is taken out from that dataset instead. We do this for 10 different subsequences. This is our query “outside of the dataset” part of the experiment. For each of them we run each individual query in the specific dataset 5 times and average the time response per query. We then compute the average time response for the 20 queries for each dataset.
As shown in Fig. 6, ONEX consistently outperforms both Standard DTW and PAA by several orders of magnitude and exhibits better running times than Trillion. The left chart in Fig. 6 displays values multiplied by 100 on a logarithmic scale, while the right chart in Fig. 6 zooms into the comparison of ONEX and Trillion. Although for small datasets ONEX and Trillion have fairly close response times, as the dataset size increases ONEX becomes on average 1.8 times faster than Trillion.

Furthermore, ONEX is highly accurate returning the best match for a query, regardless of the length of the match. Since Trillion only returns the best match of the same length as the query sequence, we designed an additional experiment by adjusting ONEX to only search for the best match of the same length as the query. We found that in this restricted situation the ONEX time response further improves, becoming on average 3.8 times faster than Trillion. The results for this particular experiment are displayed in Table 4, where ONEX-S denotes our ONEX returning the solution with the same length as the query similar to Trillion.
Next we evaluate scalability i.e., the impact of increasing the number of time series on the running time for the StarLightCurves dataset. We randomly take subsets containing time series of length 100. The number of time series varies between 1000 and 5000 in increments of 1000. We employ the same methodology described above using query sequences both present and not present in the dataset. ONEX and Trillion’s running times are close in magnitude and thus superimposed in Fig. 7. As depicted on the chart on the left in Fig. 7, the running times for Standard DTW and PAA increase drastically as the number of time series increases, while the time increase for ONEX and Trillion “seems” nearly constant. In reality, the chart on the right in Fig. 7 the zoomed detail shows that the Trillion time response is up to 4 times slower than ONEX.

**Solution Accuracy.** We measure the accuracy of the solution as follows: we retrieve the
DTW between the solution and the query sequence for ONEX, PAA and Trillion. We compute the error in retrieving each individual solution for each system as the difference between the distance computed by that system and the exact solution as provided by the brute force Standard DTW. We take the average of the error for each individual system and compute the accuracy as \((1 - \text{average(error)}) \times 100\). We use the same 20 queries as in the above experiment for similarity queries. The average accuracy for each system is displayed in Table 6. Since the brute-force always retrieves the best match possible and it is used as “accurate”, we omit its results from the table.

As shown in Table 6, the ONEX accuracy is superior by an average of 19.5% to Trillion and 2% to PAA. While PAA is closer in accuracy with ONEX, the difference in the time response renders PAA impractical, that is, ONEX is 3 to 4 orders of magnitude faster than PAA. It is interesting to observe that Trillion has a high accuracy if the query sequence is “in the dataset”, as it is the case for the first 10 queries used. That is due to the fact that Trillion, as described by its authors, performs an “exact search”. However, the accuracy drops significantly for queries that do not exist in the dataset, i.e., the last 10 queries used. As before we also conduct a separate experiment restricting ONEX to retrieve the solution of the same length as the query sequence. In this experiment shown in Table 5 ONEX shows increased accuracy by an average of 12.6% compared to Trillion.
3.5.2 Seasonal Similarity Queries

This class returns the similar subsequences of a specific length that belong to a sample time series. Neither Standard DTW or PAA have been designed to search for this kind of similarity, thus we omit them from this experiment. Trillion is omitted because its optimizations solve the problem of finding only one closest sequence of the same length as the sample and cannot be applied to find subsequences whose similarity recurs. Since the length is specified by the analyst, we cover a wide range of lengths from the smallest to the largest. We randomly select 5 time series from each dataset and we use 5 different lengths for each sample. For each sample time series, we run 5 times the seasonal similarity query for each chosen specific length and compute the average response time per length. We then average those response times for each sample time series. We repeat the experiment for each dataset and report the average running times in Fig. 10. We select 5 random lengths for each dataset and retrieve the groups with similar sequences of that length. We run 5 times for each length and compute the average time for retrieving the groups for that specific length. The average response times are reported in Fig. 10.

For a better intuition of seasonal similarity queries we also provide a visual result using one sample time series in Fig. 11. This shows that for the 13th time series sample in the ItalyPower dataset, there are two similar subsequences of length 5, one starting at position 11 and the other one starting at position 15. In this figure we intentionally “shifted” the sample time series for better display, so they don’t overlap with the other
two subsequences.

![Figure 11: Visual example of seasonal similarity](image)

3.6 Evaluation of Preprocessing Performance

Since Standard DTW, PAA and Trillion don’t involve a preprocessing phase, we display the preprocessing times and the size of the ONEX base for all datasets and for varying similarity thresholds. Fig. 8 displays the ONEX **offline construction time** for our datasets while varying the similarity threshold $ST$. As expected, for low similarity thresholds the construction time is higher because many groups are created. As the similarity threshold increases, fewer groups are created and thus more subsequences are grouped together. After a certain threshold the construction time remains constant. In Fig. 6 we display the **size of the pregenerated information** in ONEX in terms of number of representatives for varying similarity thresholds.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Representatives</th>
<th>Subsequences</th>
<th>Size in MB</th>
</tr>
</thead>
<tbody>
<tr>
<td>ItalyPower</td>
<td>1228</td>
<td>18492</td>
<td>1.14</td>
</tr>
<tr>
<td>ECG</td>
<td>3532</td>
<td>931200</td>
<td>21.53</td>
</tr>
<tr>
<td>Face</td>
<td>4896</td>
<td>4768480</td>
<td>36.75</td>
</tr>
<tr>
<td>Wafer</td>
<td>3489</td>
<td>11476000</td>
<td>183.02</td>
</tr>
<tr>
<td>Symbols</td>
<td>3424</td>
<td>78660985</td>
<td>1210.32</td>
</tr>
<tr>
<td>Two Pattern</td>
<td>3961</td>
<td>33024000</td>
<td>513.41</td>
</tr>
</tbody>
</table>

As observed in the four charts in Fig. 12, each dataset has a particular similarity threshold for which the size of the pregenerated information and the construction time are best “balanced”. We use this in combination with the observation of the best “trade-
off” between accuracy and time to choose the most appropriate ST for specific datasets. For example, for most of our datasets these similarity thresholds are around 0.2. We indeed used these thresholds for our experiments reported in this paper. Due to space constraints we only showcase the results for four datasets.

![Accuracy vs Running Time](image1)

![Accuracy vs Running Time](image2)

Figure 12: Tradeoff accuracy vs time varying ST for ItalyPower, ECG, Wafer, Face datasets in clockwise order

### 3.7 ONEX Visual Analytics

We use ONEX on real world economic and social data sets to illustrate how our web interface enables analysts to interactively explore similarity and its recurring patterns. We display in Fig. 13 the way the web interface is integrated with the ONEX framework to enable analysts to get insights into real time series collections through visual analytics. The pre-processing step (top) encodes similarity relationships between time series into “ONEX similarity groups” using Euclidean Distance. Exploration of these groups is achieved by DTW-empowered operations via the query processor (middle). The middle of the figure indicates the diverse exploratory operations performed by ONEX which include retrieving of the best match for a given sample sequence, finding repeating patterns of specific time series or seasonal similarity, and showing the
changes in the similarity between sequences for varying parameters. Lastly, the visual analytics interface, depicted at the bottom, enables the user to view, analyze and interpret information for interactive time series exploration. An array of complementary visualization techniques from stacked lines charts to connected scatter plots compose our ONEX web interface. Visualizations are critical for interactive time series analytics as they allow for intuitive interactions by analysts with large data sets. For the robust alignment of sequences of different lengths or alignments, a warping path constructed by ONEX corresponds to a set of indices marking which points are most similar, including possible multiple matchings, contrary to the use of point-wise distances. Our ONEX views uniquely display these “warped” points to highlight the shape matching. We showcase here an example of two ECG time series whose matched points are connected to better display the warping of the sequences based on their shapes and time misalignment. Our Multiple Lines Charts display dotted lines between corresponding points of the sequences highlighting the role of the time-warped matching. Radial Plots compact the time series to a radial display that allows analysts to evaluate how close the shapes are aligned. Similarly, the Connected Scatter Plots showcase the ordering of a sequence by connecting consecutive points.
Analysts can interact with ONEX via this intuitive visual web interface to understand how it assists analysts in addressing complex societal and economical questions, such as the ones described in our motivating examples. We showcase here the use of real datasets from diverse domains such as economic, census and tax datasets from MATTERS\textsuperscript{13} and a power usage dataset ElectricityLoad\textsuperscript{14}. Our interface empowers analysts to draw insights with ease from these collections, as described below.

**Data Loading into ONEX.** With a click of a button, analysts can load new data sets

\textsuperscript{13}http://matters.mhtc.org/
\textsuperscript{14}www.cs.ucr.edu/eamonn/time_series_data/
into ONEX. Loading a new dataset, such as the MATTERS GrowthRate, triggers the preprocessing of this data at the server side and its loading into the respective ONEX Base. Thereafter, the ONEX server provides near real-time responsiveness to the analyst exploring the data via a client-server architecture.

**Making Sense of Overall Time Series Trends.** To offer an overview of the data, the *Overview Pane* (Fig. 15 left top) displays the representatives of the similarity groups, color-coded such that the color intensity increases proportional with the cardinality of sequences in the group. This gives a quick sense of the typical patterns within the data set as well as the overall data distribution. Each representative is shown as a small graph that captures the general shape of the group. This supports analysts in finding the states with similar growth rates. Upon drill-down, analysts can scroll through the states in the *Query Selection Pane*, each visualized by its name and a small line graph displaying the growth rate over the last 6 years. In Fig. 15 left bottom, MA is selected from this group.

**Honing in On Specific Temporal Trends.** The *Similarity View* shown in Fig. 15 on the right assists the analyst in finding states with a similar economic growth rate to that of MA, while the *Seasonal View* in Fig. 18 enhances the understanding of a specific time series by highlighting repeating patterns. The *Query Preview Pane* displays the chosen sample query in more detail. Brushing the second half of the graph will focus the attention on the recent trends in MA. As the first preview graph is brushed, the upper chart is updated to show the selected subsequence in more detail.

**Highlighting Time-Warped Shape Matching.** When the analyst performs a similarity search, the best match sequence in the dataset is displayed along with the sample query subsequence in the *Results Pane* (Fig. 15 top right). The default “multiple lines” chart displays both time series on a single graph. The “matched points” are connected with dotted lines helping the analyst get a better intuition of how similar the time series shapes are and their relative warping.

**Contrasting Trends Across Multiple Linked Perspectives.** Different visualizations illustrate different aspects of similarity. For example, to get a richer understanding of the similarity between MA and ARK, the analyst can switch to different visuals by
selecting the mode via the right menu bar of the Results Pane.

Figure 16: Seasonal view displaying Patterns in the power usage dataset

Figure 17: Visualizing MATTERS tech employment

The same pair of time series can now be viewed in a compacted Radial Chart (Fig. 16). This view allows a consistent compression of the data, providing the analyst with alternative views to compare sequences. Further, in the Connected Scatter Plot (Fig. 17), the shape is close to a 45 degree angle. This indicates that the match is extremely close – when a point in such plot lies on the diagonal, it has the exact same value in both series. This observation coupled with the fact that all values are very close in range indicates that the subsequences are a close match.
Exploring Re-occurrence of Motives Within Time Series. Our ONEX exploratory tool can work with data from diverse domains. We showcase now its use in exploring electrical usage data using the ElectricityLoad collection. The similarity view provides a wealth of information about repeated patterns in electricity usage. Focusing on the electrical consumption of a single household, Fig. 18 shows a single time series across one year in Portugal and finds repeated patterns within it. The alternating blue and green coloration are used to clarify instances of consecutive segments. The top graph displays a monthly pattern indicating that this household tends to use electricity in a consistent manner throughout the summer months. The bottom pattern shows that winter months too have similar trends, empowering the analyst to determine that a few small habit changes could have a large savings impact.

In summary, ONEX is a truly interactive time series exploration tool that enables efficient exploration of time series datasets based on the combination of two similarity distances. This leads to shorter time responses compared to the fastest known state-of-the-art method. Complemented by novel visual analytics, ONEX offers actionable insights into similarity through rich classes of operations.
4 Generalized Model for Time Warping

In this chapter we introduce a new framework for warping point-to-point distances. This research work resulted in the following paper:

4.1 Generalized DTW Distance

To leverage the power of the DTW approach [10] for handling alignment shifting, locally out-of-phase sequences and different length time sequences, we now introduce a generalization of DTW, called the Generalized DTW Framework (GDTW). The GDTW framework preserves all the advantages of DTW, while also supporting the transformation of a wide array of popular point-to-point distances $d$ into their warped counterparts $GDTW_d$. Better yet, unlike previous work, our GDTW approach “empowers” analysts to warp any existing point-to-point distance $d$ of their own choosing. We offer efficient strategies for computing these warping paths for distances meeting the recursive and symmetry properties described below (Sec. 4.3). As validated by our case study in Sec. 4.6, the resulting GDTW framework “expands” the classic DTW to accommodate a wide array of distances. To accomplish this, our approach fundamentally changes the algorithm for computing the weight of the warping path by generalizing it to allow the embedding of alternate distances.

While our generalized DTW can be applied conceptually to any point-to-point distance, it is important in practice to compute it efficiently. Thus, we also adopt and adapt the dynamic programming strategy utilized for the classical DTW algorithm to work in our generalized context of computing warping paths for the targeted distance.

4.2 Fundamentals of the DTW Warping Path

We now generalize the classic warping algorithm described in the Preliminaries Sec. The construction of the GDTW path follows the same steps as the classic algorithm, but the way the weight of the path is chosen is different. We define this new general path
and explain how to incorporate new functions in computing it. Given two sequences
\(X = (x_1, x_2, \ldots, x_n)\) and \(Y = (y_1, y_2, \ldots, y_m)\), with \(n \geq m\), we construct an \(n \times m\) grid graph \(G\), which is a generalization of the matrix \(\Gamma\) previously described in the classic DTW. Similarly to the classic DTW algorithm and shown in Fig. 19, we define a warping path \(P\) as a sequence of elements that forms a contiguous path from \((1, 1)\) to \((n, m)\). The \(t^{th}\) element of \(P\) namely \(p_t = (i_t, j_t)\), refers to the indices \((i_t, j_t)\) of this element \((x_{i_t}, y_{j_t})\) in the path. Thus a path \(P\) is \(P = (p_1, p_2, \ldots, p_t, \ldots, p_T)\), where \(n \leq T \leq 2n - 1\), \(p_1 = (1, 1)\) and \(p_T = (n, m)\) and \(n \geq m\). By “decoding” this general warping path and extracting the values for \(x_{i_k}\) and \(y_{j_k}\) at every position on the path, we conceptually construct the following two equal-length vectors: \(X_P = (x_{i_1}, x_{i_2}, \ldots, x_{i_T})\) and \(Y_P = (y_{j_1}, y_{j_2}, \ldots, y_{j_T})\), where some of the \(x_{i_k}\) and \(y_{j_k}\) are repeated while advancing on the path. Considering an arbitrary distance measure \(d\), the weight of the warping path \(P\) is then defined as the distance between \(X_P\) and \(Y_P\), which is computed using \(d\). That is, we have \(w(P) = d(X_P, Y_P)\).

![General warping path](image)

**Figure 19: General warping path**

**Definition 22** The Generalized Dynamic Time Warping Distance corresponding to a distance \(d\), denoted by \(GDTW_d\), is the weight of the path \(P\) with the minimum weight, namely:

\[
GDTW_d(X, Y) = \min_P(d(X_P, Y_P)).
\]

Clearly the generalized dynamic time warping distance, as defined in Def. 22, accommodates any distance measure, not just based on sums, but on maximum and minimum, fractions of sums, products, etc. However, as written, it requires us to find all the warping paths first, then determine their weight, and lastly pick the one path with
the minimum weight. This is not feasible in practice.

Thus, following the principles of the DTW framework, the key idea is that we must be able to construct the distance function recursively by indicating how to incorporate the \(n^{th}\) coordinates in the distance measure based on the previous \(n-1\) coordinates. For this, we introduce a recursive property that must be identified and then utilized to define and then compute the weight of the warping paths.

**Definition 23** The distance measure \(d\) in Definition 22 must satisfy the following **recursive condition**: There exists a 3-variable function \(f_d : \mathbb{R}^+ \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}^+\) where \(\mathbb{R}\) denotes the set of real numbers and \(\mathbb{R}^+\) denotes the set of non-negative real numbers with respect to a distance \(d\) such that for vectors \(X_P = (x_1, x_2, ..., x_n)\) and \(Y_P = (y_1, y_2, ..., y_n)\) \((n \geq 2)\), we have:

\[
d(X_P, Y_P) = d((x_1, \ldots, x_n), (y_1, \ldots, y_n)) = f_d(d((x_1, \ldots, x_{n-1}), (y_1, \ldots, y_{n-1})), x_n, y_n).
\]

(21)

The \(f_d\) function tells us, given the distance measure on the first \(n-1\) coordinates \((x_1, x_2, ..., x_{n-1}, y_1, y_2, ..., y_{n-1})\), how to incorporate the \(n^{th}\) coordinates \((x_n, y_n)\). This expression assumes that the distance measure is symmetric in the coordinates.

To illustrate this concept with a concrete example, we now re-examine the well-known Euclidean Distance (ED) [4], previously applied in the classic DTW, in our proposed new context. That is, we give the recurrence for ED as per Def. 23.

**Euclidean Distance Example:** Given the Euclidean distance (ED) between two sequences \(X\) and \(Y\) defined as

\[
d_{ED}(X, Y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}.
\]

(22)

The **recursive expression** of ED according to Def. 23 is:

\[
f_{d_{ED}}(a, x_n, y_n) = \sqrt{a^2 + |x_n - y_n|^2},
\]

where \(a\) is the value of the function \(d_{ED}\) for the first \(n-1\) coordinates. As our case study
in Sec. 4.6 reveals, all distance measures in [15] can be expressed in terms of the above recursive property as per Def. 23.

Fig. 20 shows an example of computing the classic DTW warping path for two sequences X and Y as depicted with values in bold font. The leftmost matrix is the matrix M described in the classic DTW algorithm and contains the pairwise ED between the elements of the sequences. The matrix Γ in the middle showcases the dynamic programming strategy used for computing the path. For example, the element 0 in the matrix M is summed with the minimum of the three elements (left, down and diagonally-down) in Γ leading to the value 0 in the Γ matrix. The gray values hint that in reality this matrix is not always computed in its entirety, but rather the elements are calculated “as needed” to find the path efficiently. Lastly, the matrix on the right is the final Γ matrix displaying the warping path marked in blue, or the path with the minimum weight.

4.3 Devising Efficient Strategies for Computing the GDTW Warping Path

Let us assume that we have a distance measure \( d \) satisfying Equation (21). Then we propose to adopt and adapt the dynamic programming strategy designed for the classical DTW (see Sec. 2.2) by altering its recursive expression as described below.

**Definition 24** The general recursive expression amendable for dynamic programming for warping a distance \( d \) is:

\[
\gamma(i, j) = \min \begin{cases} 
  f_d(\gamma(i - 1, j - 1), x_i, y_j), \\
  f_d(\gamma(i - 1, j), x_i, y_j), \\
  f_d(\gamma(i, j - 1), x_i, y_j).
\end{cases}
\]

(23)

with \( \gamma(1, 1) = d(x_1, y_1) \).
Definition 25 Using Equation 23, the “warped” version of a distance $d$ returns a general dynamic warping distance defined as:

$$GDTW_d(X, Y) = \gamma(n, m)$$ (24)

Given a distance measure $d$, we propose to first design the function $f_d$ in Equation 21. Thereafter, we plug the former into Equation 23 to derive a dynamic programming solution that computes the warped version of distance $d$. Given that the recurrence above uses the three conditions from the classic DTW dynamic programming for progressive iterative traversal across the matrix, the complexity of our proposed GDTW process can be shown to be the same as in the classical DTW algorithm. This process thus enables us to efficiently compute the warped version of a distance $d$.

To illustrate, we apply the above process to our running example of Euclidean distance. Namely, by modifying the general DP expression in Equation 23, we derive the following dynamic programming recurrence for warping the Euclidean distance:

$$\gamma(i, j) = \min\begin{cases} 
(\gamma(i - 1, j - 1)^2 + |x_i - y_j|^2)^{\frac{1}{2}}, \\
(\gamma(i - 1, j)^2 + |x_i - y_j|^2)^{\frac{1}{2}}, \\
(\gamma(i, j - 1)^2 + |x_i - y_j|^2)^{\frac{1}{2}}.
\end{cases}$$

We note that the dynamic programming recursive expressions that are mentioned above, when applied to ED, are indeed identical to those in the classic DTW. That is, for the particular case of $d$ being the Euclidean Distance, this defaults to the classic DTW.

4.4 Proposed GDTW Methodology

In brief, our GDTW methodology for creating a corresponding warping distance $GDTW_d$ for a given distance $d$ is now accomplished through the following three steps:

1. **Select** a desired distance $d$ as the potential warping candidate. If the distance $d$ satisfies the recursive condition (Sec. 4.3), then the following steps provide the strategy to efficiently compute the warping path\textsuperscript{15}

\textsuperscript{15}If the distance $d$ is not already in our repository and it does not satisfy this condition, other strategies can be devised to compute the warping path.
2. **Design** the function $f_d$ for the recursive expression in Def. 21 to serve as the weight of the corresponding GDTW warping path (Sec. 4.3).

3. **Find** the recursive dynamic programming expression by plugging this function $f_d$ into Equation (23) to devise an efficient strategy for computing the weight of the path (Sec. 4.3).

We note that DTW is a type of elastic distance. It works by aligning the two time series and combining the individual aligned “point-to-point” distances, which may be Euclidean distance or some other distance. Where there is a possibility of confusion, we call the latter the base distance in contrast with the warped distance that incorporates it.

### 4.5 Practical Requirements for Newly Designed Distances

We now examine our newly “warped distances” in the light of practical desiderata for designing effective distance functions [2].

1. **User-centered.** In data mining applications the choice of similarity distance rests with the analysts and their understanding of the domain, their experience and perception of the results. This is one of the reasons we proposed the GDTW framework which now enables analysts to choose distances that are appropriate to their application domain but that would otherwise be overlooked because of their limitations in comparing sequences with different lengths and alignments. Incorporating domain knowledge in the design of new distances is very important, as these distances aim to model the user requirements better than existing distances. Our experimental evaluation will reveal the improvement in accuracy that our new warped distances offer compared to their respective traditional point-wise versions.

2. **Compactness.** A new distance should be efficiently computable because its computation will be likely repeated many times, especially when dealing with large datasets. We show in Sec. 4.3 that we devise effective strategies for computing our new distances, as well as a general strategy that can be applied towards achieving “compactness” of any new warped distance.

3. **Interpretability.** Being able to know which distance provides the most influential
insights into datasets is crucial for analysts. This interpretability of a distance mostly comes from the ability to compare the results based on different distances. This way, analysts can decide the best similarity model to be used in specific applications.

4.6 GDTW Case Study

We now validate our proposed GDTW framework by selecting and warping a wide range of distances using our three-step methodology. For this case study, we focus on distances collected in the highly cited survey paper [15] due to its large coverage of popular point-to-point distance functions. In particular, we showcase examples of well-known distance measures such as the $L_p$ norm distances [4] which include the Euclidean, Manhattan and Minkowski, popular in studying similarity of time series. We also show distances that could never work using the classic DTW, such as the Sorensen and the Cosine distances.

Our case study achieves three objectives:

- It demonstrates the utility of the proposed GDTW methodology for designing a warping solution for a rich diversity of distances, including those composed of arithmetic expressions including sum, difference, division, square root, maximum and minimum, fractions of sums, products, etc.

- Our work constructs a repository of off-the-shelf warped important metrics ready to use.

- The availability of this rich diverse and illustrious examples will help distance measure designers in the future with designing recurrence expressions for their own metrics.

4.7 Warping $L_p$ Norm Distances

We start with the prevalent $L_p$ norm distances, noting that $L_1$ is also known as the Manhattan Distance, $L_2$ as the ubiquitous Euclidean Distance reviewed earlier in Sec. 4.3 (and thus not repeated here), and $L_\infty$ as the Minkowski or also the Chebyshev Distance.
**L_p-distances in general**: Given the $L_p$ distance between two time series $X$ and $Y$ defined as:

$$d_{L_p}(X, Y) = \left( \sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}},$$

(25)

the recursive expression of the $L_p$ distance is stated as:

$$f_{d_{L_p}}(a, x_n, y_n) = (a^p + |x_n - y_n|^p)^{\frac{1}{p}},$$

where $a$ is the total value of the distance measured for the first $n-1$ coordinates. The $p$ in the $L_p$ can be plugged in accordingly to model specific $L_p$ norms, as mentioned above. The **dynamic programming** recurrence for warping $L_p$ distances is:

$$\gamma(i, j) = \min \left\{ \begin{array}{ll}
(\gamma(i - 1, j - 1) + |x_i - y_j|^p)^{\frac{1}{p}}, \\
(\gamma(i - 1, j) + |x_i - y_j|^p)^{\frac{1}{p}}, \\
(\gamma(i, j - 1) + |x_i - y_j|^p)^{\frac{1}{p}}.
\end{array} \right.$$

As a particular case for the most well-known $L_p$ distances, but skipping Euclidean, previously described, we review next some of the other popular distances.

**Manhattan Distance**: Given the Manhattan distance $d_{MD}$ between two time series $X$ and $Y$, defined as

$$d_{MD}(X, Y) = \sum_{i=1}^{n} |x_i - y_i|,$$

(26)

its recursive expression is:

$$f_{d_{MD}}(a, x_n, y_n) = (a + |x_n - y_n|)$$

and the **recursive dynamic programming** is:

$$\gamma(i, j) = \min \left\{ \begin{array}{ll}
(\gamma(i - 1, j - 1) + |x_i - y_j|), \\
(\gamma(i - 1, j) + |x_i - y_j|), \\
(\gamma(i, j - 1) + |x_i - y_j|).
\end{array} \right.$$

Similarly to Fig. 20, we give an example for computing the warping path for the same
pair of sequences using $GDTW_{MD}$ in Fig. 21. We note here that the resulting path differs than the path found by the classic DTW.

![Figure 21: Computing the warping path with $GDTW_{MD}$](image)

**Minkowski Distance:** Given the Minkowski distance $d_{Mink}$ between two time series $X$ and $Y$ defined as

$$d_{Mink}(X, Y) = \max_{i=1}^{n} |x_i - y_i|,$$

(27)

its **recursive expression** is:

$$f_{d_{Mink}}(a, x_n, y_n) = \max(a, |x_n - y_n|).$$

with the **dynamic programming recursive** expressions:

$$\gamma(i, j) = \min \left\{ \begin{array}{l}
\max(\gamma(i - 1, j - 1), |x_i - y_j|), \\
\max(\gamma(i - 1, j), |x_i - y_j|), \\
\max(\gamma(i, j - 1), |x_i - y_j|).
\end{array} \right.$$  

Similarly to Fig. 20, we give an example for computing the warping path for the

![Figure 22: Computing the warping path with $GDTW_{Mink}$](image)

same pair of sequences using $GDTW_{Mink}$ in Fig. 22. We note two main differences: the resulting path is different than all previous paths, and the dynamic programming strategy used to fill the matrix $\Gamma$ uses a different expression.
4.8 Warping the Sorensen Distance

The Sorensen distance, used in ecology [53], is an example that could not be accommodated by the classic DTW because of its complex form (a fraction of sums). We now show how it is handled by our GDTW methodology.

Given the Sorensen distance $d_{sor}$ between two sequences $X$ and $Y$ defined as:

$$d_{sor}(X, Y) = \frac{\sum_{i=1}^{n} |x_i - y_i|}{\sum_{i=1}^{n} |x_i + y_i|}$$  \hspace{1cm} (28)

Its recursive expression is:

$$f_{d_{sor}}(\frac{a}{b}, x_n, y_n) = \frac{a + |x_n - y_n|}{b + |x_n + y_n|} = \frac{a'}{b'}$$

where $a$ and $b$ denote the total value of the differences and respectively the sums of the first n-1 coordinates.

The dynamic programming recursive expressions are:

$$\gamma(i, j) = \min \begin{cases} \frac{a_1 + |x_i - y_j|}{b_1 + |x_i + y_j|}, \\ \frac{a_2 + |x_i - y_j|}{b_2 + |x_i + y_j|}, \\ \frac{a_3 + |x_i - y_j|}{b_3 + |x_i + y_j|}, \end{cases}$$

where $\gamma(i - 1, j - 1) = \frac{a_1}{b_1}, \gamma(i - 1, j) = \frac{a_2}{b_2}, \gamma(i, j - 1) = \frac{a_3}{b_3}$.

4.9 Warping the Cosine Distance

The Cosine distance, which measures the angles between two vectors, corresponds to the normalized Inner Product. In the literature, other names for this popular distance include: angular metric [24], Ochiai [24] and Carbo [57].

Given the Cosine distance $d_{cos}$ between two sequences $X$ and $Y$ defined as:

$$d_{cos}(X, Y) = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \sqrt{\sum_{i=1}^{n} y_i^2}}$$  \hspace{1cm} (29)
its recursive expression is:

\[ f_{d\cos}(\alpha_{bc}, x_n, y_n) = \frac{a + x_n y_n}{\sqrt{b^2 + x_n^2 \sqrt{c^2 + y_n^2}}} = \frac{a'}{b'c'}, \]

with \( a, b \) and \( c \) denoting respectively the sum of the product of the first \( n-1 \) coordinates, the sum of squares for the first \( n-1 \) values of \( x \) and the sum of squares for the first \( n-1 \) values of \( y \). Lastly, its dynamic programming recurrence becomes:

\[
\gamma(i, j) = \min \left\{ \frac{a_1 + x_i y_j}{\sqrt{b_1^2 + x_i^2 \sqrt{c_1^2 + y_j^2}}}, \frac{a_2 + x_i y_j}{\sqrt{b_2^2 + x_i^2 \sqrt{c_2^2 + y_j^2}}}, \frac{a_3 + x_i y_j}{\sqrt{b_3^2 + x_i^2 \sqrt{c_3^2 + y_j^2}}} \right\}
\]

\[ \gamma(i - 1, j - 1) = \frac{a_1}{b_1 c_1}, \gamma(i - 1, j) = \frac{a_2}{b_2 c_2}, \gamma(i, j - 1) = \frac{a_3}{b_3 c_3}. \]

4.10 Warping Other Distances

As shown by the examples above, all the distances in [15] can be theoretically warped by our GDTW methodology. These distances which are essential for classification, clustering and retrieval problems, are reviewed and categorized syntactically and semantically into eight families in [15]. Different distances in our case study correspond to different families described in this survey. Aside from the distances that we demonstrate, many other popular distances such as Jaccard, Dice and Pearson – which are based on similar arithmetic expressions – can be warped using our methodology.
4.11 Experimental Evaluation of GDTW

4.12 Experimental Methodology

Our experimental study now sets out to explore if our newly warped distances indeed can be valuable for exploring time series beyond the existing metrics. For this, we have implemented a select subset of our proposed warped distances namely, $GDTW_{ED}$ (classic DTW), $GDTW_{Mink}$, and $GDTW_{MD}$. We then apply these new distances to several important data mining problems such as classification, best match retrieval, and other similarity-based operations to assess their effectiveness and utility. *Our code, repository of newly warped distances, our experimental results, and detailed instructions to reproduce our experiments will be publically made available, once the paper is accepted.*

**Data Sets.** We use the largest public collection of time-series datasets that we are aware of, namely the UCR time-series collection\(^{16}\). This archive contains 85 benchmark datasets from a rich variety of domains. While in some experiments, we work with all data sets in this archive, in other experiments we generate additional data sets in a controlled fashion by applying distortions to evaluate other characteristics of these metrics, as further explained below.

**Four Classes of Evaluation.** We devised four classes of experiments to explore the impact that different GDTW distances have in the context of important research questions.

**Experiment 1: Time Series Classification.** We evaluate the relative effectiveness of our newly warped distances for solving the important time series classification problem. For this, we apply our newly warped distances over the training and test sets of each of the 85 datasets in the UCR archive, using them as (parameter-free) 1-NN classifiers. We compute the classification accuracy, i.e., number of correctly classified instances over all instances, and the error rate in performing 1-NN classification. Because the 1-NN classifier is deterministic, we only perform this computation once. We compare the results of these warped distances to determine their classification accuracies. This experiment, as we show, leads to important observations about one of the

\(^{16}\) [www.cs.ucr.edu/eamonn/time_series_data/](http://www.cs.ucr.edu/eamonn/time_series_data/)
newly warped metrics being more effective than the state-of-the-art classic DTW for this task.

**Experiment 2: Best Match Retrieval.** We show the impact of “warped” distances on retrieving the best match for a sample sequence. We confirm that using diverse distances leads to finding different best matches for the same sequence. First we find the best match (or the nearest neighbor) for a given sample query sequence in a dataset using a “point-to-point” distance and then by using its “warped” counterpart. We then compare these matches. Our experiment aims to show that: (1) the warped distances tend to return different results than their point-wise versions, as expected when the sequences are not aligned in time, and (2) each newly warped distance returns often different results than the other warped distances. The diversity of results demonstrates that our new distances can be useful for best match retrieval, by revealing best matches that would not be found otherwise by the traditional distances. In addition, we showcase the impact of using these new distances on an average linkage hierarchical clustering problem.

**Experiment 3: Evaluation of Warping Characteristics.** Similarly to the well-known Derivative Dynamic Time method [46], which studies the “over-warping” produced by the classic DTW, we study the amount of warpings produced by our different GDTW variants for the same pair of sequences. This can be very helpful for analysts who might be interested in finding similar sequences with fewer warpings, depending on the application domain. Recall that for two sequences of different lengths m and n, with \( m \leq n \), the length \( l \) of the warping path \( P \) has to be \( \max(m, n) \leq l \leq m + n - 1 \). While for the case of \( m = n \), we get \( n \leq l \leq 2n - 1 \). Inspired by [46], we define the amount of warping \( W \) as:

\[
W = \frac{(l - \text{average}(m, n))}{\text{average}(m, n)} \tag{30}
\]

where \( 0 \leq W \leq 1 \) and \( m, n \) are the lengths of the compared sequences. \( W = 0 \) if the algorithm does not find a warping between two sequences. \( W \) increases to a maximum value of 1 as the warping “discovered” by the algorithm increases.

We also measure the sensitivity of our warped distances to local distortion. For this, we introduce distortions in a controlled fashion in pairs of synthetic sequences, similarly to
Then we compare the warping paths computed by the different warped distances to show their respective sensitivity to local distortion. In particular, we compare the matchings produced by these diverse distances and determines which distance leads to less warping.

**Experiment 4: Real-world Use Case Studying Heart Arrhythmia.** We conduct an experimental use case where three point-wise distances and their warped versions are being used to explore a medical dataset with the help of a domain expert. We compare the best match retrieved by our diverse distances to the one chosen by an expert cardiologist.

### 4.13 Experimental Results

#### 4.13.1 Experiment on Time Series Classification

*Time series classification* [18][26][7] is an important problem where a distance is used as a subroutine in the K-Nearest Neighbor (K-NN) algorithm. This simple algorithm has been shown to be surprisingly competitive, by consistently outperforming rival methods such as decision trees, neural networks, Bayesian networks, and Hidden Markov Models [7, 82]. Moreover, time series classification has thus far been one of the few tasks to resist significant progress from “deep learning” [49].

Given this, the choice of “which” time series distance measure to use is important. Literally dozens of distances have been proposed (see [27, 49] and the references therein). However, an extensive recent empirical comparison (performing 36 million experiments) has confirmed the excellent performance of DTW-based 1-NN, which is only beaten by Ensemble Classifiers [49]. The DTW in question was the classic DTW or $GDTW_{ED}$ in our case.

It is natural to ask if other warped distances such as $GDTW_{MD}$ or $GDTW_{Mink}$ could be even more efficient for classifying time series. To test this research question, we performed classification experiments on all 85 datasets from the UCR Archive. The train/test splits were identical. We fixed the warping window to 100% for all experiments, thus any differences can be attributed solely to the effect of changing the base distance measure. A spreadsheet with the raw results is archived along with our code.
4.13 Experimental Results

and the rest of the details of the experiments. More detailed pairwise comparisons using error-rate binary plots can be found in the Additional Experimental Results (Sec. 4.15) in Fig. 31, 32, and 33.

For brevity, we present here a compact visual summary displaying a comparison between the results obtained using the classic DTW (our $GDTW_{ED}$), $GDTW_{MD}$ and $GDTW_{Mink}$. Pairwise comparisons of distance measures are often presented as 2D-scatter-plots [26]. As we are interested in comparing three algorithms, in Fig. 23 we present the results as a trivariate plot. In this plot, the locations of the points do not correspond to the actual error rate, but are proportional to them.

This ternary plot shows the error rate distribution for the three distances and marks the areas where each individual distance performs better than the others. The red dots in the light purple area indicate the wins for $GDTW_{MD}$ (in 56 cases), while the ones in the pink area show where $GDTW_{ED}$ is better (in 44 cases) and the ones in the green area correspond to $GDTW_{Mink}$ performing better (in 15 cases). The results are surprisingly diverse, with 44 “wins” for $GDTW_{ED}$, 56 “wins” for $GDTW_{MD}$, and 15 “wins” for $GDTW_{Mink}$. There are fewer dots in the green area, indicating that $GDTW_{Mink}$ generally performed poorer than the other two distances. The error rates for $GDTW_{MD}$ and $GDTW_{ED}$ are fairly close, as shown by the high concentration of points in the center.

Figure 23: A trivariate plot comparing $GDTW_{ED}$, $GDTW_{MD}$, and $GDTW_{Mink}$. For points close to the center of the figure, all three methods produce similar error rates. For points away from the center, at least one method is performing poorly.
These results are important and “surprising,” given that several papers published in the last decade introducing novel distance measures claimed to improve over $GDTW_{ED}$\cite{7}, yet the improvements offered are typically no greater that the improvements garnered by simply changing the base distance measure in DTW. While our results show that we could improve on the standard benchmark of 1-NN $GDTW_{ED}$, simply by switching to $GDTW_{MD}$, a more careful analysis (which is beyond the scope of this paper) is warranted. Our results suggest several promising future work directions to use our framework to produce substantial improvements on the state-of-the-art for time series classification, which we elaborated upon in our conclusion.

4.13.2 Experiment on Best Match Retrieval

In each dataset, we randomly select a subsequence and “promote” it to be a query; similarly to the methodology in \cite{28} we take this sequence out of that dataset. Then we find the best match for this query sample in the dataset by using three point-wise distances ($ED, MD, M_{ink}$) and their warped counterparts ($GDTW_{ED}, GDTW_{MD}, GDTW_{Mink}$) as the sequence with the smallest of each respective distance to the sample. We repeat this experiment for 10 random sample sequences and record the best match sequence identifier and the respective distance between the sample sequence and the retrieved best match.

Tables 13, 14, and 15 in the Additional Experimental Results (Sec. 4.15) show the best match sequences in the ECG dataset using ED compared to classic DTW, then using Manhattan Distance (MD) compared to $GDTW_{MD}$, then using Chebyshev or Minkowski Distance (Mink) compared to $GDTW_{Mink}$.

The results can vary significantly when using different point-to-point distances and their warped versions, as expected. The point-wise distances can theoretically be at most as good as their warped versions. Their highest performance is achieved for sequences that are aligned in time. In 20% of the cases ED and DTW returned the same sequence as the best match. MD and $GDTW_{MD}$ also return the same match in 20% of the scenarios. Mink and $GDTW_{Mink}$ never returned the same result. We com-
pare the results retrieved by the newly warped distances. In 10% of the cases DTW and \( GDTW_{Mink} \) returned the same result, \( GDTW_{MD} \) and \( GDTW_{Mink} \) found the same match in 20% of the scenarios, while DTW and \( GDTW_{MD} \) had the same result in 50% of the cases. The fact that the results are different in most cases attests to the usefulness of our distances in finding matches that otherwise might be overlooked.

In Fig. 6 we show a visual display of an example of the best match in ECG based on using the point-wise and their respectively warped counterpart distances. In this specific case, two of the point-wise distances returned the same best match (ED and Mink), while MD returned a different best match. Each warped distance returned a different match. If our distances would have all returned the same result or even the same result as their warped counterparts, then they would not be useful. It is the diversity of the results that proves the usefulness of these distances.

![Figure 24: Best matches in ECG retrieved with point-wise distances and their warped counterparts](image)

Lastly, we highlight one additional experiment that reveals new insights that can be
uncovered by our newly warped distances using **hierarchical clustering**. We select five sequences from the ECG dataset. Two of the five sequences are randomly chosen samples (red/bold) from class 1, while the other three are randomly selected from class -1 (blue/none-bold). We cluster these sequences using our three newly warped distances. As the dendogram in Fig. 25 shows, $GDTW_{MD}$ correctly classifies the red sequences from start, while $GDTW_{ED}$ and $GDTW_{Mink}$ do not. This experiment reaffirms that our new distance $GDTW_{MD}$ has a higher accuracy than the classic DTW.

In summary, based on these results, we conjecture that other warped distances, aside from the ones we showcase here, might lead to even further improvements in accuracy for tasks from classification to clustering.

### 4.13.3 Experiment on Warping Characteristics

**Evaluating Cardinality of Warpings.** For the ECG dataset, we randomly select 10 pairs of sequences of the same length, and 10 pairs of sequences of different lengths. For each pair of the same length, we find the matching elements of the sequences first by using the point-wise distances (ED, MD, $Mink$) then their corresponding warped versions. For the pairs of sequences of different lengths we only use the warped distances ($GDTW_{ED}$, $GDTW_{MD}$, and $GDTW_{Mink}$).

Warpings indicate points of a sequence that are either matching or are being matched to more than one point of the other sequence. Points that are matched one-to-one are referred to as matchings. The warpings created by $GDTW_{MD}$ and $GDTW_{Mink}$ are fewer and more intuitive than the ones created by $GDTW_{ED}$, which is a similar conclusion with the conclusion of the experiments of [46]. This shows that indeed the classic DTW can “over-warp”, mainly due to the fact that it incorporates the ED as base distance. This knowledge can be useful to analysts who might choose to use distances that produce fewer warpings.

We find no warpings when using the point-wise distances, as expected, in Fig.15(a). The chart in (b) shows that the classic DTW (same as our $GDTW_{ED}$) warps 23 times. Examining the charts in (c) and (d), we see that $GDTW_{MD}$ warps 21 times for the same pair of sequences, while
**Experimental Results**

(a) Point-to-point matching

(b) Classic DTW warping

(c) $GDTW_{MD}$ warping

(d) $GDTW_{Mink}$ warping

Figure 26: Warping for a pair of sequences in ECG.

$GDTW_{Mink}$ warps only 17 times. It is observable on the charts that the warpings produced by this last distance are to points that are closer to each other and are therefore more intuitive. $GDTW_{Mink}$ is always smaller than the classic DTW and $GDTW_{MD}$, making it a better choice for analysts in application domains where only very small distances are considered to express similarity.

We display here the visual representation for one pair of sequences in the ECG dataset, while an additional example on the ItalyPower can be found in Sec. 4.15.

We also compute the average amount of warpings for each version of GDTW for specific datasets. Consistently, the classic DTW creates more warpings than $GDTW_{MD}$ and $GDTW_{Mink}$, while $GDTW_{MD}$ warps more than the $GDTW_{Mink}$. We report these in Tables 8 and 9 for four datasets.

**Table 8: Average warpings for sequences of the same length**

<table>
<thead>
<tr>
<th>Dataset</th>
<th>DTW</th>
<th>$GDTW_{MD}$</th>
<th>$GDTW_{Mink}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ItalyPower</td>
<td>0.39</td>
<td>0.3</td>
<td>0.24</td>
</tr>
<tr>
<td>ECG</td>
<td>0.46</td>
<td>0.35</td>
<td>0.2</td>
</tr>
<tr>
<td>Wafer</td>
<td>0.46</td>
<td>0.32</td>
<td>0.21</td>
</tr>
<tr>
<td>Face</td>
<td>0.41</td>
<td>0.35</td>
<td>0.15</td>
</tr>
</tbody>
</table>
Table 9: Average warpings for sequences of different lengths

<table>
<thead>
<tr>
<th>Dataset</th>
<th>DTW</th>
<th>GDTW_MD</th>
<th>GDTW_Mink</th>
</tr>
</thead>
<tbody>
<tr>
<td>ItalyPower</td>
<td>0.42</td>
<td>0.31</td>
<td>0.23</td>
</tr>
<tr>
<td>ECG</td>
<td>0.41</td>
<td>0.33</td>
<td>0.14</td>
</tr>
<tr>
<td>Wafer</td>
<td>0.53</td>
<td>0.45</td>
<td>0.46</td>
</tr>
<tr>
<td>Face</td>
<td>0.24</td>
<td>0.21</td>
<td>0.08</td>
</tr>
</tbody>
</table>

In summary, many possible alignments of the same pair of sequences arise, depending on the chosen distance. Analysts can choose the easier to interpret alignments for specific datasets by deciding what distance to use.

Evaluating Sensitivity to Local Distortions. We test the ability to find the correct warpings by using different GDTW distances for the pairs of sequences for which the “warping” is known. We insert some distortions in each of the sequences. That is, similarly to [46], we distort the y-axis by adding or subtracting a Gaussian bump on randomly chosen anchor points of the sequences. As shown in Fig. 16 we find the warping paths for these modified sequences using our three warped distances. Each distance leads to a warping path different than the other two distances. DTW “over-warps” the distorted sequences, while GDTW_{MD} and GDTW_{Mink} find a shorter warping path. The performance for DTW and GDTW_{MD} can degrade even for small distortions of the y-axis, while GDTW_{Mink} maintains a better warping performance.

4.14 Use Case Study in Heart Arrhythmia

In collaboration with an expert cardiologist, we explore the MIT-BIH Arrhythmia Database, completed by Beth Israel Deaconess Medical Center and MIT, which supports research into arrhythmia analysis and related subjects. This database offers a set of standard test materials for evaluation of arrhythmia detectors. We compare the findings of an expert cardiologist with the ones retrieved using our point-wise distances and their warped counterparts to see which of the newly warped distances best match the semantics in this domain.

The MIT-BIH Arrhythmia Database [58], [31] contains 48 half-hour excerpts of two-channel ambulatory ECG recordings obtained from 47 subjects studied by the BIH Arrhythmia Laboratory between 1975 and 1979. Twenty-three recordings were chosen at random from a set of 4000 24-hour ambulatory ECG recordings collected from a mixed
Figure 27: Warpings for distorted sequences.

(a) shows the sequences. (b), (c), and (d) show the warpings using respectively the classic DTW, $GDTW_{MD}$ and $GDTW_{Mink}$.

A population of inpatients (about 60%) and outpatients (about 40%) at Boston’s Beth Israel Hospital; the remaining 25 recordings were selected from the same set to include less common but clinically significant arrhythmias that would not be well-represented in a small random sample.

The recordings were digitized at 360 samples per second per channel with 11-bit resolution over a 10 mV range. Two or more cardiologists independently annotated each record; disagreements were resolved to obtain the computer-readable reference annotations for each beat (approximately 110,000 annotations). Exploring similarity of ECG sequences is crucial for diagnosing arrhythmia which refers to any change from the normal sequence of electrical impulses. The electrical impulses may cause the heart to beat too fast, too slowly, or erratically. When the heart does not pump blood effectively, the lungs, brain and other organs cannot work properly and may shut down or be damaged.

Clinical diagnoses and basic investigations depend on the ability to record and analyze...
physiological signals. Examples of such signals include ECG and heart rate recordings from patients at a high risk of sudden death and healthy control subjects as seen in Fig. 28. Other examples include fluctuations of hormone and other molecular biological signal messengers, transducers in neuroendocrine dynamics, multiparameter dynamics, and multiparameter recordings in sleep apnea and epilepsy.

We use our newly warped distances to explore this database and find the best match for a given ECG shape. For this experiment, we chose the sample heart rate shape of the record labeled 107 and explored the dataset to find its best match using our three warped distances. This male patient (age 63) has a complete heart block condition in which the impulse generated in the sinoatrial node in the atrium of the heart does not propagate to the ventricles. We randomly selected 20 records from the dataset, including that of the patient with record 107 and asked our cardiologist collaborator to find the best match for the second part of this ECG shape. The cardiologist identified the second part of the ECG for the patient with record number 113, as having the closest heart rate, meaning average heart rate in beats per minute. The doctor attributed this similarity to the presence of a pacemaker, which regulates abnormal heart rhythms and heart rate by using low-energy electrical pulses to prompt the heart to beat at a normal rate. Independent of her findings, we retrieved the best match for this sequence.

![Heart Rate Time Series](image)

Figure 28: Heart rate time series
by using each of our distances, respectively: ED, MD, M_{ink}, GDTW_{ED}, GDTW_{MD}, and GDTW_{M_{ink}}. As seen in Fig.29, different best matches are returned by different distances. Sequence 113 was returned as best match by GDTW_{MD}. All point-wise distances, DTW, and GDTW_{M_{ink}} returned different matches. In this case GDTW_{MD} was more appropriate for finding the best match for an ECG shape.

In summary, this experiment exploring an ECG dataset with GDTW_{MD} led to finding the best match, which was confirmed by an expert in the field. However, this match would have been missed by the classic DTW.

Figure 29: Case study best match sequences

4.15 Additional Experimental Results.

In this section we offer extended results of our experiments and details supporting some of the summaries included in Sec. 4.13.
4.15.1 Experiment on Time Series Classification.

We include binary plots showing the comparisons $GDTW_{ED}$ vs $GDTW_{Mink}$, $GDTW_{MD}$ vs $GDTW_{Mink}$, and $GDTW_{ED}$ vs $GDTW_{MD}$ in Fig. 31, 32, and 33.
Fig. 31 is based on the error rates for $GDTW_{ED}$ and $GDTW_{MD}$ over the 85 datasets during the classification experiment. The plot indicates the areas where $GDTW_{ED}$ performs better than $GDTW_{MD}$, as well as the area where $GDTW_{MD}$ performs better than $GDTW_{ED}$. We note that although the performance seems very close, $GDTW_{MD}$ “wins” more times than $GDTW_{ED}$. Fig. 32 displays a pairwise comparison of $GDTW_{ED}$ and $GDTW_{Mink}$ based on error rates. The area where $GDTW_{Mink}$ wins is much smaller than the area where $GDTW_{ED}$ wins, but it is important to note that there are 15 times where $GDTW_{Mink}$ has higher accuracy for classifying the time series in the 85 datasets. Fig. 33 offers the pairwise comparison between $GDTW_{MD}$ and $GDTW_{Mink}$. We note again that the area where $GDTW_{Mink}$ wins is much smaller than the one where $GDTW_{MD}$ wins.

4.15.2 Experiment on Best Match Retrieval

We display in Tables 10, 11, and 12 the best match sequences in the ItalyPower dataset using ED compared to classic DTW, then using Manhattan Distance (MD) compared to $GDTW_{MD}$, then using Chebyshev or Minkowski Distance (Mink) compared to $GDTW_{Mink}$. The “best match” as defined in Sec. 4.12 is displayed as the sequence “TS” followed by a number indicating the position of the time series in the dataset and the start and end points of the match sequence inside the square brackets.
Analyzing and summarizing the results, we note that the distance between the

Table 10: Best match comparison ED vs DTW in Italy

<table>
<thead>
<tr>
<th>Query</th>
<th>ED</th>
<th>Match ED</th>
<th>DTW</th>
<th>Match DTW</th>
</tr>
</thead>
<tbody>
<tr>
<td>TS0[0,23]</td>
<td>0.36</td>
<td>TS22[0,23]</td>
<td>0.22</td>
<td>TS53[1,23]</td>
</tr>
<tr>
<td>TS1[18,23]</td>
<td>0.06</td>
<td>TS3[18,23]</td>
<td>0.06</td>
<td>TS3[18,23]</td>
</tr>
<tr>
<td>TS2[0,11]</td>
<td>0.05</td>
<td>TS4[2,13]</td>
<td>0.05</td>
<td>TS4[2,13]</td>
</tr>
<tr>
<td>TS3[2,13]</td>
<td>0.09</td>
<td>TS3[2,13]</td>
<td>0.08</td>
<td>TS4[5,11]</td>
</tr>
<tr>
<td>TS4[2,13]</td>
<td>0.10</td>
<td>TS18[2,23]</td>
<td>0.35</td>
<td>TS8[3,14]</td>
</tr>
<tr>
<td>TS5[10,23]</td>
<td>0.07</td>
<td>TS4[10,23]</td>
<td>0.07</td>
<td>TS8[12,23]</td>
</tr>
<tr>
<td>TS6[5,23]</td>
<td>0.16</td>
<td>TS5[5,23]</td>
<td>0.12</td>
<td>TS8[5,23]</td>
</tr>
<tr>
<td>TS7[0,23]</td>
<td>0.08</td>
<td>TS2[0,23]</td>
<td>0.82</td>
<td>TS2[0,23]</td>
</tr>
<tr>
<td>TS8[0,23]</td>
<td>0.13</td>
<td>TS7[0,23]</td>
<td>0.11</td>
<td>TS8[0,23]</td>
</tr>
<tr>
<td>TS9[0,21]</td>
<td>0.20</td>
<td>TS3[0,21]</td>
<td>0.10</td>
<td>TS8[0,21]</td>
</tr>
</tbody>
</table>
Table 11: Best match comparison MD vs $GDTW_{MD}$ in ItalyPower

<table>
<thead>
<tr>
<th>Query</th>
<th>MD Match</th>
<th>GDTW$_{MD}$ Match</th>
</tr>
</thead>
<tbody>
<tr>
<td>TS0[0,23]</td>
<td>0.90</td>
<td>TS22[0,23]</td>
</tr>
<tr>
<td>TS1[18,23]</td>
<td>0.13</td>
<td>TS37[18,23]</td>
</tr>
<tr>
<td>TS2[11,13]</td>
<td>0.13</td>
<td>TS43[2,13]</td>
</tr>
<tr>
<td>TS3[2,13]</td>
<td>0.26</td>
<td>TS31[2,13]</td>
</tr>
<tr>
<td>TS4[2,13]</td>
<td>0.26</td>
<td>TS43[2,13]</td>
</tr>
<tr>
<td>TS5[10,23]</td>
<td>0.23</td>
<td>TS45[10,23]</td>
</tr>
<tr>
<td>TS6[5,23]</td>
<td>0.52</td>
<td>TS36[5,23]</td>
</tr>
<tr>
<td>TS7[0,23]</td>
<td>0.31</td>
<td>TS29[0,23]</td>
</tr>
<tr>
<td>TS8[0,23]</td>
<td>0.49</td>
<td>TS7[0,23]</td>
</tr>
<tr>
<td>TS9[0,21]</td>
<td>0.33</td>
<td>TS31[0,21]</td>
</tr>
</tbody>
</table>

Table 12: Best match comparison Mink vs $GDTW_{Mink}$ in ItalyPower

<table>
<thead>
<tr>
<th>Query</th>
<th>Mink Match</th>
<th>GDTW$_{Mink}$ Match</th>
</tr>
</thead>
<tbody>
<tr>
<td>TS0[0,23]</td>
<td>0.16</td>
<td>TS22[0,23]</td>
</tr>
<tr>
<td>TS1[18,23]</td>
<td>0.08</td>
<td>TS37[18,23]</td>
</tr>
<tr>
<td>TS2[11,13]</td>
<td>0.13</td>
<td>TS43[2,13]</td>
</tr>
<tr>
<td>TS3[2,13]</td>
<td>0.05</td>
<td>TS17[2,13]</td>
</tr>
<tr>
<td>TS4[2,13]</td>
<td>0.04</td>
<td>TS18[2,13]</td>
</tr>
<tr>
<td>TS5[10,23]</td>
<td>0.04</td>
<td>TS45[10,23]</td>
</tr>
<tr>
<td>TS6[5,23]</td>
<td>0.07</td>
<td>TS33[5,23]</td>
</tr>
<tr>
<td>TS7[0,23]</td>
<td>0.05</td>
<td>TS19[0,23]</td>
</tr>
<tr>
<td>TS8[0,23]</td>
<td>0.07</td>
<td>TS7[0,23]</td>
</tr>
<tr>
<td>TS9[0,21]</td>
<td>0.05</td>
<td>TS34[0,21]</td>
</tr>
</tbody>
</table>

sample query sequence and the best match is minimal, as expected, when using warped distances instead of their counterpart point-wise versions. More detailed analysis reveals that using diverse distances leads to retrieving different best matches for the same sample sequence. In 40% of the scenarios, the ED and the classic DTW found the same best match. Also in 40% of the cases, MD and $GDTW_{MD}$ found the same best match. The best match was the same for Mink and $GDTW_{Mink}$ in only 20% of our trials. The results confirm that warped distances are better suited than point-to-point distances to perform similarity searches, except for the special situations when the sequences are aligned and have the same length. In that case, both versions (warped and point-wise) of a specific distance return the same result.

We provide an additional comparison of the three warped distances that are created with our framework. In 70% of the cases, the classic DTW and $GDTW_{MD}$ returned the same match, but in only 10% of the scenarios, the matches retrieved by the classic DTW and $GDTW_{Mink}$ were matched. 20% of the scenarios returned the same match for $GDTW_{MD}$ and $GDTW_{Mink}$. All three warped versions returned the same result only 10% of the time. This is another indication that the best match for a sample se-
quence is highly dependent on the chosen distance. In few cases the results produced by the warped distances are the same, which means these new distances can be useful in "discovering" best matches that otherwise would not be found. Below we offer the results in retrieving the best match for the ECG dataset.

Table 13: Best match comparison ED vs classic DTW in ECG

<table>
<thead>
<tr>
<th>Query</th>
<th>ED</th>
<th>Match ED</th>
<th>DTW</th>
<th>Match DTW</th>
</tr>
</thead>
<tbody>
<tr>
<td>T0</td>
<td>0.15</td>
<td>T39</td>
<td>0.10</td>
<td>T39</td>
</tr>
<tr>
<td>T1</td>
<td>0.13</td>
<td>T37</td>
<td>0.11</td>
<td>T19</td>
</tr>
<tr>
<td>T2</td>
<td>0.07</td>
<td>T14</td>
<td>0.07</td>
<td>T17</td>
</tr>
<tr>
<td>T3</td>
<td>0.10</td>
<td>T125</td>
<td>0.07</td>
<td>T188</td>
</tr>
<tr>
<td>T4</td>
<td>0.19</td>
<td>T39</td>
<td>0.06</td>
<td>T117</td>
</tr>
<tr>
<td>T5</td>
<td>0.18</td>
<td>T30</td>
<td>0.13</td>
<td>T30</td>
</tr>
<tr>
<td>T6</td>
<td>0.24</td>
<td>T46</td>
<td>0.13</td>
<td>T46</td>
</tr>
<tr>
<td>T7</td>
<td>0.11</td>
<td>T66</td>
<td>0.08</td>
<td>T131</td>
</tr>
<tr>
<td>T8</td>
<td>0.14</td>
<td>T131</td>
<td>0.08</td>
<td>T86</td>
</tr>
<tr>
<td>T9</td>
<td>0.007</td>
<td>T134</td>
<td>0.004</td>
<td>T82</td>
</tr>
</tbody>
</table>

Table 14: Best match comparison MD vs $GDTW_{MD}$ in ECG

<table>
<thead>
<tr>
<th>Query</th>
<th>MD</th>
<th>Match MD</th>
<th>$GDTW_{MD}$</th>
<th>Match $GDTW_{MD}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>T0</td>
<td>0.40</td>
<td>T39</td>
<td>0.33</td>
<td>T39</td>
</tr>
<tr>
<td>T1</td>
<td>0.59</td>
<td>T133</td>
<td>0.44</td>
<td>T103</td>
</tr>
<tr>
<td>T2</td>
<td>0.20</td>
<td>T140</td>
<td>0.14</td>
<td>T178</td>
</tr>
<tr>
<td>T3</td>
<td>0.37</td>
<td>T125</td>
<td>0.41</td>
<td>T188</td>
</tr>
<tr>
<td>T4</td>
<td>1.20</td>
<td>T39</td>
<td>0.43</td>
<td>T117</td>
</tr>
<tr>
<td>T5</td>
<td>0.75</td>
<td>T30</td>
<td>0.61</td>
<td>T85</td>
</tr>
<tr>
<td>T6</td>
<td>1.82</td>
<td>T46</td>
<td>1.14</td>
<td>T46</td>
</tr>
<tr>
<td>T7</td>
<td>0.36</td>
<td>T123</td>
<td>0.28</td>
<td>T89</td>
</tr>
<tr>
<td>T8</td>
<td>0.99</td>
<td>T131</td>
<td>0.58</td>
<td>T131</td>
</tr>
<tr>
<td>T9</td>
<td>0.01</td>
<td>T24</td>
<td>0.007</td>
<td>T52</td>
</tr>
</tbody>
</table>

Table 14 shows the best match sequences found using Manhattan distance and its warped version $GDTW_{MD}$.

Table 15: Best match comparison Mink vs $GDTW_{Mink}$ in ECG

<table>
<thead>
<tr>
<th>Query</th>
<th>Mink</th>
<th>Match Mink</th>
<th>$GDTW_{Mink}$</th>
<th>Match $GDTW_{Mink}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>T0</td>
<td>0.08</td>
<td>T33</td>
<td>0.06</td>
<td>T39</td>
</tr>
<tr>
<td>T1</td>
<td>0.06</td>
<td>T60</td>
<td>0.04</td>
<td>T77</td>
</tr>
<tr>
<td>T2</td>
<td>0.04</td>
<td>T140</td>
<td>0.02</td>
<td>T178</td>
</tr>
<tr>
<td>T3</td>
<td>0.04</td>
<td>T125</td>
<td>0.02</td>
<td>T188</td>
</tr>
<tr>
<td>T4</td>
<td>0.05</td>
<td>T63</td>
<td>0.01</td>
<td>T191</td>
</tr>
<tr>
<td>T5</td>
<td>0.07</td>
<td>T65</td>
<td>0.05</td>
<td>T30</td>
</tr>
<tr>
<td>T6</td>
<td>0.08</td>
<td>T46</td>
<td>0.03</td>
<td>T137</td>
</tr>
<tr>
<td>T7</td>
<td>0.06</td>
<td>T56</td>
<td>0.04</td>
<td>T65</td>
</tr>
<tr>
<td>T8</td>
<td>0.04</td>
<td>T131</td>
<td>0.02</td>
<td>T131</td>
</tr>
<tr>
<td>T9</td>
<td>0.005</td>
<td>T134</td>
<td>0.002</td>
<td>T82</td>
</tr>
</tbody>
</table>

In 10% of the scenarios, the best match found was the same for the Manhattan distance and its warped version.
4.15 Additional Experimental Results.

(a) Point to point matching  
(b) Classic DTW warping  
(c) $GDTW_{MD}$ warping  
(d) $GDTW_{Mink}$ warping

Figure 34: Warping for a pair of sequences in ItalyPower.

Table 15 showcases the results for finding the best match using Minkowski or Chebyshev distance and its warped version. In this case, the best match was never the same for the point-to-point distance and its warped version. In 50% of the scenarios, the three point-wise distances returned the same best match, while only 10% the same best match was returned by the warped versions of these distances. In no scenario did all distances, point-wise and warped return the same match.

4.15.3 Experiment on Warping Characteristics

Fig. 29 displays another visual example of warpings for a pair of sequences from ItalyPower. Similarly to the previous example, the classic DTW produces more warpings than the $GDTW_{MD}$ and $GDTW_{Mink}$. 
5 Knowledge Discovery Powered by Multiple Distances

This chapter introduces our work binding together the first two research tasks. We extend the ONEX framework to include combinations of point-wise distances and their warped counterparts created with the GDTW framework. This research work led to the following paper:


5.1 General Exploration of Time Series Using Multiple Distances

As described in our motivating examples, different application domains favor different distances and have different interpretations of similarity. It is of utmost importance to enable analysts to understand the changes in similarity and compare the results provided by many distances. We introduced in Chapter 4 a large array of distances that can be warped and used for comparisons of sequences with different lengths and alignments. Although we devised general strategies for efficiently computing the warping path using dynamic programming, the use of these distances over large datasets is overshadowed by the high computation cost leading to increased response times. We propose here a general framework for making the exploration of large datasets using multiple distances efficient. This framework incorporates a general strategy for reducing the cardinality of data and efficient exploring datasets through rich classes of operations and using multiple distances.

Our exploratory tool empowers analysts to get unique insights into time series datasets by interactively applying rich classes of operations instantiated with multiple distances. Having the ability to use multiple distances and get answers rapidly empowers analysts to learn what distances work best to solve specific data mining tasks as well as understand the changes in similarity due to the use of diverse similarity distances. This invaluable experience comes with many challenges including: (1) integrating multiple distances in the same system (2) guaranteeing interactivity through short response times (3) enabling analysts to compare the results retrieved by specific
distances. To tackle these problems, we introduce our General Exploration of Time Series framework, or in short GENEX, which incorporates a general strategy for efficient exploration of datasets using multiple distances. Our GENEX theoretical foundation is based on demonstrating a general triangle inequality between pairs of point-wise distances and their warped counterparts. The GENEX operations include traditional similarity searches for the best match to a given sequence, discovery of motifs of different lengths and finding correlated sequences, all with accurate results and very fast response times. In addition, we provide analysts with the ability to compare results based on diverse distances and to incorporate new distances of their choice at much reduced mathematical and programming effort.

1. We create a data discovery tool that enables analysts to get better insights into time series datasets based on a large array of distances that are symmetric in coordinates and monotonically increasing. In addition, our GENEX framework enables the exploration of time series collections using domain-specific distances, while also offering a uniform method for incorporating new similarity distances.

2. Our generalized data exploration is based on applying diverse time-warping distances over compact similarity groups constructed using simple-to-compute point-wise distances instead of the raw data. Our theoretical foundation rests on proving a generalized triangle inequality between pairs of point-wise distances and their warped counterparts. This allows us to explore compact similarity groups constructed with inexpensive point-to-point distances using elastic distances only when absolutely needed, while guaranteeing accurate results and very fast response times.

3. Our compact GENEX bases leverage similarity relationships between sequences computed using easy-to-compute pairwise distances and preserved in the form of representatives. These bases play a major role in exploring sequences with fast time responses and high accuracy.

4. Our modular framework can be extended to incorporate new similarity distances with guaranteed results. Our interfaces based on template patterns insure that
adding new distances can be done with some mathematical and little program-
ing effort.

5. Our extensive experimental evaluation over the 85 datasets in the UCR archive shows how GENEX can assist analysts in finding answers to complex questions through rich classes of exploratory operations including similarity searches, best match retrieval, motif discovery, finding correlated sequences. Experiments also evaluate the size and construction times for the GENEX bases, as well as offer guidance through trade-off experimental results for specific datasets.

5.2 Theoretical Foundation of the GENEX Framework

5.2.1 GENEX Similarity Model

Definition 26 Time Series: A time series \( X = (x_1, x_2, ..., x_n) \) is an ordered set of \( n \) real values. A dataset \( D = \{X_1, X_2, ..., X_N\} \) is a collection of \( N \) such time series.

There are many distances that can be used to measure time series similarity. From the mathematical point of view, distance is defined as a quantitative degree of how far apart two objects are. The smaller the distance between two time series, the more similar they are. Generally, similarity measures are indicative of the degree of similarity (i.e. positive correlation) by considering two time series to be similar when the value of the measure is very high [15]. The similarity measures can be expressed in terms of distances, so for the remaining of this paper we will not make the distinction between the two categories and will refer to them as “distances” or “similarity distances”.

Definition 27 The subsequence of a time series \( X_p \), denoted \( (X_p)_{ij} \), is a time series of length \( i \) starting at position \( j \) where \( 1 \leq i \leq n \) and \( 0 \leq j \leq n - 1 \).

Definition 28 We define the normalized distance \( \bar{d} \) between two sequences of the same length \( n, X = (x_1, x_2, ..., x_n) \) and \( Y = (y_1, y_2, ..., y_n) \) as

\[
\bar{d}(X, Y) = \frac{d(X, Y)}{f(n)},
\]
5.2 Theoretical Foundation of the GENEX Framework

where \(d(X,Y)\) is a point-to-point distance and \(f(n)\) is specific for each distance. This function is dependent on the mathematical expression of the distance and the length of the time series.

We present in Table 16 a summary of well-known point-wise similarity distances and their normalized counterparts as used in this dissertation.

<table>
<thead>
<tr>
<th>Definition</th>
<th>Normalized distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>ED</td>
<td>( \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} )</td>
</tr>
<tr>
<td>MD</td>
<td>( \sum_{i=1}^{n}</td>
</tr>
<tr>
<td>Mink</td>
<td>( \max_{i=1}^{n}</td>
</tr>
</tbody>
</table>

**Definition 29 Similar Time Series.** Two time series \(X\) and \(Y\) are said to be similar if the chosen normalized distance \(d\) between them is within a user specified similarity threshold \(ST\).

\[ d(X,Y) \leq ST. \]

In Table 17 we provide a list of the notations used throughout the paper.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>(D)</td>
<td>Data set</td>
</tr>
<tr>
<td>(X)</td>
<td>Time series</td>
</tr>
<tr>
<td>((X_p)_i)</td>
<td>Subsequence of (X) of length (i) starting at position (j)</td>
</tr>
<tr>
<td>(G_i^k)</td>
<td>(k^{th}) similarity group of length (i)</td>
</tr>
<tr>
<td>(R_i^k)</td>
<td>Representative (k) of a similarity group of length (i)</td>
</tr>
<tr>
<td>(ST)</td>
<td>Similarity Threshold</td>
</tr>
<tr>
<td>(L)</td>
<td>Length of Subsequence</td>
</tr>
</tbody>
</table>

**Definition 30** We define the normalized Generalized Dynamic Time Warping distance \(GDTW_d\) between two time series \(X = (x_1, x_2, \ldots, x_n)\) and \(Y = (y_1, y_2, \ldots, y_m)\) as

\[
GDTW_d(X,Y) = \frac{GDTW_d(X,Y)}{2n},
\]

where again \(GDTW(X,Y)\) is the generalized \(GDTW_d\) distance. For simplicity of notations we consider here \(m \leq n\), but the two cases are interchangeable. We divide by \(2n\) in the normalization because the warping path may have length up to \(2n\).
We now introduce the notion of GENEX similarity groups. We generalize our ONEX model described in Sec. 3.1 and extend it to work with a general distance $d$, which can be any point-wise distance as defined in Definition 23.

**Definition 31** Given a group $G^i_k$ of time series of equal length $i$, then the representative $R^i_k$ of $G^i_k$ is defined as the point-wise average of the sequences in the set $G^i_k$ [25]. That is, $R^i_k = \text{avg}((X_p^i_j)_j)$, for all $(X_p^i_j)_j$ that belong to $G^i_k$.

We now introduce the notion of GENEX similarity groups that encode similarity relationships between subsequences by imposing several key requirements that, as we prove later, assure that these groups instead of the raw data can be safely explored through their representatives. The similarity groups are defined for a general distance $d$, which can be any “warpable distance”, as previously discussed in Sec. 4.3. We discuss in this chapter the $L_p$ norms, but other distances can be included as well. The ONEX similarity groups described in Sec. 3.1 are now a particular case of the generalized GENEX similarity groups.

**Definition 32** Given the set $T$ of all possible subsequences $(X_p^i_j)_j$ of the time series of dataset $D$, assume these subsequences $(X_p^i_j)_j \in T$ are grouped into similarity groups with their respective representatives $R^i_k$, such that all subsequences $(X_p^i_j)_j \in T$ are in one and only one group $G^i_k$. These similarity groups are defined to be **GENEX similarity groups**, denoted by $G^i_k$, if the following three properties hold:

1. all subsequences $(X_p^i_j)_j$ in a group $G^i_k$ must have the same length $i$,
2. $\overline{d}$ between any subsequence $(X_p^i_j)_j$ in $G^i_k$ and the representative $R^i_k$ of this group $G^i_k$ is smaller than half of the similarity threshold $ST$ used by the system, that is $\overline{d}((X_p^i_j)_j, R^i_k) \leq ST/2, \forall i, j \in [1, n] \forall p \in [1, N]$.
3. $\overline{d}$ between the subsequence $(X_p^i_j)_j$ and the representative $R^i_k$ of the group $G^i_k$ is the smallest compared to $d$ of $(X_p^i_j)_j$ and all other representatives $R^i_l$ of the same length $i$ defined over $D$, or $\overline{d}((X_p^i_j)_j, R^i_k)) \leq \overline{d}((X_p^i_j)_j, R^i_l) (\forall i, j \in [1, n]) (\forall p \in [1, N]) (\forall l \in [1, g])$, where $g$ denotes the number of representatives of length $i$.

The key requirements for placing sequences into the same GENEX similarity group are two-fold. First, $\overline{d}$ of the sequences to the representative of the group must be the
smallest compared to the $d$ to any other representative, and second, it is also smaller than $ST/2$.

**Lemma 3** For any two subsequences $X$ and $Y$ belonging to the same group $G^i_k$, with $G^i_k$ defined in Def. 32, the $d(X,Y)$ defined in Def. 13 is within the threshold $ST$, that is, $d(X,Y) \leq ST$, for all $X, Y \in G^i_k$.

The proof is very intuitive for any distances that are metrics and thus comply with the triangle inequality. This includes all $L_p$ norms. We give here the proofs for Manhattan and Minkowski distances, while the proof for the Euclidean Distance can be found in Sec. 3.1.1.

**Proof for Manhattan Distance.** According to Def. 18, we have: $\overline{MD}(X,R) \leq ST/2$ and $\overline{MD}(Y,R) \leq ST/2$

Using Def. 14, this means:

\[
\sum_{k=i}^{j} |x_k - r_k| \leq ST/2 \tag{31}
\]

\[
\sum_{k=i}^{j} |y_k - r_k| \leq ST/2 \tag{32}
\]

We want to prove that: $\overline{MD}(X,Y) \leq ST$ which means:

\[
\sum_{k=i}^{j} |x_k - y_k| \leq ST \tag{33}
\]

We have:

\[
x_k - y_k = x_k - r_k + r_k - y_k = (x_k - r_k) + (r_k - y_k).
\]

Then using this, Def. 13 and the triangle inequality equations (31) and (32), we get:

\[
\sum_{k=i}^{j} |x_k - y_k| \leq \sum_{k=i}^{j} |x_k - r_k| + \sum_{k=i}^{j} |y_k - r_k| \leq ST/2 + ST/2 = ST.
\]  \tag{34}

**Proof for the Minkowski Distance.** Under the same same assumptions and according
to Def. 18, we have:

$$\overline{M_{\text{ink}}}(X, R) \leq ST/2$$

and

$$\overline{M_{\text{ink}}}(Y, R) \leq ST/2$$

Using Def. 15, this means:

$$\max_{k=i}^j |x_k - r_k| \leq ST/2 \quad (35)$$

$$\max_{k=i}^j |y_k - r_k| \leq ST/2 \quad (36)$$

We want to prove that: $$\overline{M_{\text{ink}}}(X, Y) \leq ST$$ which means:

$$\max_{k=i}^j |x_k - y_k| \leq ST \quad (37)$$

We have:

$$x_k - y_k = x_k - r_k + r_k - y_k = (x_k - r_k) + (r_k - y_k).$$

Using Def. 13 and equations (35) and (36) we get:

$$\max_{k=i}^j |x_k - y_k| = \max_{k=i}^j |(x_k - r_k) + (r_k - y_k)| \leq \max_{k=i}^j (|x_k - r_k| + |y_k - r_k|)$$

$$\leq \max_{k=i}^j |x_k - r_k| + \max_{k=i}^j |y_k - r_k|$$

$$\leq ST/2 + ST/2 = ST/2 \leq ST. \quad (38)$$

We now generalize the idea that for any general distance $d$ we “represent” each group constructed over a data set $D$ by only one single sequence, namely, the group’s representative. We collect the representatives for all groups over $G$ for each specific distance into a collection, called the **MDRepresentative Space** ($M\text{DR}$-Space).

**Definition 33** Given a a data set $D$ and a collection of mutually exclusive groups $\{G_k^i\}$ covering all sequences of all lengths $i \in \mathcal{L}$, then the set of representatives, with $\{R_k^i\}$ the representative of $\{G_k^i\}$, along with their associated subsequences $(X_p)_j^i$ in $\{G_k^i\}$, is called the **MD**
Representative space ($\textit{MDR}$-Space).

5.2.2 General Methodology for Time-Warped Solution Retrieval based on Generalized Triangle Inequality

The cornerstone of our GENEX time-warped retrieval framework is this unique conceptual solution based on proving a triangle inequality between a general point-wise distance $\tilde{d}$ and its warped counterpart $\textit{GDTW}_d$. This allows us to create compact GENEX bases using the point-wise distance $\tilde{d}$ and explore these bases using the warped counterpart, namely $\textit{GDTW}_d$. We prove that the similarity between a sample sequence $\textit{seq}$ provided by the user and the representative of a GENEX similarity group as defined in Def. 32 "extends" to all the subsequences in that group. This empowers GENEX to perform time warped comparisons of the sample sequence over the compacted $\textit{MDR}$-Space instead of the entire dataset $\textit{D}$.

More specifically, for a general distance $d$, if $\textit{GDTW}_d$ between a sample sequence $\textit{seq}$ and the representative $R^i_k$ is smaller than $\textit{ST}/2$, then we can guarantee that all sequences in that group $G^i_k$ are similar to this sequence $\textit{seq}$ and that $\textit{GDTW}_d$ between $\textit{seq}$ and any of these sequences is within the similarity threshold $\textit{ST}$. This important property holds for any general distance $d$ that satisfies the following three ‘qualifying’ conditions:

1. $d$ is symmetric in the coordinates, i.e. if we swap some coordinates in $X$ and we make the same swaps in $Y$, then the value of $d(X, Y)$ does not change.

2. $d$ satisfies the triangle inequality, i.e. $d(X, Z) \leq d(X, Y) + d(Y, Z)$.

3. $d$ is monotone increasing in the following sense: Let us pick a subvector $X'$ of $X$ (we keep some of the coordinates from $X$) and let $Y'$ be the respective subvector from $Y$ (we keep the same coordinates). Let $\bar{X} = (X, X')$ (so we get $\bar{X}$ from $X$ by repeating the coordinates in $X'$) and let $\bar{Y} = (Y, Y')$. Then we have the following:

$$d(X, Y) \leq d(\bar{X}, \bar{Y}) \leq d(X, Y') + d(X', Y').$$
These are natural assumptions. First, without the triangle inequality, a distance \( d \) would not even be a metric. The monotonicity condition is also satisfied by many distances. For example, any distance that is a sum of base distances qualifies
\[
d(X, Y) = \sum_{i=1}^{n} d_{\text{base}}(x_i, y_i)
\].
Distances that are based on max such as \( d(X, Y) = \max(x_i, y_i) \) qualify as well.

**Lemma 4** Let \( d \) be a general distance satisfying the above 3 properties. Given \( Y' = (y'_1, \ldots, y'_n) \) an arbitrary sequence of length \( n \) in any group as per Def. 32, with the representative of the group \( Y = (y_1, \ldots, y_n) \) and a sample sequence \( X = (x_1, \ldots, x_m) \), then the following is true: If \( d(Y, Y') \leq ST/2 \) and \( \text{GDTW}_d(X, Y) \leq ST/2 \), then we have \( \text{GDTW}_d(X, Y') \leq ST \).

**Proof:** (Case: subsequences of the same length:) From the assumptions of Lemma 4 and the definition of normalized distances we have:
\[
d(Y, Y') \leq \frac{ST}{2}.
\]
(39)
Furthermore, from the definition of \( \text{GDTW}_d \) we know that there is a warping path \( P \) between \( X \) and \( Y \) from \( (1, 1) \) to \( (n, n) \) with the \( \text{GDTW}_d \) weight at most \( 2n \frac{ST}{2} = nST \). More precisely, \( P \) is a contiguous path in the \( n \times n \) grid graph from \( (1, 1) \) to \( (n, n) \). The \( t^{th} \) element of \( P \) is \( p_t = (i_t, j_t) \). Thus \( P = (p_1, p_2, \ldots, p_T, \ldots, p_T) \), where \( n \leq T \leq 2n - 1 \), \( p_1 = (1, 1) \) and \( p_T = (n, n) \). By “decoding” this path and extracting the values \( x_{i_k} \) and \( y_{j_k} \) at every position on the path, we construct the two equal-length vectors: \( X_P = (x_{i_1}, x_{i_2}, \ldots, x_{i_T}) \) and \( Y_P = (y_{j_1}, y_{j_2}, \ldots, y_{j_T}) \), where some of the \( x_i \) and \( y_j \) are repeated while advancing on the path. Then for this path \( P \) we have
\[
\text{GDTW}_d(X, Y) = d(X_P, Y_P) \leq 2n \frac{ST}{2} = nST.
\]
(40)
We now have to show that there is a warping path from \( (1, 1) \) to \( (n, n) \) between \( X \) and \( Y' \) with GDTW weight at most \( 2nST \). In fact we will show that the same warping path
$P$ will be good, i.e. we have

$$GDTW(X_P, Y'_P) \leq d(X_P, Y'_P) \leq 2nST. \quad (41)$$

From the triangle inequality we know that:

$$d(X_P, Y'_P) \leq d(X_P, Y_P) + d(Y_P, Y'_P).$$

Here from (40) we know for the first term that

$$d(X_P, Y_P) \leq nST.$$ 

Thus in order to prove (41), all we need is to prove the same for the second term, i.e. to prove

$$d(Y_P, Y'_P) \leq nST. \quad (42)$$

We get $Y_P$ (resp. $Y'_P$) by repeating some coordinates in $Y$ ($Y'$), where each fixed coordinate is repeated at most $(n - 1)$ times. Using the monotonicity condition we get an upper bound if we repeat every coordinate in $Y$ ($Y'$) exactly $n$ times. Thus we get the following upper bound using (39) and the fact that the distance is symmetric and monotone increasing:

$$d(Y_P, Y'_P) \leq d\left((Y, \ldots, Y), (Y', \ldots, Y')\right) \leq nd(Y, Y') \leq n\frac{ST}{2} \leq nST.$$

This proves (42) (here in the second expression we repeat $Y$ (and $Y'$) $n$ times).

**Proof sketch (Case: subsequences of different lengths.)** Let $Y$ and $Y'$ be subsequences of length $n$ where $Y$ is the representative of the group, $Y'$ an arbitrary sequence in the group and $X$ a query sequence of length $m$, with $m \leq n$. Without loss of generality we consider here the case of $m \leq n$ but the proof is very similar for $n \leq m$. In the $GDTW_d$ defined in Def. 30 we divide by $2n$ because the warping path may have length up to $m + n \leq 2n$. Then the matrix $M(X, Y')$ is an $m \times n$ matrix and the warping path connects $(1, 1)$ to $(m, n)$. Other than this, the proof for sequences of different lengths and the proof
for sequences of the same length are the same.

Examples of such “qualifying” distances include the $L_p$ norms, Inner Product, Intersection, Gower, Canberra, Wave Hedges, and many other distances based on sums and respectively maximums and defined in [15]. There are possibly other distances that can work with our framework, outside of the ones based on sums and maximums – we are only showcasing the ones for which a general proof exists. In addition, analysts can prove the Lemma for other specific distances on an individual basis.

Below are two examples of proving the Lemma for specific pairs of distances such as (MD, GDTW$_{MD}$) and (Mink, GDTW$_{Mink}$).

**Proof of Lemma 4 for Manhattan Distance. (Case: subsequences of the same length:)**

We know from Def. 14, Lemma 3 and assumptions of Lemma 4:

$$MD(Y, Y') = \sum_{i=1}^{n} |y_i - y'_i| \leq \frac{ST}{2}.$$

We define matrices $M(X, Y)$ and $M(X, Y')$ as in Sec. 4.2. Given the assumptions related to this case we know that there is a warping path $P$ in $M(X, Y)$ from $(1, 1)$ to $(n, n)$ with the GDTW weight at most $2n\frac{ST}{2} = nST$. We now have to show that there is a warping path from $(1, 1)$ to $(n, n)$ in $M(X, Y')$ with weight at most $2nST$. In fact we will show that the same warping path $P$ from $M(X, Y)$ will be good. Let $P$ be $P = (p_1, p_2, \ldots, p_t, \ldots, p_T)$, where $n \leq T \leq 2n - 1$, $p_1 = (1, 1)$, $p_T = (n, n)$, $p_t = (i_t, j_t)$. Then from the assumptions of Lemma 4, we have $\sum_{t=1}^{T} |x_{i_t} - y_{i_t}| \leq 2n\frac{ST}{2} = nST$. We want to prove that

$$\sum_{t=1}^{T} |x_{i_t} - y'_{i_t}| \leq 2nST.$$

We have:

$$x_i - y'_i = x_i - y_i + y_i - y'_i = (x_i - y_i) + (y_i - y'_i)$$
5.2 Theoretical Foundation of the GENEX Framework

Then using this we get:

\[
\sum_{t=1}^{T} |x_{it} - y'_{it}| \leq \sum_{t=1}^{T} |x_{it} - y_{it}| + \sum_{t=1}^{T} |y_{it} - y'_{it}|
\]

\[
\leq nST + \sum_{t=1}^{T} |y_{it} - y'_{it}|.
\] (43)

In order to estimate the second term, note that it is \(\sum_{i=1}^{n} |y_i - y'_i|\) with some of the terms repeated. The total number of repetitions is at most \(n\), since the length of the warping path is at most \(2n\). Each fixed term is repeated at most \(n - 1\) times. Thus using Equation (39) we have:

\[
\sum_{t=1}^{T} |x_{it} - y'_{it}| \leq nST + n \sum_{i=1}^{n} |y_i - y'_i| \leq nST + nST = 2nST
\] (44)

Proof sketch (Case: subsequences of different length.) Let \(Y\) and \(Y'\) be subsequences of length \(n\) where \(Y\) is the representative of the group, \(Y'\) an arbitrary sequence in the group and \(X\) a query sequence of length \(m\), with \(m \leq n\). Without loss of generality we consider here the case of \(m \leq n\) but the proof is very similar for \(n \leq m\). In the \(\text{DTW}\) defined in Def. 16 we divide by \(2n\) because the warping path may have length up to \(m + n \leq 2n\). Then the matrix \(M(X, Y)\) is an \(m \times n\) matrix and the warping path connects \((1, 1)\) to \((n, n)\). Other than this, the proof for sequences of different lengths and the proof for sequences of the same length are the same.

Proof of Lemma 4 for Minkowski Distance. (Case: subsequences of same length:)

We know from Lemma 3 and assumptions of Lemma 4:

\[
M_{\text{ink}}(Y, Y') = \max_{i=1}^{n} |y_i - y'_i| \leq \frac{ST}{2}.
\]

We define matrices \(M(X, Y)\) and \(M(X, Y')\) as previously. Given the assumptions related to this case we know that there is a warping path \(P\) in \(M(X, Y)\) from \((1, 1)\) to \((n, n)\) with the DTW weight at most \(2ST/2 = ST\). We now have to show that there is a warping path from \((1, 1)\) to \((n, n)\) in \(M(X, Y')\) with weight at most \(2nST\). In fact
we will show that the same warping path $P$ from $M(X,Y)$ will be good. Let $P$ be $P = (p_1, p_2, \ldots, p_t, \ldots, p_T)$, where $n \leq T \leq 2n - 1$, $p_1 = (1,1)$, $p_T = (n,n)$, $p_t = (i_t, j_t)$. Then from the assumptions of Lemma 4, we have $\max_{t=1}^{T} |x_{i_t} - y_{i_t}| \leq ST$. Because the Minkowski distance is based on keeping the maximum difference between pairs of coordinates, the length of the sequences does not have an impact on the similarity distance, like it does for the case of ED and MD where the distances involves summing such differences, so normalization of Minkowski is not needed. We just the Minkowski distance without normalizing it. We want to prove that

$$\max_{t=1}^{T} |x_{i_t} - y'_{i_t}| \leq ST.$$  

We have:

$$x_i - y'_i = x_i - y_i + y_i - y'_i = (x_i - y_i) + (y_i - y'_i)$$

Then using this and the above equation, we get:

$$\max_{t=1}^{T} |x_{i_t} - y'_{i_t}| = \max_{t=1}^{T} |x_{i_t} - y_i + y_i - y'_{i_t}| \leq \max_{t=1}^{T} |x_{i_t} - y_i| + \max_{t=1}^{T} |y_i - y'_{i_t}|$$

$$\leq ST/2 + \max_{t=1}^{T} |y_i - y'_{i_t}| \leq ST/2 + ST/2 \leq ST.$$  

(45)

Proof sketch (Case: subsequences of different lengths) Let $Y$ and $Y'$ be subsequences of length $n$ where $Y$ is the representative of the group, $Y'$ an arbitrary sequence in the group and $X$ a query sequence of length $m$, with $m \leq n$. Without loss of generality we consider here the case of $m \leq n$ but the proof is very similar for $n \leq m$. In the $DTW$ defined in Def. 16 we divide by $2n$ because the warping path may have length up to $m + n \leq 2n$. Then the matrix $M(X,Y)$ is an $m \times n$ matrix and the warping path connects $(1, 1)$ to $(m, n)$. Other than this, the proof for sequences of different lengths and the proof for sequences of the same length are the same.

5.3 GENEX Overview

Based on the theoretical foundation introduced in Sec. 3 we construct an exploratory tool that enables researchers, distance designers and analysts perform many important data mining tasks. As depicted in Fig. 35, GENEX is a complex system that facilitates
time series exploration instantiated by multiple distances through the following core modules:

- The Distance Manager enables analysts to search the repository of existing distances and use them for exploration. It also supports the extension of the repository by incorporating new distances in a consistent manner. If a new distance is needed, our system provides distance designers with the ability to add a chosen point-wise distance, enable it to perform time warping and therefore use it for exploration, as shown later in Sec. 6.

- The GENEX Base Manager coordinates the construction and efficient exploration of the GENEX Base according to our theoretical foundation described in Sec. 3.

- Based on the distance chosen by the analyst, the GENEX Base Constructor module pre-processes the time series collection and it places sequences in similarity groups, thus reducing the cardinality of data as shown in Sec. 5.1.

- The efficient exploration of the GENEX Base is supported by specific storage and indexing structures as described in Sec. 5.2.

- Our GENEX Query Processor enables analysts to perform rich classes of opera-
tions as shown in Sec. 7.1 based on efficient processing strategies described in Sec. 7.2.

- Our interactive analytics are complemented by a Visual Interface that empowers analysts to conduct visual analytics of the results.

In summary, our GENEX system emerges as an exploratory tool that (1) enables analysts to explore time series datasets using multiple distances collected in a distance repository and compare side-by-side the results. (2) empowers distance designers to extend the distance repository by incorporating new point-wise distances of their choice and transform them into robust alignment tools. (3) facilitates efficient exploration of time series data based on elastic distances by seamlessly incorporating these distances into our data cardinality reduction framework. (4) helps analysts discover insights that would be missed by exploring datasets based on a single specific distance and better understand the changes in similarity revealed by a plethora of distances.

5.4 GENEX Base

5.4.1 Strategies for Constructing the GENEX Base

Next we present the process of constructing the similarity groups. The goal is to construct groups that satisfy the following requirements: (1) all the sequences in the group are similar ($d$ between any two sequences in the group is within the similarity threshold) and (2) any sequence in a groups is similar to the representative of the groups within half of the similarity threshold. There are many possible strategies that can be employed to build such groups. We describe a potential strategy below.

We show in Algorithm 3 a strategy for building the GENEX Base, namely for finding the groups for specific length subsequences and their representatives. This extends [61] through optimizations to the construction strategy. Although we construct GENEX bases for each specific distance, the first two steps are independent of the choice of distance. We first decompose the existing time series in the dataset D into subsequences of all possible lengths. We then randomize the order of subsequences of each length using the well-known RANDOMIZE-IN-PLACE method [22] to remove data-related
Algorithm 3: Construction of GENEX Similarity Groups and Representatives

Input: Similarity threshold $ST$, TimeSeries $\{X\}$, Length $L$
Output: Representatives $\{R\}$, Groups $\{G\}$

begin
\{G\} = \emptyset, \{R\} = \emptyset
RandomizedINPlace(X);
\{X\} = all subsequences of length $L$
minSM=0, mink=0
for $X_p \in X$ do
    if ($G = \emptyset$) then
        $G \leftarrow G_1$
        $G_k \leftarrow X_p$
        $R \leftarrow X_p$
    else
        for $k = 1$ to Representative.count do
            minSM=closest Representative distance
            mink=Representative index
            if ($minSM \leq (\sqrt{L} * ST/2)$) then
                $G_{mink} \leftarrow X_p$
            else
                $G \leftarrow G_{K+1}$ // new Group
                $R_{k+1} \leftarrow X_p$

bias (Lines 1-5). The remaining steps have to be performed for each specific distance $d$. In lines 7-10, we construct the first group by randomly selecting a subsequence and designating it as the representative of this first group. Choosing the first representative randomly ensures that the groups are not biased by the order in which subsequences are supplied [9]. In lines 12-20, a new randomly chosen subsequence of the same length is compared with the previous representatives.

Among all representatives for which their $\bar{d}$ is smaller than $ST/2$, the group with the minimum distance $\bar{d}$ is chosen. Then the subsequence is placed into this group. If none exists, then it will be placed into a new group and designated as the representative of the new group. We repeat this until all subsequences of each specific length are placed into a similarity group. We optimize the process of finding the group where the sequence will be placed by using the following early abandoning technique. During the computation of the distance $d$ between a representative and a sequence, if we find that the distance computed so far exceeds $ST/2$, we abandon the computation and move on to the next representative. In the case that the distance exceeds $ST/2$ and we had to abandon all representatives, we just start a new group and designate this sequence as the representative of the new group. The final result is a panorama of all groups $G^i_k$ and their representatives $R^i_k$ for all possible lengths.
The complexity of the GENEX Base for each specific distance $d$ is in the worst case $O(nl^2g)$ where $l$ is the number of distinct lengths that each time series is decomposed into, $g$ the total number of groups and $n$ the number of time series in the dataset. Similar to [61], we can show that the complexity is indeed $O(n^{3/2})$, which is much better than $O(n^3)$.

5.4.2 Storage and Indexing of GENEX Base

GENEX uses a hierarchy of in-memory structures to index similar sequences of each length. Each dataset of time series which is organized into groups of similar subsequences. All groups are partitioned into sets, $S_i$, of all different lengths $i$. This allows for direct indexing into each set. Each set is a vector of groups of length $K$. This describes our Group Retrieval Index (GRI) indexed by length for quick retrieval of the set of groups for each specific length.

For each group $G_{ik}$, we maintain a Local Sequence Structure (LSS). Each group $G_{ik}$, where $k$ and $i$ index this group into $S_i$ at location $k$, maintains a list of nodes that index a sequence in the original time series set. This is done by maintaining an index field and start field. The index field determines which time series is in the time series set, and the start determines where the sequence begins. The end is determined by the length of the set it belongs to. A list is effective here because the only operations performed are adding new members and searching through the members of a group. The memory requirements are not dependent on the similarity or sparsity of the groups, which is variable dependent on the data, the threshold, and the length of the sequences.

Each group is represented by its representative. The group maintains averages and a running sum of members in the group. This representative $R_{ik}$ is used for the grouping operation to determine membership of sequences and it is also used for the similarity search to determine which group has the most similar time series. The linear formation of each list allows for a fast search within the group.

In addition, we maintain vectors for the envelopes around each representative $R_{ik}$ using $LB_{(Keogh)}$ [65], which is a well-known lower bound for pruning unpromising candidates. The envelopes built around the representatives are used in combination
with upper and lower bounds for fast retrieval of the best match representative for a given query sample.

5.4.3 Methodology for Incorporating New Distances

Our theoretical framework has the ability to support a large array of distances; accordingly we have built GENEX to enable researchers to experiment with this in practice. We also provide a simple interface from which researchers can extend and implement their own distances. We demonstrate the use of our interface on two example distances. Specifically, we first show the example of $GDTW_{Mink}$ (warped Minkowski or Chebyshev), then the more complex Cosine distance. The reason for choosing the first example is three-fold: (1) it is a popular distance in the research community (2) we documented in the introduction that its point-wise version is valuable in diverse domains, and we empowered it to perform flexible sequence matchings (3) although valuable, its computation has quadratic complexity, so it doesn’t easily scale for large datasets, but it works well under the GENEX framework. Further, we also show how to incorporate more complex such as cosine distance[15] using this methodology.

Figure 36: Class diagram of the general distance framework

To incorporate multiple distances we design a scheme based on the template pattern which supports generalizing core steps in the dynamic time warping algorithm
while offering the flexibility of incorporating distances with diverse mathematical expressions. As shown in Fig. 36, mathematical expressions defining the distances are abstracted into functions. These functions are encapsulated in classes extended from the DistanceMetric interface. The DistanceMetric interface requires researchers to implement five methods: getName(), getDescription(), init(), reduce(), and norm(). The first two functions enable the analyst to name and describe the chosen distance. The name of reduce() is inspired from functional programming. We give the specifications of these functions below.

- **init():** Cache
  
  Returns the initial Cache value.

- **reduce(Cache a, data_t Xi, data_t Yi):** Cache
  
  Combines the accumulated value a and data points from two time series Xi and Yi (data_t is the data type of each data point) according to the recursive expression of the distance function $f_d$ defined in Def. 23 to produce a Cache value.

- **norm(Cache a, TimeSeries X, TimeSeries Y):**
  
  data_t

  Calculates the numeric value of a Cache. This value can also be normalized here. Two original time series are provided so that information for normalization, such as the length, can be extracted.

In addition to the DistanceMetric interface, researchers also need to implement the companion Cache interface, whose semantics are expressed as the accumulated value in the distance recursive function (the $a$ defined in Def. 3.2). For most distances, their Cache object only contains one numeric value. However, for more complex distances, such as cosine, it contains several accumulated values in the Cache object. The method that needs to be implemented in this interface is lessThan(), which compares two Cache objects.

For example, for Minkowski or Chebyshev distance, we have the following implementation of DistanceMetric (parameters are omitted for brevity):
• \texttt{init()} = ChebyshevCache()

• \texttt{reduce()} = max(a.x, abs(Xi - Yi))

• \texttt{norm()} = a.x

In this case, ChebyshevCache contains only one value \( x \) and its \texttt{lessThan()} method only compares two caches’ values.

For the Cosine distance whose mathematical formula is more complicated (as shown below), we need to fully utilize the cache.

\[
\text{cos}(x, y) = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \sqrt{\sum_{i=1}^{n} y_i^2}}
\]

Focusing on the right side of this equation, we see that there are three separate terms, the running sum of products and the running sum of individual squares of \( x \) and \( y \) respectively. The cache will store these three values. Here the \texttt{norm} function plays a critical role.

• \texttt{init()} = CosineCache()

• \texttt{reduce()} = new\_a where {
  \begin{align*}
  \text{new\_a.sumSqXi} &= \text{a.sumSqXi} + Xi^2 \\
  \text{new\_a.sumSqYi} &= \text{a.sumSqYi} + Yi^2 \\
  \text{new\_a.sumXiYi} &= \text{a.sumXiYi} + Xi \times Yi
  \end{align*}
}

• \texttt{norm()} =
  \[
  \frac{\text{a.sumXiYi}}{\sqrt{\text{a.sumSqXi} \times \sqrt{\text{a.sumSqYi}}}}
  \]

The implementation of the CosineCache’s \texttt{lessThan()} function compares the value produced by the formula in \texttt{norm()} above.

With this framework set up, we construct both the point-wise distance \( d \) and its warping variant \( GDTW_d \) using the \texttt{init()}, \texttt{reduce()}, and \texttt{norm()} methods. We note here that the dynamic programming expression is the same for all monotonic increasing functions. The analyst only has to define \( f_d \) inside \texttt{reduce()}, while the remaining
work for “warping” $d$ is done automatically by the system. Moreover, the normalization can also implement the normalized function $\tilde{d}$ inside the $\text{norm()}$ method.

The previously defined functions are incorporated into modules of the system. The “grouping module” uses the appropriate normalized function $\tilde{d}$ to construct the similarity groups. The “similarity search module” uses the $GDTW_d$ warped variant of the chosen function $d$ to explore the similarity groups; the “motif discovery module” uses the chosen function $d$ to retrieve the repeated patterns.

In summary, our modular system enables analysts to construct point-wise distances via recursive expressions and calculate the warped counterparts in a uniform manner, then use them for specific online operations.

### 5.5 GENEX Online Query Processor

#### 5.5.1 GENEX Query Classes

All the operations performed by ONEX and described in [61] can also be performed by the GENEX query processor, with the added feature that now the user can specify what distance to use for exploring the dataset. We note that the Similarity Distance is now replaced by a Knowledge Discovery distance indicating that diverse distances can now be used for exploration. Th GENEX analytics allow analysts to control the exploration by either providing a target sample sequence $seq$ or without a sample sequence. The user-supplied sample sequence may or may not exist in the dataset as we show in our use cases below.

```sql
Q: OUTPUT X[k]
   FROM D
   WHERE Sim <= ST, [Corr >= 1 - ST^2/2], [seq = q|NULL],
   KD in {ED, MD, Mink}
   MATCH = Exact(L)|Any
```

The above general GENEX query syntax uses clauses described in Table 18.

**Class I: Similarity queries** return the best match to a user-supplied sample sequence $seq$. If “Match=Any” then sequences of every possible length are searched, otherwise
Table 18: List of clauses for GENEX

<table>
<thead>
<tr>
<th>Clause</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MATCH</td>
<td>Exact((L)) refers to a specific length (L). Any refers to any length.</td>
</tr>
<tr>
<td>(X_p)</td>
<td>Subsequence of the time series (p)</td>
</tr>
<tr>
<td>seq</td>
<td>NULL means no sample sequence is given. (X_p, q) – samples sequences provided by user</td>
</tr>
<tr>
<td>KD</td>
<td>Knowledge Discovery Distance</td>
</tr>
<tr>
<td>ST</td>
<td>Similarity Threshold</td>
</tr>
<tr>
<td>corr</td>
<td>Correlation Threshold</td>
</tr>
<tr>
<td>(L)</td>
<td>Length of Sequence</td>
</tr>
</tbody>
</table>

only sequences of length \(l\) indicated by the sample are considered.

```sql
Q1 OUTPUT \(X_k\)
FROM D
WHERE Sim <= \text{min}\ ST, seq = q
MATCH = Exact(\(L\)|Any
KD in \{ED, MD, Mink\}
```

**Use Case:** A financial analyst may want to retrieve the stock similar to the stock fluctuations of the Apple Stock for a specific time period. If the length is not specified, a similar stock of any length is retrieved. This illustrates the case when the sample sequence is a sequence present in the dataset. Alternatively, an analyst can “design” the desired stock fluctuation of interest and search for the best match for that sample sequence in the dataset. Such sequence is likely not to exist in the dataset, in which case the closest match sequence will be retrieved.

**Class II: Motif Discovery** queries allow the analyst to gain insights into the dataset by identifying similarity patterns. For example, queries like \(Q2\), given a sample time series, return the “recurring” similarity by retrieving all similar subsequences of a specific length that belong to this sample time series.
Q2: OUTPUT MotifSL \{X_p\} 
FROM D 
WHERE seq=X_p 
MATCH = Exact(L) 
KD in \{ED, MD, Mink\}

**Use Case:** In the Stock Market application, an analyst can find all 30 days long subsequences of the Apple stock having similar prices.

**Discovery of motifs of different lengths.** This extends the “motif discovery” to motifs of different lengths based on the preferred distance specified by the analyst.

Q3: OUTPUT MotifDL \{X_p\} 
FROM D 
WHERE seq=X_p | NULL 
MATCH = Exact(L)|Any 
KD in \{ED, MD, Mink\}

**Use Case.** An analyst can find all the repeating patterns of the Apple stock, regardless of length using a distance of their choice.

**Class III: Knowledge Discovery Queries.** This class of queries allows analysts to explore a dataset without using a sample query seq and find sequences whose Pearson correlation coefficient is over a specified value.

Q1: OUTPUT X_{[k]} 
FROM D 
WHERE Corr >= 1-ST^2/2, [seq = NULL], 
KD=ED 
MATCH = Exact(L)|Any

**Use case:** An analyst might be interested in finding groups of stocks whose prices are correlated for a period of three months.
5.5.2 GENEX Query Processing Strategies

Based upon our solid formal foundation (Sec. 5.2.2), our GENEX query processor (inspired from Algo. 2, but extended to use multiple distances) now applies time-warped strategies on the compact point-to-point $d$-based GENEX bases. We have discussed in detail the processing strategies and costs for the Similarity queries (Class I) and Seasonal similarity (Class II) in [61]. We focus here on the additional classes of queries specific to GENEX.

For **Discovery of motifs of different lengths queries.** The analyst provides a sample time series $seq$ and a specific length. GENEX explores the $MDRSpace$ first using GRI (Sec. 5.4.2) to retrieve the groups of that specific length. After the groups are retrieved, they are explored using LSS (Sec. 5.4.2) based on the identifier of the sample time series. GENEX returns only the sequences in each group having the same sequence identifier as the sample $seq$. Lastly, any one of this sequences is used as a query sample to find its best match representative of a different length. According to our Lemma 2, now this sequence is similar to all sequences in that best match group. We select only the sequences in the group that belong to our originally given time series. Together these sequences and the previously retrieved ones are motifs of different lengths for the given time series.

The **processing costs** for this class include: Cost(getGroups) and Cost(getSubsequences). The complexity of Cost(getGroups) is constant for each specific length. The cost for finding sequences Cost(getSubsequences) is $O(n)$ where $n$ is the number of subsequences in the group. In addition, we incur the cost of finding the best match representative of a different length, which we showed to be $O(g)$, where $g$ is the number of groups. So the total cost is $O(g)+O(n)$. There is no additional storage overhead because all needed information is already contained in the index structures.

GENEX handles **Knowledge Discovery queries** by exploring the $MDRSpace$ using GRI (Sec. 5.4.2) to retrieve groups of specific lengths. We adapt the result from [59] which allows us to explore our ED-based similarity groups and “translate” the results to express the correlation of sequences. The correlation coefficient of two sequences $X$ and $Y$ is $corr(X, Y) = 1 - \frac{ED^2(x, y)}{2m}$, where $m$ is the length the of the sequences[59].
In this case, for our sequences that are placed in GENEX similarity groups constructed using ED, according to Def. 32 this means $ED(X,Y) \leq ST$. We replace that in the equation above and get $corr(X,Y) \geq 1 - \frac{ST^2}{2}$. This establishes that all sequences in each GENEX similarity group have a correlation factor greater than $1 - \frac{ST^2}{2}$. This allows us to retrieve correlated sequences in a dataset above this correlation threshold.

**Processing costs** for this class include: Cost(getGroups) and Cost(getSeqId). The complexity of Cost(getGroups) is constant for each specific length. Thus the total complexity is constant.

**Optimizations for Processing Queries.**

We aim to devise a general strategy to work on a general distance $d$ that will efficiently retrieve the best match sequence to a given sample sequence $seq$, by designing optimizations for retrieving the best group representative and the best-match sequence within this group. Similar to [61], we develop the following optimizing strategies for GENEX:

- For a given sample sequence of length $L$, we start the search for the best match representative with the ones of the same length as the query. If we don’t find the “best match” representative for this length (the one with $d$ to $seq$ within $ST/2$), we continue by searching the representatives in decreasing order of their length until we reach the smallest length, followed by the search in increasing order of the length.

- Our strategy to find the “best match” representative of a specific length is optimized as well. We use the $LB_{Keogh}$ lower bound to build envelopes around the representatives. This only needs to be done once, and thus saves the time and space overhead that we would need if we built the envelope around each candidate instead, which allows us to “prune” many unpromising representative candidates.

### 5.6 GENEX Experimental Evaluation

Our GENEX framework can incorporate a large array of distances, but here we aim to show that diverse distances can solve specific data mining problems in diverse appli-
cation domains, rather than highlighting the merits of individual distances. Thus, we implement a select subset of distances namely, $GDTW_{ED}$ (DTW), $GDTW_{Mink}$ (warped Minkowski or Chebyshev), and $GDTW_{MD}$ (warped Manhattan). The reason for choosing these is two-fold: (1) they are well known to the research community (2) we documented in the introduction that their point-wise versions are valuable in diverse domains, yet can not perform flexible sequence matchings. Experiments are conducted on a Windows machine with 3.35 GHz Intel Core i5 processor and 64GB of RAM. The system is implemented in C++ using Qt Creator with Clang 64-bit compiler. We select our datasets from the largest public collection of time-series datasets, namely the UCR time-series collection\footnote{www.cs.ucr.edu/~eamonn/time_series_data/}. In order to make meaningful comparisons between time series, they need to be normalized \[65\]. We normalize each sequence based on the maximum (max) and minimum (min) values in each dataset. For any sequence $X = (x_1...x_n)$, we compute the normalized values for each point $x_i$ as $\frac{x_i - \text{min}}{\text{max} - \text{min}}$.

Four classes of experiments.

1. **Experiment on similarity search.** Our study focuses on comparing the best match sequences retrieved based on different pairs of similarity distances: $ED$ – $DTW$, $MD$ – $GDTW_{MD}$ and $Mink$ – $GDTW_{Mink}$. We perform two kinds of comparisons:

   - We compare the accuracy and response time of the best match retrieved using each of the above pairs of distances with the solution retrieved by the following competitors:
     
     (1) a brute-force algorithm using each respective warped distance referred to as Standard $GDTW_{ED}$, and respectively Standard $GDTW_{MD}$ and Standard $GDTW_{Mink}$ which computes all the pair-wise comparisons.
     
     (2) Piece Aggregate Approximation [45] or PAA, which finds an approximate solution by reducing the dimensionality of the data using an average approximation of the time series.

   - We analyze the tradeoff between accuracy and time response in retrieving the best match for a sample sequence when varying the similarity threshold $ST$.

2. **Experiment on Motif discovery.** We compare the motifs retrieved based on different
similarity distances: ED, MD and Mink. We show a comparison of the time responses for finding motifs using our three distances. In addition, we offer a visual display of diverse motifs found by our three distances using a specific time series.

3. **Experiment in evaluating preprocessed GENEX bases.** We create GENEX bases for the three distances (ED, MD and Mink) for all 85 datasets in the UCR collection. We evaluate the GENEX bases by measuring the size and the preprocessing time of our pregenerated information when varying both datasets and similarity threshold ranges.

4. **Using GENEX to help study and diagnose heart conditions.** We conduct a study using the public dataset MIT-BIH Arrhythmia Database [58], [31] in collaboration with an expert cardiologist to similar sequences that are only revealed by specific warped variants and would be missed by the classic DTW, leading to potential mis-diagnosis of arrhythmia related issues.

5.7 **Experiment in Similarity Search**

The similarity query class retrieves the sequence closest to a given query sample seq. Standard $GDTW_d$ compares the query sample with all sequences in the dataset using the appropriate variant of $GDTW_d$. In our experiment we vary the length of the query sequence to cover a wide range of lengths. We compute the average accuracy for each system using 20 queries of different lengths chosen to cover a wide range from the smallest to the largest length. Further, our aim is to experiment both with query sequences that might not be present in the dataset. As our use cases motivate, it is important to find the best match for a query, regardless if the sequence is present or not in the dataset. In both cases we look to find the best match, namely the solution with the closest $GDTW_d$ to the query sequence. For each specific pair of distances we repeat the following methodology:

First we randomly select 10 subsequences of different lengths from each dataset and “promote” them to become query sequences. This is our query “in the dataset” part of the experiment. Then we adopt the methodology proposed in [28], where a random subsequence is chosen from each dataset to act as the query and is taken out from that dataset instead. We do this for 10 different subsequences. This is our query “outside of
Experiment in Similarity Search

"the dataset" part of the experiment. For each of them we run each individual query in the specific dataset 5 times and average the time response per query. We then compute the average time response for the 20 queries for each dataset.

1. Accuracy. We measure the accuracy of the solution as follows: we retrieve the $GDTW_d$ between the solution and the query sequence for GENEX. We compute the error in retrieving each individual solution as the difference between the distance computed by that system and the exact solution as provided by the brute force Standard $GDTW_d$. We take the average of the error for each individual system and compute the accuracy as $(1 - \text{average(error)}) \times 100$. Since the brute-force always retrieves the best match possible and it is used as "accurate", we only discuss our findings about the accuracy of GENEX and PAA. We find that both methods lead to highly accurate results, with over 99% accuracy for each specific distance $GDTW_d$ and across all datasets. The real difference between these methods stems from the huge difference in the response time, as we describe below.

2. Response Time We show in Fig. 37, 38 and 39 the average time response of GENEX compared to brute force and PAA for each of our three distances for four datasets in increasing order of their sizes. For better visuals the time response values have been multiplied by ten and then the log value is displayed.

We note that GENEX is on average 5-700 times faster than PAA for ED, 3-340 times for MD and 2-10 times for Mink across the six datasets we used. GENEX is faster than the brute-force method 15-7500 times for ED,10-7500 times for MD and 10-130

![Figure 37](image-url)
times for Mink across the same datasets. As the datasets increase in size, the difference in the response times between GENEX and the other two methods increases, making GENEX a viable interactive system that leads to very small response times even for large datasets.

3. Trade-off between accuracy and time response As observed in the charts below, each dataset has a particular similarity threshold for which the size of the pregenerated information and the construction time are best “balanced” for each specific distance. We use this in combination with the observation of the best “trade-off” between accuracy and time to choose the most appropriate ST for specific datasets. For example, for most of our datasets these similarity thresholds are around 0.3. We indeed used these thresholds for our experiments reported in this dissertation. As shown in [61], different datasets benefit from being processed with different similarity thresholds to get the best exploratory results. Our detailed experimental results list the specific thresholds...
used to preprocess specific datasets.

Figure 40: Tradeoff accuracy vs time varying ST for ItalyPower using $GDTW_{ED}$

Figure 41: Tradeoff accuracy vs time varying ST for ItalyPower using $GDTW_{MD}$

Figure 42: Tradeoff accuracy vs time varying ST for ItalyPower using $GDTW_{Mink}$

5.8 Experiment in Motif discovery

The Motif Discovery class returns the similar subsequences of a specific length that belong to a given sample time series. Since the length is specified by the analyst, we cover a wide range of lengths from the smallest to the largest. We randomly select 5 time series from each dataset. We find motifs based on 5 different lengths for each sample. For each sample time series, we run 5 times the motif discovery query for each
chosen specific length and compute the average response time per length. We then average those response times for each sample time series. We repeat the experiment
Figure 47: Tradeoff accuracy vs time varying ST for Face using $GDTW_{MD}$

Figure 48: Tradeoff accuracy vs time varying ST for Face using $GDTW_{Mink}$

Figure 49: Tradeoff accuracy vs time varying ST for Symbols using $GDTW_{ED}$

Figure 50: Tradeoff accuracy vs time varying ST for Symbols using $GDTW_{MD}$

for each dataset and report the average running times in Fig. 52. We note that in general the response times for the three distances are very close, with th response time
Figure 51: Tradeoff accuracy vs time varying ST for Symbols using $GDTW_{Mink}$

Figure 52: Response times for motif discovery using ED, MD, Mink

for $GDTW_{ED}$ slightly larger than the other two distances, but all smaller than 2 sec, even for large datasets such as Symbols.

We compare the motifs for a specific length using ED, MD and $Mink$ and report in Table 19 the percentages of scenarios when more than one distance returned the same motif. We include here only the scenarios when all the returned motives are the same.

To give a better intuition of the motif discovery queries we also provide a visual

Table 19: Percentages of scenarios for returning the same motif

<table>
<thead>
<tr>
<th>Dataset</th>
<th>ED-MD</th>
<th>ED-Mink</th>
<th>MD-Mink</th>
<th>ED-MD-Mink</th>
</tr>
</thead>
<tbody>
<tr>
<td>ItalyPower</td>
<td>83.3</td>
<td>50</td>
<td>33.3</td>
<td>16.6</td>
</tr>
<tr>
<td>ECG</td>
<td>0</td>
<td>33.3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Face</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Symbols</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

result of motifs discovered by each of our three distances for one sample time series in the Face dataset (Fig. 53, 54, 55). In Fig. 53, the blue dashed line represents the time series selected for motif discovery (in this case is the first time series in ECG), the red shape corresponds to the first motif (here the subsequence of the first time series
starting at 0 and ending at 49 and the green shape corresponds to the second motif retrieved (the subsequence of the first time series starting at 56 and ending at 105). In Fig. 53: Seasonal similarity using ED in Face dataset

![Figure 53: Seasonal similarity using ED in Face dataset](image)

Fig. 54, the blue dashed line represents the time series selected for motif discovery (in this case is the first time series in ECG), the red shape corresponds to the first motif (here the subsequence of the first time series starting at 0 and ending at 49 and the green shape corresponds to the second motif retrieved (the subsequence of the first time series starting at 0 and ending at 99). In Fig. 55, the blue dashed line represents the time series selected for motif discovery (in this case is the first time series in ECG), the red shape corresponds to the first motif (here the subsequence of the first time series starting at 17 and ending at 66 and the green shape corresponds to the second motif retrieved (the subsequence of the first time series starting at 73 and ending at 122).

![Figure 54: Motif discovery using MD in Face dataset](image)
5.9 Experiment in evaluating preprocessed GENEX bases.

All 85 datasets in the UCR archive have been successfully pre-processed in memory using our three distances.

We display in Fig. 56 the detailed results including preprocessing times and sizes of the GENEX bases, namely the number of representatives, the total number of subsequences and the index sizes for each specific distance: ED, MD and Mink. The index sizes (in MB) correspond to the number of representatives with the average space required per representative. We include here a summary of these results through Fig. 57 depicting the GENEX bases for our three distances. The figure illustrates the number of datasets leading to preprocessed bases of specific size ranges for each of the three distances: ED, MD, Mink. We note that most datasets (83%) lead to bases of less than 833MB for all three distances, while fewer datasets (less than 9%) correspond to bases between 834-1667, 2% datasets for 1668-2500 and an average of 5% datasets lead to bases over 2500MB.

In addition, we showcase the variation in the number of representatives while using varying similarity thresholds for six datasets using our three distances: ED, MD, Mink in Fig. 58, 59 and 60. As the figures indicate, the number of representatives decreases as the similarity threshold ST increases.

We summarize the offline construction times for our pregenerated information in Fig. 61. We note that most datasets (79% for ED, MD, and 60% for Mink) take less than 666 seconds or eleven minutes to process, while only very few datasets (less than
10% for ED, MD, and less than 20% for Mink) respectively take longer than thirty-three minutes. These preprocessed bases can subsequently support any number of online operations with response times smaller than two seconds.

The GENEX bases take different amounts of time when using varying similarity thresholds. We offer in Fig. 62, 63 and 64 depictions of the variation in the construction times for varying similarity thresholds for six datasets using ED, MD and Mink.

As expected, for low similarity thresholds the construction time is higher because many groups are created. As the similarity threshold increases, fewer groups are created and thus more subsequences are grouped together. After a certain threshold the construction time remains constant.

5.10 Using GENEX to help study and diagnose heart conditions.

We explore the MIT-BIH Arrhythmia Database [58, 31] which contains 48 half-hour excerpts of two-channel ambulatory ECG recordings obtained from 47 subjects. We use our warped distances to explore this database and find the best match for a given ECG shape. For this experiment we adopt the methodology proposed in [28], where a random ECG sample is chosen from the dataset to act as the query and is taken out from that dataset. We do this for 5 different samples picked randomly from the dataset. We run each individual query in the dataset to retrieve the best match ECG using our three distances. We then ask our cardiologist collaborator to find the best match ECG for the given sequences as returned by the three distances.

Random subsequences from the sample records 100, 101, 107, 111 and 208 are selected as query sequences. Out of five samples, \(GDTW_{MD}\) returns the match as the cardiologist three times, while \(GDTW_{ED}\) returns the same sequence as the expert twice. \(GDTW_{ED}\) returned the subsequence from record 100 as the closest match to the query, \(GDTW_{MD}\) returned a subsequence from record 103, while \(GDTW_{Mink}\) returned a subsequence from record 102 as closest match.

In addition to that, we select random subsequences from the sample records 107, 108, 111, 202 and 208 as query sequences. Out of 5 samples, \(GDTW_{MD}\) returns the same best match as the expert cardiologist 3 times, while \(GDTW_{ED}\) only matches the
expert choice twice. The patient with record 111 is a female of age 47 who has been diagnosed with first degree AV block. $GDTW_{ED}$ returned as best match a subsequence from record 104, who is a female patient (age 66) whose record displays many pacemaker fusion beats with multiforms PVCs. $GDTW_{MD}$ returned a subsequence from record 114, who is a female patient (age 72) having uniform PVCs. $GDTW_{Mink}$ returned subsequence from record 102 who is a female patient (age 84) with a demand pacemaker and multiform PVCs. We then ask our cardiologist collaborator to find the best match ECG for the given sequences as returned by the three distances.

Random subsequences from the sample records 107, 108, 111, 202 and 208 are now selected as query sequences. Out of 5 samples, $GDTW_{MD}$ returns the same best match as the cardiologist three times, while $GDTW_{ED}$ returns the same match as the expert twice. Fig. 65 shows the ECG matches retrieved by each of our distances for the ECG query sample of the patient with record 107. Record 107 indicates a male patient age 63 who has a complete heart block. His ECG displays a paced rhythm with some premature ventricular contraction. Both $GDTW_{ED}$ and $GDTW_{MD}$ return as the closest match to the query sample a subsequence from record 100, a male patient of age 69. $GDTW_{Mink}$ returned as the closest match a subsequence of the record 116, a male patient of age 68 having uniform PVCs. This patient shows a sinus rhythm and some premature ventricular contraction similar to that of record 107. The cardiologist identified the same record as the one retrieved by $GDTW_{Mink}$ as having the closest heart rate, meaning the average heart rate in beats per minute. This best match identified by $GDTW_{Mink}$ would have been missed by the classic DTW and the $GDTW_{MD}$, which speaks to the need to explore the same dataset using multiple distances.

The sample EKG of patient 107 is indicative of having a pacemaker in situ. There is underlying complete heart block rhythm present. Hence the pacemaker helps the heart pump more effectively. The EKG displays a paced rhythm with some premature ventricular contractions. The premature ventricular contractions are mainly isolated beats of ventricular ectopy. The total number of isolated beats is 59. and they are arising from different points, therefore the term multiform premature ventricular contractions would be more appropriate to describe the EKG. k The sample EKG for patient 116
shows a sinus rhythm and some premature ventricular contraction. In this EKG the contractions are ventricular ectopy as well but they contain isolated beats and couplets. The total number of isolated beats is 105. There are two couplets as well. Here the premature ventricular contractions are arising from two different points, by contrast to the sample 107 in which they are multiform.

Our new distances reveal best matches that are relevant to expert cardiologists and that otherwise would not be discovered just by using the classic DTW. This result confirms our previous findings that our newly warped distances can reveal new valuable and actionable insights into medical datasets and do that now more efficiently using the GENEX framework.
Figure 56: Detailed results for preprocessed 85 datasets using $GDTW_{ED}$, $GDTW_{MD}$, $GDTW_{Mink}$.
Using GENEX to help study and diagnose heart conditions.

Figure 57: Size of GENEX base in MB for 85 datasets using ED, MD, Mink

Figure 58: Number of representatives varying ST using ED

Figure 59: Number of representatives varying ST using MD
Figure 60: Number of representatives varying ST using Mink

Figure 61: Construction times for GENEX base for 85 datasets using ED, MD, Mink

Figure 62: Offline construction time varying ST using ED
5.10 Using GENEX to help study and diagnose heart conditions.

Figure 63: Offline construction time varying ST using MD

Figure 64: Offline construction time varying ST using Mink

Figure 65: Similar ECG sequences for arrhythmia study record 107
6 Related Work

6.1 Time Series Similarity: Data Representation, Dimensionality Reduction and Indexing Methods

Many **data representation** techniques have been used for time series data mining. Although the Euclidean Distance metric is known to be sensitive to distortion along the time axis [19], [38], [45], it remains one of the most frequently used distances [27],[42],[83]. Its ubiquity is due to its ease of implementation and its time and space efficiency. The problem of time distortion is addressed by other distances including the Dynamic Time Warping (DTW)[10], Longest Common Subsequence (LCSS)[35], Edit Distance with real penalty (ERP) [18]. DTW allows non-linear alignments between two time series to accommodate sequences that are similar, but locally out of phase [66]. Its popularity in measuring time series similarity is only dampened by its high computational complexity.

To reduce the time response of DTW, **indexing** techniques [27], [43], [83] and other optimizations like early abandoning of DTW [65], cascading lower bounds to prune unpromising candidates [26], and reversing the query/data role by creating an envelope around the query sequence instead of the data [65] have been developed. These techniques are orthogonal to our work, and we indeed leverage them in our system.

**Dimensionality reduction** techniques include Discrete Fourier Transformation (DFT) [4], Single Value Decomposition (SVD) [81] and Piece Aggregate Approximation (PAA) [41]. The basic idea is to approximate the original sequences by mapping them into low-dimensional points. Such techniques typically focus on guaranteeing no false dismissal and precision rather than tackling efficiency as their main goal. Some [4] are indexable, while others [41] are not, making them not scalable for large datasets.

Like ONEX, [6] uses a **preprocessing** phase. In their case this step is necessary for converting subsequence matching to vector matching using an embedding based on DTW. However, numerous parameters have to be supplied, such as the number of reference sequences and the number of split points. Furthermore, they support queries in a range that has to be predetermined.
Range searches and nearest neighbor searches [4] remain the target queries for time series data. [26] answers such queries by finding representatives for groups of objects. The idea that the average of a set of objects is more representative than any object from that set is popular. For instance, methods like the nearest centroid classifier [77] and k-means clustering generalize the nearest neighbor classifier by replacing the set of neighbors with their centroid. Conceptually similar, [9] and [33] reduce the data cardinality by grouping similar time series. [33] represents time series as trajectories in a multidimensional space and compares their structural similarity using the multiscale comparison technique. A more recent warping-invariant-averaging based condensation framework [63] creates a classifier based on DTW leading to improvements in accuracy and reduced computation at run time. Closer conceptually to our ONEX framework, this method uses cluster centers based on the “DTW-average“ of the sequences, while ONEX uses ED for clustering and a point-wise average sequence for representatives. However, they do data editing in order to make the classifier faster and more accurate, while we have to keep the original data so we can perform accurate similarity searches. Another recent baseline [8] introduces clusters constructed using DTW and a modified DP algorithm to improve the time response. By augmenting the DP clustering and using upper and lower bounds of DTW, TADPole is an anytime algorithm that can prune unnecessary computations leading to one order of magnitude speedup over the brute-force method. However the paper is only about clustering, not about supporting similarity search, which is our task at hand.

While ONEX adopts the general idea of finding representatives for groups of objects from [33] and [23], its solution for constructing similarity groups and for further exploring them is unique, combining two well-known distances and showing to be highly effective compared to the state-of-the-art techniques.

6.2 Dynamic Time Warping Modifications

Dynamic Time Warping has been very popular in a large range of application domains including medicine [12], spoken word recognition [69], image retrieval [30] and many others. Its ability to allow more robust calculations in comparing time series is only
shadowed by the very high level of needed computations. Many optimizations of the DTW algorithm have been proposed in time. [45] introduces a modification named PDTW which applies the classic algorithm to a higher level abstraction of the data (Piece Aggregate Approximation), outperforming DTW with no loss of accuracy. Other indexing methods like [43] and [79] much improved the time response in retrieving similar sequences.

[21] focused on modifications of the original time warping algorithm by using a variable penalty to the distance metric whenever a non-diagonal step is taken, dramatically reducing the number of non-diagonal moves and improvement the alignment of chromatogram signals.

Another penalty-based method was introduced by [37] under the name of Weighted Dynamic Time Warping or WDTW, penalizing points with higher phase difference between a reference point and a testing point in order to prevent minimum distance distortion caused by outliers.

The Symmetric Time Warping algorithm [48] addressed the slope weighting and constrained the warping by changing the value of X. The Derivative Dynamic Time Warping [46] produces superior alignments between time series by replacing the Euclidean Distance with the square of the difference of the derivatives of the sequences in computing the warping path and gaining more information about the shape.

In the literature of computer graphics, [34] proposed the iterative motion warping, a method that finds a spatial temporal warping between two instances of motion captured data. [52] proposed a probabilistic method for simultaneously aligning and normalizing sets of time series using continuous profile models. [86] introduces an extension of the canonical correlation analysis for spatial temporal alignment of human motion between two subjects. [85] is an extension of dynamic time warping for temporally aligning multi-modal sequences from multiple subjects performing similar activities. This GTW provides a feature weighting layer to adapt different modalities (like video and motion capture data), and extends DTW by allowing a more flexible time warping as combination of monotonic functions.

Although extremely valuable, none of these modifications changed fundamentally
the way the warping path is computed. We propose a generalization of the Dynamic Time Warping distance, by using a general distance as a measure of the warping path.

6.3 Domain-Specific Distances.

Although many systems use the ubiquitous ED and DTW, some application domains prefer specific distances. For example, recent applications of knn to compound classification focus on selecting the most relevant set of chemical descriptors are compared under standard Minkowski distance. [40] uses the optimal weighted Minkowski distance for maximizing the discrimination between active and inactive compounds with regard to bioactivities of interest.

It has been shown [56] that the Chebyshev approximation is almost identical to the optimal minimax polynomial, and is easy to compute. Thus, the Chebyshev polynomials have been used as a basis for indexing d-dimensional trajectories.

For image retrieval, the desirable distance measure should reflect human perception. That is to say, perceptually similar images should have smaller distance between them and perceptually different images should have larger distance between them. Therefore, for a given shape feature, the higher the retrieval accuracy, the better the distance measure. Various distance measures have been exploited in image retrieval, they include Manhattan distance [75], [76], Euclidean distance [80], and Mahalanobis distance [73].
7 Conclusion and Future Work

ONEX emerges as a truly interactive time series exploration system. Our unique approach based on the combination of two similarity distances leads to improvements in accuracy of up to 19% and up to 3.8 times shorter time responses compared to the fastest known state-of-the-art method. ONEX renders more practical the exploration of large time series datasets and helps analysts better understand similarity by supporting novel classes of operations. Complemented by novel visual analytics, ONEX offers actionable insights into similarity through rich classes of operations.

Our generalization of the classic dynamic time warping approach, called Generalized Dynamic Time Warping (GDTW), preserves all its advantages, while supporting the warping of a wide range of popular point-to-point distances. Our GDTW framework allows the use of distances that are not just based on sums, but also on products, averages, minimums, and maximums. Beyond our case study which showcases the use of GDTW applied to families of well-known distances, GDTW is proven to be a powerful methodology that enables others to easily extend their own favorite point-wise distance to misaligned and different length sequences. In addition, our new $GDTW_{MD}$ distance tested on 85 benchmarks datasets emerges as a viable contender for improving the time series classification task. While our work attempts to look beyond the competitive “apple polishing” task of improving the accuracy of time series classification [18, 26], it opens the avenue for important new research. That is, other studies leveraging variants of GDTW could now further capitalize on distances created by our framework. Possible future studies on time series exploration and classification include but are not limited to:

- If we could predict which version of GDTW is most suitable on a domain-by-domain basis, we could use only this variant for that domain to assist analysts by providing the suitable distance for their application. One might achieve this by doing cross-validation on the training set; however, this may be difficult to do robustly for time series domains, which typically have very small training sets.

- If we could predict which version of GDTW is most likely to be correct on “an
exemplar-by-exemplar” basis (much like a “gating network”), we could choose the class label of the most “confident” variant [36] for classification. This idea has been used in many settings, but to our knowledge it has not been explored in the time series context. This idea might come with the overhead of computing more distance measures at run time, but all variants of $GDTW$ can exploit the lower bounding and early abandoning ideas in [65]. Thus the time considerations become inconsequential in most practical settings.

- Finally, the current state-of-the-art classifiers for time series are all Ensemble Classifiers [49]. The base methods in the Ensemble include existing variants of DTW, such as derivative DTW [46] and weighted DTW [37]. It is quite possible, according to our results, that increasing the diversity of the Ensemble with additional distances, such as our newly developed $GDTW_{MD}$, $GDTW_{Mink}$ or other of the distances that we can now warp, will further improve the accuracy of the state-of-the-art ensemble classifiers.

GENEX is a data discovery system based on our unique framework incorporating diverse pairs of point-wise and their warped variants of similarity distances to achieve real time responsiveness and yielding highly accurate results. In addition, our extensive empirical evaluation reveals novel insights into the changes in similarity revealed by the use of diverse time warped distances. GENEX emerges as a versatile interactive exploratory tool that:

1. enables analysts to explore time series datasets using multiple distances collected in a distance repository and compare side-by-side the results.
2. empowers distance designers to extend the distance repository by incorporating new point-wise distances of their choice and transform them into robust alignment tools.
3. facilitates efficient exploration of time series data based on elastic distances by seamlessly incorporating these distances into our data cardinality reduction framework.
4. helps analysts discover insights that would be missed by exploring datasets based on a single specific distance and better understand the changes in similarity revealed by a plethora of distances.
(5) empowers analysts to answer complex questions through rich classes of analytics.

Among the avenues to be explored in the future we mention the idea of expanding GENEX to perform data discovery in large public heterogeneous datasets and streaming data. Other general directions for future research include establishing hypergraph foundations for tracing high order correlations in subsets of data and extending warping of diverse distances to multiple sequences and multidimensional time series.
8 Bibliography

References


