Iterative and Expressive Querying for Big Data Series

Anna Gogolou

université PARIS-SACLAY

UNIVERSITÉ PARIS SUD
Comprendre le monde, construire l'avenir

NNT : 2019SACLS415

Thèse de doctorat

Iterative and Expressive Querying for Big Data Series

Thèse de doctorat de l'Université Paris-Saclay
préparée à l'Université Paris-Sud

École doctorale n°580 Sciences et technologies de l'information et de la communication (STIC)
Spécialité de doctorat : Informatique

Thèse présentée et soutenue à Paris, le 15 Novembre 2019, par

ANNA GOGOLOU

Composition du Jury :

M. Jean-Daniel FEKETE Senior INRIA Researcher (DR), HDR	Président
M. Christophe HURTER Professeur, ENAC Toulouse	Rapporteur
M. Pierre-François MARTEAU Professeur, Université de Bretagne-Sud	Rapporteur
M. Marco PATELLA Associate Professor, University of Bologna	Examinateur
Mme Yvonne JANSEN Senior CNRS Researcher (CR)	Examinateur
M. Theophanis TSANDILAS Senior INRIA Researcher (CR)	Examinateur
M. Themis PALPANAS Professeur, Université de Paris (Univ. Paris-Descartes)	Co-encadrant
Mme Anastasia BEZERIANOS Maître des Conférences, Université Paris-Saclay	Directrice de thèse

ABSTRACT

Time series are becoming ubiquitous in modern life, and given their sizes, their analysis is becoming increasingly challenging. Time series analysis involves tasks such as pattern matching, anomaly detection, frequent pattern identification, and time series clustering or classification. These tasks rely on the notion of time series similarity. The data-mining community has proposed several techniques, including many similarity measures (or distance measure algorithms), for calculating the distance between two time series, as well as corresponding indexing techniques and algorithms, in order to address the scalability challenges during similarity search.

To effectively support their tasks, analysts need interactive visual analytics systems that combine extremely fast computation, expressive querying interfaces, and powerful visualization tools. We identified two main challenges when considering the creation of such systems: (1) similarity perception and (2) progressive similarity search. The former deals with how people perceive similar patterns and what the role of visualization is in time series similarity perception. The latter is about how fast we can give back to users updates of progressive similarity search results and how good they are, when system response times are long and do not support real-time analytics in large data series collections. The goal of this thesis, that lies at the intersection of Databases and Visualization/Human-Computer Interaction, is to answer and give solutions to the above challenges.

In the first part of the thesis, we studied whether different visual representations (Line Charts, Horizon Graphs, and Color Fields) alter time series similarity perception. We tried to understand if automatic similarity search results are perceived in a similar manner, irrespective of the visualization technique; and if what people perceive as similar with each visualization aligns with different automatic similarity measures and their similarity constraints. Our findings indicate that Horizon Graphs promote as invariant local variations in temporal position or speed, and as a result they align with measures that allow variations in temporal shifting or scaling (i.e., dynamic time warping). On the other hand, Horizon Graphs do not align with measures that allow amplitude and y-offset variations (i.e., measures based on z-normalization), because they exaggerate these differences, while the inverse seems to be the case for Line Charts and Color Fields. Overall, our work indicates that the choice of visualization affects what temporal patterns humans consider as similar, i.e., the notion of similarity in time series is visualization-dependent.

In the second part of the thesis, we focused on progressive similarity search in large data series collections. We investigated how fast first approximate and then updates of progressive answers are detected, while we execute similarity search queries. Our findings indicate that there is a gap between the time the final answer (best answer) is found, and the time when the search algorithm terminates, resulting in inflated waiting times without any improvement. Computing prob-

abilistic estimates of the final answer could help users decide when to stop the search process. We developed and experimentally evaluated using benchmarks, a new probabilistic learning-based method that computes quality guarantees (error bounds) for progressive k-Nearest Neighbour (k-NN) similarity search results. Our approach learns from a set of queries and builds prediction models based on two observations: (i) similar queries have similar answers; and (ii) progressive best-so-far (bsf) answers returned by the state-of-the-art data series indexes are good predictors of the final k-NN answer. We provide both initial and incrementally improved estimates of the final answer.

RÉSUMÉ

Les séries temporelles deviennent omniprésentes dans la vie moderne et leur analyse de plus en plus difficile compte tenu de leur taille. L'analyse des grandes séries de données implique des tâches telles que l'appariement de modèles (motifs), la détection d'anomalies, l'identification de modèles fréquents, et la classification ou le regroupement (clustering). Ces tâches reposent sur la notion de similarité. La communauté scientifique a proposé de plusieurs techniques, y compris de nombreuses mesures de similarité pour calculer la distance entre deux séries temporelles, ainsi que des techniques et des algorithmes d'indexation, afin de relever les défis de l'évolutivité lors de la recherche de similarité.

Les analystes, afin de s'acquitter efficacement de leurs tâches, ont besoin de systèmes d'analyse visuelle interactifs, extrêmement rapides, et puissants. Lors de la création de tels systèmes, nous avons identifié deux principaux défis: (1) la perception de similarité et (2) la recherche progressive de similarité. Le premier traite de la façon dont les gens perçoivent des modèles similaires et du rôle de la visualisation dans la perception de similarité. Le dernier point concerne la rapidité avec laquelle nous pouvons redonner aux utilisateurs des mises à jour des résultats progressifs, lorsque les temps de réponse du système sont longs et non interactifs. Le but de cette thèse est de répondre et de donner des solutions aux défis ci-dessus.

Dans la première partie, nous avons étudié si différentes représentations visuelles (Graphiques en courbes, Graphiques d'horizon et Champs de couleur) modifiaient la perception de similarité des séries temporelles. Nous avons essayé de comprendre si les résultats de recherche automatique de similarité sont perçus de manière similaire, quelle que soit la technique de visualisation; et si ce que les gens perçoivent comme similaire avec chaque visualisation s'aligne avec différentes mesures de similarité. Nos résultats indiquent que les Graphes d'horizon s'alignent sur des mesures qui permettent des variations de décalage temporel ou d'échelle (i.e., ils promeuvent la déformation temporelle dynamique). En revanche, ils ne s'alignent pas sur des mesures autorisant des variations d'amplitude et de décalage vertical (ils ne promeuvent pas des mesures basées sur la z-normalisation). L'inverse semble être le cas pour les Graphiques en courbes et les Champs de couleur. Dans l'ensemble, nos travaux indiquent que le choix de la visualisation affecte les schémas temporels que l'homme considère comme similaires. Donc, la notion de similarité dans les séries temporelles est dépendante de la technique de visualisation.

Dans la deuxième partie, nous nous sommes concentrés sur la recherche progressive de similarité dans de grandes séries de données. Nous avons étudié la rapidité avec laquelle les premières réponses approximatives et puis des mises à jour des résultats progressifs sont détectées lors de l'exécuton des requêtes progressives. Nos résultats indiquent qu'il existe un écart entre le moment où la réponse finale s'est trouvée et le moment où l'algorithme de recherche se termine, ce qui entraîne

des temps d'attente gonflés sans amélioration. Des estimations probabilistes pourraient aider les utilisateurs à décider quand arrêter le processus de recherche, i.e., quand l'amélioration de la réponse finale est improbable. Nous avons développé et évalué expérimentalement une nouvelle méthode probabiliste qui calcule les garanties de qualité des résultats progressifs de k-plus proches voisins (k-NN). Notre approche apprend d'un ensemble de requêtes et construit des modèles de prédiction basés sur deux observations: (i) des requêtes similaires ont des réponses similaires; et (ii) des réponses progressives renvoyées par les indices de séries de données sont de bons prédicteurs de la réponse finale. Nous fournissons des estimations initiales et progressives de la réponse finale.

LIST OF PUBLICATIONS

The following publications and submissions under review are included in this thesis (in parts or in an extended version):

- Anna Gogolou, Theophanis Tsandilas, Themis Palpanas, and Anastasia Bezerianos. "Comparing Similarity Perception in Time Series Visualizations." In: *IEEE Trans. Vis. Comput. Graph.* 25.1 (2019), pp. 523–533. DOI: 10.1109/TVCG.2018.2865077. URL: https://doi.org/10.1109/TVCG.2018.2865077.

- Anna Gogolou, Theophanis Tsandilas, Themis Palpanas, and Anastasia Bezerianos. "Comparing Time Series Similarity Perception under Different Color Interpolations." In: *Inria Research Report, RR-9189*. 2018. URL: https://hal.inria.fr/hal-01844994v2.

- Anna Gogolou, Theophanis Tsandilas, Themis Palpanas, and Anastasia Bezerianos. "Progressive Similarity Search on Time Series Data." In: *Proceedings of the Workshops of the EDBT/ICDT 2019 Joint Conference, EDBT/ICDT 2019, Lisbon, Portugal, March 26, 2019*. 2019. URL: https://bigvis.imsi.athenarc.gr/bigvis2019/papers/BigVis_2019_paper_5.pdf.

- Anna Gogolou, Karima Echihabi, Theophanis Tsandilas, Anastasia Bezerianos, and Themis Palpanas. "Data Series Progressive Similarity Search with Probabilistic Quality Guarantees." In: *Under submission*. 2019.

Toute science est une connaissance certaine et évidente.

– René Descartes

ACKNOWLEDGMENTS

During the three years of my PhD, I have been very fortunate to interact with several amazing people. I believe that each one of them has contributed in their own way towards the completion of this thesis.

First and foremost, I would like to thank my advisors **Anastasia Bezerianos**, **Theophanis Tsandilas**, and **Themis Palpanas** without whom this dissertation would not have been possible. They have been excellent advisors, who helped me gain valuable knowledge and contribute to the progress of research in data series management and visualization. I really admired their ability to direct me into the right path producing work with novel and meaningful results. Their passion and enthusiasm were motivational for me, especially during the tough times of the PhD.

Furthermore, I would like to express my gratitude to the distinguished professors and researchers who accepted to be part of my PhD thesis defense committee: **Jean-Daniel Fekete**, **Christophe Hurter**, **Pierre-François Marteau**, **Marco Patella**, and **Yvonne Jansen**. I would like to especially thank the reviewers of this dissertation: Prof. **Christophe Hurter** and Prof. **Pierre-François Marteau**; their comments helped me finalize the final version of this thesis.

I have been truly lucky to interact with several brilliant people at Université Paris-Descartes and Inria-Saclay. I would like to thank all the members (current and past) of the diNo group, part of LIPADE, at Université de Paris (Univ. Paris-Descartes) for everything I learned next to them: Kostas, Michele, Botao, Paul, Federico, Qitong, Pavlos, Wissam, Sabiha, Khodor, Karima, and Salima. I thank them for the beautiful work environment and their unconditional support in the lab, as well as the plentiful meals we had together! I would also like to thank the members of the ILDA team of Inria-Saclay: Marie, Vanessa, Eugénie, Raphaël, Tong, Dylan, Manu, Hugo, Caroline, Emmanuel, Olivier, and Alexandra; as well as Paola, Xiao, and Sarkis from the neighbor AVIZ team.

Next, I would like to thank the funding source of my PhD, the Inria-CORDI PhD fellowship program, that made my research work possible.

On a personal level, I made a lot of new friends in Paris who have a special place in my heart. I would like to thank Evangelia and Katerina, my two wise friends, who have always been there to advise me and support me in every aspect of my life. In addition, special thanks go to Maria, Mike, Apo, Alina, Marily, Elektra, Lenia, Anna, Kostas, Martin, and Hector for their infinite support and the beautiful time we had together out of work; Mereke for the beautiful summer walks in Paris; and Maria and Revekka for the precious gym time together and the beautiful sushi nights.

Last but not least, I would like to deeply and infinitely thank my parents and my brother. They have been supporting me unconditionally all these years. Thank you for always being next to me. I love you very much!

Anna Gogolou
Paris, November 2019

To all the people who helped me
reach my goals.

Ευχαριστώ πολύ!

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1

INTRODUCTION

The development of sensor technologies in a wide range of domains, such as neuroscience, genome sequencing, earth observation, and astronomy, has led to an explosion in monitoring activities. Based on these technology advances, we are able to record large amounts of data values. These sequences of values are called *data series*. Formally, a data series $T = (p_1, ...p_n)$ is defined as an ordered sequence of points $p_i = (v_i, d_i)$, where each point is associated with a value v_i in the dimension d_i, and n is the size (or length) of the series [89, 90]. When the dimension of ordering is time, we call them *time series* (Figure 1.1). Though, a data series can also be defined over other dimensions (e.g., angle in radial profiles in astronomy, mass in mass spectroscopy, frequency in infrared spectroscopy, position in genome sequences, etc.) [90]. In the rest of this dissertation, we use the terms time series, data series, and sequences interchangeably.

It is not unusual for data series collections across many different domains to grow in the order of multiple terabytes (TBs) in size. Analysts need to interactively explore and analyze these large data series collections, formulate sophisticated queries to test predictions and rough hypotheses about emerging patterns in the data, visually assess their results, and progressively refine their hypotheses in an iterative manner. Their data exploration and analysis involve tasks such as pattern matching, anomaly detection, frequent pattern identification, and time series clustering or classification. These tasks rely on the notion of *time series similarity* [90]. Database and data-mining research has developed a wide range of techniques to automate such tasks [46]. These techniques are based on automatic similarity search algorithms, which measure the similarity between time series, and are called similarity measures.

To effectively support their tasks that require similarity search, analysts need interactive visual analytics systems that combine extremely fast computation, expressive querying interfaces, and powerful visualization tools. This thesis addresses two main challenges when considering the creation of such systems:

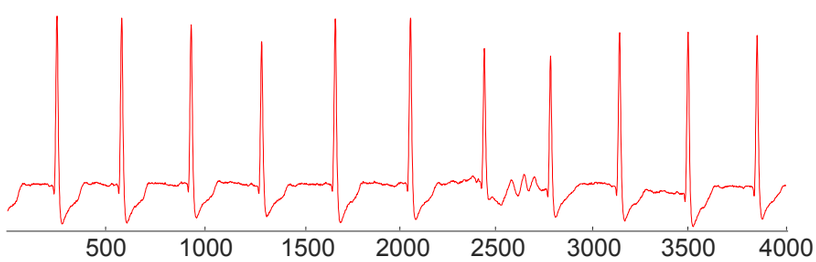


Figure 1.1.: An electrocardiogram (ECG) is a time series. It is a measurement of the electrical activity of the heart using sensors placed on the skin. The image shows the recorded values of a patient's ECG for a period of 4000 milliseconds (ms). (Image courtesy of E. Keogh)

(1) *time series similarity perception* and (2) *progressive similarity search*. In the next section, we present these two challenges and give two example scenarios that motivate them.

1.1 CHALLENGES

Before addressing the challenges of this thesis, it is worth mentioning the problems that feed them and why they are challenging.

Visual Assessment of Similarity Search Results. We demonstrate the first problem with a scenario in clinical neuroscience. Neuroscientists are looking for tools to improve the detection of abnormal epileptiform patterns [112]. They capture electroencephalography (EEG) signals from their patients using many sensors at different areas of the brain (Figure 1.2). These signals are time series data, i.e., sequences of values ordered along the dimension of time. In order to detect abnormal patterns, they often visually inspect these large data series collections and compare patterns. Although automatic solutions based on machine learning algorithms exist for this problem, the neuroscientists do not trust them (see Section 3.2), as they yield too many false positives, which makes their task even harder. Unfortunately, identifying and characterizing whether two patterns are similar requires a lot of experience, thus some of their decisions remain subjective. The way such patterns are shown to users may affect their decisions, so there is a more general research question that previous work has not yet addressed. *Do different time series visualizations affect human similarity perception and how?*

`Challenge 1: Time Series Similarity Perception.` In particular, we ask the question whether different visualizations change what patterns humans view as similar. If yes, how does each visualization communicate the similarity between patterns? Do some visualizations favor or penalize the results of similarity search algorithms? We answer all these questions in Chapter 3 by conducting a user study in the lab consisted of two main experiments and one follow-up. In this study, we compared three different visualization techniques and the results of three similarity search algorithms.

Progressive Similarity Search. We explain and motivate the second problem with a scenario from astronomy research. The scenario is inspired by a real analysis task as described by an astronomer in a design workshop* that we organized back in 2016 [91]. An astronomer needs to explore and analyze streams of data collected from several detectors over a period of a couple of years, amounting to several TBs of data series [23]. These data, which are photon counts as a function of time, represent space-time deformation or "strain" (Figure 1.3). Her goal is to look for gravitational-wave signals in the strain data, i.e., specific patterns in the strain created by the motion of very dense astrophysical objects (for instance, black-holes). She has simulated data series from the output of noise-free simulations of gravitational-wave signals (Figure 1.4). The astronomer wants to discover

* Questionnaire filled in by an astronomer during a design workshop [91], December 2016, Appendix A

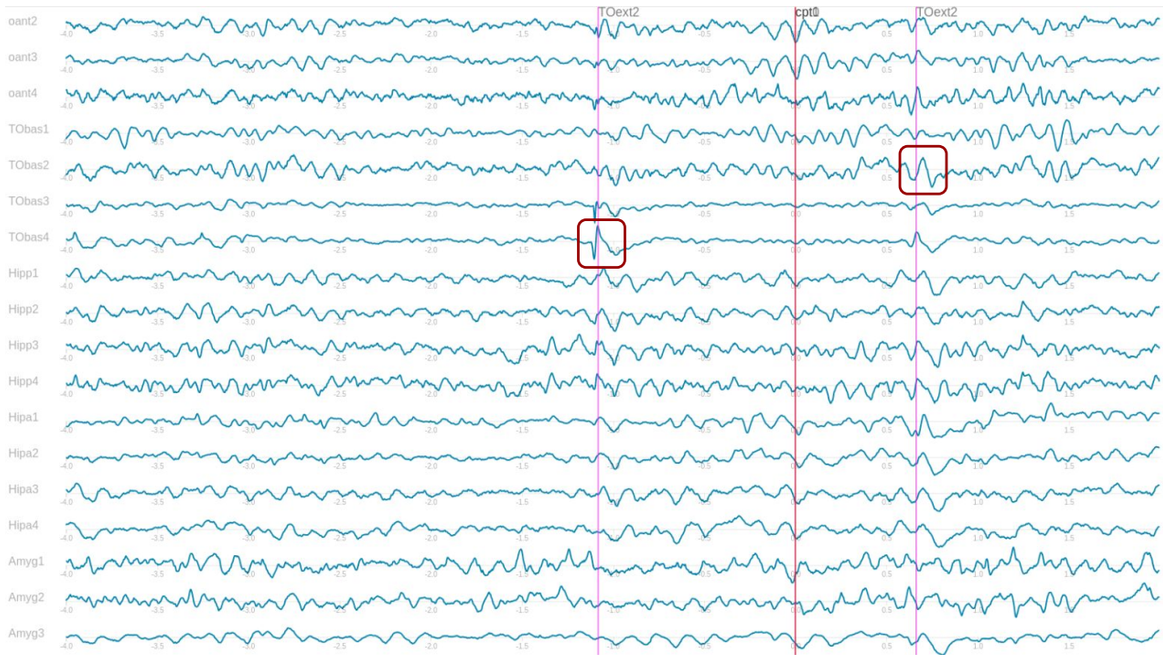


Figure 1.2.: Neuroscientists often visually inspect large collections of EEG signals, which are time series data. Vertical lines indicate manual annotations of epileptiform discharges (abnormal patterns) that neuroscientists have detected on different sensors. The particular discharges are highlighted in a red oval. Neuroscientists characterized these discharges as similar giving them the same name TOext2.

whether the simulated (predicted) signals are present in the strain. She accomplishes her task by performing pattern matching, the full analysis of which can take minutes (for the fastest algorithms making strong simplifying assumptions for the sake of speed) to months (for the slowest, most complex and computationally expensive analyses).

Therefore, she needs tools that first provide fast and approximate similarity search results and then give back updates of *progressive results*. Progressive results may not be precise, but gradually improve as the algorithm is executed (Figure 1.5). Such results should come with probabilistic error bounds that indicate how close they are to the exact solutions. The astronomer could then judge these "rough" results based on their probabilistic error bounds and other user-defined bounds of domain-specific parameters, such as the position of the signal in the sky. But two challenges arise: (i) how fast we can give back to users first approximate results (e.g., in the order of milliseconds, seconds, or minutes) and how good they are (i.e., how close to the final result); and (ii) how we can efficiently and effectively compute quality guarantees (probabilistic error bounds) of progressive results and communicate them to users.

`Challenge 2: Quality Guarantees of Progressive Results.` How to compute progressive results and estimates of probabilistic distance bounds, in order to help users decide when to stop the search process, is an open research question. In particular, we examine the quality of progressive query answering and what probabilistic guarantees we can provide users. In Chapter 4, we demonstrate through experiments using several large data series datasets (synthetic and real) and the state-of-the-art data series indexing techniques that we are

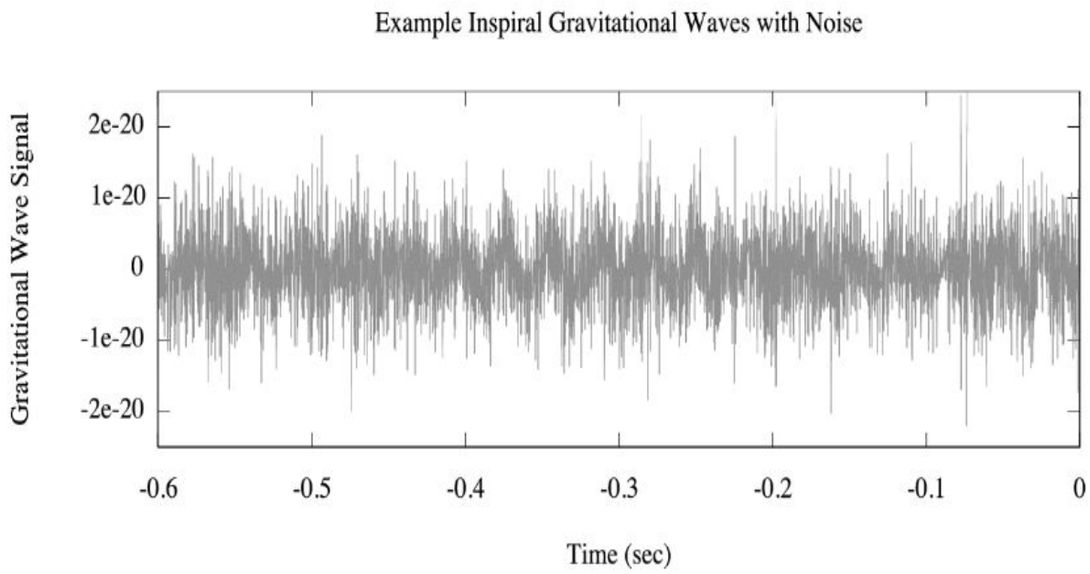


Figure 1.3.: The plot shows a **stretch** (0.6 sec) of noisy strain data collected from a detector. They are plotted as a function of time in seconds. They are usually filtered to remove the noisy low and high frequency content. (Image source: www.ligo.org)

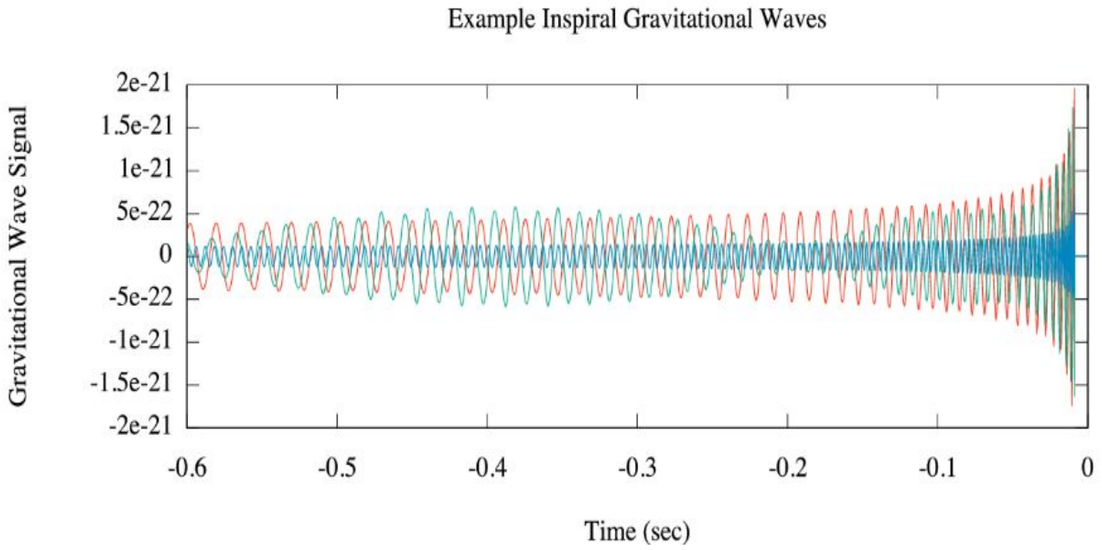


Figure 1.4.: Three noise-free simulated gravitational-wave series of length 0.6 sec. These signals are specific patterns in the strain created by the motion of very dense (for instance, black-holes) astrophysical objects. The astronomers' goal is to look for these patterns in the strain data (Figure 1.3). (Image source: www.ligo.org)

able to give back to users first approximate results almost immediately (in the order of milliseconds), and then updates of progressive results that fast converge to the final solution. Next, we develop and present a new probabilistic method for efficient and effective computation of distance distributions and error bounds of progressive similarity search results. We leave for future work the still open challenge of how to communicate visually and statistically such probabilistic error bounds to users without misleading them and misguiding them to wrong decisions.

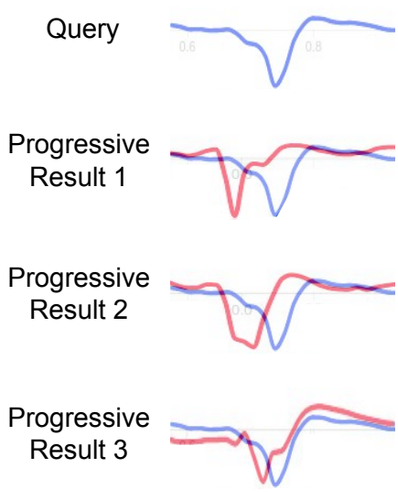


Figure 1.5.: Example of progressive results, which are not the final and exact ones, but are getting better (closer to the query) as progressive similarity search is executed.

This PhD thesis lies at the intersection of different research areas of Computer Science: *Database Management Systems*, *Human-Computer Interaction*, and *Data Visualization*.

1.2 THESIS STATEMENT AND OVERVIEW OF CONTRIBUTIONS

The thesis main contributions are:

- The first investigation of whether different time series visualizations affect similarity perception. We studied which visualizations promote or penalize results from similarity search algorithms, and provided guidelines on visualizations to use according to domain-dependent definition and notion of similarity.
- An investigation of the quality of early progressive results through a set of computational benchmarks using large data series datasets, both synthetic and real.
- The introduction of a scalable probabilistic method that provides quality guarantees (error bounds) for progressive similarity search query answering in massive data series collections, by building prediction models; and the comparison of different such models when it comes to their speed and quality.

In the next Chapters, we provide details of the results of this dissertation with respect to the above points.

1.3 OUTLINE OF THE THESIS

The rest of the dissertation is organized as follows. In Chapter 2, we present background material and prior work on data series visualizations and analytics that span from user studies related to time series perception to similarity

search algorithms, database management systems for fast similarity search in large data series collections, querying interfaces for interactive similarity search, and progressive visual analytics solutions. In Chapter 3, we present our work on similarity perception under different time series visualizations; we present the two main experiments of this study (experimental design, results), as well as the results of a follow-up experiment. In Chapter 4, we focus on progressive similarity search; in particular, we demonstrate through experiments why progressive similarity search is useful (and necessary) in large data series collections and how fast we can provide first approximate results to users, and then updates of progressive results. We propose a new probabilistic method for computing quality guarantees (error bounds) for progressive similarity search results. Finally, in Chapter 5, we offer concluding remarks and future insights on the next steps of data series visualizations and analytics.

2

RELATED WORK

We discuss now previous work on time series visualizations, similarity search, and perception, in particular with respect to time series similarity. We start off by introducing existing time series visualizations and studies that have been done with them investigating perceptual aspects. Next, we present research from the data mining community on data series similarity search. Data mining research has developed plethora of similarity search algorithms to measure time series similarity. The choice of the right measure depends on the data domain, which defines sometimes roughly and other times rigorously the similarity between patterns. All these works compose the background material for our time series perceptual study, which we present in Chapter 3.

In the second half of the chapter, we discuss previous work on database management systems for fast similarity search, interactive querying interfaces for interactive similarity search, and progressive visual analytics systems for progressive query answering. In particular, the database community has developed advanced techniques (e.g., indexes) for fast similarity search in large data series collections. Many querying interfaces take advantage of these back-end solutions. They combine interactive interfaces of time series visualizations with effective back-end techniques for fast and interactive similarity search. However, these systems cannot support the analysis and visualization of massive data series collections without the adoption of progressive technologies. The last section is devoted to progressive query answering and progressive visualizations, which are widely called progressive visual analytics. These last sections compose the background material for our work on data series progressive similarity search, which we present in Chapter 4.

2.1 TIME SERIES VISUALIZATIONS

Time series are sequences of values changing over time. Line Charts [117] are the first visualization for time-varying data dating back to 18th century, when Lambert and Playfair started using them (Figure 2.1). Today, Line Charts are widely known and established as the simplest and most common visualization for time series. They usually map time to the horizontal x-axis, and value to the vertical y-axis, and utilize position to place connected-by-line points in a two-dimensional (2-D) Cartesian coordinate system. However, Line Charts are not the only visualization for time series. Several visualizations have been introduced to plot data as a function of time (see [2, 84] for an overview of time oriented visualizations).

In particular, different visualizations serve different characteristics of the data (abstract or spatial, univariate or multivariate), of the time (linear or cyclic, instant or interval), and of the visual representation itself (static or dynamic, 2D

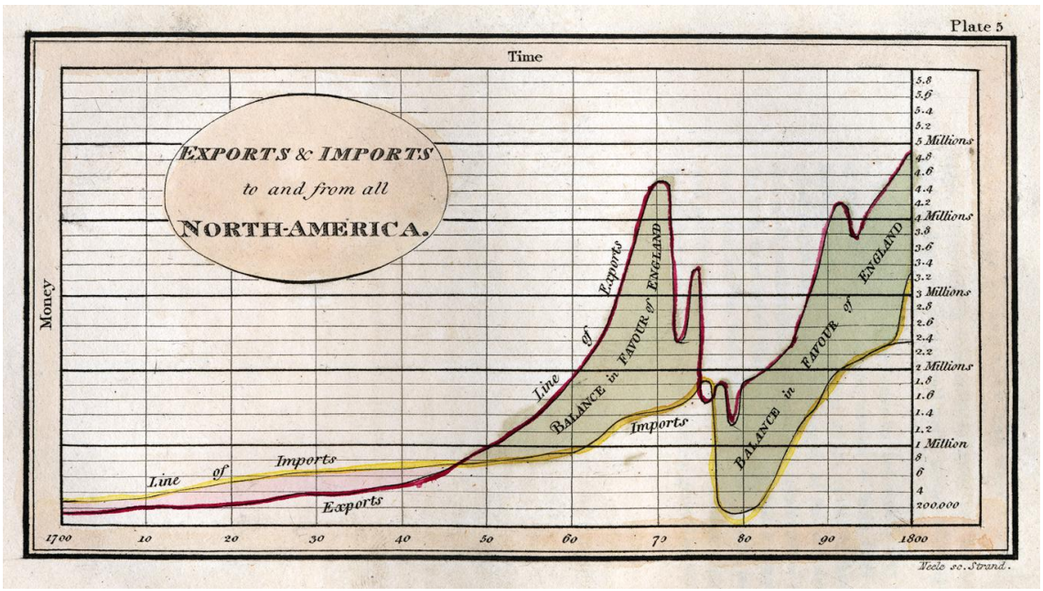


Figure 2.1.: Playfair's trade-balance time-series chart between England and North America in the 18th century, Plate 5, Commercial and Political Atlas and Statistical Breviary, 1786.

or 3D) [2]. In this thesis, the main focus is given on linear, multivariate data and 2D time series visualizations. This is the kind of data that come from multiple sensors placed on the scalp (neuroscience), or earth and space observation detectors (astronomy), that neuroscientists and astronomers explore in 2D linear plots (see Chapter 1). Nonetheless, other visualizations would also be of interest for this kind of data, e.g., visualizations that communicate the periodical nature of the data using spirals [13, 115, 127] (Figure 2.2). Through these visual representations neuroscientists might be able to spot periodical epileptiform abnormal patterns or astronomers periodical events in the sky, but they don't use them, because they do not scale well for large cardinalities of series.

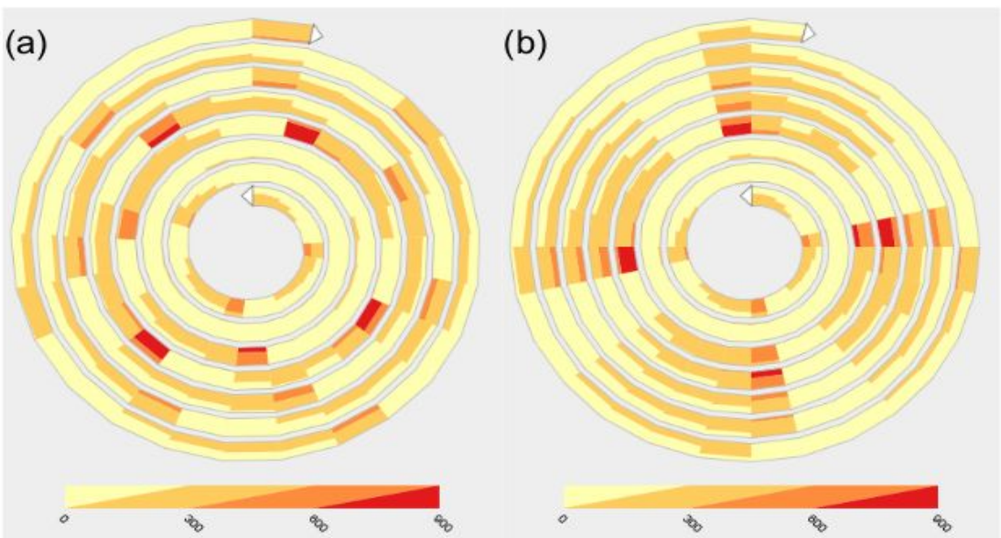


Figure 2.2.: (a) A time series in a spiral (cyclic) view. Color is used for the encoding of values. No periodical patterns are visible. (b) The same time series in a spiral with different cycle length. The example shows that with the right parameterization periodical patterns are disclosed. (Image source: Tominski et al., Enhanced Interactive Spiral Display, 2008)

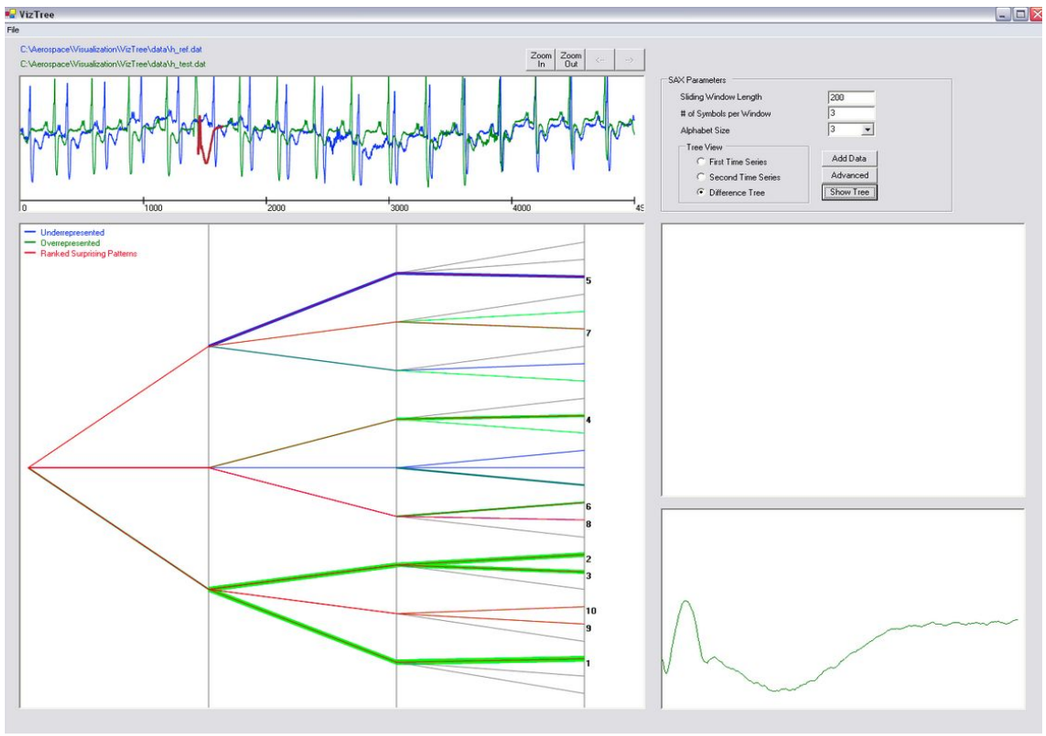


Figure 2.3.: VizTree. A tree-based clustering visualization of similar patterns. The image shows the differences in pattern distributions of two ECG datasets (blue and green). The surprising/abnormal patterns are highlighted in red. Clicking on the branch ranked 1, the anomalous heartbeat in the green time series is shown (also highlighted in the time series window). (Image source: https://cs.gmu.edu/~jessica/viztree.htm)

Scalability is one aspect that has received considerable attention in time series visualizations. Some visualizations aggregate points of long time series (e.g., [12, 67]), others aggregate multiple time series or their segments trough clustering (e.g., [74, 76, 120]), and yet others focus on examining how to interactively explore and compare a set or a subset of time series (e.g., [132, 133]) (Figure 2.3). One of the oldest visualization approaches is to present Line Charts in small multiples [117] or sparklines [80], i.e., small time series charts embedded in tables, text, or other graphs. More recent approaches extend the Line Chart representation itself. For example, the two-tone pseudo coloring and Horizon Graphs [93, 99, 102] split the vertical range of values in a Line Chart into a few horizontal bands, that are then colored and superimposed (Figure 2.4). This representation saves vertical space, while maintaining the overall line shape. Others address scalability using color-based representations, often referred to as heatmaps or Color Fields (Figure 2.5). Instead of using position to encode the range of values over time (as is done in Line Charts), these visualizations use vertical color strips, whose color saturation or brightness encodes value. This approach is seen in many systems [3, 32, 86, 102] and scales well as multiple such sequences of small height can be stacked together [69, 111]. For example, Line Graph Explorer [69] (Figure 2.6) colors Line Charts based on their y-value and allows users to focus on some of them, while the rest are each collapsed into Color Fields colored based on the y-value at any point. As remarked by Javed et al. [63], in order to represent multiple time series, the above representations split the space (mainly

vertically) and attempt to optimize the vertical footprint of each individual time series.

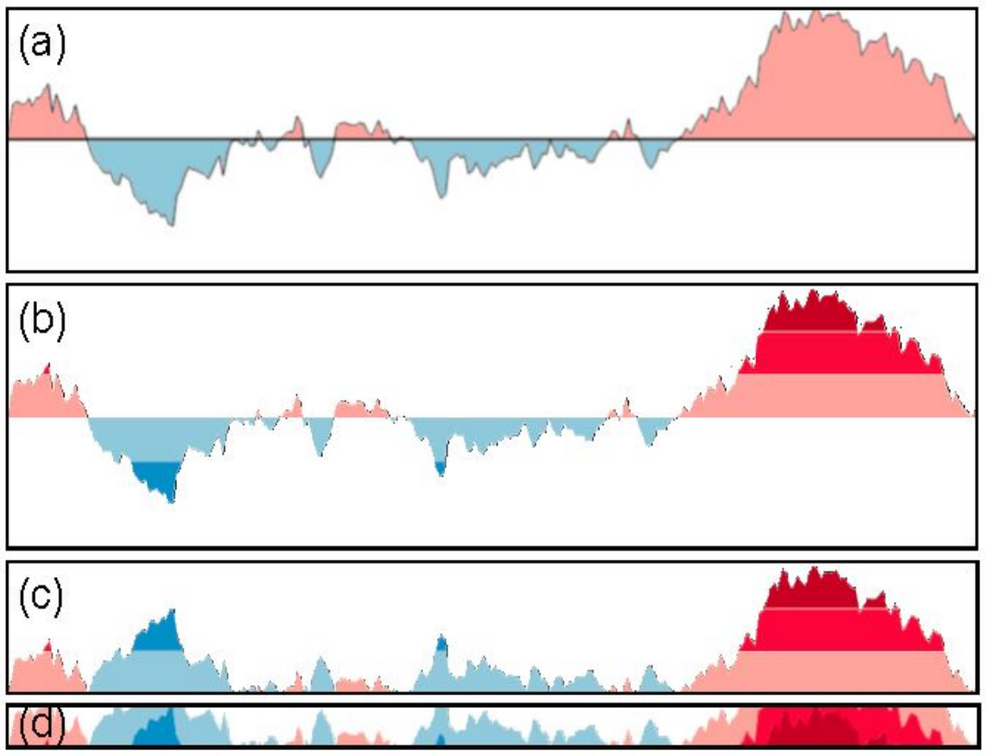


Figure 2.4.: The development of Horizon Graph. (a) The Line Chart is divided into negative (blue) and positive (red) values. (b) 6 horizontal, same-width bands (3 negative and 3 positive) split the Line Chart, that are then colored from light to darker saturations for the extreme low and high values. (c) The negative bands are mirrored above the x-axis, and (d) all bands are superimposed. The final Horizon Graph occupies 1/6 of the original Line Chart's vertical space, offering a compact visualization for time series. (Image source: Perin et al., Interactive Horizon Graphs: Improving the Compact Visualization of Multiple Time Series, 2013)

Alternatively, multiple visualizations can occupy the same space [63]. Multiple Line Charts, often of different colors, can be superimposed or can be replaced by variations of area charts that attempt to optimize space (e.g., stacked [18] or braided [63] graphs). For example, stacked graphs or their variation streamgraphs [18, 56] (Figure 2.7a) are colored area graphs that do not use a common baseline, instead each time series is drawn using the previous series as a baseline. The series are displaced around a central axis resulting in a flowing (river) shape. Braided graphs [63] (Figure 2.7b) are area graphs with a common baseline, that ensure that all curves are visible by sorting the curves so that the highest values are rendered at the back, resulting in a braiding effect. The majority of these space sharing techniques do not scale well for a large number of time series due to clutter. Moreover, the stacked variations, which do not have a common baseline, could complicate comparison tasks such as determining similarity. We focused on techniques that split space, as our motivating scenario for similarity search and similarity perception (see Section 3.2) revealed that it is important to be able to see a large number of time series together.

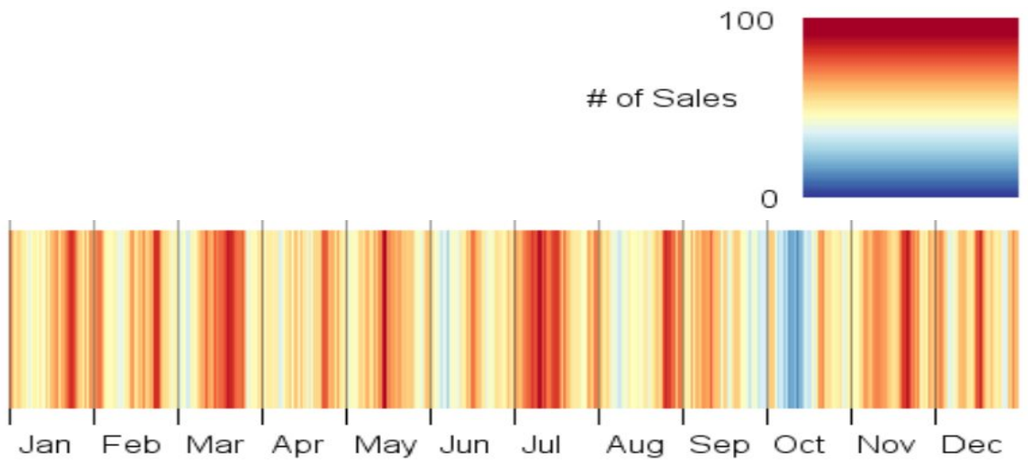


Figure 2.5.: A time series visualized as a Color Field, where y-values are encoded as colors. The color mapping is based on a continuous, usually interpolated, color scale of one, two, or multiple colors. In the example, the 3-tone color scale goes from blue for the lowest to red for the highest values utilizing yellow shades for the median values. (Image source: Correll et al., Comparing Averages in Time Series Data, 2012)

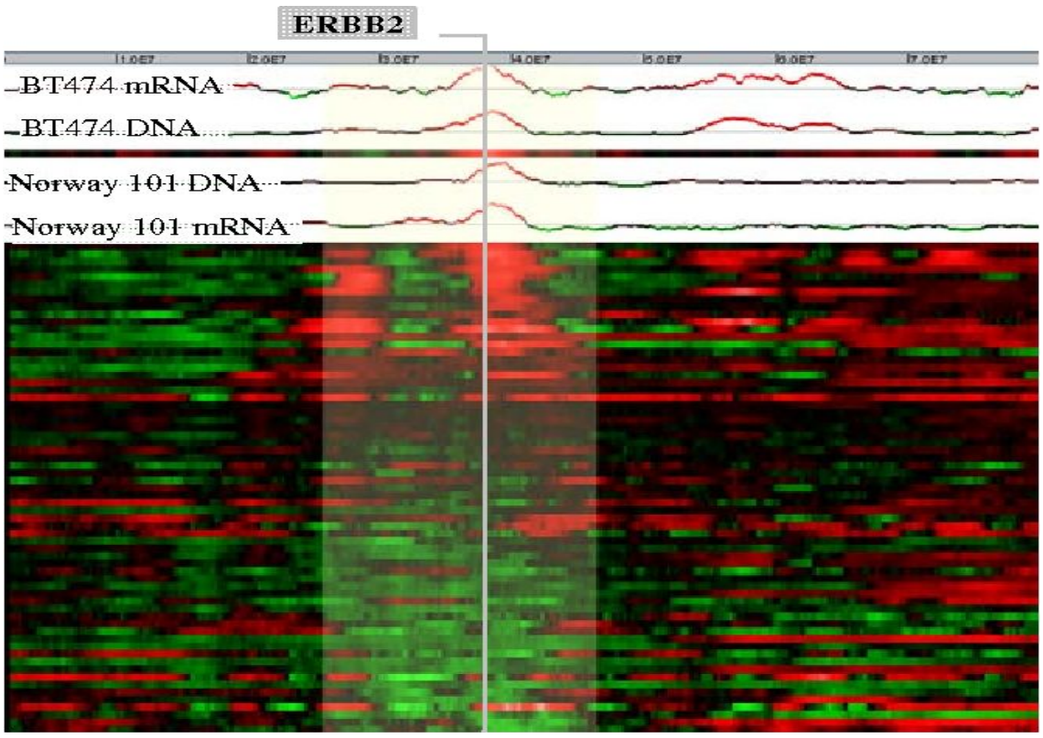


Figure 2.6.: Line Graph Explorer. Gene data visualized as heatmaps or Color Fields. Line Charts are colored based on their y-value, allowing users to focus on some of them, while the rest are each collapsed into Color Fields colored based on the y-value at each point. The expanding area shows four open Line Charts. Thousands of time series are visualized utilizing the minimum possible space. (Image source: Kincaid & Lam, Line Graph Explorer: Scalable Display of Line Graphs using Focus+Context, 2006)

2.2 STUDIES ON TIME SERIES PERCEPTION

A number of perception studies have compared different time series visualizations under a variety of tasks, in particular visualizations that use positional

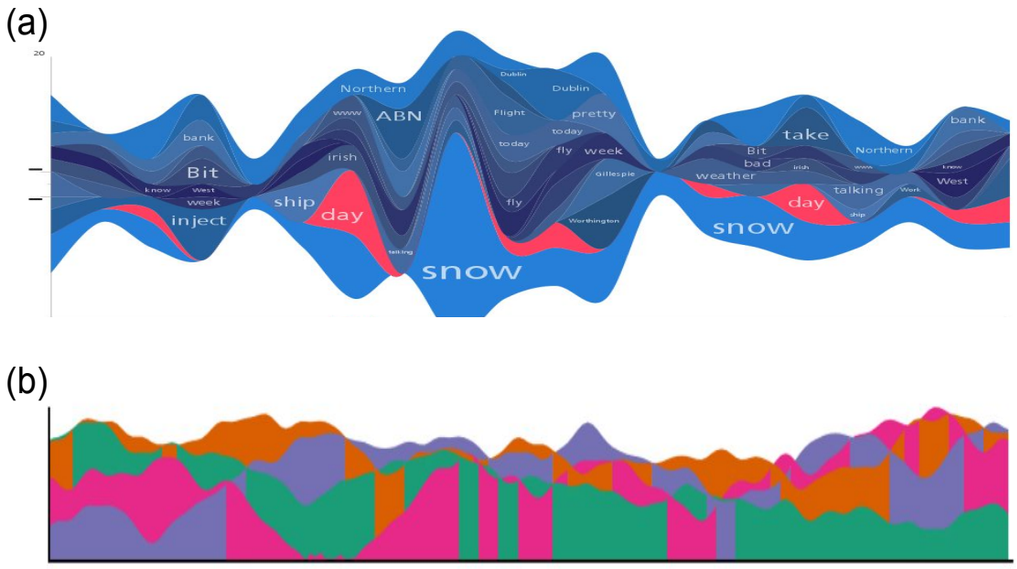


Figure 2.7.: Shared-space techniques for the visualization of multiple time series. (a) Stacked graphs or streamgraphs are colored area graphs disturbed around a central axis resulting in a flowing (river) shape. The image shows Twitter keyword trends over a period of time. (Image source: Ronan Lyons, 2009) (b) Braided graphs are area graphs with a common baseline. Areas are cut at points where the curves change hierarchy and are sorted with the highest values rendered first in the back, ensuring that all curves are visible. (Image source: Javed et al., Graphical Perception of Multiple Time Series, 2010)

vs. color encodings, linear vs. cyclic settings, and shared-space vs. split-space techniques.

`Position vs. Color.` Correll et al. [32] investigated the efficiency of representations using either position (Line Charts) or color (Color Fields) when estimating averages. They found that people are better at estimating high-level statistical overview tasks, such as averages, when using Color Fields. Albers et al. [3] compared eight different time series visualisations that used both positional and color encodings (among other variations). They found that positional visualizations like Line Charts were more efficient for tasks requiring point comparisons (e.g., minima, maxima, range), whereas color once again performed better for summary comparisons (e.g., averages, spreads from average). On the other hand, Heer et al. [57] compared Line Charts with variations of Horizon Graphs for a value comparison and estimation task. In particular, subjects were comparing two points, a point on one graph to a point on another of the same-type graph, reporting which point represented the greater value and then estimating the absolute difference between the two. Heer et al. focused mainly on the effects of chart size and layering and found that Horizon Graphs performed better than Line Charts for small chart sizes. Later, Perin et al. [93] improved the efficiency of Horizon Graphs for maxima, value comparison, and series identification tasks by allowing an interactive adjustment of the band baseline.

`Linear vs. Cyclic.` Adnan et al. [1] compared the effectiveness of three time series visualisations that utilize position, color, and area for the encoding of time series. They tested the same visual encodings in both linear and polar (cyclic) layouts (see Figure 2.8). Their results, which agree with previous study results [3],

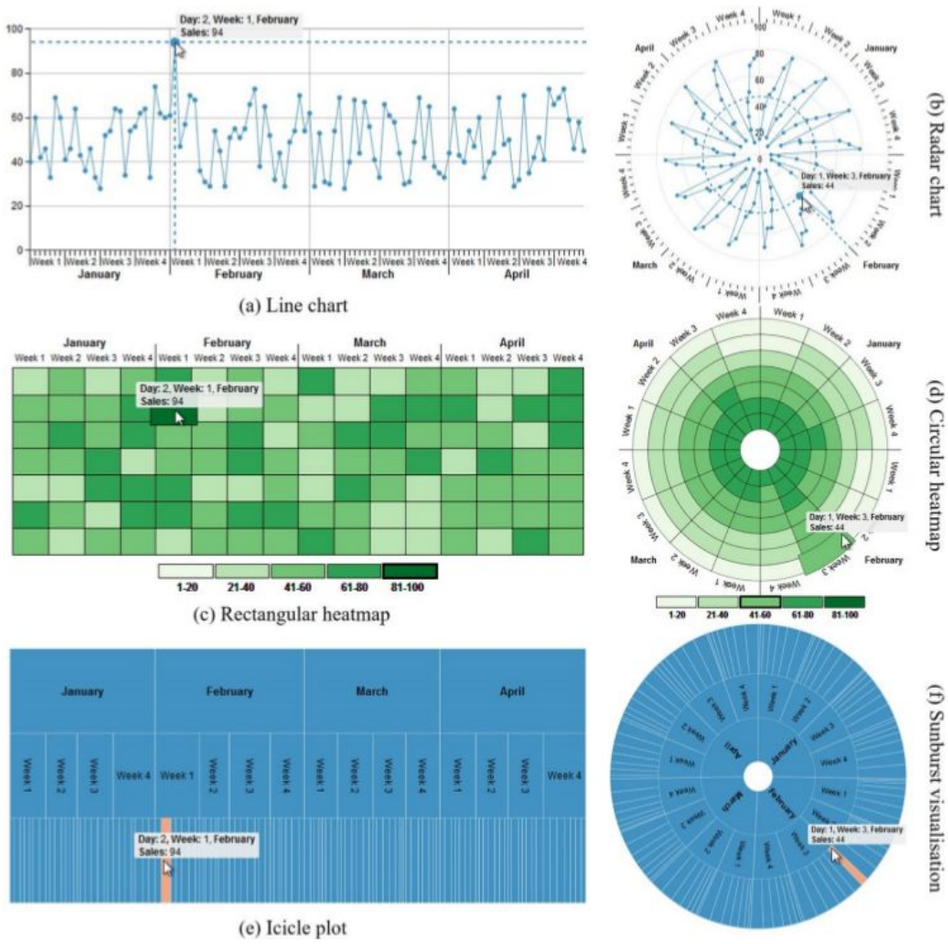


Figure 2.8.: Three visualizations that utilize (a) position, (c) color, and (e) area to visualize time series in both linear (a, c, e) and polar (cyclic) (b, d, f) layouts – tested for maxima, minima, trend detection, and aggregate estimation tasks. (Image source: Adnan et al., Investigating Time Series Visualisations to Improve the User Experience, 2016)

indicate that for point comparison tasks, such as maxima, minima, and trend detection, positional visual encodings are more effective, while for aggregate estimation tasks, area visual encodings perform better than the previous-studied color visual encodings. They also found that linear settings are generally comparable or more effective than the polar ones. Several works have also considered tasks on multiple time series. Fuchs et al. [47] studied glyphs, presented in small multiples (see Figure 2.9). Position/length and color were among the variations used for the different glyph designs for the encoding of data values. For temporal-location encoding, radial (cyclic) vs. linear layouts were considered. They did not test aggregation tasks, but they found that for peak and trend detection tasks, line glyphs worked best. On the other hand, radial encodings were better for discrimination tasks (i.e., comparing values at specific temporal locations).

`Shared-space vs. Split-space.` Javed et al. [63] compared visualization techniques which split (small multiples, horizon graphs) or share the same space (line graphs, braided graphs) using color to differentiate the different series. They tested them under peak, trend, and discrimination (value comparison) tasks. They found that while shared-space (superimposed) techniques worked well for small numbers of time series, they did not scale well due to both clut-

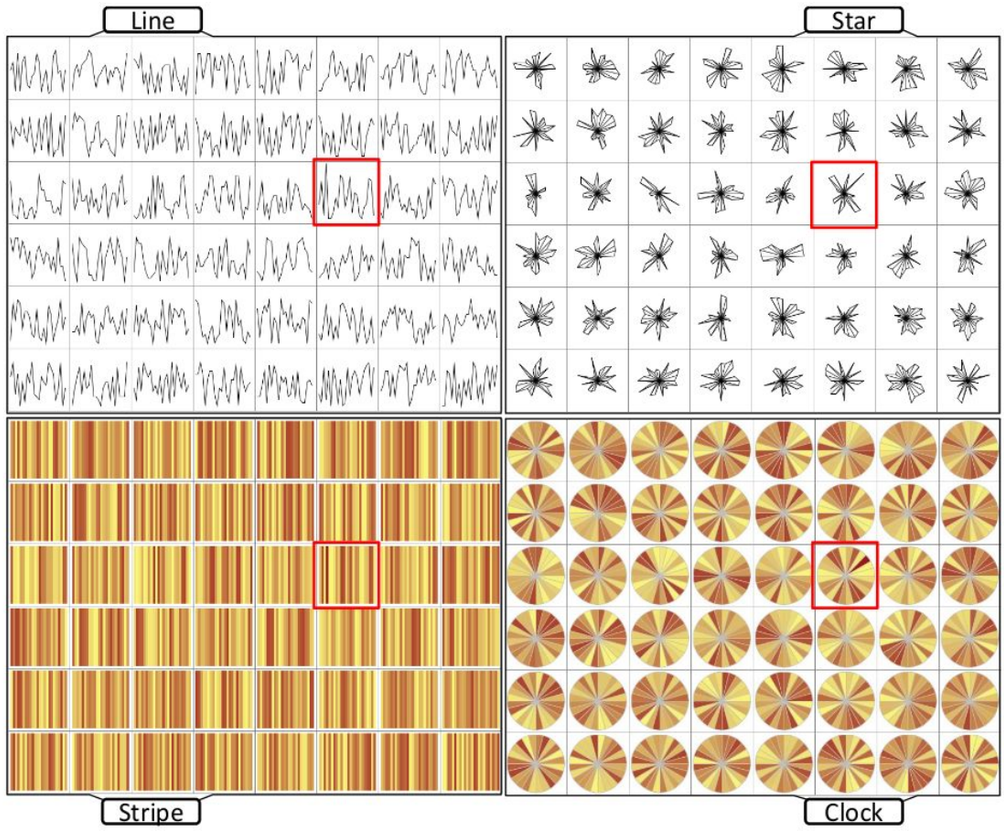


Figure 2.9.: Four different visualizations in small multiples – tested for peak, trend, and discrimination (value comparison) tasks. Upper row: position/length encodings, lower row: color encodings, left column: linear settings, right column: radial (cyclic) settings. (Image source: Fuchs et al., Evaluation of alternative glyph designs for time series data in a small multiple setting, 2013)

ter and difficulty in color disambiguation. On the other hand, split-space ones worked better for large numbers. Horizon Graphs were faster than Line Charts for discrimination tasks, but slower for peak and trend detection tasks. Table 2.1 summarizes the tasks and the characteristics of the best performing visualizations according to the previous studies.

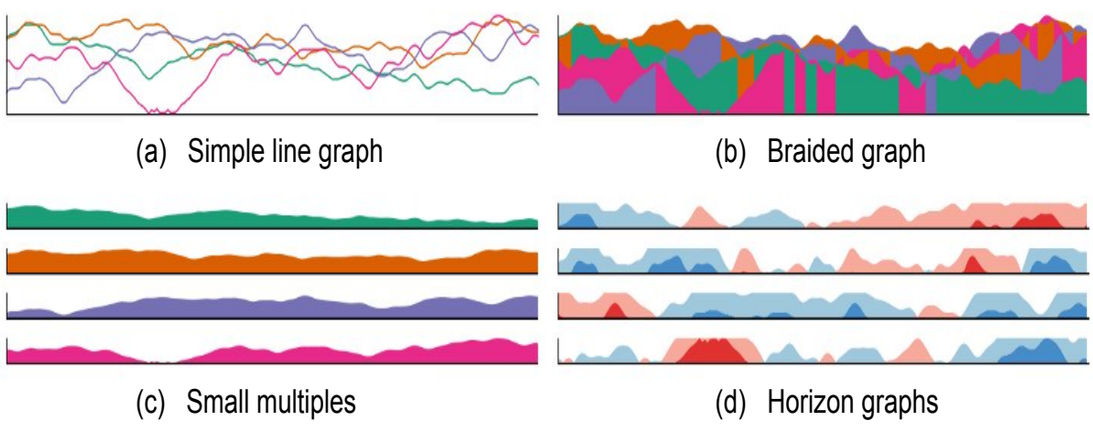


Figure 2.10.: Shared-space (a, b) vs. split-space (c, d) techniques – tested for peak, trend, and discrimination (value comparison) tasks. (Image source: Javed et al., Graphical perception of multiple time series, 2010)

Table 2.1.: Time series perception: Tasks and best-performing visualizations.

Task	Best-performing visualizations
`Peak detection (maxima, minima)`	position (Line Charts) [1, 3, 47, 63]
`Trend detection (range)`	position (Line Charts) [1, 3, 47, 63]
`Discrimination (value comparison)`	position and color (Horizon Graphs) [57, 63, 93]
`Aggregate estimation`	color (Color Fields) [3, 32]

Similarity search likely involves both point comparisons, such as finding maxima and minima, and overview comparisons, such as comparing the overall shape of patterns. It is thus unclear if position or color-based visualizations are best suited for similarity tasks. In Chapter 3, we focus on three linear, split-space visualization techniques that rely on position (Line Charts), color (Color Fields), or both (Horizon Graphs). These techniques can also scale well to multiple time series when presented as small multiples, supporting our motivating domain (neuroscience).

2.3 TIME SERIES SIMILARITY

Analysts often define a subsequence of interest as *query* and use automated tools to search for similar patterns. We discuss now data-mining research on similarity search algorithms, and then visualization research on how to specify similarity between queries and evaluated results. We highlight the few studies that perform subjective user evaluation of similarity search results.

2.3.1 *Similarity Measures*

Data-mining research has proposed a plethora of algorithms (distance measures) that assess the distance between time series. Each one computes similarity between patterns in a different way according to the data-domain definition of similarity. The difference lies on how the algorithm treats signal deformations when measuring for similarity. For example, measures are considered to be invariant to one or more signal characteristics (e.g., amplitude), when they match patterns by eliminating these signal variations. Ding et al. [35] group them in four categories:

`Lock-step`: Lock-step measures, such as the Euclidean Distance (ED) [42], perform point-by-point value comparison between two time series comparing the i-th point of one series to the i-th point of another (Figure 2.11). ED treats signals without distorting them and any variation between them adds in their distance; thus ED is a non-invariant measure. On the other hand, ED can be combined with data normalization, often called *z-normalization* [52], which transforms time series into new ones of the same length that have zero mean and standard deviation one. ED with z-normalization considers as similar, patterns that may vary in amplitude and y-offset (Figure 2.12). As a result, ED with z-normalization becomes invariant to amplitude and y-offset.

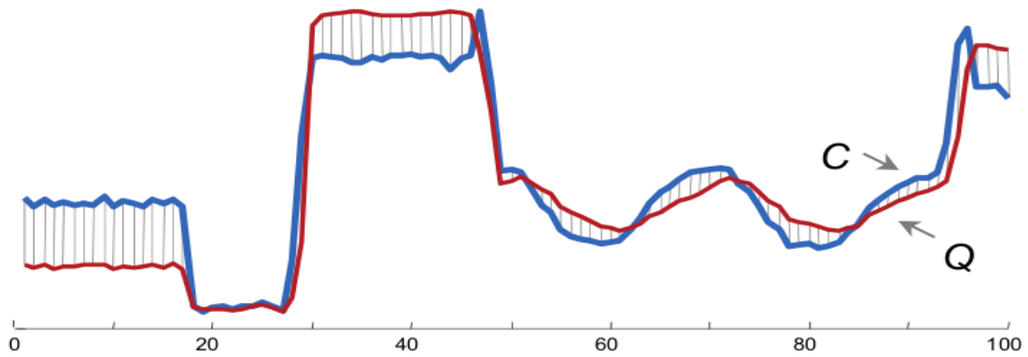


Figure 2.11.: Euclidean distance: point-by-point value comparison between time series C and Q. (Image source: Batista et al., CID: an efficient complexity-invariant distance for time series, 2014)

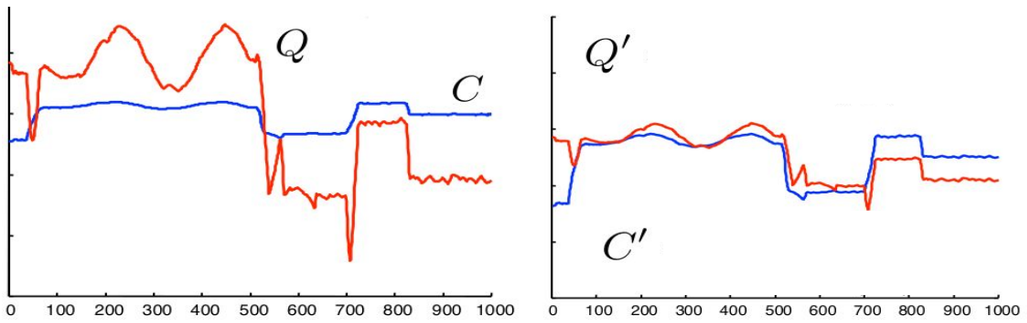


Figure 2.12.: Euclidean distance with z-normalization eliminates amplitude and y-offset variations by transforming time series into new ones that have zero mean and standard deviation one. (Image courtesy of E. Keogh)

`Elastic`: Elastic measures compare one-to-many or one-to-none points between two time series. For example, Dynamic Time Warping (DTW) [11] allows local, horizontal "stretching" and/or "compression" of time series patterns when searching for similar ones (Figure 2.13). DTW accounts for similar sequences that may vary in speed or are shifted temporally; thus DTW supports temporal-warping invariance. In contrast to the local temporal scaling of DTW, there is another measure for global scaling, which is called uniform scaling. Uniform scaling aligns two time series by stretching or compressing uniformly the entire series altering their size.

`Threshold-based`: Threshold-based measures are less common and more specialized. For example, TQuEST [7] transforms time series into a sequence of time intervals which cross a given threshold. Then, similarity is measured by treating these time intervals as points in a two dimensional space, where the starting time and ending time constitute the two dimensions.

`Pattern-based`: Pattern-based measures, e.g., SpADe [26], look for matching patterns (subsequences) between two time series within their entire length. They focus on the shape of the patterns allowing both temporal shifting and amplitude scaling. It can be considered that they are a combination of DTW and z-normalization.

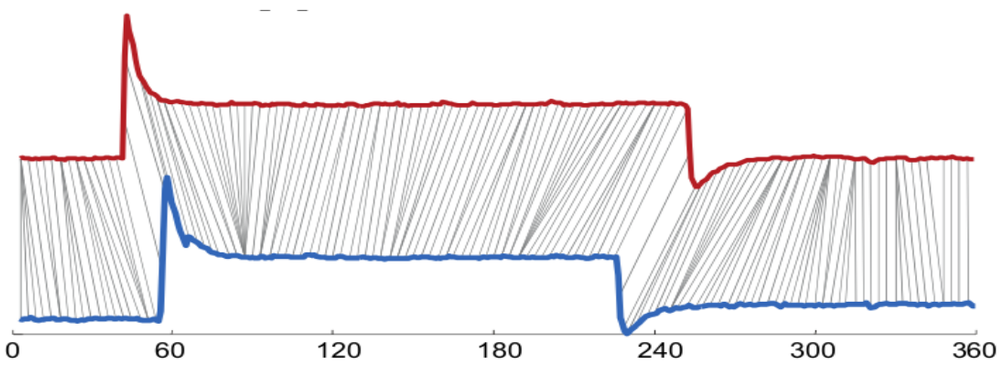


Figure 2.13.: Dynamic time warping eliminates speed or temporal shift variations by stretching and/or compressing time series' patterns. (Image source: Batista et al., CID: an efficient complexity-invariant distance for time series, 2014)

Even more specialized similarity measures eliminate variations in phase, complexity, or differences due to missing values supporting invariance to those signal characteristics.

To evaluate similarity measures, Ding et al. [35] performed a nearest neighbor classification (1-NN) by using distances of nine different similarity measures that belong to the above categories. Then they compared their classification accuracy in pre-labeled datasets coming from different domains [25]. Based on their analysis, they concluded that there is no superior measure, as their classification accuracy depends on the dataset and its domain. Among their findings is that, in small datasets, DTW can be significantly more accurate than ED, but, as the size of the dataset increases their accuracies converge. In our similarity perception work (see Chapter 3), we focus on ED, DTW and its variations because: (i) they are the most commonly used measures in the visualization and data-mining literature; (ii) they are efficient [31, 35]; and (iii) they are appropriate for our motivating domain (see Section 3.2). In our progressive similarity search work (Chapter 4), we use only the ED because [35, 39]: (i) it is fast; and (ii) it leads to efficient solutions for large datasets. (We plan to examine other measures, e.g., DTW, in our future work.)

2.3.2 *Studies on Similarity Perception*

Few studies have investigated subjective user evaluation of similarity search results. TimeSketch [41] proposed a crowdsource procedure where crowdworkers ranked time series with regards to their similarity to a small set of sketched queries (Figure 2.14). The goal was to produce a human-generated ranking and then compare it to the ranking of similarity algorithms. They found DTW to be the closest to human ranking, with ED performing worse or similarly, and SpADe performing badly for small queries. This procedure helps derive human-driven similarity measures and provides insights about how close they are to algorithmic measures, but it is unclear how it can apply to non-sketched queries. Mannino and Abouzied [79] compared their own matching algorithm with ED and DTW by using again simplified query patterns sketched by hand (Figure 2.15). Their algorithm tolerates local distortion errors in amplitude and scale of the sketched queries made by hand, by rescaling the hand-drawn patterns.

Their studies showed that the results of their scale-free, matching algorithm were ranked higher than those of DTW (and ED), but focused on a small set of sketched queries rather than a large set of real time series patterns as is our case. Correll and Gleicher [31] in turn examined whether similarity perception is *invariant* [10] to signal deformations, in the same way that similarity measures are invariant to them. In particular, they examined how humans rated the similarity between a simplified pattern (the query) and a target that was the original query transformed in different ways (e.g., different levels of amplitude, noise, size, temporal position) (see Figure 2.16 for the full list of signal deformations they tested). They also used three similarity measures to rate automatically the similarity between the original patterns and the transformed ones. The algorithms they used were the ED, DTW, and a pattern-based measure. Their results indicated that no single algorithm could match human judgements and that multiple algorithms are required. This work again used Line Chart visualizations, while we consider similarity across different visual representations. We explain finer differences to this study in Section 3.4.6.

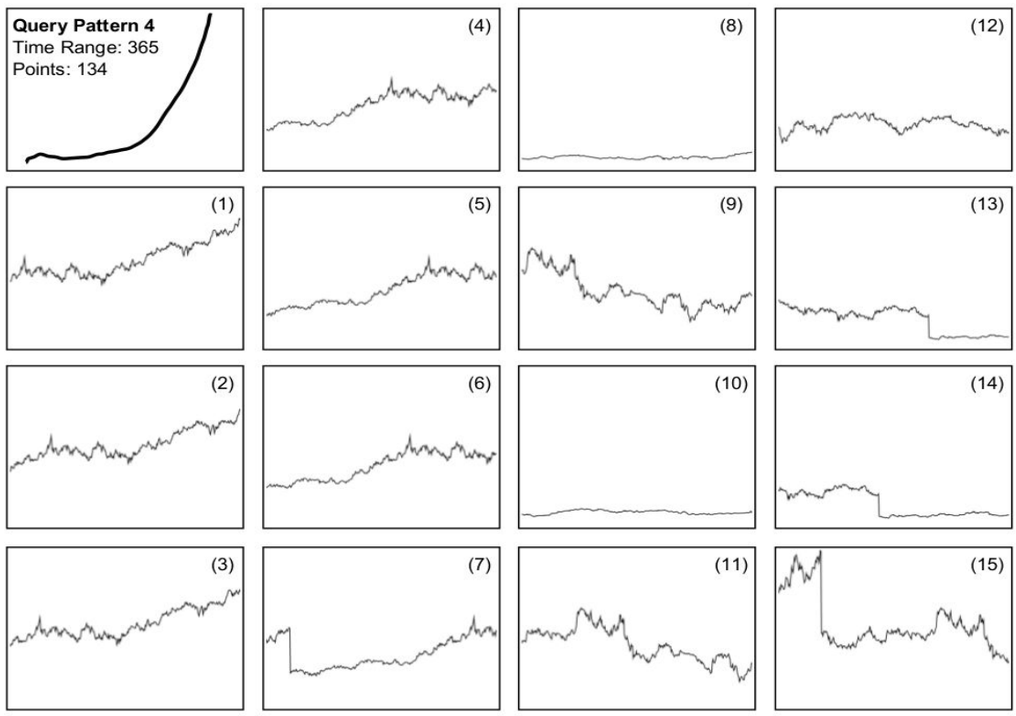


Figure 2.14.: Example of a sketched query and a human-annotated ranking created by crowdworkers. (Image source: Eichmann and Zgraggen, Evaluating Subjective Accuracy in Time Series Pattern-Matching Using Human-Annotated Rankings, 2015)

2.4 SIMILARITY SEARCH AND INTERACTIVE QUERYING

2.4.1 *Similarity Search*

While similarity measures calculate similarity between time series, advanced data series similarity search techniques need to enable scalability. The database community has optimized time series similarity search by developing index structures [19, 29, 125, 135], or by directly optimizing sequential scans [97]. Re-

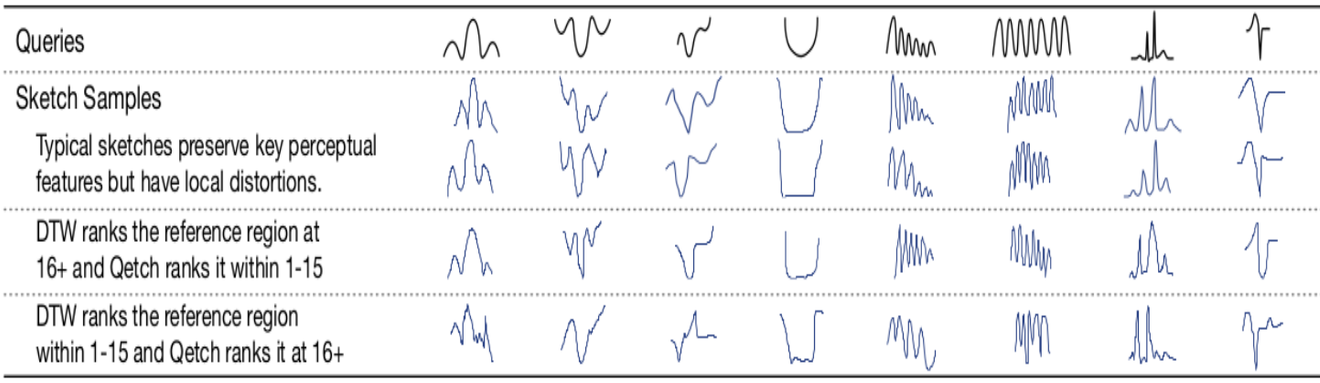


Figure 2.15.: First row: Original queries. Second row: Samples of sketched queries drawn by crowdworkers based on the original queries. The authors ran their own matching algorithm (Qetch) and DTW to measure the similarity between the original queries and the sketched ones. Third row: Sketched queries that have high similarity to the original ones with Qetch algorithm, but low with DTW. Last row: Sketched queries that have high similarity to the original ones with DTW, but low with Qetch. **Note:** We observe that the sketched queries of the third row look more similar to the original queries, while only a few sketches from the last row are similar to those, i.e., Qetch performs better. (Image source: Mannino and Abouzied, Expressive Time Series Querying with Hand-Drawn Scale-Free Sketches, 2018)

cently, Echihabi et al. [39] compared these methods in terms of efficiency under a single, unified experimental framework. They ran 1-NN similarity search queries and assessed index scalability and efficiency by measuring the wall clock time and the number of random disk accesses (one random disk access corresponds to one leaf access in the index tree). Their work indicates that there is no single best method that outperforms all the rest. The dataset size, time series length, disk-resident or in-memory search determine which method performs better. In our progressive similarity search work (Chapter 4), we use the state-of-the-art ADS [135] and DSTree [125] indices, which provide high-quality approximate answers almost immediately, and then updates of progressive results that converge fast to the exact answer. Next, a brief description of the ADS and DSTree approaches follows.

The ADS [135] index organizes the data in a tree structure, where the leaf nodes contain the raw data and each internal node contains summarized data series under the Symbolic Aggregate Approximation (SAX) [75] representation. SAX splits the data series into equi-length segments, and using Piecewise Aggregate Approximation (PAA) [68], associates each segment with the mean value of its points. Then, SAX discretizes the time series and minimizes their footprint by representing the mean values of each segment with a discrete set of symbols (see Figure 2.17a).

Similarly to ADS, the DSTree [125] is also a tree-based index that stores raw data in the leaves and summaries in internal nodes. The main difference with ADS is that DSTree does not support bulk loading, continues to segment the data in both dimensions (horizontally and vertically) while indexing, and uses Extended Adaptive Piecewise Approximation (EAPCA) [125] instead of SAX.

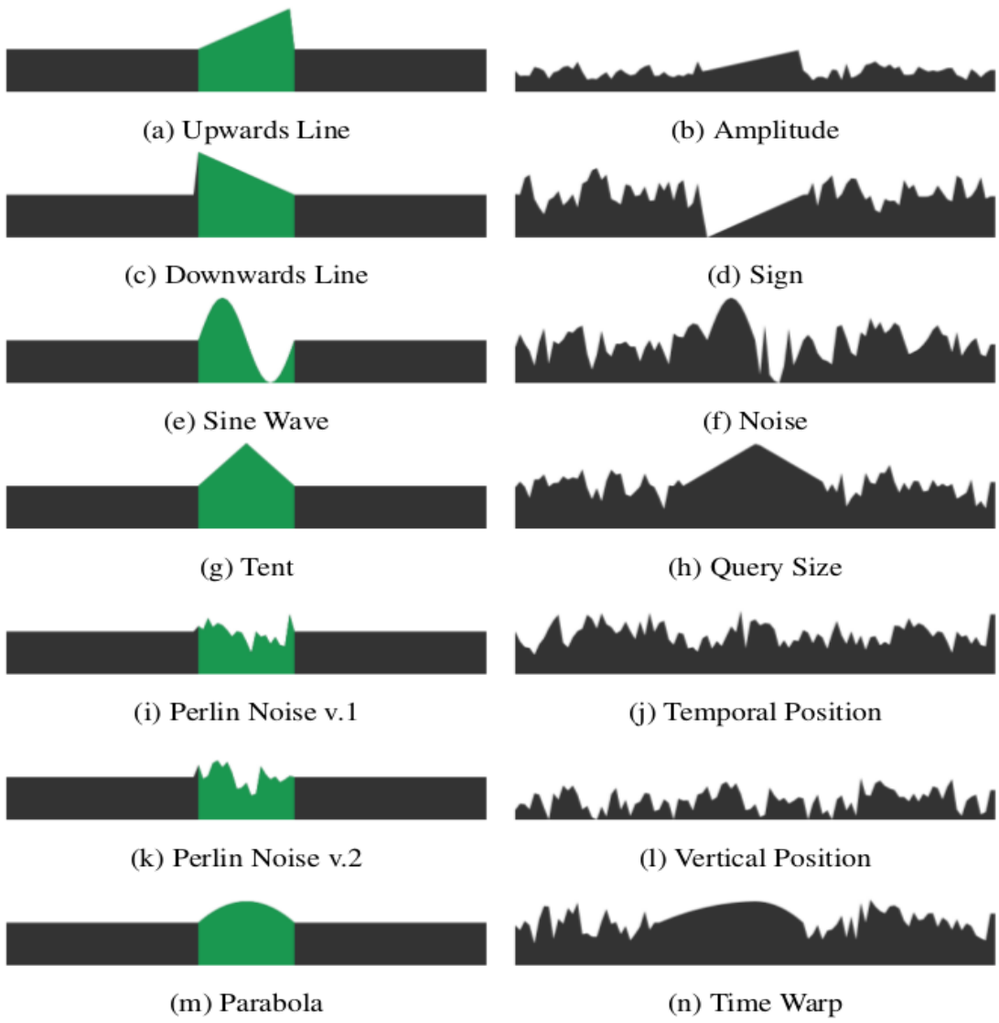


Figure 2.16.: List of the original queries (left column) and examples of signal transformations they applied (right column). They tested three different levels of signal transformation, adding noise to the rest of the series. (Image source: Correll and Gleicher, The semantics of sketch: Flexibility in visual query systems for time series data, 2016)

EAPCA splits the data series into varying-length segments, and using Adaptive Piecewise Constant Approximation (APCA) [22], each segment is represented with the mean and standard deviation values of its points (see Figure 2.17b).

Index search with both the ADS [135] and DSTree [125] indices works by traversing a single path of the tree structure to visit the most promising leaf (i.e., the leaf that is more likely to contain the final answer). The search continues with the next most promising leaf, while a pruning algorithm decides which leaves will be visited and in what order. Each time there is an answer closer (more similar) to the query, we update the progressive results. On the other hand, optimized sequential scan [97] cannot guarantee high-quality approximate results, because the updates depend on the position of random series in the dataset and how early they are accessed. Note that in our progressive similarity search study, we are focusing on the popular centralized solutions. Nevertheless, our results naturally extend to parallel and distributed solutions that have recently appeared in the literature [92, 129], since these solutions are based on the same principles and mechanisms as their centralized counterparts.

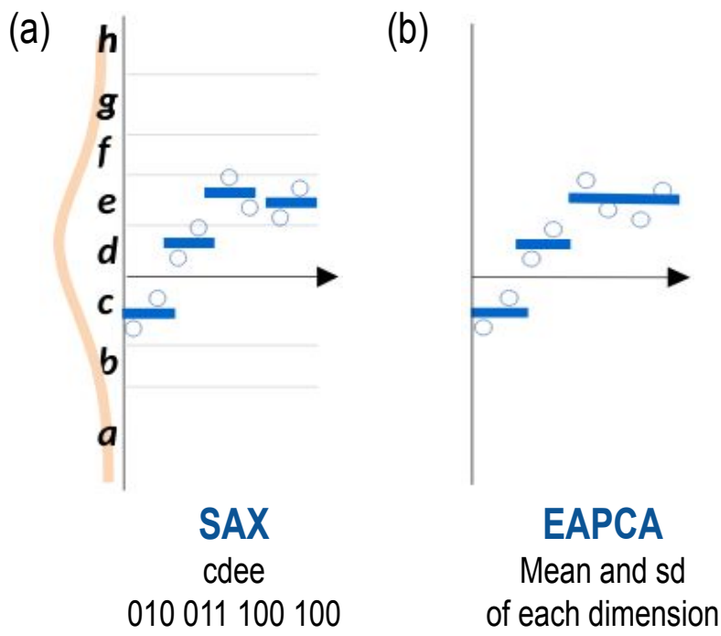


Figure 2.17.: (a) SAX representation (used by the ADS index) splits the data series into equi-length segments and each segment is represented with a discrete set of symbols. (b) EAPCA representation (used by the DSTree index) splits the data series into varying-length segments and each segment is represented with its mean and standard deviation values. (Image source: Echihabi et al., The Lernaean Hydra of Data Series Similarity Search: An Experimental Evaluation of the State of the Art, 2018)

2.4.2 *Interactive Querying*

The human-computer interaction community focuses on the interactive visual exploration and querying of data series. There has been a growing research interest in this direction. In particular, they are interested in how to form interactive similarity search queries. Existing querying approaches on top of Line Chart visualizations rely either on the interactive selection of part of an existing time series [16, 17] (Figure 2.18), or on sketching of patterns to search for [31, 79, 105, 126, 134] (Figure 2.19). Other examples express queries through visual filtering. For example, TimeSearcher [59] allows users to specify their queries through "time box" selections (rectangle regions), which filter the time series and keep only those that go through all the time boxes (Figure 2.20). In Querylines [101], instead of boxes, users can create line segments to define the constraints for the queries that need to be filtered (Figure 2.21). Yet others allow refinement and interactive adjustment of the tolerance around the query (e.g., [60]) (Figure 2.22).

Most selection- and sketch-based systems use ED [16, 60], but more recent works [31, 79, 105, 134] have considered additional similarity measures. Later approaches focus on algorithmic similarity based on visual cues, for example through the automatic detection of specific "motifs", simple shapes such as spikes or sinks that users can combine to form queries [53]. Others [85] examine how to automatically extract a grammar to express time series approximately and simplify the search of matches to a sketched query. These works mainly perform visual matching of series. Zoumpatianos et al. have focused on algorithmic per-

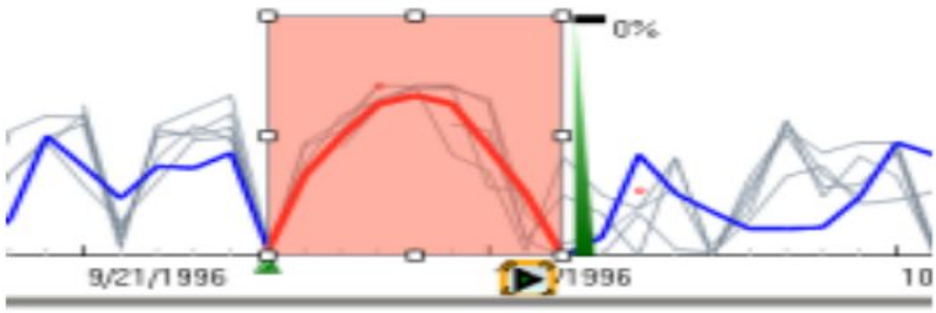


Figure 2.18.: Interactive pattern selection as a query (red-highlighted pattern in the red box). (Image source: Buono et al., Interactive Pattern Search in Time Series, 2005)

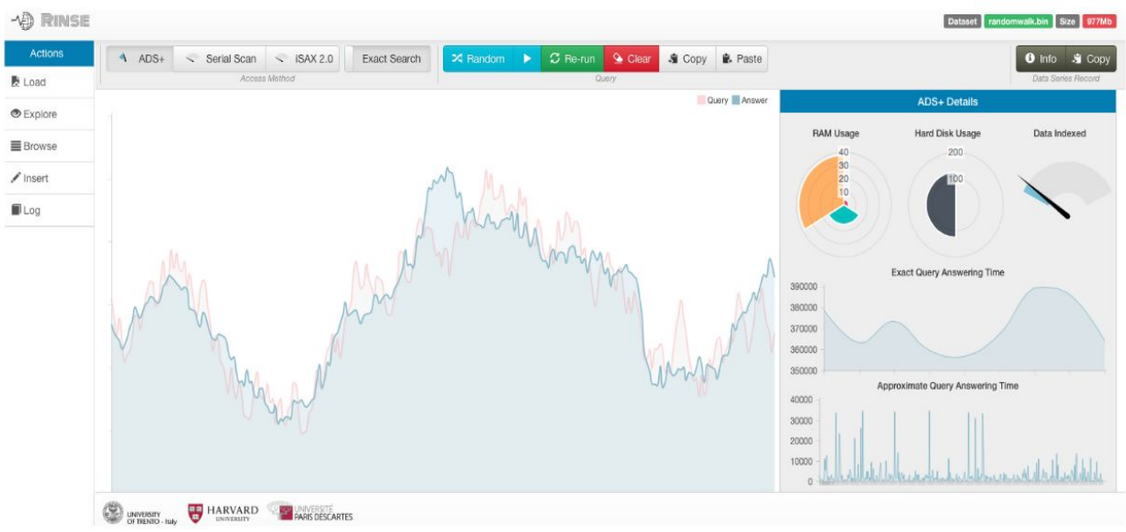


Figure 2.19.: RINSE: an interactive query-sketching system for similarity search in large data series collections. Red line: the sketched query, blue area graph: the 1-NN answer. (Image source: Zoumpatianos et al., RINSE: interactive data series exploration with ADS+, 2015)

formance and scalability aspects of similarity search (millions of data series) in their query-sketching system using ED-, DTW- with or without z-normalization as similarity measures [134, 135]. Recently, Qetch [79] presented a sketch-based querying system and a similarity algorithm that is scale independent. Qetch algorithm works by diminishing local distortion errors in the hand-drawn query patterns. With few exceptions [79, 134], these approaches have not been evaluated through user studies and have not been tested for their scalability to millions of data series.

All the above works rely on Line Chart visual representations. While we do not study mechanisms of querying, this line of work motivates our similarity perception research work (see Chapter 3), as we want to understand how people assess similarity in the results of their queries. This line of work is also orthogonal to our progressive similarity search approach (see Chapter 4), that considers approximate and progressive results from these queries when interactive search times are not possible.

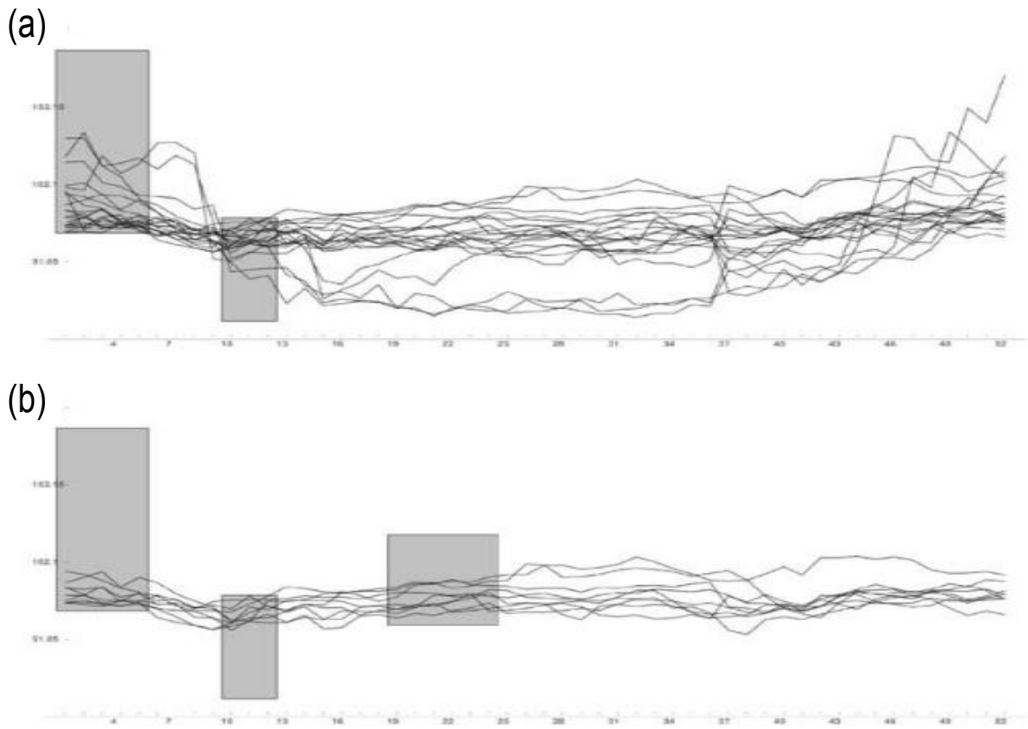


Figure 2.20.: TimeSearcher: an interactive time-series filtering system. Time boxes filter time series and keep only those that go through them. Three time boxes (b) are a stricter filter than two (a). (Image source: Hochheiser and Shneiderman, Dynamic query tools for time series data sets: timebox widgets for interactive exploration, 2004)

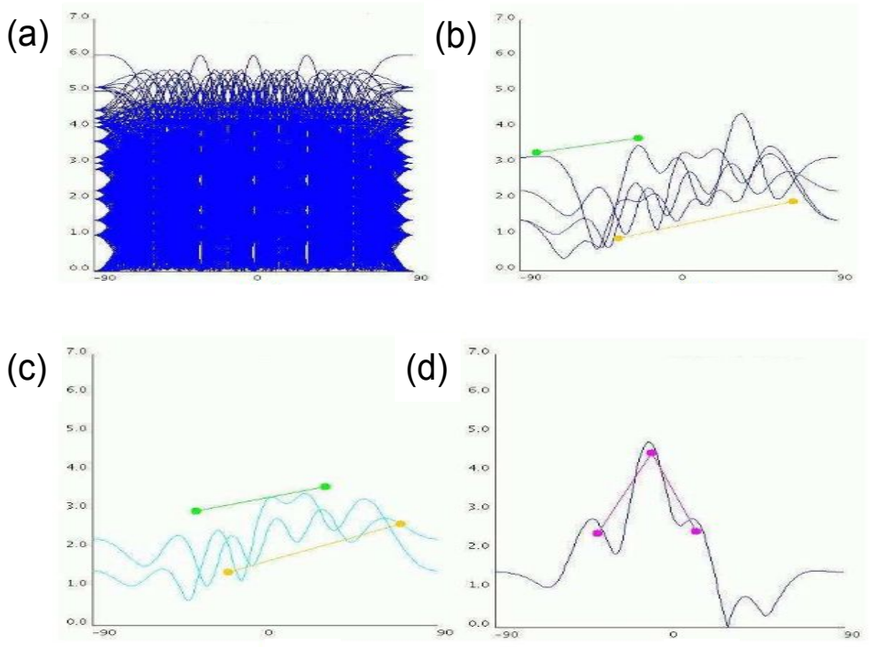


Figure 2.21.: QueryLines: Instead of boxes, lines are used to filter time series. (a) All the series share the same space. (b) Two "hard" lines (green and yellow) define "strict" max and min filtered criteria for specific time intervals. (c) A more relaxed query that gives approximate matches, e.g., time series with an upward trend in the given time interval. (d) A peak-shaped query asks for time series with a peak in the query-defined space. (Image source: Ryall et al., QueryLines: Approximate Query for Visual Browsing, 2005)

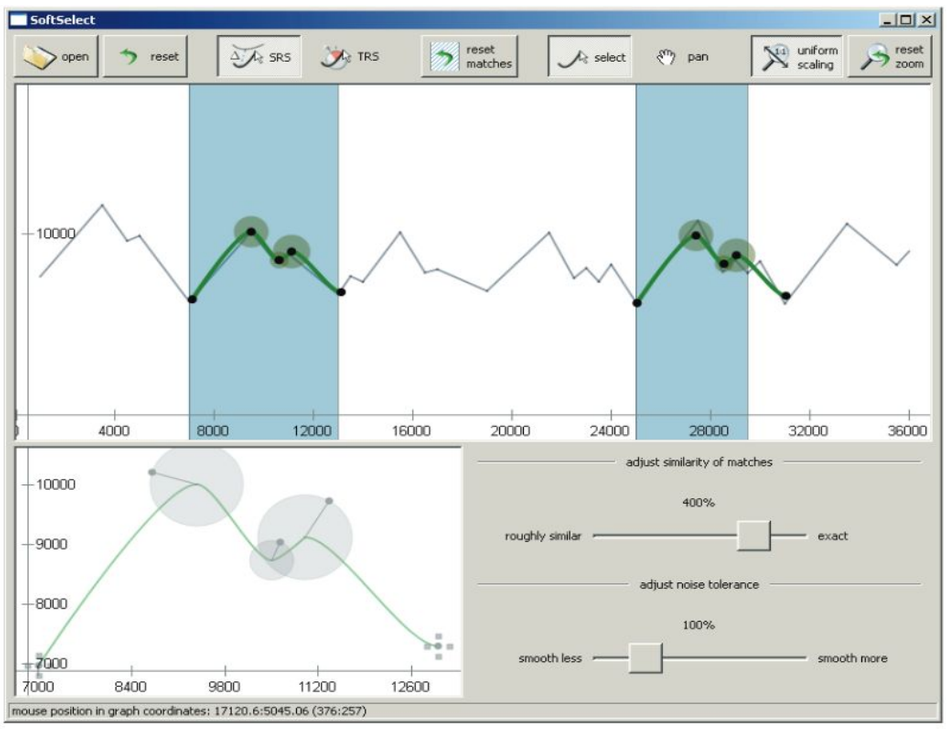


Figure 2.22.: Users can refine a query and adjust the tolerance of specific points (bottom part of the screen). Tolerances are visualized as circles. For a match, all corresponding points need to lie within the circles. (Image source: Holz and Feiner, Relaxed selection techniques for querying time-series graphs, 2009)

2.5 PROGRESSIVE VISUAL ANALYTICS

A recent research direction studies the problem of how we can support interactive, real-time visual analytics when back-end computations cannot be performed instantaneously, as is the case of our work in Chapter 4. To this effect we can use progressive and iterative methods in order to produce fast, but approximate, computational results and visualizations, that are refined over time with increasing precision. Fekete and Primet [43] provide a summary of the features of a progressive system. Here we focus on a subset, namely how to provide: (i) progressively improved answers; (ii) feedback about the state and costs of the computation; and (iii) guarantees of time and error bounds for progressive and final results. We address these features in Chapter 4.

The state-of-the-art in big data exploration takes advantage of the power of distributed systems, indexing, and sampling methods, and different works utilize one or more of these techniques in order to provide progressive results for different kinds of queries and data. Zenvisage [105] defines a new query language that allows the execution of multiple aggregate queries for the visual exploration of large datasets, taking advantage of the inherent parallelism in database systems. Falcon [82] optimizes brushing and linking actions over aggregate visualizations by utilizing indexing and data prefetching. IncVisage [96] incrementally improves trendline and heatmap visualizations by progressively splitting and sampling revealing salient features first (Figure 2.23). Very recently, PANENE [66] was proposed to progressively index (batches of data points) and query for approximate k-Nearest Neighbors, enabling users to access results

in interactive times, while the index is still being built/updated. This work focuses on approximate query answering algorithms for in-memory data, while the focus of our current work is on exact query answering algorithms for out-of-memory data.

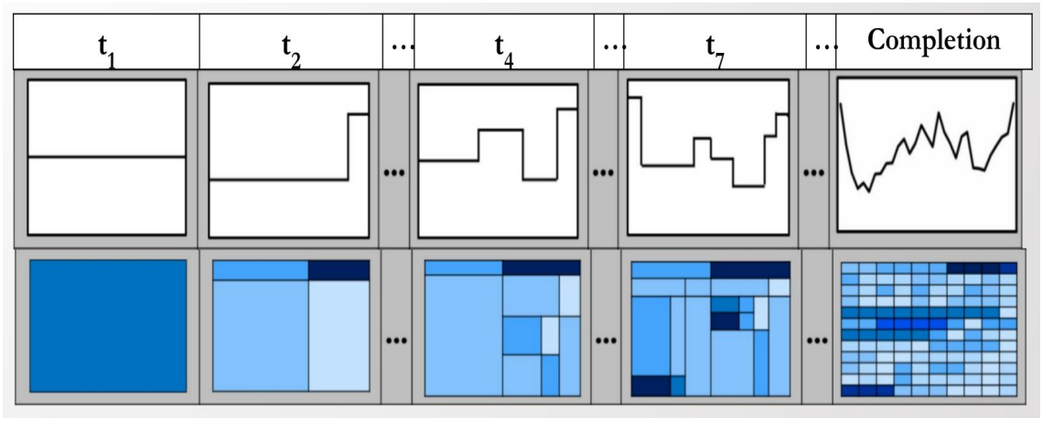


Figure 2.23.: IncVisage: An incremental visualization system for line chart and heatmap visualizations. At each time point, the algorithm samples new data with the most prominent features revealed first. (Image source: Rahman et al., I've seen "enough": incrementally improving visualizations to support rapid decision making, 2017)

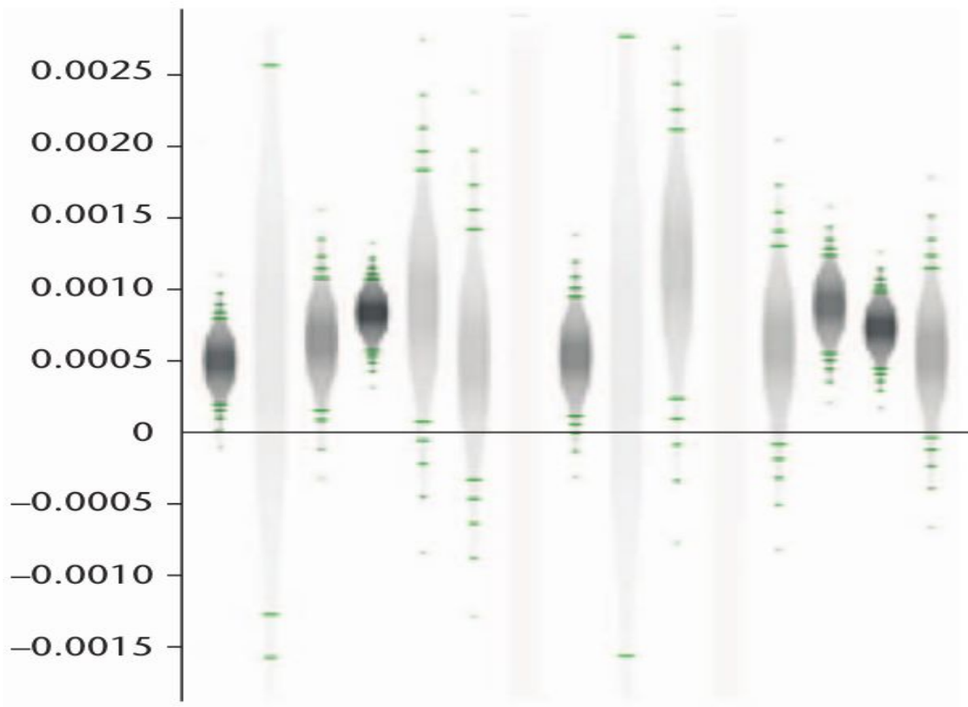


Figure 2.24.: Width and color indicate high confidence about the convergence of incremental aggregate estimates. (Image source: Fisher et al., Exploratory Visualization Involving Incremental, Approximate Database Queries and Uncertainty, 2012)

Systems that provide progressive results and incremental visualizations are appreciated by users due to their quick feedback [8, 130]. Nevertheless, there are some caveats. Users can be mislead into believing false patterns [83, 118] with early progressive results. It is thus important to communicate the progress

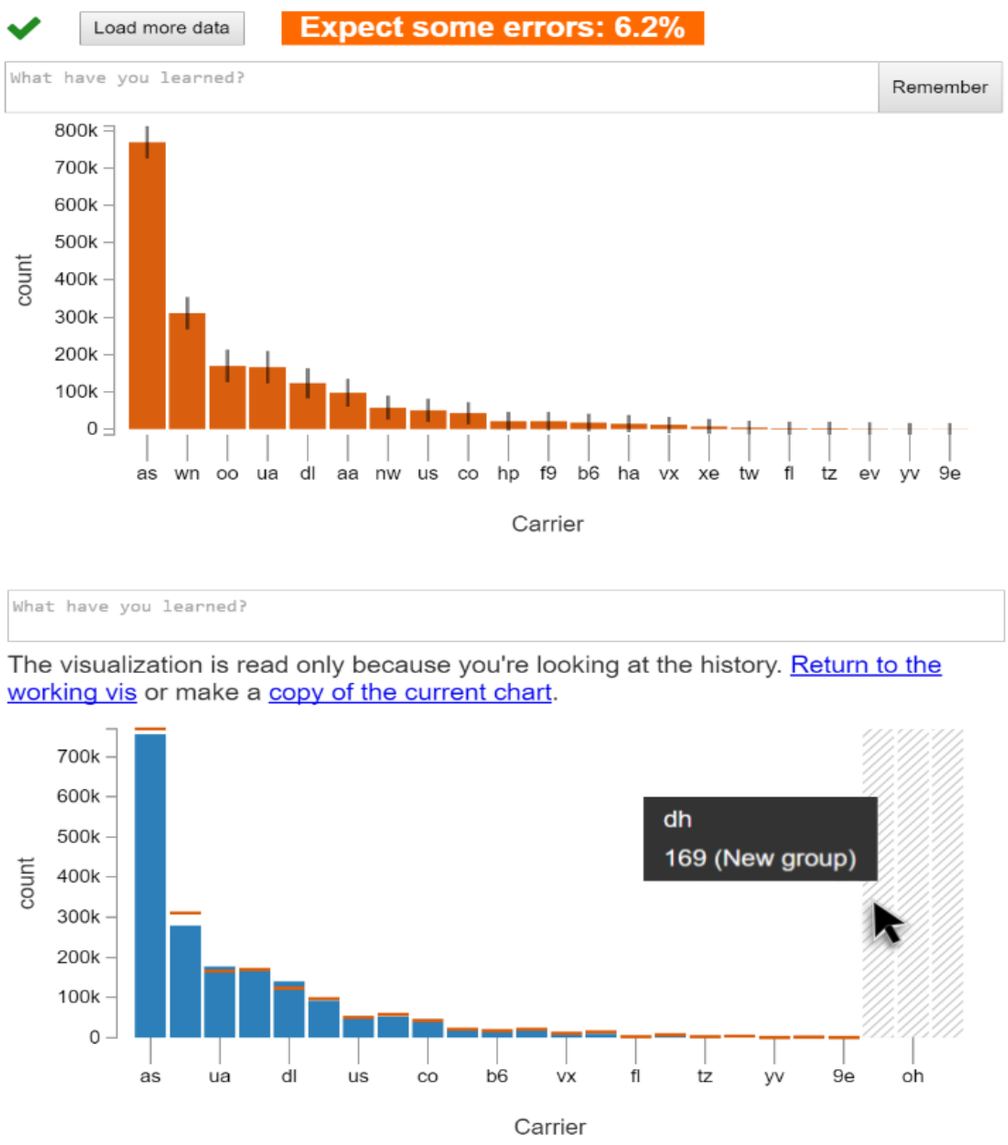


Figure 2.25.: Above: Approximate histogram visualization of a group-by query. Below: The precise result for the same chart. Blue bars show the exact values, orange lines show the approximate results, and highlighted bars, at the end, show results that were missing from the approximation. (Image source: Moritz et al., Trust, but Verify: Optimistic Visualizations of Approximate Queries for Exploring Big Data, 2017)

of ongoing computations [5, 103], including the uncertainty and convergence of results [5] and guarantees on time and error bounds [43]. Previous work has attempted to provide such uncertainty and guarantees in relational databases and aggregation type of queries [58, 64, 128]. In the human-computer interaction community, Fisher et al. [45] studied if analysts trust data that come from incremental samples with added statistical measures such as means and confidence intervals built on top of a standard SQL database. They explored two different ways of visualizing confidence: (a) width and (b) saturation (color). In Figure 2.24 they combine both, with the wider and darker areas of the bars to indicate largely converged aggregate estimates of high confidence. Later, Moritz et al. [83] used an existing algorithm [34] which also exploits sampling methods for incremental, approximate query processing of multiple aggregate database queries at the same time. In that work, they communicated errors and confidence

intervals by presenting approximate visualizations of bar charts and heatmaps (Figure 2.25). At the end of each query, users could compare those approximate visualizations with the precise and final ones.

Closer to the context of data series and similarity search, Ciaccia and Patella [28] studied progressive similarity search queries over general multi-dimensional spaces and proposed a probabilistic approach for computing the uncertainty of partial similarity search results. Their approach is based on sampling methods. Their query-independent method [29] draws pairwise distance distributions between sampled vectors of a multi-dimensional space. Their query-sensitive method [27] draws distance distributions between the query and sampled vectors. However, large samples of distances are required for credible probabilistic distance estimates of the final results. They tested their approach with a few thousands of series (up to 20K) and dimensionality between 2-64 points. However, their approach does not scale to the dataset sizes and number of dimensions that we target. We discuss their approach in detail in Chapter 4.

We focus on very large collections (i.e., in the order of GBs) of data series (where the dimensionality of each series is in the order of hundreds to thousands), and how we can develop approaches to support progressive visual analysis in a fully interactive system. Our ultimate goal is to study how users decide to terminate a search that is progressive in nature (and thus reduce waiting times), when they are provided with approximate answers and information about their uncertainty. We are interested in the quality of approximate answers and how to communicate to users when no improvement is expected to be obtained even if the search algorithm is still running.

2.6 SUMMARY

We presented prior work on data series visualizations and analytics. We saw that there are many different ways to visually encode data series values, such as color, position, or both, in linear or cyclic time settings. For multiple time series, glyph and small multiple techniques split the space devoting a different area to each data sequence, while shared-space techniques place sequences on the same axes. The research literature has previously studied all these visual representations of data series with regards to human perception when subjects (humans) perform common analysis tasks, such as finding maxima and minima, detecting trends, comparing different values, and estimating aggregates. However, a common challenge faced by many domain experts who visually inspect their data is how to compare similar patterns. Although previous work has tested tasks that require similarity comparison – for results of different similarity measures – under Line Chart visualizations, it has never considered alternative visual encodings. Chapter 3 presents a series of studies that investigate similarity perception under different time series visualizations.

Data mining has developed a plethora of similarity measures for automatic similarity search and comparison between data series, that satisfy different similarity constraints. Similarity search is a fundamental task in data series analytics. Database researchers have focused on fast similarity search by developing techniques, such as indexes, that enable scalability. On the other hand, human-

computer interaction researchers have focused on interactive querying interfaces for the creation and definition of similarity search queries. Existing systems dealing with large data series try to combine both fast, back-end computations and expressive querying interfaces backed up by powerful visualizations. However, due to the increasing volumes of data series data, existing solutions cannot support interactive similarity-search query answering. Therefore, for real-time analytics, state-of-the-art approaches seek progressive query- answering mechanisms that are built on top of indexing structures, distributed systems, and sampling methods. Such approaches provide approximate, progressive results, that progressively converge to the final answer.

Progressive results should come with quality guarantees (error bounds) of how close they are to the final answer. Visual analytics systems need to efficiently compute and visually communicate such bounds to users. Prior work has studied the computation and visual communication of error bounds for progressive aggregate query results. For approximate similarity results in data series, prior work computes distance bounds based on samples of distance distributions, but is not scalable to millions of high-dimensional data series, while there is no prior work on how to visualize such distance bounds to communicate uncertainty of similarity measures. In Chapter 4, we present a new probabilistic method for the efficient computation of progressive estimates that scales to millions of data series.

3

COMPARING SIMILARITY PERCEPTION IN TIME SERIES VISUALIZATIONS

A common challenge faced by many domain experts such as neuroscientists working with time series data is how to identify and compare similar patterns. This operation is fundamental in high-level tasks, such as detecting recurring phenomena or creating clusters of similar temporal sequences. While automatic measures exist to compute time series similarity, human intervention is often required to visually inspect these automatically generated results. The visualization literature has examined similarity perception and its relation to automatic similarity measures for Line Charts, but has not yet considered if alternative visual representations, such as Horizon Graphs and Color Fields, alter this perception. It is thus unclear whether the role of visualization has an impact on how people perceive similar patterns. Do people give more attention to the slopes, the extreme points (i.e., peaks and valleys), or the general shape of the patterns when they compare for similar series? Does their focus and their degree of attention to different characteristics of the signal depend on the visualization technique? We note that similarity measures work by amplifying or diminishing different signal characteristics (e.g., time warping, amplitude scale, y-offset, etc.) when computing for similarity. Thus, we seek to understand if visualizations have the same effect on similarity perception.

In particular, we seek to understand if the time series results returned from automatic similarity measures are perceived in a similar manner, irrespective of the visualization technique; and if what people perceive as similar with each visualization aligns with different automatic measures and their similarity constraints. Our findings indicate that Horizon Graphs align with similarity measures that allow local variations in temporal position or speed (i.e., dynamic time warping) more than the two other techniques. On the other hand, Horizon Graphs do not align with measures that allow variations in y-offset and amplitude scaling (i.e., measures based on z-normalization), but the inverse seems to be the case for Line Charts and Color Fields. Overall, our work indicates that the choice of visualization affects what temporal patterns humans consider as similar, i.e., the notion of similarity in time series is **visualization-dependent**. We published the results of this study at IEEE VIS 2018 [49].

In the last section, we also report the results of a follow-up experiment, which we published as an INRIA technical report [48]. In this follow-up, we studied whether different color interpolation techniques (the linear RGB and the more perceptually uniform CIE L*a*b*) in visual encodings that utilize color maps, such as Color Fields, affect time series similarity perception. For the patterns we tested, we didn't find any statistically significant differences between these two color interpolation techniques on data series similarity perception.

3.1 INTRODUCTION

Time series derive from measurements and recordings of a range of natural processes (e.g., seismic activity per ms), human functions and activities (e.g., systolic and diastolic blood pressure per hour, number of steps per day), and tracking of mechanical and technological equipment (e.g., temperature of an airplane engine per second). Large time series collections are becoming increasingly commonplace [89], and users need to analyze them in order to extract useful knowledge. Their analysis involves a diverse range of tasks, such as searching for pattern templates or anomalies, identifying reoccurring waveforms, or classifying time series subsequences into clusters of similar patterns, all of which involve the notion of similarity between time series. Data-mining research has developed a wide range of techniques to automate such tasks [46].

In many situations however, automated techniques fail to produce satisfactory results, thus experts rely on visual analytic tools to perform their tasks. For example, in EEG data, comparing time series to identify epileptiform discharges is difficult [65]. These temporal patterns take a variety of different forms that are very specific to individual patients, while very similar patterns appear in normal background activity. Although several techniques claim to automatically detect such patterns [61], medical experts still visually inspect the EEG data of their patients. This process is especially time consuming, as experts need to visually scan a large number of temporal signals recorded from multiple EEG sensors, find, and compare these patterns.

In such scenarios, the use of visualization techniques that accurately and effectively communicate similar patterns between time series becomes important. Time series are commonly represented as Line Charts, but a considerable amount of work in Information Visualization has examined alternative visual encodings, such as Horizon Graphs [57, 63, 93, 99, 102] and Color Fields [3, 32, 86, 102, 111]. This literature has focused on elementary visual tasks that require estimation, e.g., estimation of averages, or point comparison and discrimination tasks. Visual pattern matching is a more complex task that requires the simultaneous comparison of a large number of features and likely incorporates many of these previously mentioned tasks. Thus, previous results say very little about how people access the similarity of two or more time series when using different time-series visualizations.

In this Chapter, we examine how *line*- and *color*-encoding techniques affect what time series humans perceive as similar. Specifically, we present the results of two laboratory experiments that compare three representative techniques: (1) **Line Charts**, (2) **Horizon Graphs**, and (3) **Color Fields**. In addition to task performance, we assess the reliability (or subjectivity) of participants' answers and examine whether the above techniques penalize or favor similarity invariances [10, 31, 41] that are often required by certain application domains. For example, two patterns might be considered as similar, irrespective of their amplitudes (*amplitude invariance*) or their stretching along the time dimension (*time-scale invariance*). We want to understand whether the three visualizations exaggerate or deemphasize such deformations. To this end, we assess the perception of similarity between time series, with respect to representative similarity distance measures

that are well known to be invariant to certain properties of a time series [10]. Our first experiment investigates *local time-scale* (or *time-warping*) *invariance* by contrasting similarity perception with **Euclidean Distance** (**ED**) and **Dynamic Time Warping** (**DTW**). Our second experiment in turn investigates *amplitude* and *y-offset invariance* by contrasting similarity perception with and without **z-normalization**.

In contrast to previous studies, that used human sketched [41, 79] or artificially generated [31] query patterns, the queries in our experiments are extracted from annotated EEG data and express real patterns of interest. A major challenge is how to derive patterns that are representative of real data and tasks, but that also highlight the differences of the tested similarity measures. We address this challenge by selecting query patterns for which the different distance similarity measures produce clearly distinct answers. This enables us to assess whether similarity perception with each visual encoding technique is invariant to warping as well as to amplitude and offset deformations in the signals.

To summarize, this study is the first to investigate how humans perceive similarity between time series with both line- and color-encoding visualization techniques. Our results answer two major questions: (1) how easy or difficult it is to visually identify similar patterns with different visualization techniques; and (2) whether similarity perception with these techniques is invariant to representative signal deformations.

3.2 MOTIVATION

Our motivation stems from a real problem presented to us by a team of neuroscientists, experts in the analysis of EEG recordings for the diagnosis of epileptic events. Our experimental task is inspired by the user interfaces that such experts use to visually analyze EEG data. The pool of our experimental data was also provided directly by them. In two 1h sessions we met with two and three neuroscientists respectively from the MEG/EEG Center of the ICM Brain and Spine Institute [114] in Paris. They are looking for tools to improve the detection of "epileptiform discharges". These are abnormal patterns that have been linked to various cognitive disruptions and reoccurrences of epileptic seizures [112]. They are often not isolated cases but may appear as periodic patterns [88], whose periodicity may vary significantly from one patient to another.

Epileptiform discharges are events which are characterized by a spike of 20-70 milliseconds (ms) usually followed by a sharp wave lasting 70-200 ms [33, 107, 108]. As opposed to epileptic seizures that produce large disturbances in the EEG signal of a patient, epileptiform discharges are especially hard to detect. Although data-mining research has developed algorithms to automatically detect their patterns [61], according to our experts, such algorithms result in many false positives and are not useful in practice. Main reasons for this problem is that epileptiform discharges take a range of different forms and often resemble normal background activity due to regular artifacts such as pulses of the heart, the eyes, or the muscles [65]. In addition, their patterns vary greatly across patients so machine-learning approaches cannot help.

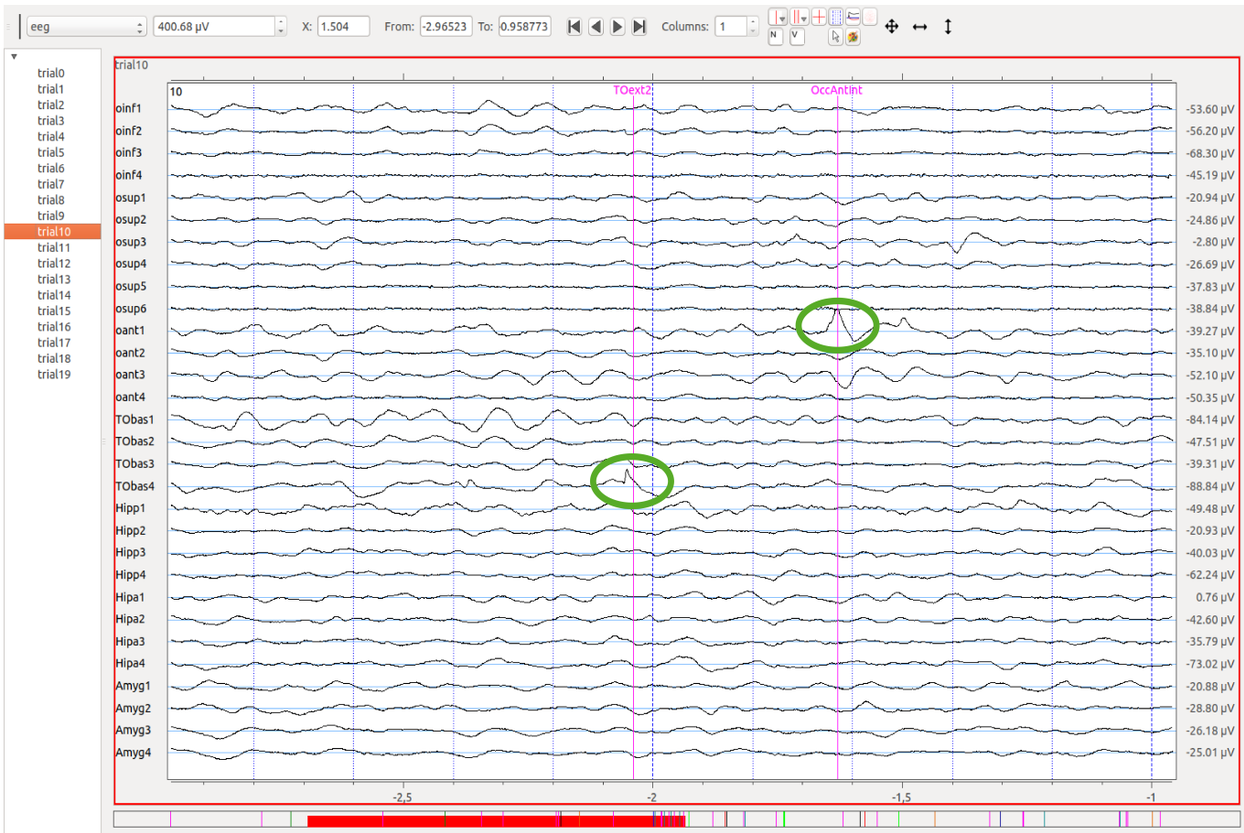


Figure 3.1.: The Muse tool used by the neuroscientists of ICM [114] to visualize measurements from 295 electrodes and sensors placed on patients' scalps. Here a neuroscientist has restricted the view to 6 groups of sensors (30 in total) from one recording trial (trial10). Purple lines indicate manual annotations of epileptiform discharges that neuroscientists have detected on different sensors. The particular discharges are highlighted in a green oval for illustration purposes only (these highlights are not part of the tool). The scroll bar in the bottom indicates what time frame of the series is currently visible, and is augmented with indications of where manual annotations exist (small colored line segments).

For these reasons, medical experts do not trust automated techniques and still visually scan the data to identify abnormal events, using tools such as the one depicted in Figure 3.1. This can be a very tedious and complex task. Experts need to visually inspect around 300 sensors and several thousand data points per sensor (see Section 3.3.2). And even when they find candidate events, they often need to consult additional resources (e.g., 3D representations of the location of the electrodes placed on the scalp) to make their decisions and annotate their data.

In an attempt to aid our users, we tried to understand if it would be possible for them to first manually identify a small number of epileptiform discharges and use them as patterns to automatically detect similar subsequences. The experts could then visually verify whether they are similar and decide if they are also potential discharges. To this end, we requested information about what types of variations or deformations in the patterns could indicate similar signals.

The experts were able to verbally describe roughly the signal they were looking for. They explained that the duration of spikes and waves can vary and are

not consistent even for a single patient, thus stressed or compressed signals are of interest (invariant to time-warping). When asked, they also explained that the height of the pattern can vary across patients (invariant to amplitude). But they could not say to what extent the amplitude of the spikes and discharges is important, i.e., to what extent signals could be considered similar if they differed in amplitude. In some cases we got the response that a spike can be too small (i.e., in some cases amplitude may play a role) but this can only be determined by looking at the background noise - the parts of the signal before and after the spike. Or that to interpret a spike they needed access to views from other sensors. The importance of context in detecting such discharges is well documented [33, 107, 108]. These are all very subtle properties that need to be evaluated case by case, and in context, stressing further the need for human intervention.

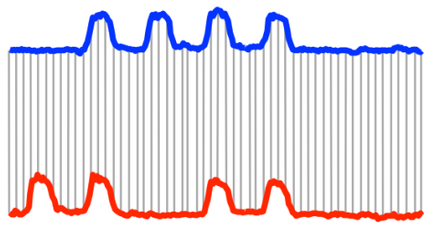
As our experts explained, identifying these types of discharges requires a lot of experience, and some of their decisions remain subjective. Past work has shown that agreement even between different experts can be particularly low [65]. While this task relies on extensive experience and involves substantial domain knowledge, it still raises an interesting question. Do visualizations actually help viewers understand what temporal patterns are similar, or are there aspects of the invariances of interest that are not communicated well? We set out to investigate if different types of visualizations communicate or de-emphasize invariances in a similar way, or if visualizations need to be chosen appropriately.

3.3 GOALS AND RESEARCH STRATEGY

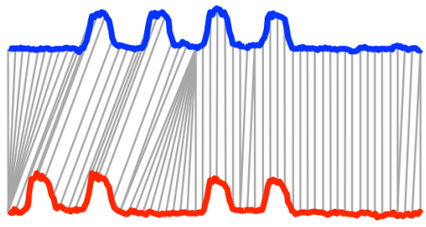
Given that users like neuroscientists rely on visualization tools to take decisions, understanding how a visualization may affect what time series are perceived as similar is important. The similarity criteria used by experts can be complex and highly uncertain, and the extent to which signal deformations satisfy such criteria often depends on thresholds that may vary from case to case. Thus, we are especially interested in knowing which visual encodings are sensitive to deformations of a time series signal and which of them are ”invariant” to those deformations. Such knowledge can help us design tools that better match the invariances required by different application domains. It can also help us support users' tasks by proposing alternative visualizations, as different visualizations may emphasize (or de-emphasize) the perception of different deformations in the signal.

3.3.1 *Experimental Approach*

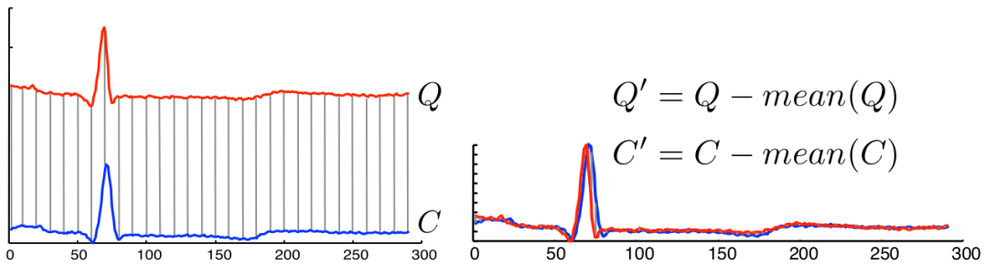
As discussed in Chapter 2, previous work has studied deformation invariances from an algorithmic perspective. Batista et al. [10] enumerate several types of invariance: temporal warping, uniform scaling, amplitude and offset, phase, trend, complexity, etc. Correll and Gleicher [31] consider these types of invariances to design a sketch-based query system that is flexible enough to accommodate algorithms with different invariance characteristics. They then present the results of an experiment that investigates how sensitive or invariant similarity perception is with respect to different deformations when using Line Charts.



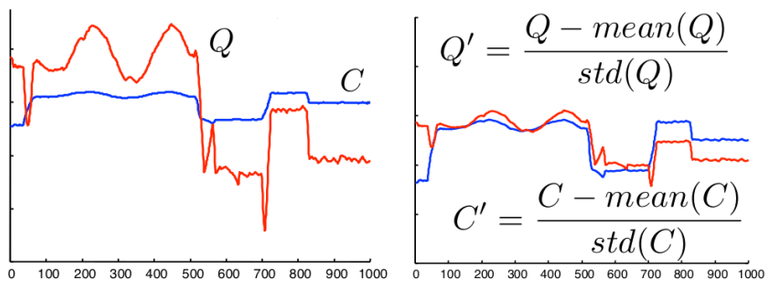
(a) Euclidean Distance (ED)



(b) DTW



(c) Z-Normalization: y-offset shifting



(d) Z-Normalization: amplitude scaling

Figure 3.2.: Overview of how the algorithms we used perform matching for similarity: (a) Euclidean Distance computes the distance between all the corresponding points of two time series of equal length. (b) DTW allows the matching of points between two time series, even if these points are not aligned on the time axis (*invariant to time-warping*). (c-d) Z-normalization transforms a time series into a new series of the same length that has zero mean and standard deviation (std) one. It enables similarity search independent of y-offset and amplitude scaling (*invariant to y-offset and amplitude*). (Images courtesy of E. Keogh)

While inspired by this research, our goal is different. We are interested in how *different visualizations* affect similarity perception, thus we treat the visualization techniques as our primary experimental factor. Although we also seek to understand how different techniques support invariances, the way we control for invariances is different. In particular, our approach is based on the observation that signal deformations emerge naturally in real data, taking complex forms that cannot be easily reproduced with artificially created patterns. Thus, as opposed to Correll and Gleicher [31], we do not directly control signal deformations as experimental factors. In Correll and Gleicher's experiment, the patterns of interest take elementary forms (upward and downward lines, sine waves, Perlin noise, etc.) and are transformed uniformly along the time dimension. This approach allows for stricter control and simplifies the experimental design but does not capture the way people compare patterns in real data. For example, when determining if two time series are similar, a user may have to assess temporal stretches or vertical shifts that occur in small portions of the signal in combination with other deformations. In such scenarios, the perception of similarity is likely to rely on a mix of very subtle signal characteristics.

Given these considerations, we decided to use real data to generate our experimental tasks, based on the application domain and scenario that we described in the previous section. We also decided to concentrate on the invariances that are most relevant to these data.

3.3.2 *Dataset*

We used a **real dataset** provided to us by our collaborating neuroscientists (see Section 3.2). The dataset contains measurements from 295 electrodes and sensors placed on patients' scalps: among them 151 signals come from Magneto-Encephalo-Graphy (MEG), 33 from Electro-Encephalo-Graphy (EEG), and 39 from intracranial Electro-Encephalo-Graphy (iEEG) sensors. Measurements last six seconds and are captured at a sampling rate of 1250 Hz. All our data come from 154 such recordings of the same patient, that each contains 295 long time series - 1 per sensor, of 7500 data-points each (~ 341 million data points in total). We used this dataset to generate experimental trials.

To understand similarity, we need to compare time series with interesting temporal patterns. How to determine interesting patterns is a difficult problem. Synthetic patterns can lead to artificially looking results, while randomly selecting ones from a real data set may result in empty or noisy patterns. How to determine interesting query patterns is a difficult problem. Synthetic query patterns can lead to artificial results while randomly selected patterns may result in empty or noisy patterns. Eichmann and Zgraggen [41] addressed this problem by collecting sketched patterns by non-expert people. However, this approach is only appropriate for simplified human-generated patterns that may capture the intricacies of real patterns in the data.

Our dataset allows for a better solution. Neuroscientists have manually annotated this dataset by adding markers at time points that correspond to potential interictal epileptiform discharges. Thus the dataset already contains real patterns of interest. An interictal epileptiform discharge is a fast paroxysmal event that

is characterized by a spike of length 20-70 milliseconds (ms) usually followed by a sharp wave lasting 70-200 ms [33, 107, 108]. We used the area around these annotated events as potential queries for our similarity search algorithms. The dataset contains a total of 205 annotations.

3.3.3 *Invariances*

When considering time series to compare against the potential queries, we focus on ones that contain deformations that are important to our experts. As they indicated in Section 3.2, patterns that are invariant to - i.e., allow for variations in (i) *time warping* and (ii) *amplitude* and *y-offset* are of interest. Time-warping invariance is important since EEG signals often vary in transient or rhythmic activity and speed [78], e.g., they may include slow delta waves with frequencies lower than 4 Hz, as well as fast beta waves with frequencies greater than 13 Hz. Amplitude and offset invariance is important because experts are often interested in clustering spikes based on their shape independently of their vertical height or shift [100]. Other invariances, such as noise and trend, are usually unwanted. Medical experts preprocess their data by applying filters that remove noise or long additive trends in the signals. Finally, global invariances such as uniform scaling are less interesting, as they can be supported by global-scaling tools that are independent of visualization.

As we do not treat invariances as experimental factors, we do not directly vary their levels. However, we control them by using similarity algorithms that are well known to support them (see Figure 3.2). For *time-warping invariance*, we use **Dynamic Time Warping** (**DTW**) [11]. This is a flexible distance measure that allows the matching of points between two time series, even if these points are not aligned on the x-axis, discovering similar patterns that may vary in speed. For *amplitude* and *y-offset invariance*, we use **z-normalization** [52]. Z-normalization transforms a time series into a new series of the same length, that has zero mean and standard deviation one, and enables similarity search independent of scaling and shifting.

Both algorithms are well established and widely used in the data-mining literature [10]. We do not consider Hough Transform [31], as it combines invariances of both DTW and z-normalization. We contrast the results of the above algorithms with the results of the simple **Euclidean Distance** (**ED**) by asking participants to choose between them. We note that in the experiment participants see the original time series and their values (not the deformed versions used by the similarity algorithms).

This approach shares similarities with that of Eichmann and Zgraggen [41], who compared how people rank the results of multiple algorithms that measure similarity. For many queries, however, similarity algorithms may return identical or similar results. To deal with this constraint, we developed an automated mechanism for selecting queries for which the algorithms produce distinct results. These cases are especially interesting because (i) they better capture the differences of the algorithms, and (ii) they represent the most difficult cases, for which careful visual inspection might be more critical. This approach also allows

us to observe the effect of the underlying invariance assumptions more clearly within an experimental setting.

Another differentiation of our approach compared to previous studies is that we also measure how different participants agree on their assessments. Measuring agreement is important for assessing similarity perception as it enables us to evaluate the level of subjectivity and diversity in participants' answers in an objective way.

3.4 EXPERIMENTS

We conducted two experiments to study if using different time series visualizations, Line Charts (LC), Horizon Graphs (HG), and Color Fields (CF), changes whether time series are perceived as similar. And if invariances in the data effect this perception. Exp-1 investigated *time-warping* invariance by asking participants to compare the results of ED and DTW. Exp-2 investigated *amplitude* and *offset* invariance by asking participants to compare the results of ED with and without z-normalization. Aspects in the setup and procedure are common in both experiments, so we present them together unless explicitly stated.

3.4.1 *Participants & Apparatus*

A total of 36 volunteers, 23 to 42 years old (mean = 29, std = 5.6), participated in the two experiments without monetary compensation. We recruited from a local university mailing list 18 participants (seven women) for Exp-1 and 18 additional participants (three women) for Exp-2. Our participants came from different scientific backgrounds, including students and researchers in Computer Science, Electrical Engineering, Physics, and Finance. As our study is perceptual in nature, we opted for a general pool of participants rather than experts. For both experiments, we used a 24" DELL monitor set to 1920 × 1080 resolution with mouse and keyboard as input. The user interface was implemented with Javascript and D3.js and was set to full screen.

3.4.2 *Visualization Techniques*

Similarity search likely involves both point comparisons, such as finding maxima, and overview comparisons, such as identifying trends. It is thus unclear how *position*- or *color*-based visualizations would affect it (see Chapter 2). We thus focused on three visualization techniques that rely on position (Line Charts - LC), color (Color Fields - CF), or both (Horizon Graphs - HG). These visualizations can also scale when arranged in small multiples [69, 99, 102], e.g., in order to support context that is important for neuroscientists (see Section 3.2). We explain how we represented time series with these visualizations.

Line Charts

Line Charts (LC) map time to the horizontal axis, and value to the vertical, placing points into particular positions in a 2-D Cartesian coordinate system. These are the simplest and most common visualization for time series and are

often seen in small multiples (e.g., in the MUSE tool that our experts use in Figure 3.1). Cleveland and McGill [30] report that positional encodings, such as Line Charts, are good for accurate values' retrieval and comparison tasks (see Section 2.2 for more details). In our implementation, the y-axis was not visible to prevent participants from trying to read exact values. Nevertheless, all axes had a common scale to aid participants compare time series. The zero value of y-axis was at the middle of the area allocated to each time series. We chose the line variation rather than *filled area charts* because it is commonly used by EEG visualization tools [65] and our own experts. It has also been used in previous studies on time series similarity [41, 79] and thus acts as a baseline.

`Horizon Graphs`

Horizon Graphs (`HG`) use a combination of color and position encodings. This representation saves vertical space using both mirroring and superimposition of the bands, while maintaining the overall line shape. In that way, they utilize space more efficiently with baselines that are specific to each time series, e.g., when the baseline is the average of the time series value range. Previous studies (see Section 2.2) have shown that Horizon Graphs were faster than Line Charts for discrimination tasks, but slower for peak and trend detection tasks [63]. Regarding the similarity task, different baselines would make comparisons for similarity challenging, that is why we used a common baseline in our experiment for all time series, set to zero. The performance of these graphs seems to deteriorate when increasing the number of bands [57], thus we used a variation of two positive bands and two negative ones, similarly to previous studies [63]. We also followed the convention of using variations of red (#ff9999, #b30000) to indicate negative, and of blue (#bdd7e7, #08519c) to indicate positive values [57, 99], with darker hues assigned to the bands furthest form the baseline (most negative and positive).

`Color Fields`

Color Fields (`CF`) use color mapping to encode time series values [3, 32, 69]. Vertical color stripes at each time point are colored based on their values. As we report in Section 2.2, this encoding can be used to create fairly dense displays and has been shown to be a promising representation for overview tasks [32]) and take up less space [69], even though color is worse than position [30]. Previous works consider color scales of two [4, 86] or more colors [102]. We opted for a simple two-color scale in our experiment to be in accordance with Horizon Graph two-color encoding scheme. We again chose red tones (#ff0000) for the most negative and blue (#0000ff) for the most positive value. Pure tones were used to maximize the distance between the two extreme colors. We used a linear RGB interpolation between the two tones. In a follow-up experiment (see Section 3.8), we used the exact same tasks to compare linear to CIE L*a*b* interpolation. As Section 3.7 and Section 3.8 discuss, CIE L*a*b* interpolation might be a worse choice.

It is worth noting that the three visualizations utilize space differently. For our experiments, we allocated the same amount of y-axis height (vertical space) per time series for all techniques, which is consistent with previous studies [63]. It is important to first understand how humans' similarity perception is affected

by the actual visual encoding before considering additional factors, such as the graph height or vertical space. As different visualizations utilize space differently, this is a topic worthy of future experimentation.

We chose a fairly large vertical size (60 pixels) to ensure that time series were clearly visible in all visualizations. For LC, we fixed the position of the time axis at the middle of its available space since our data includes both positive and negative values. Due to their encoding, HG can utilize the vertical space more efficiently, as they superimpose negative and positive values in the same space. CF do not necessarily require as much vertical space [69], nevertheless this size ensures that colors are large enough to be seen clearly [110].

We fixed the horizontal size of the time series to 501 pixels, encoding one time point per pixel (all our time series were 501-point long – see Section 3.4.5). In practice, users, e.g., medical experts, explore their data at different granularities, by keeping the vertical space fixed and compressing or decompressing the time axis. Nevertheless, we decided to avoid factors, such as over-plotting and aggregation, that might also affect similarity perception. As this is a first study comparing perceived similarity across visualizations, we have simplified the experiment by not considering interaction (including no means to change the baseline or bands in the Horizon Graphs).

3.4.3 *Similarity Measures*

Participants had to assess the similarity of time series extracted from the dataset (Section 3.3.2). For each trial, we determine one time series that serves as the query and four additional ones that serve as possible matches. These matches were extracted from the data using automatic similarity search algorithms that are sensitive or not to signal deformations, in other words, they consider or not time series invariances which are important for our data domain (see Section 3.3.3). Both experiments used the simple **Euclidean Distance** (**ED**) as control, but each investigated a different invariance.

Exp-1 (Time-Warping)
We examined *time-warping invariance* by contrasting **ED** to **DTW**. DTW supports time-warping invariance, while ED not. A main parameter of DTW is the warping size, i.e., the x-offset window size in which the algorithm searches for the best matching point. According to Ding et al. [35], constraining the warping size increases the speed of the algorithm by reducing the computation cost and enabling effective pruning. Depending on the domain, its accuracy remains the same or further improves. We set the warping window size to 10% of the time series length as this is the most common size used in the literature and larger sizes can hurt accuracy results [98].

Exp-2 (Z-Normalization)
We examined *amplitude* and *y-offset invariance* by contrasting the results of Euclidean Distance (**ED**) and Euclidean Distance in conjunction with z-normalization (**NormED**) [52]. NormED supports y-offset and amplitude invariance, while ED not. For this second case, time series are z-normalized to acquire similar amplitude and y-offset, while maintaining the shape of their patterns. The new time se-

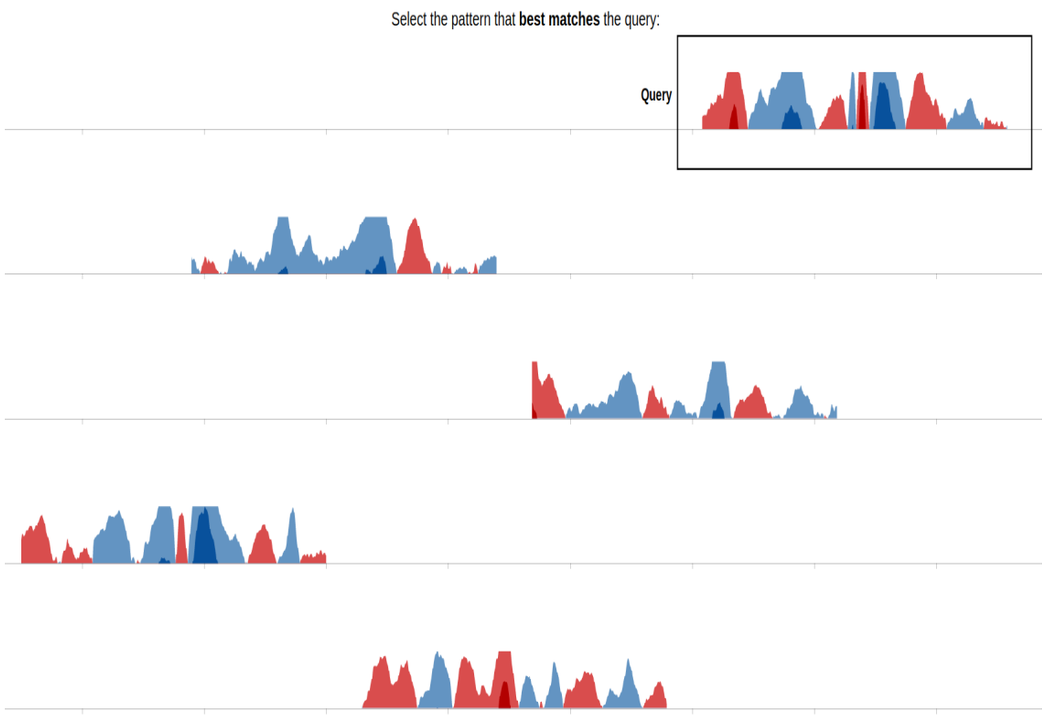


Figure 3.3.: Experimental screen for the Horizon Graph condition. The answer vertical order and horizontal shift was randomized across visualizations. From the top, the series are: Query, Outsider-ED, Top-ED, Top-DTW, Outsider-DTW.

ries have zero mean and standard deviation (std) one. Then, `ED` computes the distance between the two normalized time series.

Both the query and its resulting matches were visualized without any deformations, such as the ones the algorithms perform to access similarity. In that way, participants had no idea how the given answers have been suggested.

3.4.4 *Task*

In both experiments participants had to make subjective similarity judgments using one of the three visualizations. The goal was to determine if using different visualizations alters which time series our users perceive as similar. Participants were shown five time series, one of which was marked as "Query". Their task was to select which of the other four time series was the most similar to the query (Figure 3.3). Those four possible choices were results returned from the similarity measures presented above. In Exp-1, two choices came from **`ED`**, and two choices came from **`DTW`**. In Exp-2, two choices came from **`ED`**, and two choices came from **`NormED`**. Details on the trial generation are described in Section 3.4.5. Participants gave their answer by clicking on the time series of their choice, which became highlighted, and they rated their confidence on a 5-point scale (”very low” to ”very high”). Although there was no time limit for the task, we instructed participants to be as fast and accurate as possible.

Participants performed the same tasks across all visualizations, but we randomized the vertical order of the five time sequences, so as to not favor one measure by presenting its results always closer to the query. We also ensured that time series were not directly one below the other to ensure that certain similarity

measures, in particular DTW, were not penalized. This way participants could not perform a low-level point-by-point comparison of horizontally aligned data series. Instead, they made a more high-level subjective judgement of whether the time series were similar or not. The fact that the sequences were not vertically nor horizontally aligned is consistent with the practices of our domain experts, who often compare patterns across sensors or trials that appear in varying vertical positions, and patterns that appear in different times (horizontal positions) and at different frequencies for different patients.

Notice that the task was a subjective assessment of similarity, so there was no correct or wrong answer. Our goal was to understand if some visualizations favor some automatic similarity measures and their invariances in terms of perceived similarity.

3.4.5 *Trial Generation*

All trials were generated from the annotated dataset described in Section 3.3.2. For each trial, we had to extract a time series that serves as the query and four additional sequences as possible matches. Two of these sequences were *Top* answers of the two different algorithms that each experiment studied: ED vs DTW (Exp-1), and ED vs NormED (Exp-2). The other two sequences were *Outsiders* that resulted from the same two algorithms, but in a lower rank.

As discussed in Section 3.3.3, one challenge was how to differentiate between the similarity search algorithms, given that they may return similar results. We thus opted for a query-extraction process that ensures that the algorithms return *Top* answers that are distinct.

Step 1: *Creating Candidate Queries.* We started from the manually annotated markers to extract possible queries. Epileptiform discharges last less than 250ms [107], but we extracted a larger window of 401ms around each marker (200ms left and right). This ensured that the full pattern of interest was included in the query, and that the sequence includes background activity (context), which can be important for assessing similarity. The resulted time series of 401ms were 501 points long in size, as recordings performed with sampling rate of 1250 values per second. From 205 annotations, we extracted a pool of 202 candidate queries. We excluded three that were very close to the beginning or the end of a recording (and thus of smaller size).

Step 2: *Finding Similar Subsequences.* For each candidate query, we ran similarity searches by using the two search algorithms of interest: ED vs DTW in Exp-1, and ED vs NormED in Exp-2. We collected the first 100 Nearest Neighbor (NN) answers for each algorithm. We focused our searches on the same iEEG sensors as the query, but answers could be part of different recordings. We extended an optimized sequential scan algorithm for early subsequence pruning [97] to support k-NN instead of 1-NN similarity search. Its average time complexity for comparing two series of the same length (n points) is less than $O(n)$ for all distance measures and is the fastest sequential scan algorithm known in the literature.

Step 3: *Selecting the Final Queries.* We then checked if the best results returned by each algorithm were unique. We considered the *Top* five answers of the two

measures that we compared each time. Those were generally not the same: an average of 62% of the *Top* five answers of the two measures was different in Exp-1, and this percentage was 55% in Exp-2. We wanted to select answers that clearly highlight the differences of the two measures. In addition, we had to avoid biases that may arise when picking *Top* answers for one measure that are also highly ranked for the other measure (and therefore more probable to be selected). Thus, we looked at queries where the *Top* five answers of one measure did not appear within the *Top* ten of the other. This resulted in a set of 30 queries for Exp-1 and a different set of 31 queries for Exp-2, from which we randomly picked 30 queries.

Step 4: *Choosing the Answers to each Query.* The experimental trials were formed from those 30 queries. For each query, we had to determine the four candidate answers to present to participants as choices for the task. Two of the four possible answers presented to participants were the highest ranked answers of each algorithm from Step-3 (referred to as `Top-ED`, `Top-DTW`, and `Top-NormED`, respectively for each algorithm). Another two answers were produced in a way similar to Step-3, but looking at answers between the lower 20-30 rank of each algorithm (we refer to them as `Out-ED`, `Out-DTW`, and `Out-NormED`). *Outsiders* were expected to be perceived as less similar than *Top* answers, but were still valid answers to the query. They provided a control for assessing the accuracy of participants' answers with respect to the underlying algorithms, and acted as distractors to make the task more realistic, given that analysts may search through many subsequences to find a match.

3.4.6 *Experimental Design*

We followed a within-participants design – all participants were exposed to all three visualization techniques. The order of appearance of the three techniques was fully counterbalanced. For each technique, participants completed 5 practice and 20 main trials.

For each experiment, we generated a different set of 30 distinct trials (see Section 3.4.5). To make use of the full set of trials, we divided the trials in 3 bins of 10, and each participant saw one bin during training and the other two during the experiment (counterbalanced across participants). Overall, each trial was tested by exactly 12 participants. Each participant performed the same 20 trials for all three visualizations, but we randomized the vertical order and horizontal shift of the five time sequences, including the query. This ensured that participants could not recognize the queries or their choices between conditions.

In summary, each experiment consisted of:

18 participants
× 3 visualizations (`LC`  , `HG`  , `CF` )
× 20 query-answer trials
= 1080 trials per experiment

3.4.7 *Procedure*

The experimental procedure was similar for both experiments. Before starting, participants completed a short color blindness test using the Isihara plates. If they failed the test, they were not be allowed to proceed. They then signed a consent form and continued with a training session on how to read the respective visualization technique. Before the main experiment, participants had to pass three readability tests, where they compared values of different points in a time series.

As we were interested in participants' intuitive perception of similarity across visualizations, we gave no instructions about how to interpret similarity, did not mention invariances, and did not provide any guidelines about how to assess similarity with each technique. A similar approach was used by Correll and Gleicher [31]. Furthermore, we did not explain what the data represented or how the queries and their candidate answers were generated.

After the experiment, participants completed a questionnaire to provide background information and evaluate the three visualization techniques. The experiment lasted from 45 to 80 minutes.

3.4.8 *Experimental Measures*

We use a mix of measures to evaluate the outcome of our experiments. These measures assess the types of answers given by participants, their accuracy with respect to the similarity algorithms that we tested, and their agreement among participants. In addition, we measure participants' confidence about their answers, time performance, as well as their subjective assessment of the three visualizations.

Type of Answers: We count the number of occurrences of each type of answer. For Exp-1, we count `Top-ED`, `Top-DTW`, `Out-ED`, and `Out-DTW`. For Exp-2, we count `Top-ED`, `Top-NormED`, `Out-ED`, and `Out-NormED`. Counts provide raw information about participants' choices and are used to construct our ratio measures (next).

DTW vs ED and **NormED vs ED:** We assess participants' tendency to select the *Top* answers of one similarity measure over the other by calculating the ratio of their counts. For Exp-1, we take the ratio of the counts of `Top-DTW` to those of `Top-ED`. A ratio greater than 1 indicates a preference for the *Top* answers of `DTW`. For Exp-2, we take the ratio of the counts of `Top-NormED` to those of `Top-ED`. Here a ratio greater than 1 indicates a preference for the *Top* answers of `NormED`. We compare the difference of these ratios between techniques, a difference greater or smaller than zero provides evidence that the techniques differ.

Outsider vs Top Answers: We assess the accuracy of participants' answers with respect to the answers of the similarity measures by calculating the ratio of the counts of their *Outsiders* to the counts of their *Top* answers. A large ratio indicates a relatively large number of *Outsiders* in participants' choices.

Agreement: We assess the level of consensus in participants' choices with agreement coefficients, which are commonly used in the context of inter-rater reliabil-

ity studies [55]. High agreement demonstrates low subjectivity in participants' choices. In contrast, low agreement indicates high uncertainty when making decisions. It may also imply that similarity perception is highly subjective.

We choose the κ_q coefficient of Brennan and Prediger [14]. The coefficient assumes that all q categories are selected by chance with the same probability $p_e = 1/q$. This assumption is valid in our case, since the $q = 4$ alternative answers were presented in a random order to participants, which avoided problems of bias [116]. In addition to overall agreement, we assess agreement *specific to categories* [106]. This allows us to assess how agreement is divided across different types of answers.

Time Performance: We measured the time it takes participants to complete a task, from the moment the time series are shown on the screen to the moment participants select their final answer and validate it. Although assessing time performance was not a primary goal of our experiments, this measure allows us to compare how easy or difficult it was to perform similarity tasks with each visualization technique.

Subjective Measures: We recorded participants' self-reported level of confidence on their answers to each query. We use this measure of confidence in conjunction with agreement measures. Participants rated their confidence on a 5-point scale ("very low" to "very high"). High confidence reflects certitude about the choice of their answer, while low confidence implies incertitude about the properness of their answer. In addition, at the end of each experiment, we collected subjective evaluation of the three visualizations from our users by asking them to fill in a Questionnaire Google-Form. We recorded participants' assessment across five dimensions: visual perception (how easy or difficult it was to visually identify patterns); cognitive effort (how easy or difficult it was to make decisions); accuracy (how accurate they think that their answers were); time performance (how fast they think that they completed the tasks); and overall experience (how effective they think that each technique was for the given tasks). We used a 7-point Likert scale to collect participants' answers.

3.4.9 *Expected Outcomes*

LC are extensively used in practice, so one could expect that it is the most appropriate technique for determining similarity of time series. HG and CF have not been studied in the context of perceived similarity tasks before, thus existing evidence about how they would perform compared to LC is limited. Previous studies have shown that HG were faster than Line Charts for discrimination tasks, but slower for peak and trend detection tasks [63]. Whereas CF have been shown to be a promising representation for overview tasks [32]. Similarity search likely requires both low-level (i.e., detecting peaks) and overview tasks.

In terms of similarity measures, Dynamic Time Warping (DTW) is widely considered to give better results than Euclidean Distance (ED). For LC, Eichmann and Zgraggen [41] found that DTW generally produce rankings that are closer to human-annotated ranking, so we expected to find similar results. On the

other hand, `Z-normalization` is a recommended practice for all similarity measures [35], thus one could predict that it would produce more similar answers. However, we also expected that color encodings might be sensitive, i.e., non-invariant, to y-offset and amplitude variations.

3.5 RESULTS

We present the results of the two experiments. Our statistical analysis is largely based on interval estimation [36], as this approach better supports future replication efforts. All analyses reported were planned before data were collected.

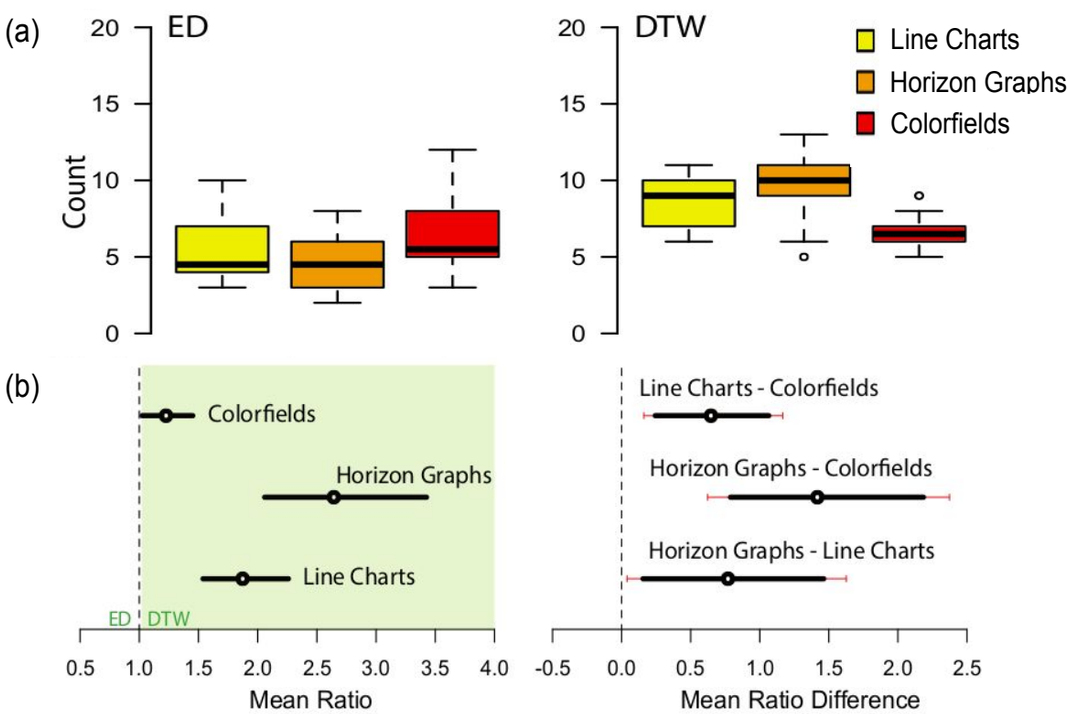


Figure 3.4.: **Experiment 1**: (a) Count of `Top-ED` vs. `Top-DTW` answers as selected by our participants under each visualization technique. The horizontal black lines show the average count. (b) Interval estimates comparing the mean ratios of `Top-DTW` to `Top-ED` answers. Error bars represent 95% CIs. For mean ratio differences, we also show (in red) CIs adjusted for three pairwise comparisons with Bonferroni correction. The dotted vertical lines show the values of reference.

3.5.1 *Invariances: Time-Warping and Z-Normalization*

We first examine how the three visual encoding techniques affected participants' choices in favor or against the two invariances of interest. Our analysis relies on ratios of counts, where counts are not independent. The sampling distribution of such measures can be complex and hard to approximate with analytical methods. We thus use bootstrapping methods to construct 95% confidence intervals (CIs) of the mean. We apply Efron's [40] bias-corrected and accelerated (BCa) bootstrap method as implemented by *R*'s *boot* package [21]. For our analyses, we construct confidence intervals with 10K bootstrap iterations.

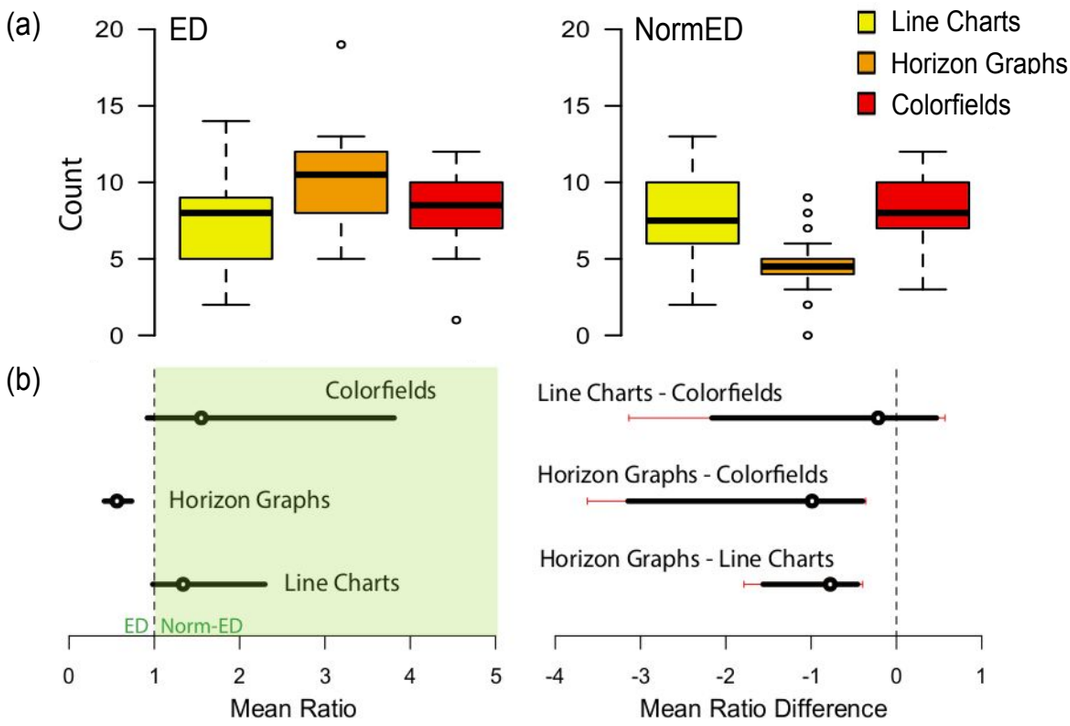


Figure 3.5.: **Experiment 2**: (a) Count of `Top-ED` vs. `Top-NormED` answers as selected by our participants under each visualization technique. The horizontal black lines show the average count. (b) Interval estimates comparing the mean ratios of `Top-NormED` to `Top-ED` answers. Error bars represent 95% CIs. For mean ratio differences, we also show (in red) CIs adjusted for three pairwise comparisons with Bonferroni correction. The dotted vertical lines show the values of reference.

Exp-1 (`DTW vs ED`): Figure 3.4a shows the number of `Top-ED` vs. `Top-DTW` answers that participants selected under each visualization technique. Based on these counts, we computed the ratios `Top-DTW` to `Top-ED` answers. Figure 3.4b presents interval estimates for individual mean ratios (left) and their differences between visualizations (right). Mean ratio greater than 1.0 indicates preference for `Top-DTW` answers. For all three techniques, we observe that participants considered as more similar to the query the `Top-DTW` answers. This trend is however different across visualization techniques. It is especially pronounced for `HG`, where `Top-DTW` answers were on average 2.64 (std = 1.49) times more frequent than `Top-ED` answers. The mean ratio of `Top-DTW` to `Top-ED` answers drops to 1.87 (std = 0.80) for `LC`, and 1.23 (std = 0.48) for `CF`.

Exp-2 (`NormED vs ED`): Figure 3.5a shows the number of `Top-ED` vs. `Top-NormED` answers that participants selected under each visualization technique. Based on these counts, we computed the ratios `Top-NormED` to `Top-ED` answers. Figure 3.5b presents interval estimates for both mean ratios (left) and their differences between visualizations (right). Mean ratio greater than 1.0 indicates preference for `Top-NormED` answers. We observe a strong tendency in `HG` for participants to not find as similar `Top-NormED` answers, where their mean ratio to `Top-ED` answers is equal to 0.56 (std = 0.36). In contrast, with the other visualizations they lean towards z-normalized answers, with mean ratios equal to 1.33 (std = 1.18) for `LC` and 1.55 (std = 2.41) for `CF`. However, due to large variance, this trend is not clearly supported by statistical evidence. We see that `HG` favor `Top-ED`

answers more than the other techniques, but we observe no clear difference between LC and CF.

3.5.2 *Outsiders vs Top Query Answers*

We further analyze the ratio of *Outsiders* to *Top* query answers by using a similar analysis procedure. We based again our statistical analysis on BCa bootstrap confidence intervals constructed with 10K bootstrap iterations.

Exp-1: Based on the counts of *Outsiders* vs. *Top* answers selected by our participants, we computed the ratios *Outsiders* to *Top* query answers for Exp-1. Figure 3.6 shows interval estimates for these mean ratios (left) and their differences for three pairwise comparisons of the visualization techniques (right). Mean ratio greater than 1.0 indicates more *Outsiders*. Clearly, the *Top* answers of the two algorithms dominated participants' choices. However, in many cases, participants perceived *Outsiders* as more similar than *Top* answers. Their ratio was 0.39 (std = .20) for HG, 0.49 (std = .22) for LC, and 0.63 (std = .36) for CF. The difference is more evident between HG and CF. The latter resulted in a relatively large number of *Outsiders*.

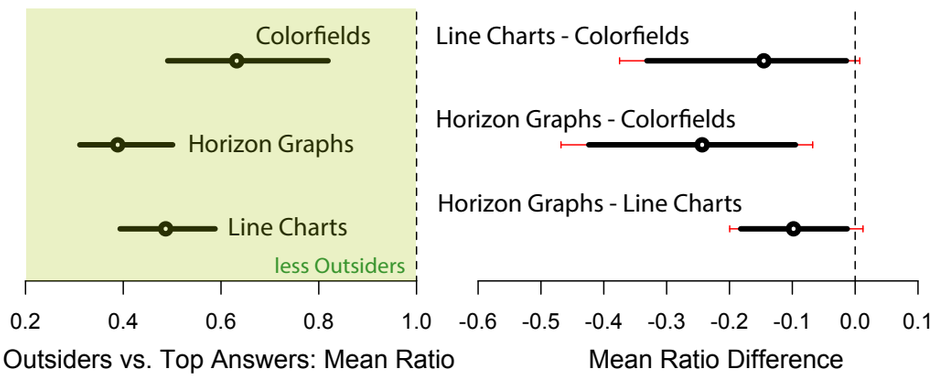


Figure 3.6.: **Experiment 1**: Interval estimates comparing the mean ratios of *Outsiders* to *Top* query answers. Error bars represent 95% CIs. Red extensions show the adjustment for three pairwise comparisons.

Exp-2: Based again on the counts of *Outsiders* vs. *Top* answers selected by our participants, we computed the ratios *Outsiders* to *Top* query answers for Exp-2. Figure 3.7 presents interval estimates for these mean ratios (left) and their differences for three pairwise comparisons of the visualization techniques (right). Mean ratio greater than 1.0 indicates more *Outsiders*. Again, the *Top* answers dominated participants' choices. However, we now observe the opposite trend, and differences between the techniques are less clear. The ratio of *Outsiders* to *Top* answers was 0.40 (std = .27) for HG, 0.31 (std = .21) for LC, and 0.27 (std = .16) for CF. CF now resulted in a lower ratio than HG. Combined with the results of Section 3.5.1, these results seem to suggest that CF are less appropriate for DTW, while HG are less appropriate for z-normalized answers.

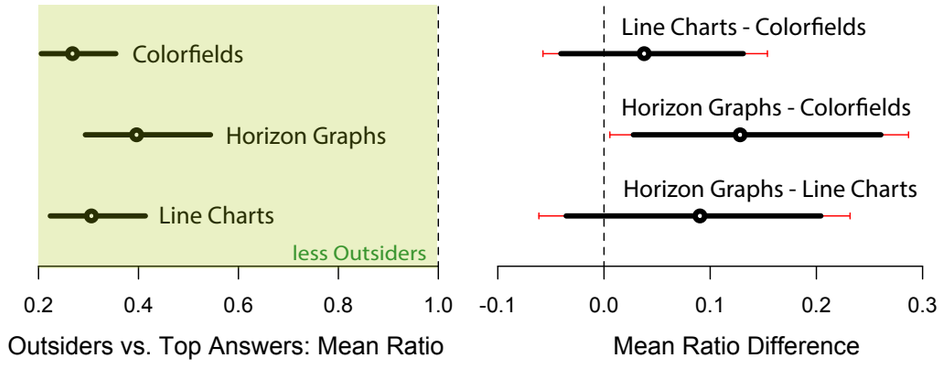


Figure 3.7.: **Experiment 2**: Interval estimates comparing the mean ratios of *Outsiders* to *Top* query answers. Error bars represent 95% CIs. Red extensions show the adjustment for three pairwise comparisons.

3.5.3 *Agreement*

To construct confidence intervals for our agreement estimates, we use the jackknife technique [55, 116] by assuming that raters, i.e., participants, are randomly sampled from a larger population, whereas the set of queries is fixed.

Exp-1: Table 3.1 summarizes the results of Exp-1. Overall, agreement is higher than zero for all three techniques. This verifies that similarity perception was not fully subjective and that participants' choices were not random. However, agreement values are generally low for `HG` and `CF`, which implies a higher subjectivity of participants' choices with these techniques. Overall, we observe a higher agreement for the choice of `Top-DTW` answers. This is especially the case for `HG` - this further shows the tendency of the technique towards `DTW`, as `Top-ED` answers were chosen with no consistency among participants. We observed a positive linear correlation between agreement values and the average confidence level reported by participants for each task (Pearson's moment correlation was $r = .45$, 95% CI = [.27, .60]). This result is not surprising – agreement or disagreement is largely due to the confidence or uncertainty with which participants make choices. The higher the confidence is, the more consistent participants are in their choices (i.e., general agreement increases).

Exp-2: Table 3.2 summarizes the results of Exp-2. Again, overall agreement is higher than zero for all techniques. Agreement values are now more balanced across techniques. We note that `HG` resulted in low agreement values for `z-normalized` answers. This further shows that the technique does not favor `z-normalization` and as a result, may not promote *amplitude* and *y-offset invariance*. For this experiment, Pearson's moment correlation between participants' self-reported confidence level and agreement was $r = .59$, 95% CI = [.43, .71].

3.5.4 *Time Performance*

Time measures are well-known to follow lognormal distributions [9, 73], thus we log-transform time values and analyze them with standard parametric methods that assume normal distributions. According to this approach, comparisons

Table 3.1.: **Experiment 1**: Specific and overall agreement values (Brennan-Prediger κ_q). Brackets show 95% jackknife CIs.

	Line Charts	Horizon Graphs	Color Fields
Top-ED:	.42 [.22, .62]	.04 [−.07, .16]	.28 [.10, .47]
Top-DTW:	.54 [.41, .68]	.41 [.28, .55]	.35 [.22, .49]
Outsider-ED:	.14 [−.09, .36]	−.06 [−.20, .07]	.21 [.00, .42]
Outsider-DTW:	.39 [.26, .52]	−.01 [−.19, .17]	.07 [−.07, .22]
Overall:	.44 [.36, .52]	.21 [.13, .29]	.26 [.18, .33]

Table 3.2.: **Experiment 2**: Specific and overall agreement values (Brennan-Prediger κ_q). Brackets show 95% jackknife CIs.

	Line Charts	Horizon Graphs	Color Fields
Top-ED:	.38 [.19, .57]	.48 [.32, .63]	.41 [.27, .55]
Top-NormED:	.43 [.29, .57]	.14 [.01, .27]	.43 [.28, .59]
Outsider-ED:	.03 [−.23, .29]	−.03 [−.19, .12]	−.01 [−.27, .24]
Outsider-NormED:	.05 [−.09, .20]	.06 [−.14, .25]	.05 [−.12, .21]
Overall:	.33 [.21, .45]	.27 [.20, .35]	.34 [.23, .45]

between techniques are based on the ratio of their *median times* rather than their mean time differences [36].

Exp-1: Mean task-completion time was 20.5 sec (std = 13.9 sec) for LC, 23.7 sec (std = 9.1 sec) for HG, and 15.6 sec (std = 7.5 sec) for CF. Figure 3.8a shows the mean and median times, and interval estimates for median times (left) and their ratios (right). We observe that CF was the fastest technique. And we have some evidence that HG were on average 33.6% slower than LC.

Exp-2: Mean task-completion time was 21.1 sec (std = 12.6 sec) for LC, 28.8 sec (std = 15.8 sec) for HG, and 21.5 sec (std = 13.2 sec) for CF. Figure 3.8b shows the mean and median times, and interval estimates for median times (left) and their ratios (right). We found no evidence of a difference between LC and CF. HG was again the slowest, on average 40% slower than the two other techniques.

3.5.5 *Subjective Evaluation*

Figure 3.9 presents a summary of participants' evaluation of the three techniques. We combine the results of both experiments, as the overall trends were similar. Overall, LC was rated high across all evaluation criteria, while HG received the lowest ratings.

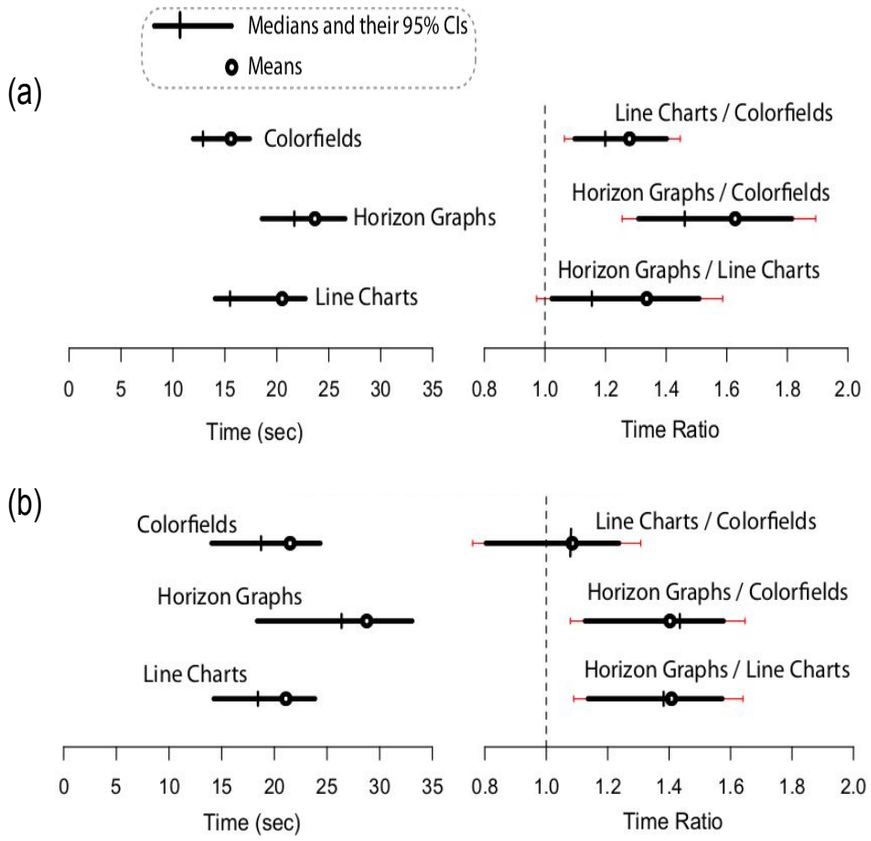


Figure 3.8.: Interval estimates comparing the median task-completion time for each visualization technique for (a) Exp-1 and (b) Exp-2. Error bars represent 95% CIs. Red extensions (right) show adjustments for three pairwise comparisons.

3.6 DISCUSSION AND DESIGN IMPLICATIONS

Results from both experiments suggest that humans may perceive similarity differently, depending on the visualization, and that different visual encodings are invariant to specific signal parameters.

In Exp-1 participants preferred results returned by Dynamic Time Warping (DTW), i.e., subsequences that can be shifted in the x-axis and locally stretched or compressed. This finding corroborates previous evidence from both data-mining and visualization communities [35, 41] that DTW is superior to Euclidean Distance (ED). Nevertheless, this effect differs across visualization techniques. It is stronger for Horizon Graphs, likely due to this technique's double encoding. Color variations often communicate high-level patterns (spikes/valleys, positive/negative ranges), while shape and position reveal details. Participants may have focused on the high-level patterns in color to determine similarity, considering shape and position (which encode warping and x-axis shifting) as secondary factors. Line Charts favored DTW but to a lesser degree, and the trend was even weaker for Color Fields. Color Fields aid the detection of ranges of similar color[3] so it is probable that participants considered both the color of the spikes and the width of the color ranges formed around them. Thus, they were likely to avoid candidates that were too stretched or compressed. The example in Figure 3.10 demonstrates this issue.

In Exp-2 we observed a clear difference between Horizon Graphs and the two other visualizations. Horizon Graphs strongly favored the answers of ED

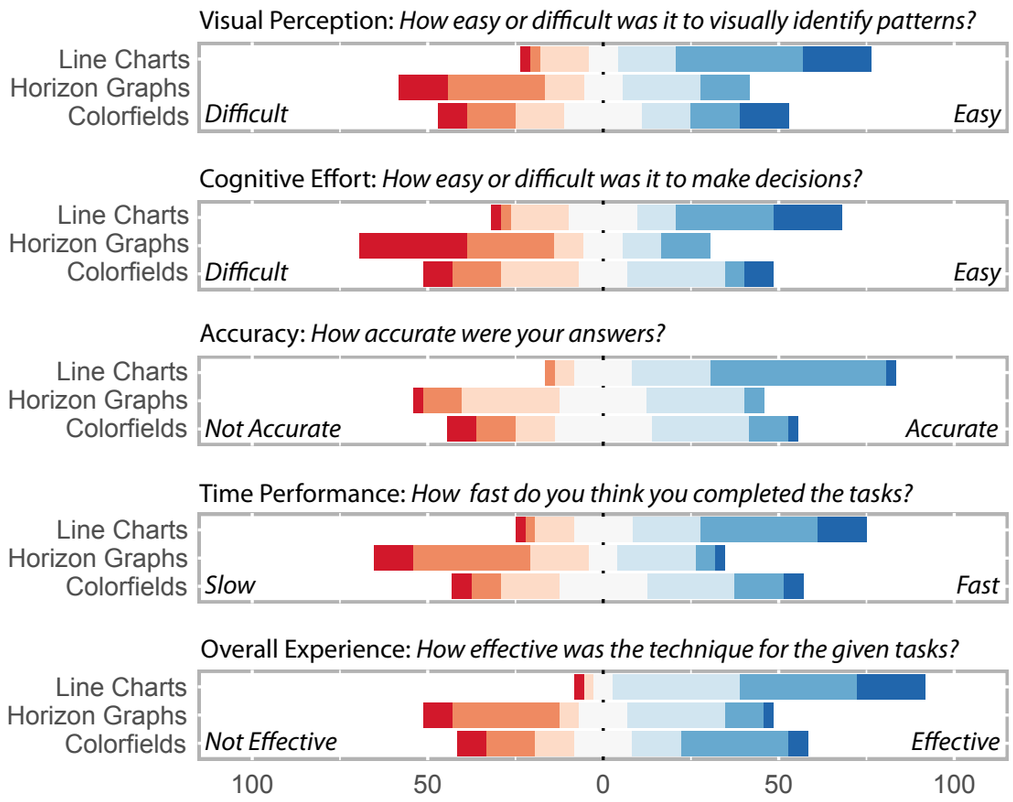


Figure 3.9.: Summary of participants' subjective evaluation of the techniques for both experiments (N = 36). For all the evaluation criteria, there were seven levels (1 = most negative to 7 = most positive).

`without z-normalization`. The opposite trend was observed for Line Charts and Color Fields, which were more favorable to `z-normalization`, i.e., invariant to amplitude scaling and y-offset shifts. The strategies mentioned before can explain these results as well. In Horizon Graphs, small amplitude and y-offset changes can fall on different sides of a band and have different colors. Thus, if participants tried to match colors rather than shape, they likely disregarded subsequences whose prominent characteristics fell on different bands (see Figure 3.11). For Line Charts and Color Fields, the exact amplitude and offset values can be less critical, as people seem to focus on relative values and overall shapes.

Overall, agreement scores were lower in Horizon Graphs and time performance was slower, which indicates this encoding can be difficult to visually identify patterns and make decisions when using it.

In both experiments, participants tended to select the top answers of the algorithms rather than their outsiders, irrespective of the visualization technique. This confirms that the rankings of these algorithms capture real differences in perceptual similarity.

`Design Implications:` Overall, our work indicates that the choice of visualization affects what temporal patterns people consider as similar, i.e., *the notion of similarity in time series is not visualization independent*. Visualization designers need to consider what invariances are important in the data domain [35] and suggest visualizations appropriately. Similarly, if designers use algorithmic distance measures, they should consider visualizations that match the invariances of those measures, or viewers could lose confidence in their results.

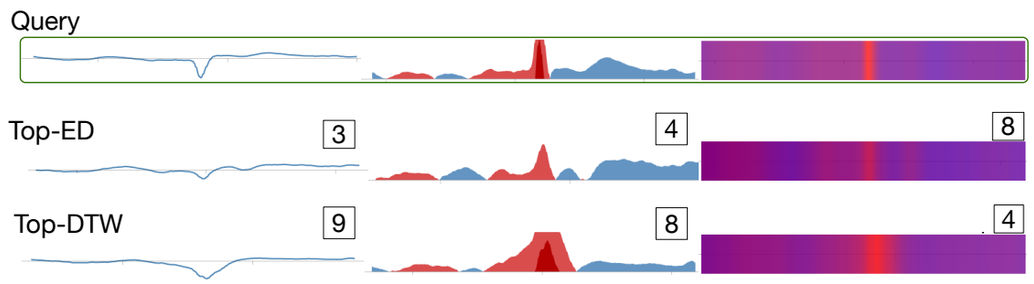


Figure 3.10.: **Experiment 1**: A query for which different visualizations resulted in different choices. Boxes show the number of participants (out of 12) who chose the specific answer. This example shows a strong preference for `Top-DTW` under Line Charts and Horizon Graphs and a strong preference for `Top-ED` under Color Fields. Overall, Color Fields can be more sensitive than Line Charts and Horizon Graphs to stretching deformations along the time axis.

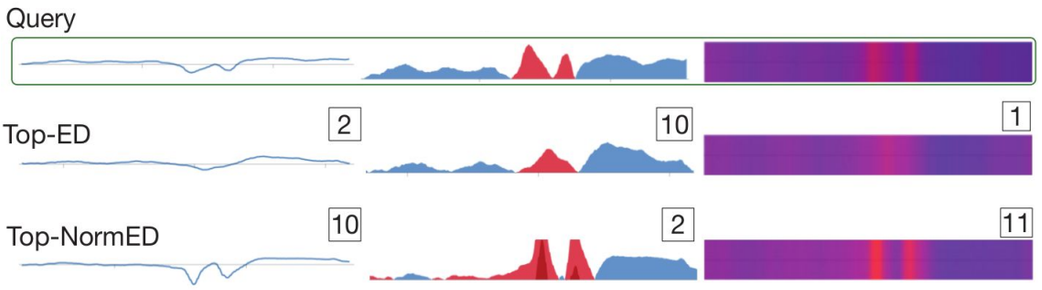


Figure 3.11.: **Experiment 2**: A query for which different visualizations resulted in different choices. Boxes show the number of participants (out of 12) who chose the specific answer. This example shows a strong preference for `Top-NormED` under Line Charts and Color Fields and a strong preference for `Top-ED` under Horizon Graphs. Overall, Horizon Graphs seem to exaggerate flat signals and are more sensitive to deformations along the y-axis.

Our results suggest that *Color Fields are less appropriate for domains that require invariance to temporal warping*, as they are sensitive to temporal warping and shifting. Here, Horizon Graphs are a viable alternative to Line Charts, as they are less sensitive to warping. Nevertheless, designers should consider the visual complexity of time series visualizations. Agreement was lower for Horizon Graphs and time performance was slower, while participants reported they found it more difficult to visually identify patterns and make decisions when using it.

In turn, *Horizon Graphs are less appropriate when amplitude and y-offset invariance is important*, as they are sensitive to value transformations along the y-axis due to the explicit limits of their bands.

Finally, as in previous work using Line Charts [35, 41], our results support that DTW, an algorithm that is invariant to temporal warping, is likely closer to what we perceive as similar in temporal patterns, and thus *DTW could be considered as a good default* unless otherwise indicated by the data domain [35].

3.7 LIMITATIONS

There are several limitations to our work. First, we focused on a small number of similarity measures. The data-mining literature has studied measures for other types of invariance [10]. Future work needs to determine what visualizations

best match such measures. Furthermore, our dataset consists of EEG data that have specific pattern characteristics, such as spikes followed by rapid discharges. Although we believe that our high-level results will hold for other types of signals, the sensitivity of visual perception to certain signal deformations may be less or more pronounced. Further studies are needed to validate our findings in a wider range of patterns and datasets from other domains.

Our implementation of Color Fields used a naive, linear RGB interpolation. This approach leads to a color space that is not perceptually uniform, i.e., differentiating variations may be harder for one of the two color extremes. On the other hand, it may extend the differences near the central range of the color space, in magenta tones which humans are more sensitive to [72]. This central range is where low-amplitude variations and spikes (which might be important for EEG signals) are located. We conducted a follow-up experiment (N = 18 participants) that compared linear RGB interpolation to a perceptually uniform CIE L*a*b color space (see Section 3.8 for details). Accuracy and agreement scores were very similar for the two techniques, while most participants (10 vs 6) found that is was easier to identify patterns with linear RGB interpolation. CIE L*a*b resulted in less pronounced differences between similarity measures, but we found no statistically significantly differences between the two interpolation techniques. We report the detailed results of this experiment in Section 3.8. Nevertheless, it is possible that differences in these color mappings exist in other types of temporal patterns. Moreover, in domains where similarity comparison is the only task of interest, one could also consider dynamic mapping variations (e.g., difference color maps, or ones based on equi-depth or equi-width binning of time series values to provide wider color ranges for the most frequent values), that nonetheless distort the original signals. The effect of color in time series similarity is an exciting future research direction.

We focused on a small number of time series to compare, with a generous vertical drawing area. While we hypothesize that our results will hold for larger number of time series, their size might affect these results. For example, we expect that Color Fields will scale well, but it is known that the choice of the aspect ratio affects readability in Line Charts [113]. Thus, for Line Charts and to a lesser degree for Horizon Graphs, a reduced vertical space could lead to a loss of small patterns and reinforce large structures (peaks, valleys) altering similarity perception.

Finally, we plan to compare additional visual encodings or variations of the ones studied in this study, such as composite visualizations that go beyond Horizon Graphs [62], and area charts with alternative designs, e.g., designs based on single or dual fill color, and mirroring.

3.8 FOLLOW-UP: COLOR INTERPOLATION TECHNIQUES

In our original study, our Color Fields implementation used a naive RGB color interpolation between red and blue hues. This approach leads to a color space that is not perceptually uniform, i.e., differentiating variations can be harder for one of the two color extremes. On the other hand, it may extend the differences near the central range of the color space (in magenta tones which humans are

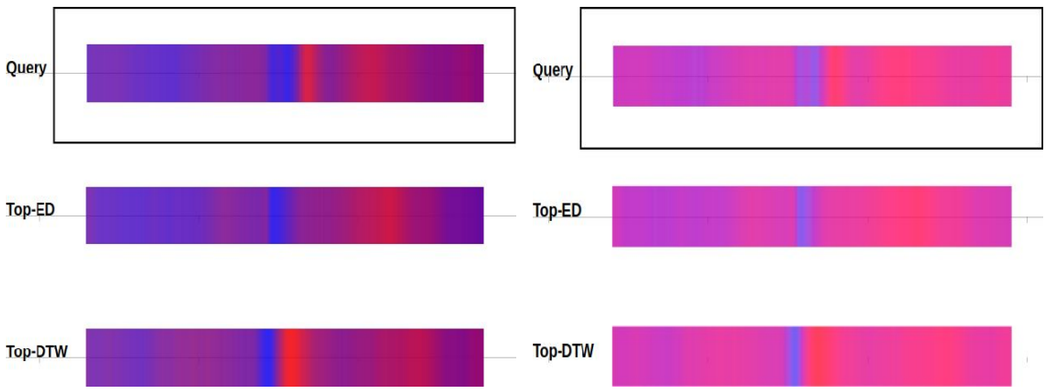


Figure 3.12.: Two color interpolation techniques for Color Field visualization (RGB left, LAB right), compared in our experiment in order to understand whether humans perceive similarity in a similar manner. This example shows a query and two of the four possible answers participants had to choose from. The answers here come from the ED and DTW automatic similarity measures.

more sensitive to [72]). This central range is where low-amplitude variations and spikes, which might be important for EEG signals, are located. Nevertheless, it is unclear how this color mapping fairs against others that are more perceptually uniform.

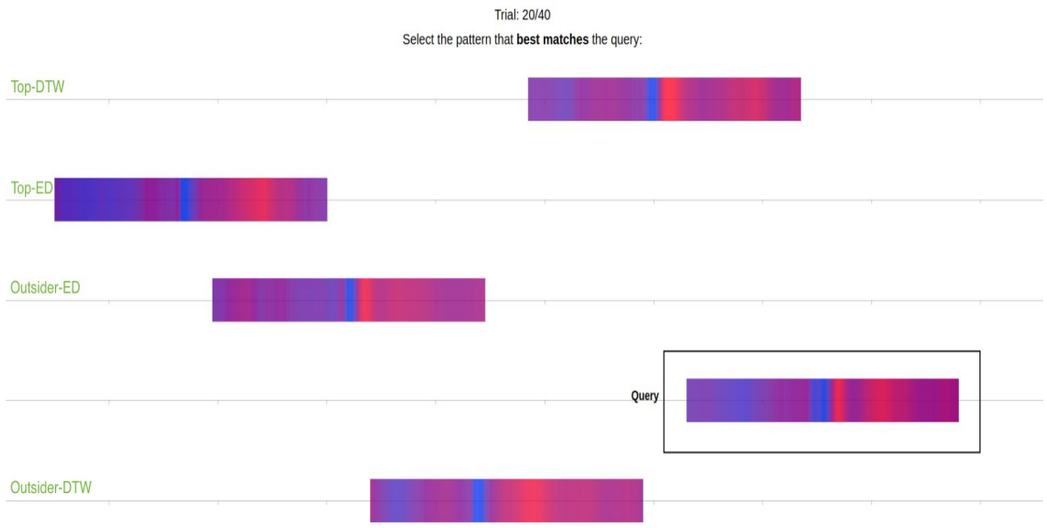
We thus conducted a follow-up experiment to study and compare the RGB interpolation to one that is perceptually uniform (in our case CIE L*a*b*) (Figure 3.12). We wanted to see if the color interpolation used changes whether time series are perceived as similar or not. As in our original study, we investigated time-warping invariance by asking participants to compare the results of Euclidean Distance (ED) [42] and Dynamic Time Warping (DTW) [11] (Exp-1 in our original study); and amplitude and offset invariance by asking participants to compare the results of ED with and without z-normalization [52] (Exp-2 in our original study). Aspects in the setup and procedure are common in the experiments of the original study and this follow-up, so we refer to the original study for details unless differences are explicitly stated.

3.8.1 *Experimental Design*

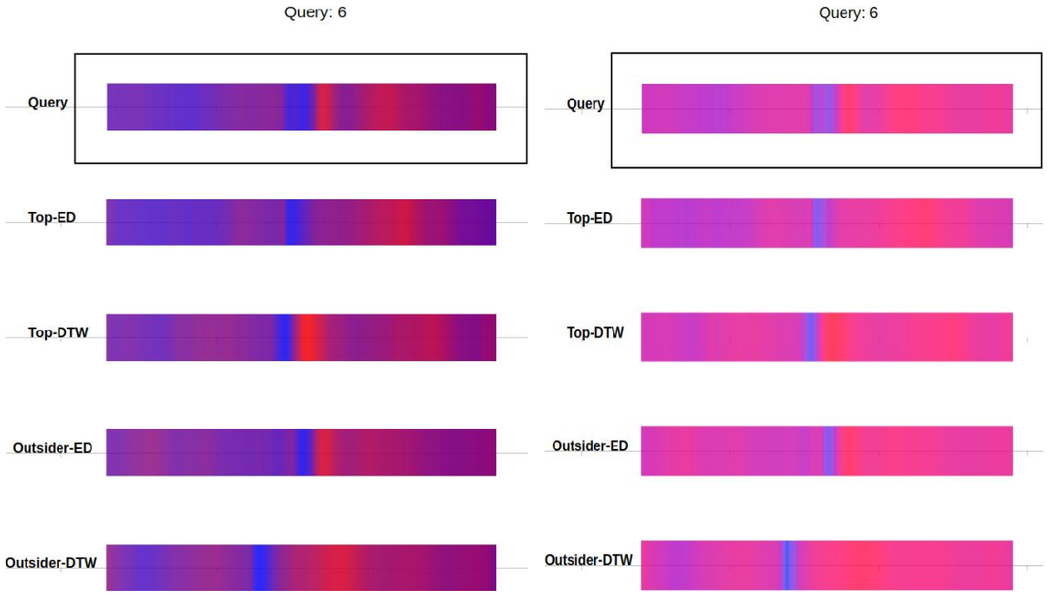
Participants & Apparatus: A total of 18 volunteers (six women), 22 to 30 years old (M = 25, SD = 2.4), participated in our follow-up study without monetary compensation. We recruited them from a local university mailing list. None of these participants had taken part in our previous studies. Our participants came from different scientific backgrounds, including students and researchers in Computer Science, Robotics, Material Engineering, and Physics.

The setup was identical to that of the original study. We used the same 24" DELL monitor set to 1920 × 1080 resolution.

Tasks and Procedure: As in the original experiments, we followed a within-participants design – all participants were exposed to both color interpolation techniques. The order of appearance of the two techniques was fully counterbalanced. For each technique, participants completed 5 practice and 40 main trials.



(a)

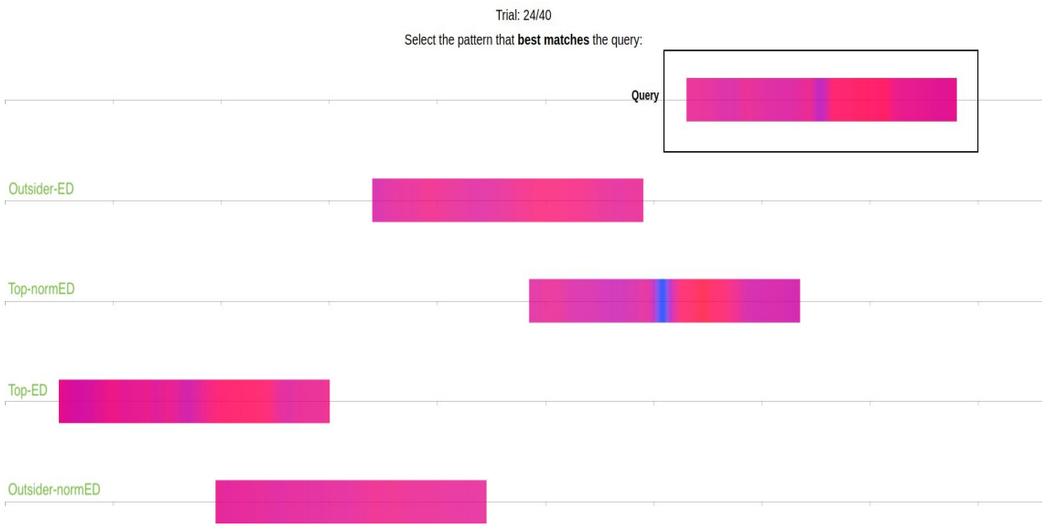


(b)

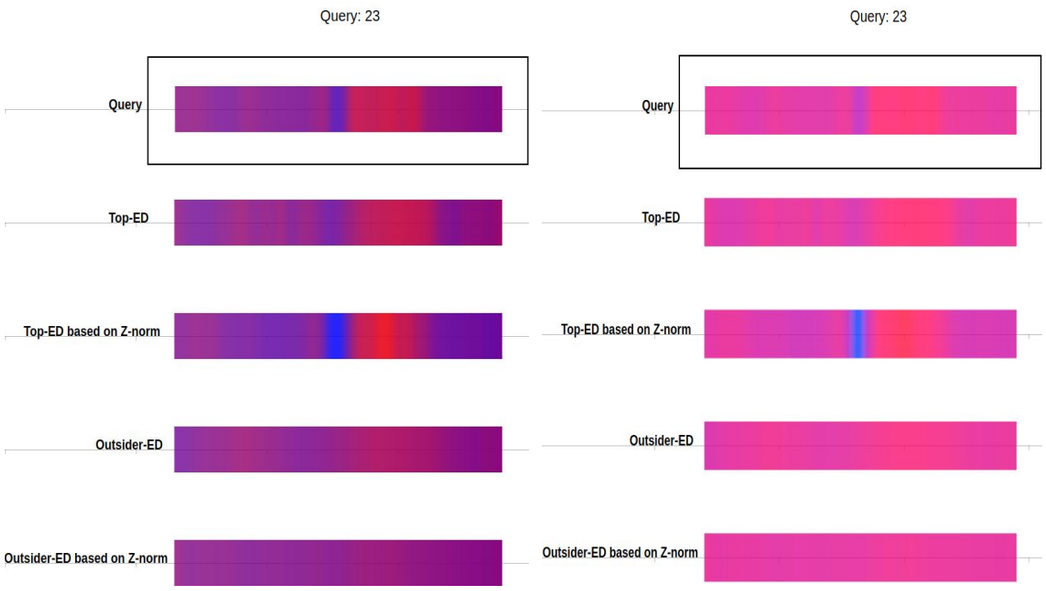
Figure 3.13.: (a) Experimental trial (stimulus) for the RGB condition. The answers come from the ED and DTW similarity measures. The answer order and horizontal shift was randomized across trials. Green annotations (indicating the type of answer) are for illustration purposes only and were not visible in the experiment. (b) The complete query-answer trial used to generate the stimulus in (a), under both the RGB (left) and LAB (right) condition.

The main difference to the previous study procedure, was that participants saw trials from both Exp-1 and Exp-2 of the original study (since the number of trials was fairly small). We decided on this combination, since similarity judgement is perceptual and subjective in nature, and the instructions we gave to our participants (here and in the original study) do not make any mention of similarity measures or invariances (the factors that are different across experiments in our main study).

To make use of the full set of queries from the original study (60 queries in total, 30 queries of Exp-1 and 30 queries of Exp-2), we divided the queries in 3 bins of 10 for each experiment, and each participant saw one bin from each



(a)



(b)

Figure 3.14.: (a) Experimental trial (stimulus) for the LAB condition. The answers here come from the ED and NormED similarity measures. The answer order and horizontal shift was randomized across trials. Green annotations (indicating the type of answer) are for illustration purposes only and were not visible in the experiment. (b) The complete query-answer trial used to generate the stimulus in (a), under both the RGB (left) and LAB (right) condition.

experiment during training and the other two bins from each experiment during the study (counterbalanced across participants). Overall, each query-answer trial was tested by exactly 12 participants (same as in the original study). Each participant performed the same 40 trials for both techniques, but we randomized the horizontal shift and vertical order of the five time subsequences, including the query (see Figure 3.13a and Figure 3.14a). For detailed justifications we refer the reader to Section 3.4.

An example of an experimental trial (stimulus) and of the query and answers, used to generate the stimulus, can be seen in Figure 3.13. The stimulus shown

here is for the RGB condition, and the similarity measures used are ED and DTW. Another example of experimental trial under LAB interpolation, where the similarity measures are ED and ED based on z-normalization (NormED), can be seen in Figure 3.14, together with the complete query-answer trial used to generate the stimulus.

In summary, the follow-up study consisted of:

18 participants
× 2 color interpolations
× 40 query-answer trials
= 1440 trials

Color Interpolation Techniques: Previous work considers color scales of two [4, 86] or more colors [102]. We opted for a simple two-color scale in our experiment, as we did in the original study. As in the original study, we chose red tone (#ff0000) for the most negative and blue (#0000ff) for the most positive value for both interpolations. Pure tones were used to maximize the distance between the two extreme colors.

`RGB interpolation:` In this condition we used a simple linear RGB interpolation between the two pure red and blue tones*. An example of a generated trial under this condition can be seen in Figure 3.13a.

`LAB interpolation:` In this condition we used a perceptually uniform interpolation between the two pure red and blue tones, based on the CIE L*a*b* space†. An example of a generated trial under this condition can be seen in Figure 3.14a.

Similarity Measures: These were identical to the ones used in the original study, i.e., ED vs DTW and ED vs NormED.

3.8.2 *Results*

Invariances: Time-Warping and Z-Normalization

As in our original study, our analysis relies on ratios of counts. We use bootstrapping methods to construct 95% confidence intervals (CI) of the mean. We apply the bias-corrected and accelerated (BCa) bootstrap method as implemented by *R*'s *boot* package [21]. We construct confidence intervals with 10000 bootstrap iterations.

Figure 3.15 summarizes our results. We split our analysis into two parts. We first compare the two color interpolation techniques for the trials for which answers are given by `ED` and `DTW` algorithms (see Figure 3.15a). These trials come from *Exp-1* (see Section 3.4.3). We then compare them for the trials for which answers are given by `ED` and `NormED` (see Figure 3.15b). These trials come from *Exp-2* (see Section 3.4.3).

For `RGB` interpolation, we observe that the results of this experiment are very close to our previous experimental results (see Section 3.5). Again, `top-DTW` answers were preferred to `top-ED` answers, in trials that compare answers returned

* D3 code for RGB interpolation comes from `http://github.com/d3/d3-interpolate#interpolateRgb`
† D3 code for LAB interpolation comes from `http://github.com/d3/d3-interpolate#interpolateLab`

by the DTW and ED measures. When it comes to trials that compare ED with ED based on z-normalization, we also observe a (non-statistically significant) trend for `top-NormED` answers. For `LAB`, these trends disappear – this color interpolation technique does not seem to favor any of the similarity measures that we compared. However, any differences between `RGB` and `LAB` were not statistically significant ($\alpha = .05$).

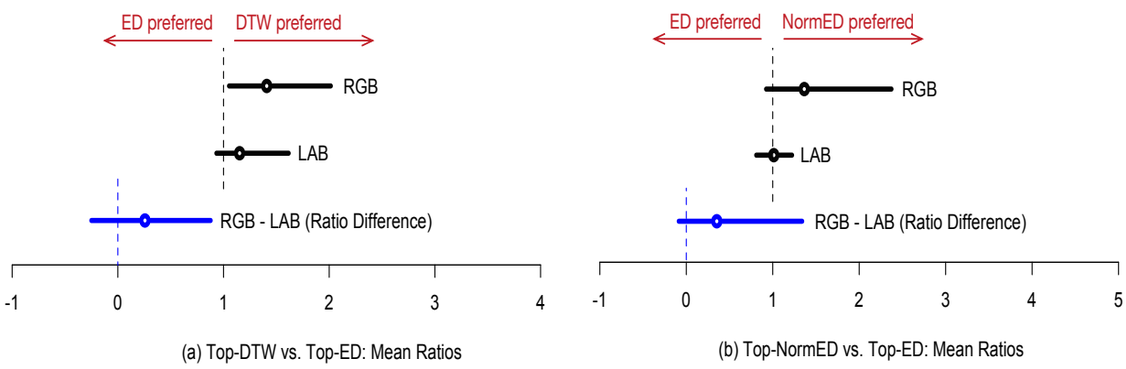


Figure 3.15.: Interval estimates comparing the mean ratios of (a) Top-DTW vs. Top-ED answers and (b) Top-NormED vs. Top-ED answers, for the two color interpolation techniques (RGB vs. LAB). In blue, we show interval estimates of the mean ratio differences of the two techniques. Error bars represent 95% CIs. The dotted vertical lines show the values of reference.

Outsiders vs Top Query Answers

We analyze the ratio of outsiders to top query answers by using a similar analysis procedure as the original study. We observe that the top answers of the two algorithms dominated participants' choices in a similar way (Figure 3.16). This indicates that choices were not made at random and that the rankings of the algorithms capture real differences in perceptual similarity. We observe that the ratio of outsiders is very similar for both color interpolation techniques – `RGB` performs at least as well as `LAB`.

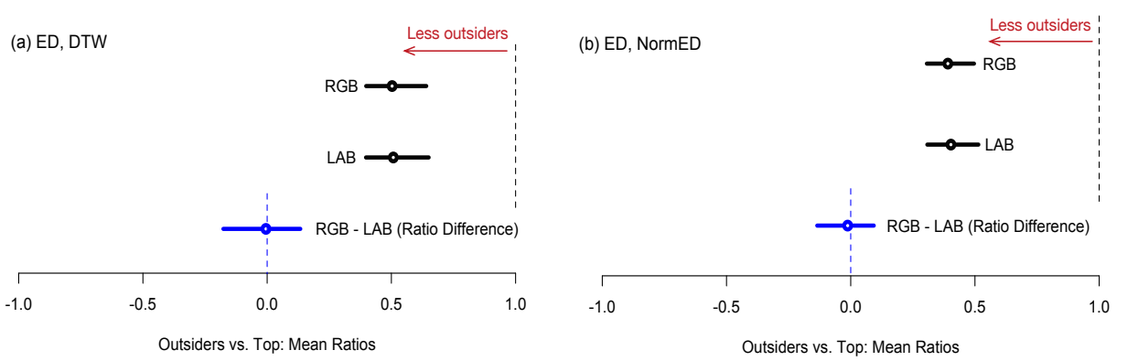


Figure 3.16.: Interval estimates comparing the mean ratios of outsiders to top query answers (a) for the ED vs. DTW trials and (b) for the ED vs. NormED trials. In blue, we show interval estimates of the mean ratio differences of the two color interpolation techniques (RGB vs. LAB). Error bars represent 95% CIs. The dotted vertical lines show the values of reference.

Agreement

We use the κ_q coefficient of Brennan and Prediger [14] to assess agreement among participants. We also use the jackknife technique [55] to construct confidence intervals by assuming that participants are randomly sampled from a larger population, whereas the set of queries is fixed. Overall agreement values are shown in Table 3.3. Agreement was above chance, while the two techniques resulted in very similar scores. These values are again consistent with the values of our previous experiments (see Section 3.5.3).

Table 3.3.: Overall agreement values (Brennan-Prediger κ_q). Brackets show 95% jackknife CIs.

	RGB	LAB
ED vs. DTW	.22 [.12, .32]	.22 [.14, .30]
ED vs. NormED	.32 [.23, .41]	.30 [.23, .38]

Time

Time measures are well-known to follow lognormal distributions [9, 73], thus we log-transform time values and analyze them with standard parametric methods that assume normal distributions. According to this approach, comparisons between techniques are based on the ratios of their *median times* rather than their mean time differences [36].

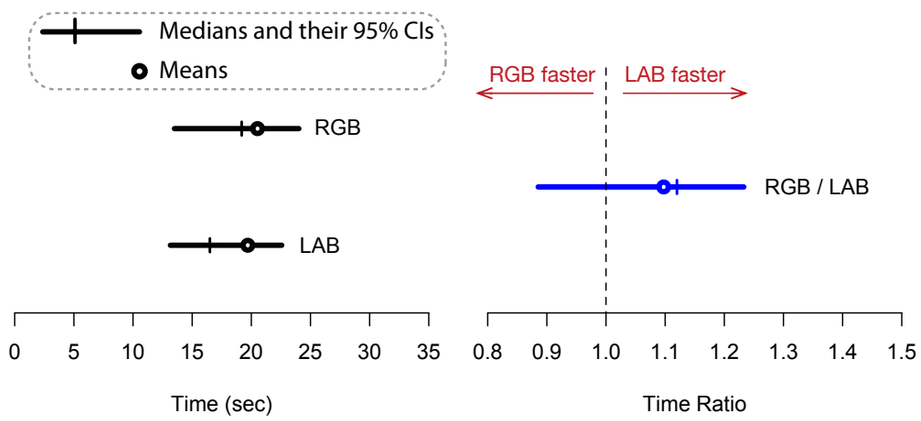


Figure 3.17.: Interval estimates comparing the median task completion time for each technique. Error bars represent 95% CIs. Red extensions (right) show adjustments for three pairwise comparisons.

Mean completion time were very close for RGB 20.5 sec (SD = 9.7 sec), and 19.7 sec (SD = 10.4 sec) for LAB. Figure 3.17 shows interval estimates for medians (left) and ratios of median times (right). We observe no clear time difference across interpolations.

User Preferences

Participants indicated in a 7-point Likert scale their preference for each technique. Lower score indicated higher preference. RGB was overall more preferred (mean score 3.61) than LAB (mean score 4.22). Thus RGB was overall more preferred (as the lower the score, the more preferred the technique). In particular 10 of the 18 participants rated RGB higher, 2 same as LAB, and 6 rated LAB higher.

3.8.3 *Discussion*

Our results show no statistical difference between RGB and LAB interpolations in similarity perception for all our comparisons of similarity measures. As in the original experiment, there is a trend to prefer DTW to ED for both color interpolations. However, this trend seems to be less clear when LAB interpolation is used. In trials comparing ED with NormED, participants tended to prefer NormED to ED for RGB interpolation. However, as in the original experiment, this trend is not statistically significant. In contrast, the LAB interpolation does not seem to favor any of the two measures. Again, the difference between the two color interpolations is non-statistically significant, so larger studies are required to verify these effects. It is possible that participants could differentiate more details in the RGB interpolation, thus favoring slightly one distance measure (DTW or NormED) over the other (ED).

Overall, the results for both interpolations are consistent with the findings from our original experiments, and the variation of color interpolation does not change our high-level results and recommendations. We can conclude that Color Fields (irrespective of interpolation) are less adapted for DTW than Horizon Graphs and Line Charts. Given the slightly different (non significant) trends for NormED and ED, recommendations are not interpolation-blind in this case. However, we can still conclude that Color Fields are better adapted to NormED than Horizon Graphs, irrespective of which color interpolation is used.

As our study does not show any significant difference between the two encodings for our similarity perception tasks, this indicates our results are fairly robust for our task and data (EEG signals). Nevertheless, this does not mean that differences do not exist for other temporal patterns. Further research is required to investigate the effect of color mapping on similarity perception for other subsequences with pattern characteristics other than EEG.

Moreover, due to our motivation domain (neuroscience), in both the main study and this follow-up experiment, we assume that viewers are interested in comparing time series but also in seeing the visualization of the raw values and their context (see Section 3.2). In other domains where similarity comparison is the only task of interest, one could also consider mapping variations that exaggerate differences. For example, one could consider taking any color map space and create an equi-depth binning of time series values. This could provide a wider color range for the most frequent values, thus exaggerating the parts of the time series with most variations. It is clear that further investigation on the choice of color space is needed when it comes to similarity judgements. We hope this work motivates future studies, and in this vain we provide our data for replication.

3.9 CONCLUSION

We presented two laboratory experiments that compare how three visualizations (Line Charts, Color Fields, and Horizon Graphs) affect how humans perceive similarity in time series. Specifically, we studied if some deformations in the data, detected by automatic similarity measures, are perceived in a differ-

ent manner depending on the visualization. Our findings indicate that all three visualizations, favor similarity results from algorithmic measures that allow flexibility in local deformations in temporal position or speed (i.e., dynamic time warping). This is the case most notably for Horizon Graphs. On the other hand, this visualization does not promote results from algorithms that are invariant to y-offset shifts and amplitude rescaling (i.e., z-normalization).

We also presented the results of a follow-up experiment. In this follow-up study, we compared the simple RGB interpolation for color mapping in Color Fields tested by our original study to one that is perceptually uniform (CIE L*a*b*). We observed that the RGB results of the follow-up study are consistent with the results from our original experiments, verifying our findings. In addition, there are no statistically significant differences between the two color interpolation techniques with regards to time series similarity perception. However, it does not mean that possible differences in these color mappings do not exist in other types of temporal patterns.

Overall, our work provides evidence that the notion of time series similarity is visualization dependent, and that when choosing visual representations, we should consider what deformations the underlying data domain considers as similar. This should be consistent with the similarity measures used in each domain. Therefore, visual analytics systems should visualize similarity search results with visualizations that effectively communicate the computed similarity. In the future, we plan to investigate how choosing appropriate visualizations to communicate similarity can affect agreement of what is similar among domain experts, and if this increases trust on the results of similarity search algorithms.

4

DATA SERIES PROGRESSIVE SIMILARITY SEARCH WITH PROBABILISTIC QUALITY GUARANTEES

Before systems visualize similarity search results, they need to compute this similarity. Due to the increasing data series size, data series analysis remains highly challenging in a wide range of human activities. Existing systems cannot guarantee interactive response times, even for fundamental tasks such as similarity search. Therefore, in this work, we develop analytic approaches that support real-time data series exploration and decision making by providing progressive similarity search results, before the final and exact ones have been computed. We investigate how fast we can provide users with first approximate, and then updates of progressive results. Our findings indicate that there is a gap between the time the most similar answer is found and the time when the search algorithm terminates. Probabilistic estimates of the final answer could help users decide when to stop the search process, i.e., deciding when improvement is unlikely, thus eliminating waiting times. We published these preliminary results in a vision paper at the BigVis workshop of the EDBT/ICDT 2019 joint conference [51]. In this workshop paper, we further developed two open challenges we identified: (i) how to efficiently compute probabilistic error estimates of progressive similarity search results in large data series collections and (ii) how to communicate them to users.

Prior work has proposed different methods for computing probabilistic error estimates of approximate similarity search results in multi-dimensional spaces. However, these methods lack both efficiency and accuracy when applied to large-scale data series collections. We developed, and experimentally evaluated using benchmarks, a new probabilistic learning-based method that provides quality guarantees for progressive k-Nearest Neighbour (k-NN) query answering results. Our approach learns from a set of queries and builds prediction models based on two observations: (i) similar queries have similar answers; and (ii) progressive best-so-far (bsf) answers returned by the state-of-the-art data series indexes are good predictors of the final k-NN answer. We provide both initial and progressive estimates of the final answer that are getting better during the execution of similarity search. Our benchmark evaluation, with synthetic and diverse real datasets, indicates that our prediction methods constitute the first practical solution to the problem, significantly outperforming competing approaches. The results of this work are currently under a second round of submission.

4.1 INTRODUCTION

In Chapter 3, we observed that time series similarity can be domain- and visualization-dependent [10, 49], and in many situations, analysts depend on time-consuming manual analysis processes. For example, we saw that neuroscientists manually inspect the EEG data of their patients, using visual analysis tools, so as to identify patterns of interest [49, 65]. In such cases, it is of paramount importance to have techniques that can operate within interactive response times [87], in order to enable analysts to complete their tasks easily and quickly.

In the past years, several visual analysis tools have combined visualizations with advanced data management and analytics techniques (e.g., [71, 96]), albeit not targeted to data series similarity search. Moreover, we note that even though the focus of the data management community is on the scalability issues related to the processing and analysis of very large data series collections, the state-of-the-art indexes currently used for scalable data series processing [20, 70, 77, 125, 135] are still far from achieving interactive response times [39].

To allow for interactive response times when users analyze large data series collections, we need to consider progressive and iterative visual analytic approaches [8, 118, 130]. Such approaches provide progressive answers to users' requests [44, 83, 109], sometimes based on algorithms that return quick approximate answers [28, 34, 43]. Their goal is to support exploration and decision making by providing progressive (i.e., intermediate) results, before the final and exact ones have been computed.

Most of the above techniques consider approximations of aggregate queries in relational databases, with the exception of Ciaccia et al. [27, 28], who provide a probabilistic method for assessing how far an approximate similarity search answer is from the exact answer in multi-dimensional metric spaces. Nevertheless, none of these works has considered data series, which have the additional characteristic of being high-dimensional: their dimensionality ranges from several hundreds to several thousands*. We note that the framework of Ciaccia et al. [27, 28] does not explicitly target progressive similarity search. Furthermore, their approach has only been tested in datasets with up to 275K vectors with dimensionality of a few dozen, while we are interested in hundreds of millions of data series with dimensionality of a few hundreds. Our experiments show that the probabilistic estimates that their methods provide [27, 28] are inaccurate and cannot support progressive similarity search in large data series collections.

Proposed Approach. In our work, we develop the first progressive approaches for data series similarity search with probabilistic quality guarantees, which are scalable to very large data series collections. Our preliminary experiments show that there is a gap between the time the 1st Nearest Neighbour (1-NN) is found and the time when the search algorithm terminates. In other words, users often wait without any improvement in their answers. Our goal is to predict how much improvement is expected when the search algorithm is still running. We can then communicate this information to users, and they can decide to terminate a progressive search in order to reduce waiting times. The challenge is

* The dimensionality of a data series is defined by its length [39], i.e., the number of points in the series

how to derive such predictions. Our experiments show that high-quality approximate answers are found very early, e.g., in less than one second. If we further inspect our results, we observe that although answers progressively improve, improvements are not radical. This implies that approximate answers are generally not very far from the 1-NN. We show that this behavior is more general and can be observed across different datasets and different similarity search algorithms [125, 135]. Our approach consists in describing this behavior through statistical models and then using these models to derive probabilistic guarantees about the k-NN distance in the form of prediction intervals. We explore query-sensitive models that can predict a probable range of the k-NN distance even before the search algorithm starts, which can then be progressively improved as approximate answers arrive.

Contributions. Our key contributions are as follows:
• We introduce the problem of *progressive data series similarity search* and formally define it.
• We demonstrate the importance of progressive results for similarity search operations in very large data series collections, and the potential benefits of such an approach.
• We investigate a family of statistical methods for supporting progressive similarity search. We show how to apply these methods to derive probabilistic distance (or distance error) bounds that improve over time.
• We perform an extensive experimental evaluation with both synthetic and real datasets, comparing our solutions to existing baselines. The results demonstrate that previous approaches cannot scale to the size and dimensionality of modern data series collections. In contrast, our solutions (each to a different extent) provide high accuracies that fit well their nominal levels. Furthermore, their probabilistic bounds become tight quickly, long before the search ends.

4.2 BACKGROUND: SIMILARITY SEARCH

Echihabi et al. [39] offer a common language to describe data series similarity search. Our definitions are largely based on their terminology.

Data Series and Data Series Collections: A *data series* $S(p_1, p_2, ..., p_\ell)$ is an ordered sequence of points with length n. Similarly, a *subsequence* $S[i : j]$ is the sequence $S(p_i, p_{i+1}, ..., p_{j-1}, p_j)$, where $1 \leqslant i \leqslant j \leqslant \ell$. A data series of length ℓ can also be represented as a single point in an ℓ-dimensional space. For this reason, the values of a data series are often called *dimensions*, and its length ℓ is called *dimensionality*.

We use $\mathbb{S}$ to denote a *data series collection* (or *dataset*). We refer to the size $n = |\mathbb{S}|$ of a data series collection as *cardinality*. We focus on datasets that contain a very large number of data series.

Distance Measures: A data series *distance* $d(S_1, S_2)$ is a function that measures the dissimilarity of two data series S_1 and S_2, or alternatively, the dissimilarity of two data series subsequences. As mentioned in Section 2.3, we chose Euclidean Distance (ED) as a measure due to its popularity and efficiency for large datasets [35].

Similarity Search Queries: Given a dataset $\mathbb{S}$, a *query* series Q, and a distance function $d(\cdot,\cdot)$, a k-*Nearest-Neighbor (k-NN) query* identifies the k series in the dataset with the smallest distances to Q. The 1st Nearest Neighbor (1-NN) is the series in the dataset with the smallest distance to Q.

Although other types of queries exist, such as *r-range* queries, we do not address them here. We further simplify our presentation by focusing on *whole-matching queries*. As Echihabi et al. [39] explain, *subsequence-matching* queries can be trivially converted to whole matching queries.

Similarity search can be *exact*, when it produces answers that are always correct, or *approximate*, when there is no such strict guarantee. Echihabi et al. [39] identify different flavors of approximate similarity search algorithms, based on what types of "soft" guarantees (probabilistic or relative distance error bounds) they provide. In particular, a δ-ϵ-*approximate algorithm* guarantees that its distance results will have a relative error no more than ϵ with a probability of at least δ. We note that only a couple of approaches [6, 28] provide such guarantees. Furthermore, the accuracy of such approaches has never been tested in the range of dimensions and dataset sizes that we examine here.

Similarity Search Methods: Most data series similarity search techniques [19, 29, 42, 70, 77, 92, 125, 129, 135] use an index, which enables scalability. The index can offer quick approximate answers by traversing a single path of the index structure to visit the single most promising leaf, from where we select the *best-so-far (bsf)* answer: this is the candidate answer in the leaf that best matches (has the smallest distance to) the query. The bsf may, or may not be the final, exact answer: in order to verify, we need to either prune, or visit all the other leaves of the index. In this process, having a good first bsf (i.e., a bsf very close to the exact answer) leads to efficient pruning of the search space.

In the general case, approximate data series similarity search algorithms do not provide guarantees about the quality of their answers. In our work, we illustrate how we can efficiently and effectively provide such guarantees, with probability bounds.

We focus on index-based approaches that support both quick approximate, and slower but exact, similarity search results. In this work, we adapt the state-of-the-art ADS index [135], which finds high-quality approximate answers almost immediately, and subsequently updates these answers that converge fast to the final, exact answer. We also adapt the DSTree index [125], which has been shown to answer queries very fast [39], and demonstrate the applicability of our techniques to this index, as well.
Table 4.1 summarizes the symbols we use in this work.

4.3 PROGRESSIVE SIMILARITY SEARCH

We define progressive similarity search for k-NN queries.

Definition 4.1. *Given a* k*-NN query* Q*, a data series collection* $\mathbb{S}$*, and a time quantum* q*, a* ***progressive similarity-search algorithm*** *produces results* $R(t_1), R(t_2), ..., R(t_z)$ *at time points* $t_1, t_2, ..., t_z$*, where* $t_{i+1} - t_i \leqslant q$*, such that* $d(Q, R(t_{i+1})) \leqslant d(Q, R(t_i))$.

Table 4.1.: Table of symbols

Symbol	Description
S	data series
$S[i:j]$	series subsequence from i to j
ℓ	length of a data series
$\mathcal{S}$	data series collection (or dataset)
$n = \lvert\mathcal{S}\rvert$	number of series in $\mathcal{S}$
$R(t)$	progressive answer at time t
Q	query series
k-*NN*	k^{th} Nearest Neighbor of Q
$d_{Q,R}(t)$	distance between Q and $R(t)$
$d_{Q,knn}(t)$	distance between Q and k-*NN*
$\epsilon_Q(t)$	relative distance error of $R(t)$ from k-*NN*
$\hat{(\bullet)}$	estimate of $\bullet$
$I_Q(t)$	information at time t
$h_{Q,t}(x)$	probability density function of Q's distance from its k-NN, given information $I_Q(t)$
$H_{Q,t}(x)$	cumulative distribution function of Q's distance from its k-NN, given information $I_Q(t)$
$f_Q(x)$	probability density function of Q's distance from a random series in $\mathcal{S}$
$F_Q(x)$	cumulative distribution function of Q's distance from a random series in $\mathcal{S}$
$G_{Q,n}(x)$	cumulative distribution function of Q's distance from its k-NN
$\mathscr{W}$	set of witness series
$n_w = \lvert\mathscr{W}\rvert$	number of witnesses in $\lvert\mathscr{W}\rvert$

We borrow the quantum q parameter from Fekete and Primet [43]. It is a user-defined parameter that determines how frequently users require updates about the progress of their search. Although there is no guarantee that distance results will improve at every step of the progressive algorithm, the above definition states that a progressive distance will *never* deteriorate. This is an important difference of progressive similarity search compared to other progressive computation mechanisms, where results may fluctuate before they eventually converge, which may lead users to making wrong decisions based on intermediate results [24, 43, 54].

Clearly, progressive similarity search can be based on approximate similarity search algorithms – a progressive result is simply an approximate (best-so-far) answer that is updated over time. A progressive similarity search algorithm is also exact if the following condition holds:

$$\lim_{t\to\infty} d(Q, R(t)) = d(Q, knn(Q)) \tag{4.1}$$

where $knn(Q)$ represents the k-NN of the query series Q.

According to the above condition, the progressive algorithm will always find an exact answer. However, there are generally no strong guarantees about how long this can take. Next, we present some preliminary experiments which show that a progressive similarity search algorithm will find good answers very fast, e.g., within interactive times, and will also converge to the exact answer without long delays. However, waiting for the algorithm to confirm that there is no better answer and finish execution will take much longer burdening users with long wasted times.

4.4 PRELIMINARY OBSERVATIONS

We examine how early we can provide approximate answers, and how good these answers are compared to the exact answers. To this end, we conducted *similarity search* experiments in three real datasets with the state-of-the-art ADS index [135], which can quickly provide good initial approximate results and can potentially support progressive similarity search within interactive time thresholds.

Scope. We examine approximate and exact 1-NN whole-matching† similarity search queries [39]. (We cover k-NN queries later. We leave r-range queries and subsequence matching for future work.)

Environment. We ran all experiments on a Dell T630 Rack Server with two Intel Xeon E5-2643 v4 3.4Ghz CPUs, 512GB of RAM, and 3.6TB (2 x 1.8TB) HDD in RAID0. The search algorithm is a single-core implementation.

Datasets. We tested diverse real datasets that have also been used in previous studies [39, 135]. These datasets have the same overall size of *100GB*, but contain different number of series with different length (i.e., number of points) (Table 4.4). The IRIS seismic dataset‡ consists of seismic instrument recordings from several stations worldwide and contains 100 million series of length 256 points. The neuroscience dataset, SALD§, consists of MRI data and contains 200 million series of length 128 points. The image processing dataset, deep1B¶, consists of vectors extracted from the last layers of a convolutional neural network and contains 267 million series of size 96 points.

Table 4.2.: Experimental datasets

Name	Description	Cardinality	TS Length
seismic	seismic records	100M	256
SALD	MRI data	200M	128
deep1B	image descriptors	267M	96

Queries. All our query workloads include 100 query series. We generated the query datasets by extracting random data series from the raw data. For the

† Query and all series in the dataset have the same length.
‡ http://ds.iris.edu/data/access/
§ http://fcon_1000.projects.nitrc.org/indi/retro/sald.html
¶ http://sites.skoltech.ru/compvision/noimi/

deep1B dataset, we used a real query workload that came with the original dataset.

Measures. For each similarity query, we recorded its overall completion time, the time for each progressive answer, as well as the time passed until the algorithm finds the exact answer to the query, i.e., the 1-NN. For each approximate and exact answer, we also recorded its Euclidean distance from the query.

Results. Table 4.3 summarizes our results. For each dataset, we present the average, minimum, and maximum time (in seconds) for the 1-NN query answering algorithm to first encounter the 1-NN answer (marked as *1-NN Time* in Table 4.3), and the corresponding times for the same algorithm to finish execution (marked as *Total Time*). We observe that that total time waiting for a single query to finish can be long, e.g., up to three minutes, which is beyond acceptable thresholds for interactive data-analysis tasks [43]. Moreover, these delays are orders of magnitude longer than the actual time needed to find the best answer (i.e., first encounter of 1-NN). This means that for most queries, the greatest cost is not locating the 1-NN, but rather confirming that there is no better answer: this is why the query answering algorithm finishes execution long after having retrieved the 1-NN value. This finding is consistent with results by Ciaccia and Patella [28], who report that most of the time spent in an exact NN search is "wasted time, during which no improvement is obtained."

Table 4.3.: Summary of experimental results

	1-NN Time (sec)			Total Time (sec)		
Dataset	Avg	Min	Max	Avg	Min	Max
seismic	8.5	0.017	48.5	92	21.3	111
SALD	0.4	0.003	5.2	49	0.24	183
deep1B	0.2	0.001	2.8	76	0.05	189

The time needed to locate the 1-NN was especially fast for the SALD and deep1B datasets, where average times were below 1 sec. However, times varied greatly in the seismic dataset, ranging from a few milliseconds to 48.5 sec. For 28% of the queries, the delay was greater than 10 sec, which is considered as a limit for keeping a user's attention focused on a dialog [43]. We expect that such delays will further increase in larger datasets, and for k-NN exact search. In these cases, providing early approximate answers will also be crucial.

Figure 4.1 presents the progressive results for four example queries in the seismic data. For these queries, the time to locate the 1-NN is relatively long (> 20 sec), while progressive answers (intermediate points in each curve) appear with various frequencies and trends. For example, for the yellow and green queries, results converge quickly (in the order of hundreds of milliseconds), and then only slightly improve. For other queries, such as the purple line, convergence is more progressive. We show the evolution of the progressive error as a percentage of the exact 1-NN distance. We observe that the algorithm provides approximate answers within a few milliseconds, and those answers gradually converge to the exact answer, which is the distance of the query from the 1-NN. The error of the first approximate answer is on average 16% (see gray line – average trend).

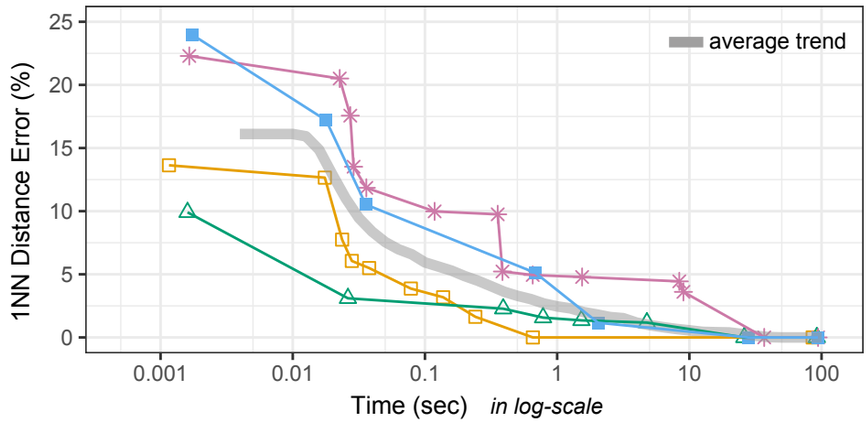


Figure 4.1.: Progression of the 1-NN distance error (Euclidean distance) for 4 example queries (seismic dataset), using the ADS index [135]. The points in each curve represent approximate (intermediate points) or exact answers (last point) given by the algorithm. The lines end when the similarity search ends. The thick grey line represents the average trend over a random sample of 100 queries.

Interestingly, the 1-NN is often found in less than 1 sec (e.g., see yellow line), but it takes the search algorithm much longer to verify that there is no better answer and terminate.

Overall, our results indicate that (i) supporting interactive similarity search in large time series datasets is feasible, and (ii) providing early progressive answers to users could drastically reduce waiting times. The challenge is how to help users assess the quality of such progressive answers and decide whether to trust these answers, or wait for a better one.

4.5 PROGRESSIVE ESTIMATES

In the absence of information, users may not be able to trust a progressive result, no matter how close it is to the exact answer. In our work, we investigate exactly this problem: how to provide progressive estimates and quality guarantees about how close a progressive answer is to the exact answer and help users assess the quality of such progressive results.

Given a progressive answer $R(t)$ to a k-NN query at time t, we are interested in knowing how far from the k-NN this answer is. For simplicity, we will denote the exact k-NN distance to the query as $d_{Q,knn}$ and the distance between $R(t)$ and the query as $d_{Q,R}(t)$. Then, the relative distance error is $\epsilon_Q(t)$, where $d_{Q,R}(t) = d_{Q,knn}(1 + \epsilon_Q(t))$. Given that this error is not known, our goal is to find an estimate $\hat{\epsilon}_Q(t)$. However, finding an estimate for the relative error is not any simpler than finding an estimate $\hat{d}_{Q,knn}(t)$ of the actual k-NN distance. We will concentrate on this latter quantity for most of our analysis. Though, since the distance $d_{Q,R}(t)$ is known, deriving the distance error estimate $\hat{\epsilon}_Q(t)$ from the k-NN distance estimate $\hat{d}_{Q,knn}(t)$ is straightforward:

$$\hat{\epsilon}_Q(t) = \frac{d_{Q,R}(t)}{\hat{d}_{Q,knn}(t)} - 1 \tag{4.2}$$

We return to this measure later (see Section 4.8) when we discuss about how to communicate results to users.

We represent progressive similarity-search estimates as probability distribution functions.

Definition 4.2. *Given a* k*-NN query* Q*, a data series collection* $\mathbb{S}$*, and a progressive similarity-search algorithm, a* ***progressive*** k***-NN distance estimate*** $\hat{d}_{Q,knn}(t)$ *of the actual* k*-NN distance at time* t *is represented by a probability density function:*

$$h_{Q,t}(x) = Pr\{d_{Q,knn} = x | I_Q(t)\} \tag{4.3}$$

which expresses the conditional probability that $d_{Q,knn}$ *is equal to* x*, given information* $I_Q(t)$.

We expect that progressive estimates will converge to the distance $d_{Q,knn}$ of the exact answer (i.e., $\hat{e}_Q(t)$ will converge to zero). Evidently, the quality of an estimate at time t largely depends on the information $I_Q(t)$ that is available at this moment. In Section 4.6, we investigate different types of information, $I_Q(t)$, that we can use in order to produce a probabilistic estimate.

Given the probability density function in Equation 4.3, we can derive a point estimate that gives the *expected* k-NN distance, or an interval estimate in the form of a *prediction interval* (PI). Like a confidence interval, a prediction interval is associated with a confidence level. Given a confidence level $1-\theta$, we expect that at least $(1-\theta) \times 100\%$ of the prediction intervals that we construct will include the true k-NN distance. We should note that although a confidence level can be informally assumed as a probability (i.e., what is the likelihood that the interval contains the true k-NN distance?), this assumption may or may not be strictly correct. Our experiments evaluate the frequentist behavior of such intervals. Our goal is to guarantee that for out of 100 queries that users, at most $\theta \times 100$ of the intervals that we construct will fail to include the true k-NN distance.

To construct a prediction interval with confidence level $1-\theta$ over a density distribution function $h_{Q,t}(\cdot)$, we derive the cumulative distribution function:

$$H_{Q,t}(x) = Pr\{d_{Q,knn} \leqslant x | I_Q(t)\} \tag{4.4}$$

From this, we take the $\theta/2$ and $(1-\theta/2)$ quantiles that define the limits of the interval.

The Problem of Sequential Tests: Presenting multiple progressive estimates to users raises some concerns. Recent work on progressive visualization [81] discusses the problem of confirmation bias, where an analyst may use incomplete results to confirm a "preferred hypothesis." For example, an analyst may choose to stop the query execution as soon as the prediction interval of the 1-NN distance excludes a low threshold value. This is a well-studied problem in sequential analysis [122]. It relates to the multiple-comparisons problem [131] and is known to increase the probability of a Type I error (false positive).

The easiest way to deal with this problem is to fix the maximum number of progressive estimates that users can look at and use a generic method, such as the Bonferroni correction, to adjust the confidence level of prediction intervals. If the maximum number of updates is $z = 5$, then confident level need to be adjusted to $(1-\theta/5)$. In this case, in order to guarantee a 95% confidence level over

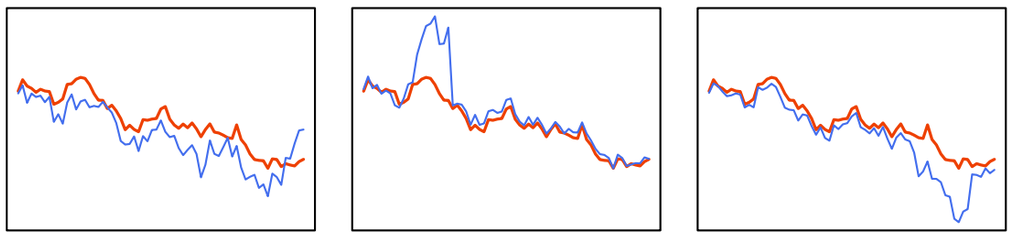


Figure 4.2.: The three series in blue represent alternative answers to the same query (in orange). All have the same distance to the query, but it is distributed differently along their length. For the first query, the distance is distributed in a rather uniform manner. In the two other cases, differences are concentrated around a smaller range.

all sequential tests, we need to set the level of all prediction intervals to 99%. Unfortunately, this approach may result in conservative (i.e., too wide) prediction intervals. Alternative methods, such as the Pocock boundary [95], require less radical adjustments and thus produce narrower intervals. However, they focus on summary statistics (e.g., the mean) over progressively growing samples and cannot generalize to our problem.

In this work, we evaluate how multiple sequential tests affect the accuracy of our methods. We defer a detailed analysis and solution to this problem to future work.

Estimates for Subsequences: An estimate of the distance error (see Equation 4.2) may be insufficient by itself to evaluate a progressive answer as it provides no information about how this error is distributed along the series length. The error may be uniform across all the points of the series, or concentrate on specific parts of the series (e.g., in the middle section) of great interest for the user. Likewise, an estimate of the k-NN distance provides incomplete information about the probable shapes of a k-NN (see Figure 4.2).

We can address this limitation by providing distance estimates about individual subsequences of the k-NN. For example, if the ending of a query series has great interest for a task, estimating the local distance of the k-NN around this part of the series can help the user decide whether waiting for a better answer, e.g., better than the third series in Figure 4.2, is worth. As for the full series, our goal again is to construct a probability density function $h_{Q[i,j],t}(\cdot)$ that estimates the distance between $Q[i:j]$ and $knn(Q)[i:j]$.

4.6 PREDICTION METHODS

We investigate a range of statistical models and probabilistic estimation methods, where each considers a different amount of information. Some methods consider constant information ($I_Q(t) = I_Q$). Such methods are especially useful for providing an initial estimate before a similarity search starts. We distinguish between *query-sensitive* methods, which take into account the query series Q, and *query-agnostic* methods, which provide a common estimate irrespective of Q ($I_Q = I$). Other methods use information that is progressively updated during the execution of a similarity search, and their predictions improve over time.

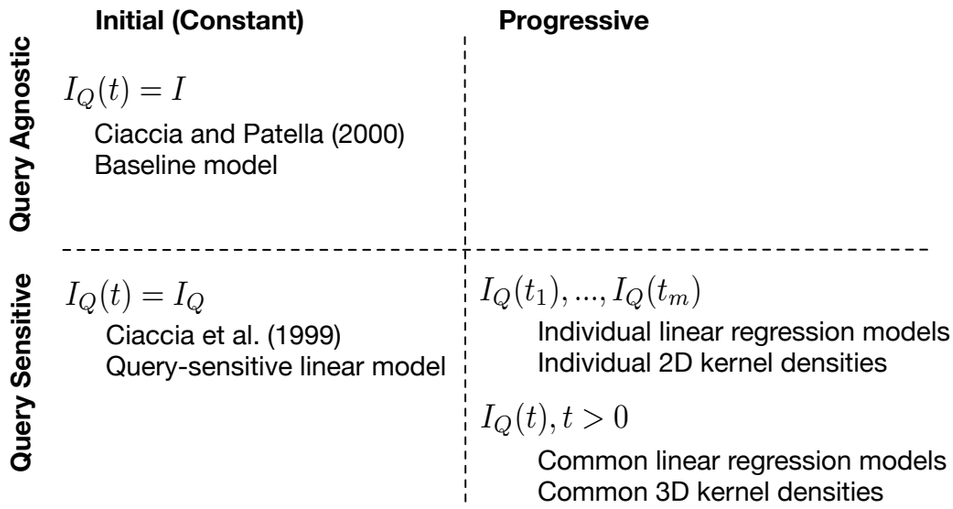


Figure 4.3.: k-NN prediction methods that we study

Figure 4.3 presents a summary of all the methods that we study. To simplify our analysis, we focus on 1-NN similarity search. Our analysis naturally extends to k-NN search; a detailed study of this case is part of our future work.

4.6.1 *Baseline Approaches*

We first describe a probabilistic approach inspired by Ciaccia et al. [27–29]. The approach was originally used to estimate the cost of similarity search in high-dimensional spaces [27, 29]. It was further used to derive stopping conditions for approximate similarity search with probabilistic quality guarantees [28]. Despite our different goals, adapting the approach to query-sensitive [27], or query-agnostic [28] k-NN distance estimation in the context of progressive data-series similarity search is straightforward.

Based on Ciaccia et al. [29], a dataset S (a data series collection in our case) can be seen as a random sample drawn from a large (or infinite) population $\mathscr{U}$ of points in a high-dimensional space. Being a random sample, a large dataset is expected to be representative of the original population.

Given a query Q, let $f_Q(x)$ be the probability density function that gives the relative likelihood that Q's distance from a random series drawn from $\mathscr{U}$ is equal to x. Likewise, let $F_Q(\cdot)$ be its cumulative probability function. Based on $F_Q(\cdot)$, Ciaccia et al. [29] show how to derive the cumulative probability function $G_{Q,n}(\cdot)$ for Q's k-NN distances in a dataset of size $n = |S|$. For 1-NN similarity search, we have:

$$G_{Q,n}(x) = 1 - (1 - F_Q(x))^n \tag{4.5}$$

If we now assign this function to $H_{Q,t}(\cdot)$ (see Equation 4.4), we have a way to construct estimates for 1-NN distances. Such estimates remain static, since $G_{Q,n}(\cdot)$ is not updated during the execution of a progressive similarity search. However, they are still valuable as they provide a reference that can help analysts evaluate whether a progressive answer is close enough to a probable range for the exact answers.

Unfortunately, $f_Q(\cdot)$, and thus $F_Q(\cdot)$, are not known. Therefore, the challenge is how to approximate them from a given dataset. We discuss two different approximation methods:

1. Query-Agnostic Approximation. Ciaccia et al. [29] show that for multi-dimensional spaces, a large enough sample from the overall distribution $f(\cdot)$ of pairwise distances in a dataset provides a reasonable approximation for $f_Q(\cdot)$. Ciaccia and Patella [28] use this approximation to evaluate their probabilistic stopping-conditions approach by taking sampling sizes between 10% and 1% (for larger datasets).

2. Query-Sensitive Approximation. The first method does not take the query into account. Ciaccia et al. [27] introduce an alternative query-sensitive approximation approach that is based on a training set of predefined reference queries, called *witnesses*. Witnesses can be randomly drawn from the dataset, or selected with the GNAT algorithm [15]. The GNAT algorithm identifies the n_w points that best cover a multidimensional (metric) space based on an initial random sample of $3n_w$ points.

Based on the rationale that close objects have similar distance distributions, Ciaccia et al. [27] approximate $f_Q(\cdot)$ by using the distance distribution of the nearest witness to Q or the weighted average of the distance distributions of all the witnesses. We concentrate on this second approach as it is a generalization of the first.

Let $W_1, W_2, ..., W_{n_w}$ be the available set of witnesses and let $d(Q, W_j)$ be the distance between the j^{th} witness and Q. Based on this, the normalized weight $a_{Q,j}$ of the witness is:

$$a_{Q,j} = \frac{d(Q, W_j)^{-exp}}{\sum_{i=1}^{n_w} d(Q, W_i)^{-exp}} \tag{4.6}$$

Based on these weights, $F_Q(\cdot)$ is approximated as follows:

$$F_Q(x) \simeq \sum_{i=1}^{n_w} F_{W_j}(x) \cdot a_{Q,j} \tag{4.7}$$

where the cumulative probability function $F_{W_j}(\cdot)$ can be directly derived from the distribution of distances between W_j and the other points in the dataset, with or without sampling. Ciaccia et al. [27] have experimented with a range of exponent values to optimize the weighting function (see Equation 4.6) and have also introduced an *adaptive* variant that determines *exp* as a function of the distances between the witnesses and the query.

Unfortunately, the above methods have the following three major limitations:

(1) Since their 1-NN distance estimates are static, they are less appropriate for progressive similarity search.

(2) A good approximation of $F_Q(\cdot)$ does not necessarily lead to a good approximation of $G_{Q,n}(\cdot)$. This is especially true for large datasets, as the exponent term n in Equation 4.5 will inflate even tiny approximation errors. Note that $G_{Q,n}(\cdot)$ can be thought of as a scaled version of $F_Q(\cdot)$ that zooms in on the range of the lowest distance values. If this narrow range of distances is not accurately approximated, the approximation of $G_{Q,n}(\cdot)$ will also fail.

Table 4.4.: Experimental datasets

name	description	number of series	series length
synthetic	random walks	100M	256
seismic [104]	seismic records	100M	256
SALD [119]	MRI data	200M	128
deep1B [121]	image descriptors	267M	96

(3) Both methods require the pre-calculation of a large number of distances. In large datasets, such pre-calculations can become prohibitively expensive. Sampling the dataset can significantly reduce this cost. However, since the approximation of $G_{Q,n}(\cdot)$ is sensitive to errors in large datasets (see above), a rather large number of samples is required in order to capture the frequency of the very small distances. Thus, even with sampling, the computational cost of distance calculations remains a major problem.

The solutions that we present next address the above three problems. For illustration purposes only, we will apply our prediction models to the four datasets shown in Table 4.4. Note that we previously used the three real datasets in our preliminary experiments (see Section 4.4).

4.6.2 *Providing Initial Estimates*

We first concentrate on how to approximate the distribution function $h_{Q,o}(x)$ (see Equation 4.3), thus provide estimates before similarity search starts.

As Ciaccia et al. [27], we rely on witnesses, which are "training" query series that are randomly sampled from a dataset. Unlike their approach, however, we do not use the distribution of raw pairwise distances $F_Q(\cdot)$. Instead, for each witness, we execute 1-NN similarity queries with a fast state-of-the-art algorithm, such as ADS [135], or DSTree [125]. This allows us to derive directly the distribution of 1-NN distances and predict the 1-NN distance of new queries.

This approach has two main benefits. First, we use the tree structure of the above algorithms to prune the search space and reduce pre-calculation costs. Rather than calculating a large number of pairwise distances, we focus on the distribution of 1-NN distances with fewer distance calculations. Second, we achieve reliable and high-quality approximation with a relatively small number of training queries ($\approx 100 - 200$) independently of the dataset size (we report and discuss these results in Section 4.7).

Query-Agnostic Model (Baseline). Let $\mathscr{W} = \{W_j | j = 1..n_w\}$ be a set of $n_w = |\mathscr{W}|$ witnesses randomly drawn from the dataset. We execute a 1-NN similarity search for each witness and build their 1-NN distance distribution. We then use this distribution to approximate the overall (query-independent) distribution of 1-NN distances $g_n(\cdot)$ and its cumulative probability function $G_n(\cdot)$. This method is comparable to Ciaccia et al. [29] query-agnostic approximation method and serves as a baseline.

Query-Sensitive Model. Intuitively, the smaller the distance between the query and a witness, the better the 1-NN of this witness predicts the 1-NN of the

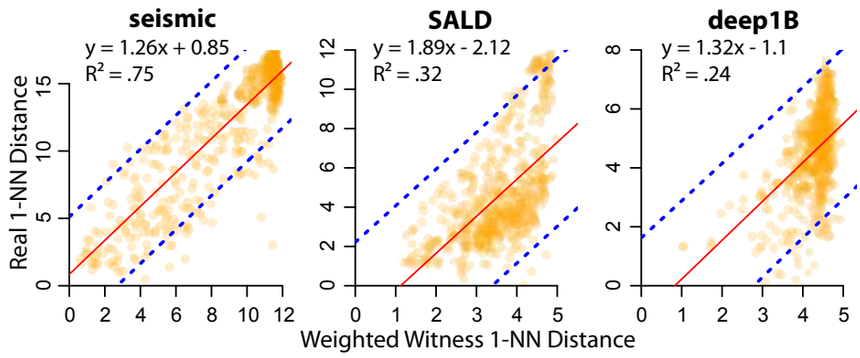


Figure 4.4.: Linear models (red solid lines) predicting the real 1-NN distance $d_{Q,1nn}$ based on the weighted witness 1-NN distance dw_Q for $exp = 5$. All models have been based on 200 random witnesses and 500 queries. The blue (dashed) lines show the range of their 95% prediction intervals.

query. We capture this relationship through a random variable that expresses the weighted sum of the 1-NN distance of all n_w witnesses:

$$dw_Q = \sum_{j=1}^{n_w} (a_{Q,j} \cdot d_{W_i,1nn}) \tag{4.8}$$

where the weights $a_{Q,j}$ are derived from Equation 4.6. We use this variable as predictor of the query's real 1-NN distance $d_{Q,1nn}$. We base our analysis on the following linear model:

$$d_{Q,1nn} = \beta \cdot dw_Q + c \tag{4.9}$$

Figure 4.4 shows the parameters of this model for the three real datasets of Table 4.4 with 100 witnesses ($n_w = 100$) and 500 queries. We conduct linear regressions by assuming that the distribution of residuals is normal (Gaussian) and has equal variance. Our tests have shown optimal results for exponents (see Equation 4.6) that are close to 5. For simplicity, we use $exp = 5$ for all our analyses. Additional tests have shown that the fit of the model becomes consistently worse if witnesses are selected with the GNAT algorithm [15, 27] (we omit these results for brevity). Therefore, we only examine random witnesses here.

Since the model parameters (β and c) and the variance are dataset specific, they have to be trained for each individual dataset. To train the model, we use an additional random sample of n_r *training queries* that is different from the sample of witnesses. Based on the distance of each training query Q_i from the witnesses, we calculate dw_{Q_i} (see Equation 4.8). We also run similarity search to find its 1-NN distance $d_{Q_i,1nn}$. We then use all pairs $(dw_{Q_i}, d_{Q_i,1nn})$, where $i = 1..n_r$, to build the model and predict the 1-NN distance of new queries. The approach allows us to construct both point estimates (see Equation 4.8) and prediction intervals (see Figure 4.4) that provide probabilistic guarantees about the range of an 1-NN distance. We should note that the time required to train a model is fully dominated by the time it takes to find the 1-NN distance of the n_w witnesses and the n_r training queries.

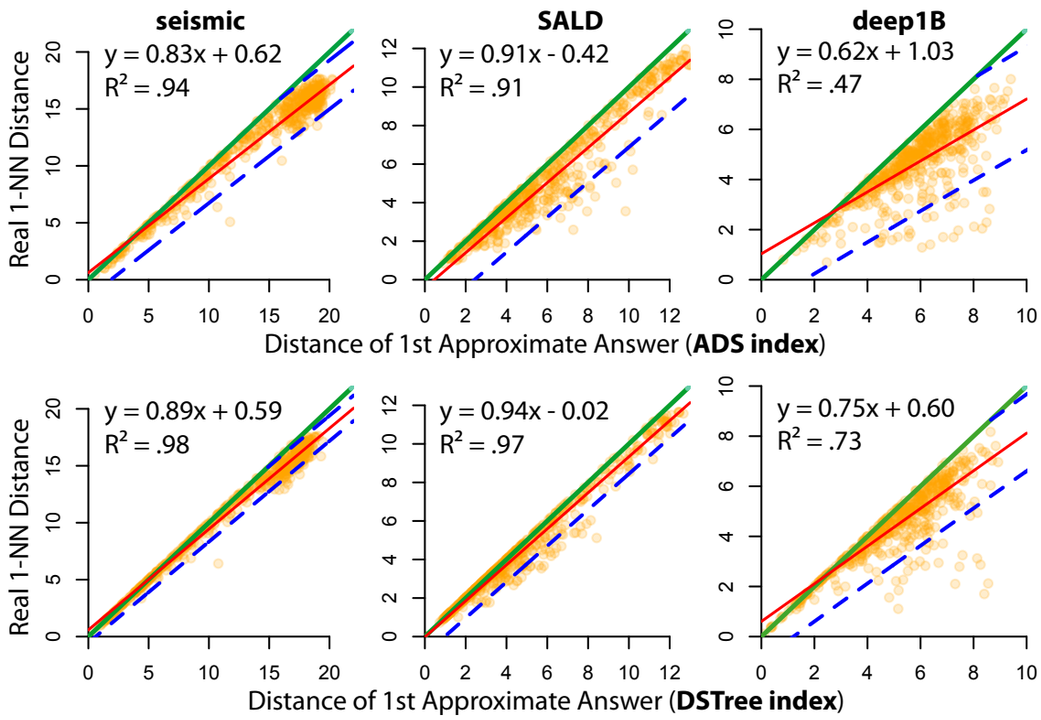


Figure 4.5.: Linear models (red/dark solid lines) predicting the real 1-NN distance $d_{Q,1nn}$ based on distance of *first* approximate answer of ADS [135] and DSTree [125]. All models trained with 200 queries. The 500 (orange) points in each plot are queries out of the training set. The green (solid) lines ($y = x$) are hard upper bounds, determined by the approximate answer. The blue (dashed) lines show the range of 95% prediction intervals for which the model provides tighter bounds than the hard ones.

4.6.3 *Providing Progressive Estimates*

So far, we have focused on initial 1-NN distance estimates. Those do not consider any information about the partial results of a progressive similarity-search algorithm. Now, given Definition 4.2, the distance of a progressive result $d(Q, R(t_i))$ will never deteriorate and thus can act as an upper bound for the real 1-NN distance. The challenge is how to provide probabilistic bounds that are tighter than the obvious hard bound $d_{Q,1nn} \in [0, d(Q, R(t_i))]$.

Our approach relies on the observation that the approximate answers of index-based algorithms are generally close to the exact answers. Figure 4.5 illustrates the relationship between the true 1-NN distance and the distance of the first progressive (approximate) answer returned by the ADS index [135] and the DSTree index [125], which follows a completely different design from ADS. We observe a strong linear relationship for both algorithms, especially for the DSTree index. We can express it with a linear model and then derive probabilistic bounds in the form of prediction intervals. As shown in Figure 4.5, the approach is particularly useful for constructing lower bounds. Those are clearly greater than zero and provide valuable information about the extent to which a progressive answer can be improved or not.

Since progressive answers improve over time and tend to converge to the 1-NN distance, we could take such information into account to provide tighter estimates as similarity search progresses. To this end, we examine different pro-

gressive prediction methods. They are all based on the use of a dataset of n_r training queries that includes information about all progressive answers of a similarity search algorithm to each query, including a timestamp and its distance.

Individual Linear Models. Let $t_1, t_2, ..., t_m$ be specific moments of interest (e.g., 100ms, 1s, 3s, and 5s). Given t_i, we can build a time-specific linear model:

$$d_{Q,1nn} = \beta_{t_i} \cdot d_{Q,t_i} + c_{t_i} \tag{4.10}$$

where d_{Q,t_i} is the distance of Q from the best approximate answer at time t_i. The advantage of this method is the fact that it produces models that are well adapted to each individual time point of interest. On the downside, it requires the pre-specification of a discrete set of time points, which may not be desirable for certain application scenarios. However, building such models from an existing training dataset is inexpensive, so reconfiguring the moments of interest at use time is not a problem.

The above model can be enhanced with an additional term $\beta \cdot dw_Q$ (see Equation 4.8) that takes witness information into account. However, this term results in no measurable improvements in practice, so we do not discuss it further.

Common Linear Model. Alternatively, we use a common multivariate linear model that has the following form:

$$d_{Q,1nn} = \beta \cdot d_{Q,t} + \gamma \cdot \log(t) + \delta \cdot (d_{Q,t} \times \log(t)) + c \tag{4.11}$$

where $d_{Q,t}$ is again the distance of Q from the best approximate answer at time t. The second term captures the main effect of time, while the third term captures its interaction with the best-so-far distance. We hypothesize that the longer it takes to find an answer with a certain distance value, the larger the 1-NN distance is expected to be.

This model has two advantages. First, it has at most four parameters (β, γ, δ, and c) compared to a total of $2 \times m$ parameters (β_{t_i} and c_{t_i}, where $i = 1 \ldots m$) required by the previous method. Second, we can use it to predict the 1-NN distance at any point in time. Nevertheless, this comes with a cost in terms of precision (see Section 4.7).

So far, we have based our analysis on time. Nevertheless, time (expressed in seconds) is not a reliable measure for training and applying models in practice. The reason is that time largely depends on the available computation power, which can vary greatly across different hardware settings. Our solution is to use alternative measures that capture the progress of computation without being affected by hardware and computation loads. One can use either the number of series comparisons (i.e., the number of distance calculations), or the number of visited leaves. Both measures can be easily extracted from the ADS index [135], the DSTree [125], or other tree-based similarity-search algorithms. Given that we are interested in logarithmic time scales, all our analyses are based on the number of visited leaves (*Leaves Visited*). We should note that for a given number of visited leaves, we only consider a single approximate answer, which is the best-so-far answer after traversing the last leaf.

When we substitute t by the number of visited leaves, the second term in Equation 4.11 has no influence (i.e., $\gamma \simeq 0$), so we omit it. Figure 4.6 shows how

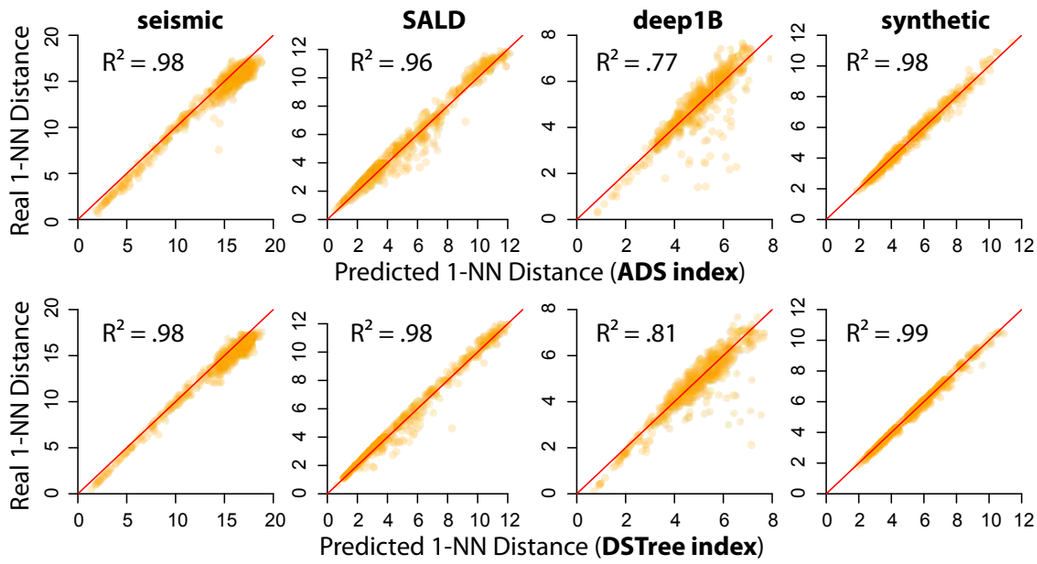


Figure 4.6.: Fit of the common temporal model (Equation 4.11). We compare the real to the predicted 1-NN distance. All models were trained with 200 queries and tested on a different sample of 500 queries.

the observed 1-NN distance values fit to the ones predicted by the model in this case. The high R^2 values demonstrate that the model can explain a large portion of the overall variance.

Kernel Density Estimation. A main strength of the previous methods is their simplicity. However, linearity is a strong assumption that may not always hold. Other assumption violations, such as heteroscedasticity, can limit the accuracy of linear regression models. As alternatives, we investigate non-parametric methods that approximate the density distribution function $h_{Q,t}(\cdot)$ based on multivariate kernel density estimation [37, 123].

As for linear models, we rely on the functional relationship between progressive and final answers. We represent this relationship as a 3-dimensional density probability function $k_Q(x, y, t)$ that expresses the probability that the 1-NN distance from Q is x, given that Q's distance from the progressive answer at time t is y. From this function, we derive $h_{Q,t}(x)$ by setting $y = d_{Q,t}$, where $d_{Q,t}$ is Q's distance from the progressive answer at time t. We use again the number of visited leaves to measure time on a logarithmic scale.

We examine two approaches for constructing the function $k_Q(\cdot, \cdot, \cdot)$. As for individual linear models, we specify discrete moments of interest t_i and then use bivariate kernel density estimation [124] to construct an individual density probability function $k_Q(\cdot, \cdot, t_i)$. Alternatively, we construct a common density probability function by using 3-variate kernel density estimation. The accuracy of kernel density estimation highly depends on the method that one uses to smooth out the contribution of points (2D or 3D) in a training sample. We use Gaussian kernels, but for each estimation approach, we select bandwidths with a different technique. We found that the plug-in selector of Wand and Jones [124] works best for our bivariate approach, while the smoothed cross-validation technique [37] works best for our 3-variate approach.

4.7 EXPERIMENTAL BENCHMARK EVALUATION

4.7.1 *Setup*

Environment. All experiments were run on a Dell T630 rack server with two Intel Xeon E5-2643 v4 3.4Ghz CPUs, 512GB of RAM, and 3.6TB (2 x 1.8TB) HDD in RAID0.

Implementation. Our estimation methods were implemented in R. We use the R's *lm* function to carry our linear regression and the *ks* library [38] for multivariate kernel density estimation. We use a grid of 200×200 points to approximate a 2D density distribution and a grid of $60 \times 180 \times 180$ points to approximate a 3D density distribution.

Datasets. We used one synthetic and three diverse real datasets from past studies [39, 135]. We used the real datasets in our preliminary experiments as well (see Section 4.4 for a detailed description). The synthetic data series were generated as random walks (i.e., cumulative sums) of steps that follow a Gaussian distribution (0,1). This type of data has been extensively used in the past [20, 42, 136] and is claimed to model the distribution of stock market prices [42].

Measures. We use the following measures to assess the estimation quality of each method and compare their results:

1. Coverage Probability: It measures the proportion of the time that the prediction intervals contain the true 1-NN distance. If the confidence level of the intervals is $1-\theta$, the coverage probability should be close to $1-\theta$. A low coverage probability is problematic. In contrast, a coverage probability that is higher than its nominal value (i.e., its confidence level) is acceptable but can hurt the intervals' precision. In particular, a very wide interval that always includes the true 1-NN distance (100% coverage) can be useless.

2. Prediction Intervals Width: It measures the size of the prediction intervals that each method constructs. Tighter prediction intervals are better. However, this is only true if the coverage probability of the tighter intervals is close to or higher than their nominal confidence level.

3. Root-Mean-Squared Error (RMSE): It evaluates the quality of point (rather than interval) estimates by measuring the standard deviation of the true 1-NN distance values from their expected (mean) values.

Validation Methodology. To evaluate the different methods, we use a Monte Carlo cross-validation approach that consists of the following steps. For each dataset, we randomly draw two disjoint sets of data series $\mathscr{W}_{\text{pool}}$ and $\mathscr{T}_{\text{pool}}$ and pre-calculate all distances between the series of these two sets. The first set serves as a pool for drawing random sets of witnesses (if applicable), while the second set serves as a pool for randomly drawing training (if applicable) and testing queries. At each iteration, we draw n_w witnesses (n_w = 50, 100, 200, or 500) and/or n_r training queries (n_r = 50, 100, or 200) from $\mathscr{W}_{\text{pool}}$ and $\mathscr{T}_{\text{pool}}$, respectively. We also draw n_t = 200 testing queries from $\mathscr{T}_{\text{pool}}$ such that they do not overlap with the training queries. We train and test the evaluated methods and then repeat the same procedure N = 100 times, where each time, we draw a

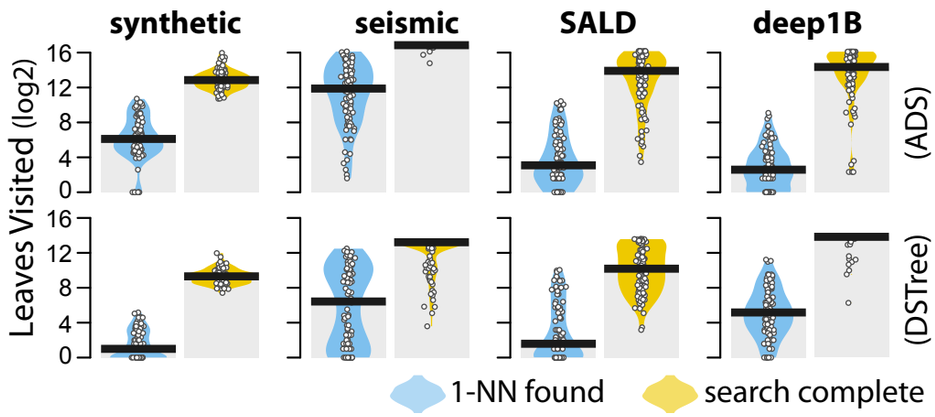


Figure 4.7.: Distribution (over 100 queries) of the number of leaves visited (in $\log_2$ scale) until finding the 1-NN (light blue) and competing the search (yellow). The thick black lines represent medians.

new set of witnesses, training, and testing queries. Thus, for each method and condition, our results are based on a total of $N \times n_t = 100 \times 200 = 20000$ test measurements.

For all progressive methods, we test the accuracy of their estimates after the similarity search algorithm has visited 1 (2^0), 4 (2^2), 16 (2^4), 64 (2^6), 256 (2^8), and 1024 (2^{10}) leaves. Figure 4.7 shows the distributions of visited leaves for 100 random queries for all four datasets.

4.7.2 *Results*

Previous State-of-the-Art Methods. We first evaluate the query-agnostic and query-sensitive approximation methods of Ciaccia et al. [27, 28]. To assess how the two methods scale with and without sampling, we examine smaller datasets with cardinalities of up to 1M data series (up to 100K for the query-agnostic approach). Those datasets are derived from the initial datasets presented in Table 4.4 through random sampling. Such smaller dataset sizes allow us to derive the full distribution of distances without sampling errors, while they are sufficient for demonstrating the behavior of the approximation methods as datasets grow.

Figure 4.8 presents the coverage probabilities of the methods. The behavior of query-agnostic approximation is especially poor. Even when the full dataset is used to derive the distribution of distances, the coverage tends to drop below 10% for larger datasets (95% confidence level). This demonstrates that the approximated distribution of 1-NN distances completely fails to capture the real one. Figure 4.9 compares the real to the approximated distributions for datasets of 100K series. We observe that the method largely underestimates the 1-NN distances for all four datasets.

Results for the query-sensitive method are better, but coverage is still below acceptable levels. Figure 4.8 presents results for $n_w = 500$ witnesses. Note that our further tests have shown that larger numbers of witnesses result in no or very little improvement, while Ciaccia et al. [27] had tested a maximum of 200 witnesses. To weight distances (see Equation 4.6), we tested the exponent values $exp = 3$, 5, and 12, where the first two were also tested by Ciaccia et al. [27],

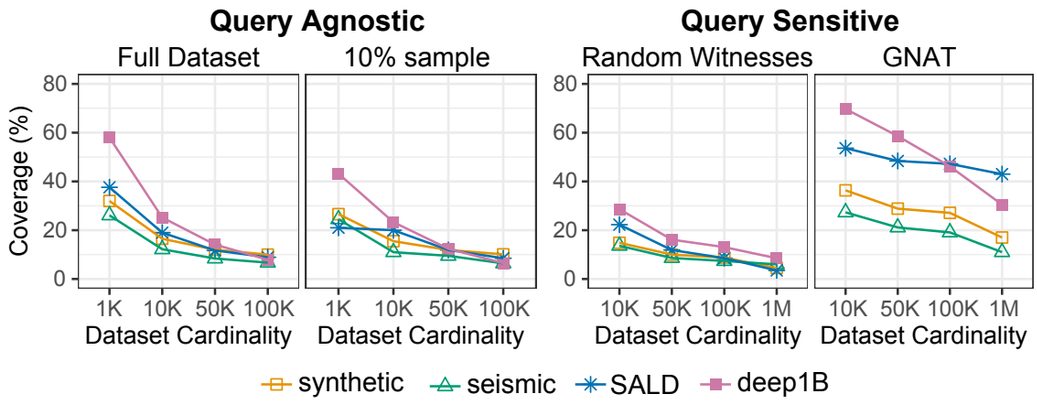


Figure 4.8.: Coverage probabilities of the query-agnostic (left) and query-sensitive (right) approximation methods of Ciaccia et al. [27, 28] for a 95% confidence level. We use 500 witnesses for the query-sensitive methods. We show best-case results, where the best *exp* is chosen (3, 5, 12, or adaptive).

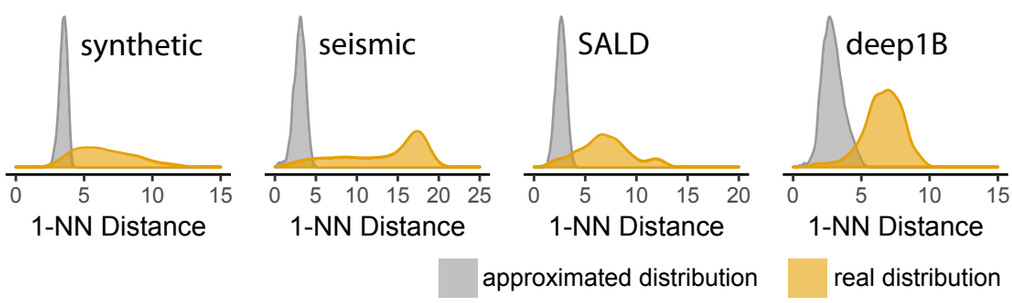


Figure 4.9.: Real distribution of 1-NN distances and its query-agnostic approximation of Ciaccia and Patella [28]. All datasets contain 100K series.

while we found that the third one gave better results for some datasets. We also tested the authors' adaptive technique. Figure 4.8 presents the best result for each dataset, most often given by the adaptive technique.

We observe that the GNAT method results in clearly higher coverage probabilities than the fully random method. This result is somehow surprising because Ciaccia et al. [27] report that the GNAT method tends to become less accurate than the random method in high-dimensional spaces with more than eight dimensions. Even so, the coverage probability of the GNAT method is largely below its nominal level. In all cases, it tends to become less than 50% as the cardinality of the datasets increases beyond 100K, while in some cases, it drops below 20% (synthetic and seismic).

For much larger datasets (e.g., 100M data series), we expect the accuracy of the above methods to become even worse. We conclude that they are not appropriate for our purposes, thus we do not study them further.

Our Estimation Methods. We simplify our analysis by focusing on the ADS index (we examine the DSTree index in the following subsection). We first analyze the coverage probability of our methods for confidence levels 95% ($\theta = .05$) and 99% ($\theta = .01$). Figure 4.10 presents our results. The coverage of the *Baseline* method reaches its nominal confidence level for $n_w = 200$ to 500 witnesses. In contrast, the *Query-Sensitive* method demonstrates a very good coverage even for small numbers of witnesses ($n_w = 50$) and training queries ($n_r = 50$). How-

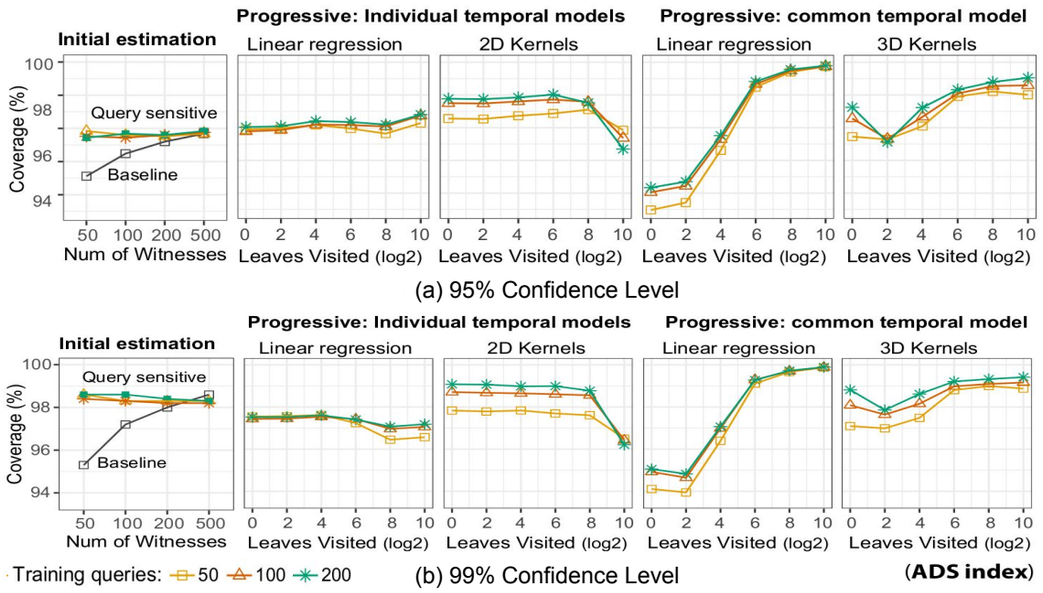


Figure 4.10.: Coverage probabilities of our estimation methods for 95% and 99% confidence levels. We show averages for the four datasets (synthetic, seismic, SALD, deep1B) and for 50, 100, and 200 training queries. The results of the temporal models are for the ADS index.

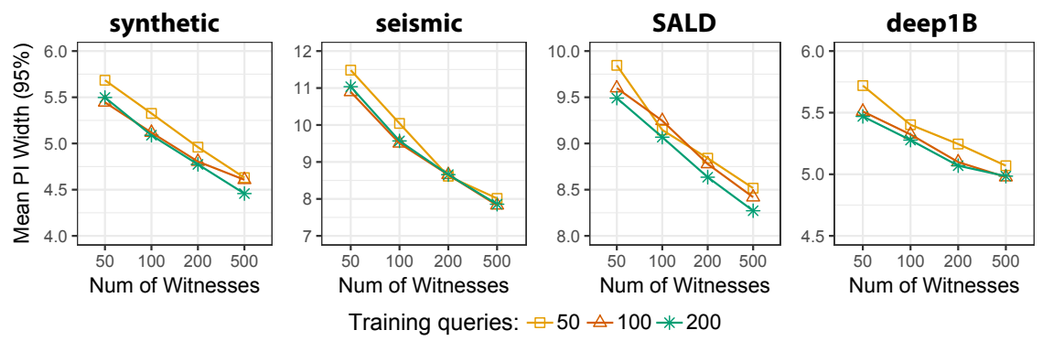


Figure 4.11.: The mean width of the 95% PI for the witness-based query-sensitive method in relation to the number of witnesses and training queries.

ever, as Figure 4.11 shows, more witnesses increase the precision of prediction intervals, i.e., intervals become tighter while they still cover the same proportion of true 1-NN distances. Larger numbers of training queries also help but to a lesser extent.

The coverage probabilities of *individual linear* models are generally stable and very close to 95% for $\theta = .05$. For $\theta = .01$, the coverage is around $97-98\%$, which suggests that the approach is relatively less accurate at high confidence levels. The 2D kernel density approach (*individual kernel*) results in higher coverage for both confidence levels (expect for $2^{10} = 1024$ visited leaves). Results for the *common linear* model are less satisfying, especially for the very first stages of the progression algorithm (< 16 visited leaves), where coverage is low. Coverage reaches very high levels ($> 98\%$ for a 95% confidence level) as more leaves are visited. As we explained earlier, this behavior may not be desirable since a very high coverage can be due to prediction intervals that are unnecessarily large. The *common kernel* density estimation method results in improved coverage levels. Coverage again tends to further increase as the number of visited leaves grows.

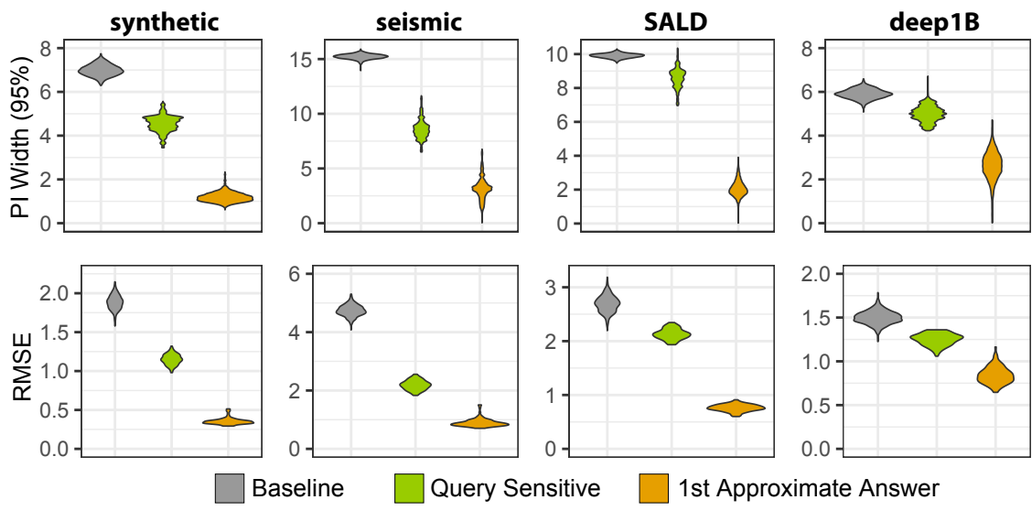


Figure 4.12.: Violin plots showing the distribution of the width of 95% prediction intervals (top) and the distribution of the RMSE of expected 1-NN distances (bottom). We use $n_w = 500$ (baseline and query-sensitive method) and $n_r = 100$ (query-sensitive method and model for the first approximate answer derived using the ADS index).

Figure 4.12 compares the two methods of initial estimates (Baseline and Query-Sensitive) against estimation based on the 1st approximate answer of the ADS index (see Figure 4.5). For this latter method, we construct individual linear models, where *Leaves Visited* = 1. For all comparisons, we set $n_w = 500$ and $n_r = 100$ because we know that for these parameters, the coverage probability of all methods is very close to 95%. We evaluate the width of their 95% prediction intervals and RMSE. Both measures show similar trends. There is a clear advantage of the query-sensitive method compared to the baseline. Estimation based on the 1st approximate answer works the best, leading to radical improvements for all datasets.

As shown in Figure 4.13, our progressive methods result in further improvements. The RMSE is very similar for all four methods. This means that they are all equally good at providing point estimates. The common linear model produces less accurate intervals though. We observe that their width improves slowly, which explains the growing coverage of this method as search progresses (see Figure 4.10). Both kernel density estimation methods provide a good balance between coverage and size but produce wider intervals than individual linear models.

DSTree Index. We observe similar patterns with the DSTree index [125]. Figure 4.14 summarizes our experimental results for this algorithm. All methods result in similar RMSE scores, but provide a different balance between coverage and interval width. The individual linear models present a less variable coverage probability that is very close to 95% and produce tighter intervals. Prediction intervals constructed with a common linear model are again the worst with very variable coverage levels. If we compare Figure 4.13 and Figure 4.14, we further observe that the DSTree index produces more precise estimates than the ADS index and very low errors at the very first leaves.

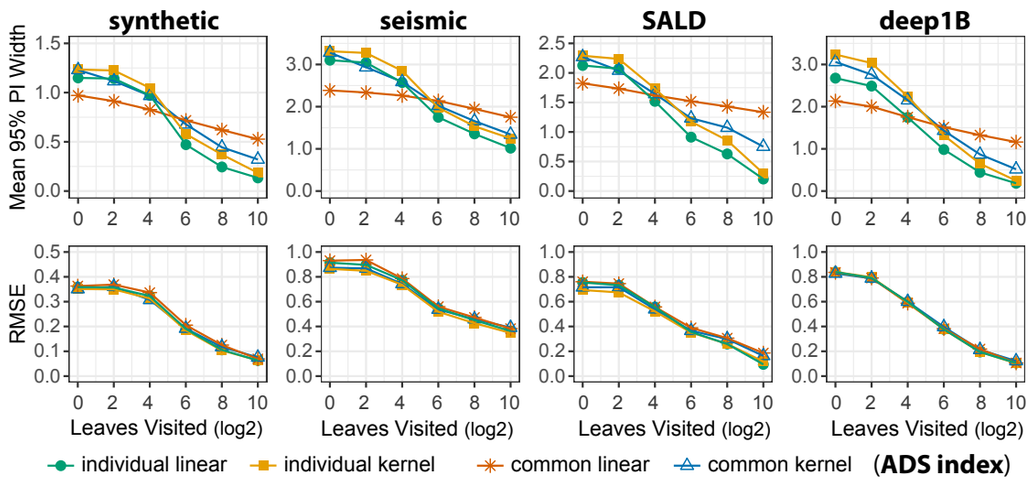


Figure 4.13.: Progressive models: Mean width of 95% prediction intervals of the 1-NN distance and its RMSE. Training is based on $n_r = 100$ queries.

Table 4.5.: Coverage Probabilities for Subsequences

Dataset	Full Series	Subseq. (mean)	Subseq. (min)
synthetic	95.2%	94.6%	94.1%
seismic	95.9%	94.0%	93.5%
SALD	94.4%	92.8%	92.1%
deep1B	95.0%	94.7%	94.3%
Overall	95.1%	94.0%	93.7%

Note: We report coverage probabilities (95% confidence level) for 16 data series subsequences when estimating the sub-distance of the 1-NN from the 1st approximate answer of the ADS index. All results are based on $n_r = 200$ training queries.

Subsequences. We can apply the same approach to provide distance estimates for the subsequences of the 1-NN (see Section 4.3). Table 4.5 reports the coverage probability of 95% prediction intervals predicting the partial distance of 16 individual segments of the 1-NN. Results are based on the fist approximate answer (*Visited Leaves* = 1) of the ADS index and $n_r = 200$ training queries when using a individual linear model. Average coverage levels are slightly lower than nominal levels ($\simeq$ 94% on average) but still acceptable.

A key question is whether distance estimation on individual subsequences helps reduce uncertainty. Figure 4.15 shows the distribution of the lower and upper bounds of the 95% prediction intervals constructed for both full series and their subsequences. We observe that the approach allows us to derive tighter bounds for subsequences. In particular, the upper bounds are greatly lower than the ones determined by the intervals that we construct for the full series distances. Likewise, for several queries, lower bounds are considerably greater than zero, which is the hard lower bound that we get in the absence of other information.

Sequential Tests. We assess how multiple sequential tests (refer to Section 4.3) affect the coverage probability of 95% and 99% prediction intervals. We focus on

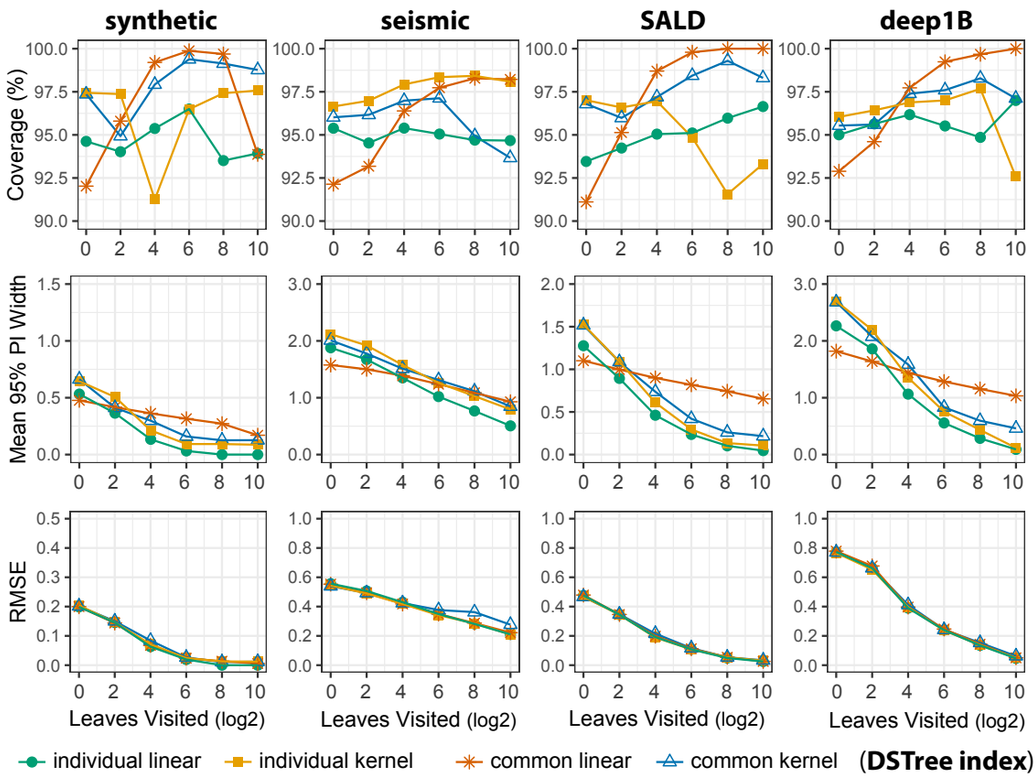


Figure 4.14.: Coverage probability (95% confidence level), mean width of prediction intervals, and RSME of progressive models for the DSTree index. All results are based on $n_r = 100$ training queries.

the ADS index and the three progressive estimation methods (individual linear, individual kernel, and common kernel) that gave the best coverage results (see Figure 4.10). We examine the effect of (i) three sequential tests when visiting 1, 16, and 256 leaves and (ii) five sequential tests when visiting 1, 4, 16, 64, and 256 leaves. For each test query, we count an error if at least one of the three or five progressive prediction intervals do not include the true 1-NN distance.

Figure 4.16 summarizes our results. The two kernel methods present similar trends. The coverage of their 95% prediction intervals drops from over 95% to about 90% for five tests. Likewise, the coverage of their 99% prediction intervals drops to a level that is slightly higher than 95%. In contrast, individual linear models are more sensitive to sequential testing and result in lower coverage lev-

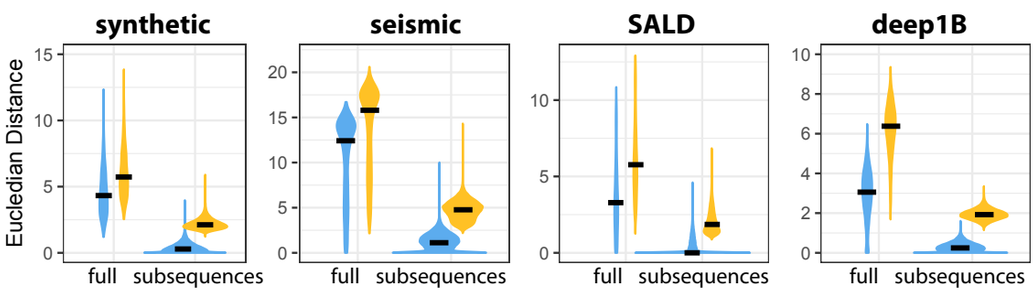


Figure 4.15.: Violin plots showing the distributions of low (blue/left) and upper (yellow/right) bounds of the 95% prediction intervals (1st approximate answer of ADS index) of the distance for the full 1-NN and for its 16 subsequences. The thick black lines show medians. Training is based on $n_r = 200$ queries.

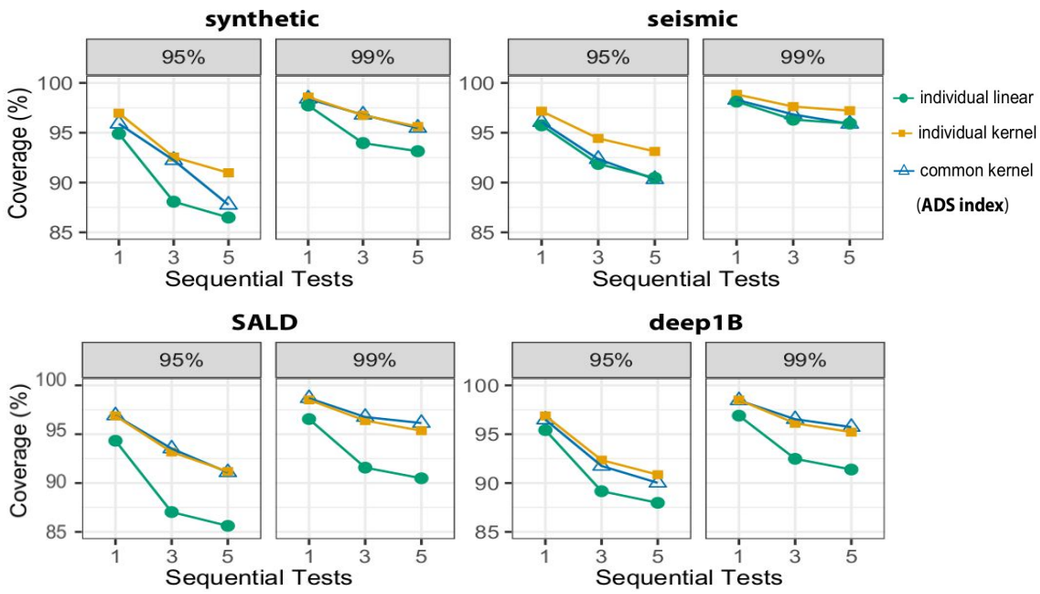


Figure 4.16.: Effect of multiple sequential tests on the coverage of 95% and 99% prediction intervals. We evaluate three (for 1, 16, and 256 visited leaves) and five sequential tests (for 1, 4, 16, 64, and 256 visited leaves).

els. In particular, the coverage of their 99% prediction intervals becomes clearly lower than 95% for all but one dataset.

When sequential testing is an issue, we recommend using either of the two kernel methods, as their coverage level is more stable across datasets. Adjusting the confidence level to account for multiple sequential comparisons is still an open question. Our results provide some rules of thumb (e.g., using a 95% level to guarantee a 90% coverage in 5 sequential tests), but such rules may depend on the estimation method and the steps at which estimates are made.

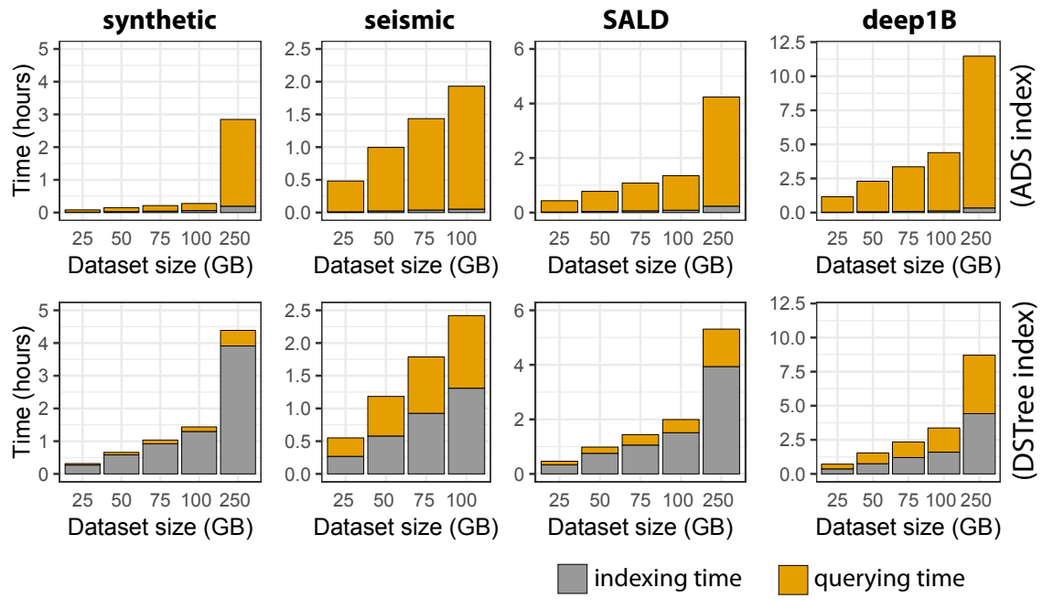


Figure 4.17.: Time to process 100 training queries, varying dataset size: ADS (top), DSTree (bottom).

Training Time. Finally, we evaluate the time that it takes to process a random set of training queries, required by our methods. Figure 4.17 presents results for ADS and DSTree for n_r=100. Our results show that DSTree results in faster

query completion, but for a higher indexing cost. Even though we report the index creation cost, we note that we could train a model on an existing index, paying only the query answering cost for the n_r queries. Overall, we can train our models in ~50min for 100GB datasets (average DSTree query answering performance), or ~2hours for 250GB datasets, and then use these models to support large numbers of progressive queries by multiple users with very high coverage (see Figure 4.10 and Figure 4.14).

4.8 VISUALIZATION EXAMPLES

Our estimation methods can help users assess how far their current answer is from the 1-NN. To this end, we provide estimates about two different measures: (i) the true 1-NN distance, and (ii) the relative (percent) distance error of the current answer (Equation 4.2). Figure 4.18 presents a query example and its progressive results. We use a variation of pirate plots [94] to communicate the 1-NN distance estimate $\hat{d}_{Q,1nn}(t)$ and the distance error estimate $\hat{e}_Q(t)$ by visualizing their probability density distribution and their 95% prediction interval. We also depict distance error estimates for the 4 subsequences (a,b,c,d) of progressive answers.

We observe that the initial distance estimate is highly uncertain, but estimates become precise at the early stages of the search. The upper bound of the full error estimate drops below 10% within 1.1sec and below 3% within 3.8sec (total query execution is 75.2sec). Such estimates can give confidence to the user that the current answer is very close to the 1-NN. In this example, the 1-NN is found within 1.1-3.8 sec. Note also that subsequences' error estimates are wider: if users have high precision requirements for a specific part of the query (e.g., for subsequences c and d), they may decide to wait longer for these error estimates to reduce further.

4.9 DISCUSSION AND FUTURE WORK

We showed that existing approaches do not scale to large collections of millions of data series. We note that, as data series indexes get faster, our proposed solutions will still be relevant, as they can eventually support larger datasets. Our methods could also be used to speedup complex analysis algorithms (e.g., k-NN classifiers) by enabling them to automatically decide when to stop a similarity search query early, without sacrificing the accuracy of the overall analysis process.

We currently examine extensions to our work. We plan to run benchmarks for the prediction of error bounds for every k-NN answer and not only for the 1-NN that we presented here. In addition, some data domains demand progressive similarity search using similarity measures other than the Euclidean Distance. We have to check if our methods work well with distance measures such as Dynamic Time Warping, which is invariant to temporal location.

We also need to study the visualization and human-computer interaction aspects that emerge in this context. The challenge of how to visually communicate progressive guarantees of similarity search results still remains open. Past work

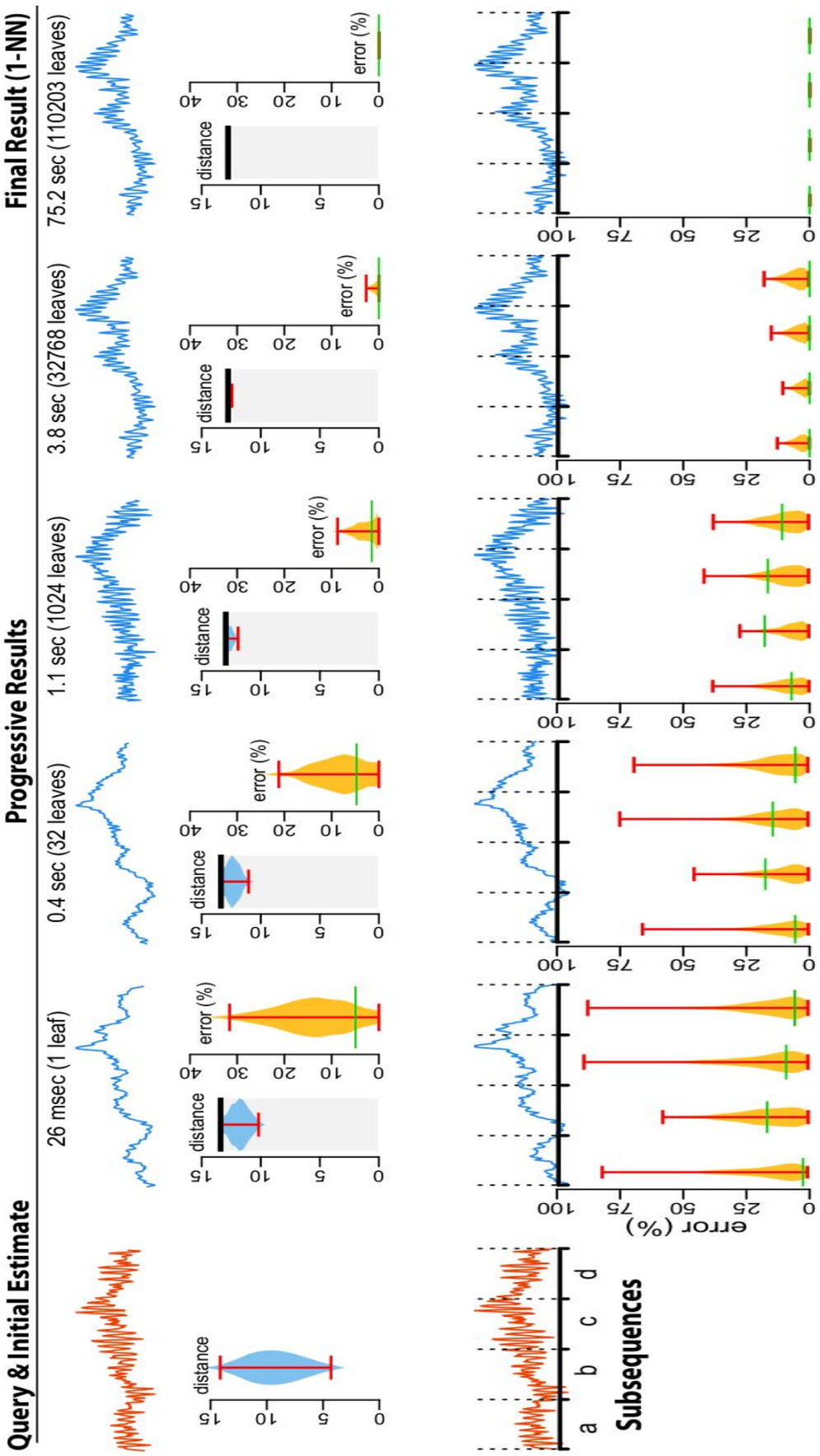


Figure 4.18.: A query example from the seismic dataset showing the evolution of 1-NN distance estimates, and estimates of the distance error (see Equation 4.2) of the full progressive answer (top) and its four subsequences a-d (bottom). The thick black lines show the distance of the current approximate answer. The red error bars represent 95% prediction intervals. The green line over the predicted distribution of errors shows the real error, which is only shown for illustration purposes (it is unknown during the search). Estimates are based on a training set of 100 queries, as well as 100 random witnesses for initial estimates. We use the ADS index.

has considered width and color (saturation) of converged bars for visually encoding confidence of incremental aggregate results (i.e., the wider and the darker the bars, the more confident the progressive aggregate estimation (high density probability)). For example, we could apply such visual variables to violin or pirate plots, like our attempt in Figure 4.18, or integrate distance bounds into the visual representation of a time series, by either using the query itself, or its progressive k-NN answer. In Line Charts, we could draw color-saturated areas, more or less tight, around patterns of progressive results to convey the uncertainty of final estimates. Though, it is not clear what needs to be done in other visualizations beyond Line Charts, such as Horizon Graphs and Color Fields, where the color variable is already used as a core element of their implementation.

We also need to integrate our solutions into a visual analytics system. Given the requirements of some users, such as neuroscientists and astronomers, to view context information (i.e., multiple time series together), creating a system that shows progressive results in-context presents many challenges. In particular, we have to think about how to show and alternate between progressive results that appear in different parts of a dataset.

Finally, and most importantly, our work is motivated by real analysts (astronomers) and their problems, but has never been validated in practice. So we do not know if analysts are willing to use our solutions in their everyday work, nor how they will influence their workflow. Given the increasing popularity of data series visual analysis tasks, these research directions are both relevant and important, offering exciting research opportunities.

4.10 CONCLUSION

We demonstrated the usefulness (and need) of progressive similarity search in large data series collections. Our preliminary findings showed that the greatest cost is not locating the 1-NN, but rather waiting for the algorithm to confirm that there is no better answer and finish execution. This behaviour results in inflated waiting times without any improvement. We can reduce waiting times by providing users with progressive results. An important research question is how to compute such progressive results and couple them with probabilistic quality guarantees. Such information can help users decide when to stop the search process, in cases where improvement in the final answer is not possible, eliminating wasted times.

We presented the first scalable and effective solutions to this computational probabilistic problem. We demonstrated their applicability using synthetic and real datasets. Our benchmark evaluation indicated that our prediction methods significantly outperform competing approaches. We are able to provide both initial and progressive estimates of the final result, that are getting better during the execution of similarity search. We are currently examining extensions of our work, that is, k-NN search and other distance measures (such as Dynamic Time Warping), as well as studying in more detail how to visually communicate such prediction measures to users without misleading them to false decisions.

5

CONCLUSION

Data series are ubiquitous in a wide range of domains, such as seismology, astrophysics, and neuroscience. Their massive volume has introduced numerous challenging problems to the research community. This dissertation focused on two research challenges we identified: (1) *time series similarity perception* and (2) *progressive similarity search*. In particular:

- We investigated for the first time in data visualization community whether different time series visualizations affect similarity perception. We studied which visualizations promote or penalize results from similarity search algorithms and provided guidelines to people who visually explore their data which visualizations to use according to their domain-dependent definition and notion of similarity.
- We investigated the quality of early progressive results through a set of computational benchmarks (or experiments) using large data series datasets, both synthetic and real.
- We developed a scalable probabilistic method that provides quality guarantees (error bounds) for progressive similarity search query answering in massive data series collections; and we compared different models of this method with regards to their speed and quality.

We provide an overview of the results of this thesis, and then discuss future research directions.

5.1 SUMMARY OF CONTRIBUTIONS AND FUTURE WORK

Time Series Similarity Perception.

As a first step (Chapter 3), we studied how different visual encodings (Line Charts, Horizon Graphs, and Color Fields) affect time series similarity perception. In particular, we investigated if the time series results returned by automatic similarity measures are perceived in a similar manner, irrespective of the visualization technique; and if what people perceive as similar with each visualization aligns with different automatic measures and their similarity constraints. Our findings suggest that Horizon Graphs align with similarity measures that allow local variations in temporal position (i.e., dynamic time warping). On the other hand, Horizon Graphs do not align with measures that allow variations in y-offset and amplitude scaling (i.e., measures based on z-normalization), while the inverse seems to be the case for Line Charts and Color Fields. Overall, our work provides some first evidence that similarity perception in time series is visualization-dependent.

`Future Work`: There are still many aspects of this work left for future investigation. Future experiments need to determine the relationship between additional

visual encodings and other similarity measures. We tested visual settings that utilize position and/or color for value encoding, are linear in time, split the space, and scale well for multiple time series; and only a few similarity measures whose similarity constraints support our data domain. Our EEG dataset stemming from neuroscience has specific pattern characteristics, such as spikes followed by rapid discharges. Further studies need to validate our findings in a wider range of patterns and datasets from other domains.

While our visual encodings scale to multiple time series, we focused on a small number of data series to compare. Thereby, it is unclear how our results generalize to larger series collections. For example, Color Fields scale well as small multiples. In contrast, the aspect ratio may greatly affect the readability of Line Charts. Thus, for Line Charts and to a lesser degree for Horizon Graphs, a reduced vertical space could lead to a loss of small patterns and reinforce large structures (peaks, valleys) altering similarity perception.

Our follow-up experiment showed no clear differences between the RGB and LAB color interpolation techniques in Color Fields with regards to time series similarity perception. However, it is possible that differences may exist in other types of temporal patterns. Moreover, dynamic color maps, such as ones based on equi-depth or equi-width binning of time series values, distort the original signals, conceivably affecting similarity perception. Studying how the choice of color mappings affects how people perceive time series similarity is an exciting future direction.

Last but not least, our future goal is to validate our results with domain experts. Our work has been motivated by how neuroscientists inspect their data and compare similar patterns, e.g., by using Line Chart visualizations in small multiples. A common problem is that there is high disagreement in their decisions. Their assessments are often very subjective. We are interested in the role of different visualizations in their decision-making process. Our goal is to investigate if choosing appropriate visualizations can improve consensus among domain experts about what is similar, and if this approach can increase their trust on the results of automatic similarity search algorithms.

Progressive Similarity Search & Quality Guarantees.

In the second part of the thesis (Chapter 4), we focused on the scalability constraints of similarity search algorithms. As datasets become larger, systems dealing with data series cannot provide users with similarity search results within interactive response times. Therefore, we sought progressive query-answering mechanisms, i.e., mechanisms that give back to users progressive, approximate results. Such results are not the final and exact ones but progressively improve during the execution of similarity search. Our experiments (benchmarks) indicated that there is a gap between the time the most similar answer is found and the time when the search algorithm terminates, resulting in inflated waiting times without any improvement. Probabilistic estimates of the final answer could help users decide when to stop the search process, i.e., decide if improvements in the final answers are not probable, thus eliminating waiting times. Our work focused on how to efficiently compute such probabilistic estimates. We developed a new probabilistic, learning-based method that provides quality guar-

antees (distance bounds) for progressive k-Nearest Neighbour (k-NN) query results. We demonstrated the applicability of our method using synthetic and real datasets of millions of data series. Our approach significantly outperforms competing approaches.

`Future Work`: There are several possible future extensions of this work. We tested and evaluated our method to predict estimates of the final 1-NN answer (i.e., the most similar answer to the query). We have to test if the same method is applicable and effective for predicting error bounds and quality guarantees for every k-NN answer.

In addition, we only used the Euclidean Distance (ED) as a similarity measure during the execution of k-NN similarity search queries. ED is a measure that does not transform data series and any variation between two raw series contributes to their further distance apart (i.e., ED is a non-invariant measure). Furthermore, ED satisfies the triangular inequality. We have to test if our method also works well with other distance measures, such as Dynamic Time Warping, which is invariant to time warping and temporal offset and does not support the triangular inequality.

We also need to study in more detail how to visually and statistically communicate error distance bounds of progressive similarity search results to users. Possible directions are to integrate such distance bounds and errors into the visual representation of a time series, by either using the query itself, or its progressive k-NN answers. We could draw color-saturated areas, more or less wide around progressive results in order to convey the confidence of the final estimate. Color and width have been used before for the visualization of uncertainty in progressive aggregate results. We could apply such visual variables on top of Line Charts, but it is not clear what alternative visual variables are appropriate for confidence encoding in other visualizations, such as Horizon Graphs and Color Fields, where color is already used. Alternatively, we could apply width and color to supplementary visualizations, such as violin or pirate plots.

Finally, and most importantly, we plan to validate our work with real users, such as astronomers who often need to analyze large volumes of data series. In particular, we do not know how our solutions could affect their decisions and workflow. We are interested in evaluating the effectiveness of alternative visualizations of progressive guarantees and measuring how fast they help users complete their visual analysis tasks, when those have to deal with large data series collections.

BIBLIOGRAPHY

[1] Muhammad Adnan, Mike Just, and Lynne Baillie. "Investigating Time Series Visualisations to Improve the User Experience." In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI '16. San Jose, California, USA: ACM, 2016, pp. 5444–5455. ISBN: 978-1-4503-3362-7. DOI: 10.1145/2858036.2858300. URL: http://doi.acm.org/10.1145/2858036.2858300.

[2] Wolfgang Aigner, Silvia Miksch, Heidrun Schumann, and Christian Tominski. *Visualization of Time-Oriented Data*. 1st. Springer Publishing Company, Incorporated, 2011. ISBN: 0857290789, 9780857290786.

[3] Danielle Albers, Michael Correll, and Michael Gleicher. "Task-driven Evaluation of Aggregation in Time Series Visualization." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '14. Toronto, Ontario, Canada: ACM, 2014, pp. 551–560. ISBN: 978-1-4503-2473-1. DOI: 10.1145/2556288.2557200. URL: http://doi.acm.org/10.1145/2556288.2557200.

[4] Danielle Albers, Colin Dewey, and Michael Gleicher. "Sequence Surveyor: Leveraging Overview for Scalable Genomic Alignment Visualization." In: *IEEE Transactions on Visualization and Computer Graphics* 17.12 (Dec. 2011), pp. 2392–2401. ISSN: 1077-2626. DOI: 10.1109/TVCG.2011.232. URL: http://dx.doi.org/10.1109/TVCG.2011.232.

[5] Marco Angelini, Giuseppe Santucci, Heidrun Schumann, and Hans-Jörg Schulz. "A Review and Characterization of Progressive Visual Analytics." In: *Informatics* 5 (2018), p. 31.

[6] Sunil Arya, David M. Mount, Nathan S. Netanyahu, Ruth Silverman, and Angela Y. Wu. "An Optimal Algorithm for Approximate Nearest Neighbor Searching Fixed Dimensions." In: *J. ACM* 45.6 (Nov. 1998), pp. 891–923. ISSN: 0004-5411. DOI: 10.1145/293347.293348. URL: http://doi.acm.org/10.1145/293347.293348.

[7] Johannes Aßfalg, Hans-Peter Kriegel, Peer Kröger, Peter Kunath, Alexey Pryakhin, and Matthias Renz. "Similarity Search on Time Series Based on Threshold Queries." In: *Proceedings of the 10th International Conference on Advances in Database Technology*. EDBT'06. Munich, Germany: Springer-Verlag, 2006, pp. 276–294. ISBN: 3-540-32960-9, 978-3-540-32960-2. DOI: 10.1007/11687238_19. URL: http://dx.doi.org/10.1007/11687238_19.

[8] Sriram Karthik Badam, Niklas Elmqvist, and Jean-Daniel Fekete. "Steering the Craft: UI Elements and Visualizations for Supporting Progressive Visual Analytics." In: *Comput. Graph. Forum* 36.3 (2017).

[9] Thomas Baguley. *Serious Stats: A guide to advanced statistics for the behavioral sciences*. Palgrave Macmillan, 2012. ISBN: 9780230363557. URL: https://books.google.fr/books?id=ObUcBQAAQBAJ.

[10] Gustavo E. Batista, Eamonn J. Keogh, Oben Moses Tataw, and Vinícius M. Souza. "CID: An Efficient Complexity-invariant Distance for Time Series." In: *Data Min. Knowl. Discov.* 28.3 (May 2014), pp. 634–669. ISSN: 1384-5810. DOI: 10.1007/s10618-013-0312-3. URL: http://dx.doi.org/10.1007/s10618-013-0312-3.

[11] Donald J. Berndt and James Clifford. "Using Dynamic Time Warping to Find Patterns in Time Series." In: *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*. AAAIWS'94. Seattle, WA: AAAI Press, 1994, pp. 359–370. URL: http://dl.acm.org/citation.cfm?id=3000850.3000887.

[12] Lior Berry and Tamara Munzner. "BinX: Dynamic Exploration of Time Series Datasets Across Aggregation Levels." In: *10th IEEE Symposium on Information Visualization (InfoVis 2004), 10-12 October 2004, Austin, TX, USA*. 2004. DOI: 10.1109/INFVIS.2004.11. URL: https://doi.org/10.1109/INFVIS.2004.11.

[13] Enrico Bertini, Patrick Hertzog, and Denis Lalanne. "SpiralView: Towards Security Policies Assessment Through Visual Correlation of Network Resources with Evolution of Alarms." In: *Proceedings of the 2007 IEEE Symposium on Visual Analytics Science and Technology*. VAST '07. Washington, DC, USA: IEEE Computer Society, 2007, pp. 139–146. ISBN: 978-1-4244-1659-2. DOI: 10.1109/VAST.2007.4389007. URL: https://doi.org/10.1109/VAST.2007.4389007.

[14] Robert L. Brennan and Dale J. Prediger. "Coefficient kappa: Some uses, misuses, and alternatives." In: *Educational and psychological measurement* 41.3 (1981), pp. 687–699.

[15] Sergey Brin. "Near Neighbor Search in Large Metric Spaces." In: *Proceedings of the 21th International Conference on Very Large Data Bases*. VLDB '95. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1995, pp. 574–584. ISBN: 1-55860-379-4. URL: http://dl.acm.org/citation.cfm?id=645921.673006.

[16] Paolo Buono and Adalberto Lafcadio Simeone. "Interactive Shape Specification for Pattern Search in Time Series." In: *AVI*. 2008.

[17] Paolo Buono, Aleks Aris, Catherine Plaisant, Amir Khella, and Ben Shneiderman. "Interactive pattern search in time series." In: *Visualization and Data Analysis 2005, San Jose, CA, USA, January 17, 2005*. 2005, pp. 175–186. DOI: 10.1117/12.587537. URL: https://doi.org/10.1117/12.587537.

[18] Lee Byron and Martin Wattenberg. "Stacked Graphs – Geometry & Aesthetics." In: *IEEE Transactions on Visualization and Computer Graphics* 14.6 (Nov. 2008), pp. 1245–1252. ISSN: 1077-2626. DOI: 10.1109/TVCG.2008.166. URL: http://dx.doi.org/10.1109/TVCG.2008.166.

[19] Alessandro Camerra, Themis Palpanas, Jin Shieh, and Eamonn Keogh. "iSAX 2.0: Indexing and Mining One Billion Time Series." In: *ICDM*. 2010.

[20] Alessandro Camerra, Jin Shieh, Themis Palpanas, Thanawin Rakthanmanon, and Eamonn J. Keogh. "Beyond One Billion Time Series: Indexing and Mining Very Large Time Series Collections with iSAX2+." In: *Knowl. Inf. Syst.* 39.1 (2014), pp. 123–151.

[21] Angelo Canty and B. D. Ripley. *boot: Bootstrap R (S-Plus) Functions*. R package version 1.3-20. 2017.

[22] Kaushik Chakrabarti, Eamonn J. Keogh, Sharad Mehrotra, and Michael J. Pazzani. "Locally adaptive dimensionality reduction for indexing large time series databases." In: *ACM Trans. Database Syst.* 27.2 (2002), pp. 188–228. DOI: 10.1145/568518.568520. URL: https://doi.org/10.1145/568518.568520.

[23] Eric Chassande-Mottin. "Data analysis challenges in transient gravitational-wave astronomy." In: *arXiv:1210.7173v2* (2013).

[24] Surajit Chaudhuri, Bolin Ding, and Srikanth Kandula. "Approximate Query Processing: No Silver Bullet." In: *SIGMOD*. 2017.

[25] Yanping Chen, Eamonn Keogh, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, and Gustavo Batista. *The UCR Time Series Classification Archive*. www.cs.ucr.edu/~eamonn/time_series_data/. 2015.

[26] Yueguo Chen, Mario Nascimento, Beng Chin Ooi, and Anthony KH Tung. "SpADe: On Shape-based Pattern Detection in Streaming Time Series." In: *Proceedings of the IEEE 23rd International Conference on Data Engineering*. ICDE'07. IEEE, 2007, pp. 786–795. DOI: 10.1109/ICDE.2007.367924.

[27] Paolo Ciaccia, Alessandro Nanni, and Marco Patella. "A Query-sensitive Cost Model for Similarity Queries with M-tree." In: *In Proc. of the 10th ADC*. Springer Verlag, 1999, pp. 65–76.

[28] Paolo Ciaccia and Marco Patella. "PAC Nearest Neighbor Queries: Approximate and Controlled Search in High-Dimensional and Metric Spaces." In: *ICDE*. 2000, pp. 244–255.

[29] Paolo Ciaccia, Marco Patella, and Pavel Zezula. "A Cost Model for Similarity Queries in Metric Spaces." In: *PODS*. 1998.

[30] William S. Cleveland and Robert McGill. "An Experiment in Graphical Perception." In: *International Journal of Man-Machine Studies* 25.5 (1986), pp. 491–501. DOI: 10.1016/S0020-7373(86)80019-0. URL: https://doi.org/10.1016/S0020-7373(86)80019-0.

[31] Michael Correll and Michael Gleicher. "The Semantics of Sketch: Flexibility in Visual Query Systems for Time Series Data." In: *2016 IEEE Conference on Visual Analytics Science and Technology (VAST)*. 2016, pp. 131–140.

[32] Michael Correll, Danielle Albers, Steven Franconeri, and Michael Gleicher. "Comparing Averages in Time Series Data." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '12. Austin, Texas, USA: ACM, 2012, pp. 1095–1104. ISBN: 978-1-4503-1015-4. DOI: 10.1145/2207676.2208556. URL: http://doi.acm.org/10.1145/2207676.2208556.

[33] Marco de Curtis, John G R Jefferys, and Massimo Avoli. "Interictal Epileptiform Discharges in Partial Epilepsy: Complex Neurobiological Mechanisms Based on Experimental and Clinical Evidence." In: *Jasper's Basic Mechanisms of the Epilepsies [Internet]. 4th edition* (2012), pp. 303–325. eprint: https://www.ncbi.nlm.nih.gov/books/NBK50785/pdf/Bookshelf_NBK50785.pdf. URL: https://www.ncbi.nlm.nih.gov/books/NBK98179/.

[34] Bolin Ding, Silu Huang, Surajit Chaudhuri, Kaushik Chakrabarti, and Chi Wang. "Sample + Seek: Approximating Aggregates with Distribution Precision Guarantee." In: *SIGMOD*. 2016.

[35] Hui Ding, Goce Trajcevski, Peter Scheuermann, Xiaoyue Wang, and Eamonn Keogh. "Querying and Mining of Time Series Data: Experimental Comparison of Representations and Distance Measures." In: *Proc. VLDB Endow.* 1.2 (Aug. 2008), pp. 1542–1552. ISSN: 2150-8097. DOI: 10.14778/1454159.1454226. URL: http://dx.doi.org/10.14778/1454159.1454226.

[36] Pierre Dragicevic. "Fair Statistical Communication in HCI." In: *Modern Statistical Methods for HCI*. Springer, 2016, pp. 291–330. DOI: 10.1007/978-3-319-26633-6_13. eprint: https://hal.inria.fr/hal-01377894/file/fairstats-last.pdf. URL: https://hal.inria.fr/hal-01377894.

[37] Tarn Duong and Martin L. Hazelton. "Cross-validation Bandwidth Matrices for Multivariate Kernel Density Estimation." In: *Scandinavian Journal of Statistics* 32.3 (2005), pp. 485–506. DOI: 10.1111/j.1467-9469.2005.00445.x.

[38] Tarn Duong, Matt Wand, Jose Chacon, and Artur Gramacki. *ks: Kernel Smoothing*. https://cran.r-project.org/web/packages/ks/. 2019.

[39] Karima Echihabi, Kostas Zoumpatianos, Themis Palpanas, and Houda Benbrahim. "The Lernaean Hydra of Data Series Similarity Search: An Experimental Evaluation of the State of the Art." In: *The VLDB Journal* 12.2 (2018).

[40] Bradley Efron. "Better Bootstrap Confidence Intervals." In: *Journal of the American Statistical Association* 82.397 (1987), pp. 171–185. DOI: 10.1080/01621459.1987.10478410. eprint: https://www.tandfonline.com/doi/pdf/10.1080/01621459.1987.10478410. URL: https://www.tandfonline.com/doi/abs/10.1080/01621459.1987.10478410.

[41] Philipp Eichmann and Emanuel Zgraggen. "Evaluating Subjective Accuracy in Time Series Pattern-Matching Using Human-Annotated Rankings." In: *Proceedings of the 20th International Conference on Intelligent User Interfaces*. IUI '15. Atlanta, Georgia, USA: ACM, 2015, pp. 28–37. ISBN: 978-1-4503-3306-1. DOI: 10.1145/2678025.2701379. URL: http://doi.acm.org/10.1145/2678025.2701379.

[42] Christos Faloutsos, M. Ranganathan, and Yannis Manolopoulos. "Fast Subsequence Matching in Time-series Databases." In: *Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data*. SIGMOD '94. Minneapolis, Minnesota, USA: ACM, 1994, pp. 419–429. ISBN: 0-89791-639-5. DOI: 10.1145/191839.191925. URL: http://doi.acm.org/10.1145/191839.191925.

[43] Jean-Daniel Fekete and Romain Primet. "Progressive Analytics: A Computation Paradigm for Exploratory Data Analysis." In: *CoRR* abs/1607.05162 (2016). URL: http://arxiv.org/abs/1607.05162.

[44] Danyel Fisher, Steven M. Drucker, and A. Christian König. "Exploratory Visualization Involving Incremental, Approximate Database Queries and Uncertainty." In: *IEEE CG&A* 32 (2012).

[45] Danyel Fisher, Igor Popov, Steven Drucker, and M. C. Schraefel. "Trust Me, I'm Partially Right: Incremental Visualization Lets Analysts Explore Large Datasets Faster." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '12. Austin, Texas, USA: ACM, 2012, pp. 1673–1682. ISBN: 978-1-4503-1015-4. DOI: 10.1145/2207676.2208294. URL: http://doi.acm.org/10.1145/2207676.2208294.

[46] Tak-chung Fu. "A Review on Time Series Data Mining." In: *Eng. Appl. Artif. Intell.* 24.1 (2011), pp. 164–181. ISSN: 0952-1976. DOI: 10.1016/j.engappai.2010.09.007. URL: http://dx.doi.org/10.1016/j.engappai.2010.09.007.

[47] Johannes Fuchs, Fabian Fischer, Florian Mansmann, Enrico Bertini, and Petra Isenberg. "Evaluation of Alternative Glyph Designs for Time Series Data in a Small Multiple Setting." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '13. Paris, France: ACM, 2013, pp. 3237–3246. ISBN: 978-1-4503-1899-0. DOI: 10.1145/2470654.2466443. URL: http://doi.acm.org/10.1145/2470654.2466443.

[48] Anna Gogolou, Theophanis Tsandilas, Themis Palpanas, and Anastasia Bezerianos. "Comparing Time Series Similarity Perception under Different Color Interpolations." In: *Inria Research Report, RR-9189*. 2018. URL: https://hal.inria.fr/hal-01844994v2.

[49] Anna Gogolou, Theophanis Tsandilas, Themis Palpanas, and Anastasia Bezerianos. "Comparing Similarity Perception in Time Series Visualizations." In: *IEEE Trans. Vis. Comput. Graph.* 25.1 (2019), pp. 523–533. DOI: 10.1109/TVCG.2018.2865077. URL: https://doi.org/10.1109/TVCG.2018.2865077.

[50] Anna Gogolou, Karima Echihabi, Theophanis Tsandilas, Anastasia Bezerianos, and Themis Palpanas. "Data Series Progressive Similarity Search with Probabilistic Quality Guarantees." In: *Under submission*. 2019.

[51] Anna Gogolou, Theophanis Tsandilas, Themis Palpanas, and Anastasia Bezerianos. "Progressive Similarity Search on Time Series Data." In: *Proceedings of the Workshops of the EDBT/ICDT 2019 Joint Conference, EDBT/ICDT 2019, Lisbon, Portugal, March 26, 2019*. 2019. URL: https://bigvis.imsi.athenarc.gr/bigvis2019/papers/BigVis_2019_paper_5.pdf.

[52] Dina Q. Goldin and Paris C. Kanellakis. "On Similarity Queries for Time-Series Data: Constraint Specification and Implementation." In: *Proceedings of the First International Conference on Principles and Practice of Constraint Programming*. CP '95. London, UK, UK: Springer-Verlag, 1995, pp. 137–153. ISBN: 3-540-60299-2. URL: http://dl.acm.org/citation.cfm?id=647484.726176.

[53] Machon Gregory and Ben Shneiderman. "Shape Identification in Temporal Data Sets." In: *Expanding the Frontiers of Visual Analytics and Visualization*. 2012, pp. 305–321. DOI: 10.1007/978-1-4471-2804-5_17. URL: https://doi.org/10.1007/978-1-4471-2804-5_17.

[54] Yue Guo, Carsten Binnig, and Tim Kraska. "What you see is not what you get!: Detecting Simpson's Paradoxes during Data Exploration." In: *Proceedings of the 2nd Workshop on Human-In-the-Loop Data Analytics, HILDA@SIGMOD*. 2017.

[55] Kilem Li Gwet. *Handbook of Inter-Rater Reliability, 4th Edition: The Definitive Guide to Measuring The Extent of Agreement Among Raters*. Advanced Analytics, LLC, 2014. ISBN: 9780970806284. URL: https://books.google.fr/books?id=fac9BQAAQBAJ.

[56] Susan Havre, Elizabeth G. Hetzler, and Lucy T. Nowell. "ThemeRiver: Visualizing Theme Changes over Time." In: *IEEE Symposium on Information Visualization 2000 (INFOVIS'00), Salt Lake City, Utah, USA, October 9-10, 2000*. 2000, pp. 115–123. DOI: 10.1109/INFVIS.2000.885098. URL: https://doi.org/10.1109/INFVIS.2000.885098.

[57] Jeffrey Heer, Nicholas Kong, and Maneesh Agrawala. "Sizing the Horizon: The Effects of Chart Size and Layering on the Graphical Perception of Time Series Visualizations." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '09. Boston, MA, USA: ACM, 2009, pp. 1303–1312. ISBN: 978-1-60558-246-7. DOI: 10.1145/1518701.1518897. URL: http://doi.acm.org/10.1145/1518701.1518897.

[58] Joseph M. Hellerstein, Peter J. Haas, and Helen J. Wang. "Online Aggregation." In: *SIGMOD*. 1997.

[59] Harry Hochheiser and Ben Shneiderman. "Dynamic Query Tools for Time Series Data Sets: Timebox Widgets for Interactive Exploration." In: *Information Visualization* 3.1 (2004), pp. 1–18. ISSN: 1473-8716. DOI: 10.1145/993176.993177. URL: http://dx.doi.org/10.1145/993176.993177.

[60] Christian Holz and Steven Feiner. "Relaxed Selection Techniques for Querying Time-series Graphs." In: *Proceedings of the 22Nd Annual ACM Symposium on User Interface Software and Technology*. UIST '09. Victoria, BC, Canada: ACM, 2009, pp. 213–222. ISBN: 978-1-60558-745-5. DOI: 10.1145/1622176.1622217. URL: http://doi.acm.org/10.1145/1622176.1622217.

[61] K.P. Indiradevi, Elizabeth Elias, P.S. Sathidevi, S. Dinesh Nayak, and K. Radhakrishnan. "A multi-level wavelet approach for automatic detection of epileptic spikes in the electroencephalogram." In: *Computers in Biology and Medicine* 38.7 (2008), pp. 805 –816. ISSN: 0010-4825. DOI: https://doi.org/10.1016/j.compbiomed.2008.04.010. URL: http://www.sciencedirect.com/science/article/pii/S0010482508000693.

[62] Ali Jabbari, Renaud Blanch, and Sophie Dupuy-Chessa. "Composite Visual Mapping for Time Series Visualization." In: *IEEE Pacific Visualization Symposium (PacificVis)*. 2018, pp. 116–124. DOI: 10.1109/PacificVis.2018.00023.

[63] Waqas Javed, Bryan McDonnel, and Niklas Elmqvist. "Graphical Perception of Multiple Time Series." In: *IEEE Transactions on Visualization and Computer Graphics* 16.6 (Nov. 2010), pp. 927–934. ISSN: 1077-2626. DOI: 10.1109/TVCG.2010.162. URL: http://dx.doi.org/10.1109/TVCG.2010.162.

[64] Chris Jermaine, Subramanian Arumugam, Abhijit Pol, and Alin Dobra. "Scalable Approximate Query Processing with the DBO engine." In: *ACM Trans. Database Syst.* 33.4 (2008), 23:1–23:54.

[65] J. Jing, J. Dauwels, T. Rakthanmanon, E. Keogh, S.S. Cash, and M.B. Westover. "Rapid annotation of interictal epileptiform discharges via template matching under Dynamic Time Warping." In: *Journal of Neuroscience Methods* 274 (2016), pp. 179 –190. ISSN: 0165-0270. DOI: https://doi.org/10.1016/j.jneumeth.2016.02.025. URL: http://www.sciencedirect.com/science/article/pii/S0165027016300061.

[66] Jaemin Jo, Jinwook Seo, and Jean-Daniel Fekete. "PANENE: A Progressive Algorithm for Indexing and Querying Approximate k-Nearest Neighbors." In: *IEEE Transactions on Visualization and Computer Graphics* (2018). ISSN: 1077-2626. DOI: 10.1109/TVCG.2018.2869149. URL: http://dx.doi.org/10.1109/TVCG.2018.2869149.

[67] Uwe Jugel, Zbigniew Jerzak, Gregor Hackenbroich, and Volker Markl. "M4: A Visualization-Oriented Time Series Data Aggregation." In: *PVLDB* 7.10 (2014), pp. 797–808. DOI: 10.14778/2732951.2732953. URL: http://www.vldb.org/pvldb/vol7/p797-jugel.pdf.

[68] Eamonn J. Keogh, Kaushik Chakrabarti, Michael J. Pazzani, and Sharad Mehrotra. "Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases." In: *Knowl. Inf. Syst.* 3.3 (2001), pp. 263–286. DOI: 10.1007/PL00011669. URL: https://doi.org/10.1007/PL00011669.

[69] Robert Kincaid and Heidi Lam. "Line Graph Explorer: Scalable Display of Line Graphs Using Focus+Context." In: *Proceedings of the Working Conference on Advanced Visual Interfaces*. AVI '06. Venezia, Italy: ACM, 2006, pp. 404–411. ISBN: 1-59593-353-0. DOI: 10.1145/1133265.1133348. URL: http://doi.acm.org/10.1145/1133265.1133348.

[70] Haridimos Kondylakis, Niv Dayan, Kostas Zoumpatianos, and Themis Palpanas. "Coconut: A Scalable Bottom-Up Approach for Building Data Series Indexes." In: *PVLDB* 11.6 (2018), pp. 677–690. DOI: 10.14778/3184470.3184472.

[71] Tim Kraska. "Northstar: An Interactive Data Science System." In: *PVLDB* 11.12 (2018), pp. 2150–2164.

[72] Haim Levkowitz and Gabor Herman. "The Design and Evaluation of Color Scales for Image Data." In: *Computer Graphics and Applications)* 12.1 (1992), pp. 82–89.

[73] Eckhard Limpert, Werner A. Stahel, and Markus Abbt. "Log-Normal Distributions Across the Sciences: Keys and Clues." In: 51 (May 2001), pp. 341–. DOI: 10.1641/0006-3568(2001)051[0341:LNDATS]2.0.CO;2.

[74] Jessica Lin, Eamonn Keogh, and Stefano Lonardi. "Visualizing and Discovering Non-trivial Patterns in Large Time Series Databases." In: *Information Visualization* 4.2 (July 2005), pp. 61–82. ISSN: 1473-8716. DOI: 10.1057/palgrave.ivs.9500089. URL: http://dx.doi.org/10.1057/palgrave.ivs.9500089.

[75] Jessica Lin, Eamonn Keogh, Stefano Lonardi, and Bill Chiu. "A Symbolic Representation of Time Series, with Implications for Streaming Algorithms." In: *Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*. DMKD '03. San Diego, California: ACM, 2003, pp. 2–11. ISBN: 978-1-4503-7422-4. DOI: 10.1145/882082.882086. URL: http://doi.acm.org/10.1145/882082.882086.

[76] Jessica Lin, Eamonn Keogh, Stefano Lonardi, Jeffrey P. Lankford, and Donna M. Nystrom. "Visually Mining and Monitoring Massive Time Series." In: *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '04. Seattle, WA, USA: ACM, 2004, pp. 460–469. ISBN: 1-58113-888-1. DOI: 10.1145/1014052.1014104. URL: http://doi.acm.org/10.1145/1014052.1014104.

[77] Michele Linardi and Themis Palpanas. "Scalable, Variable-Length Similarity Search in Data Series: The ULISSE Approach." In: *PVLDB* (2019).

[78] Erik K. St. Louis and Lauren C. Frey. *Electroencephalography (EEG): An Introductory Text and Atlas of Normal and Abnormal Findings in Adults, Children, and Infants*. American Epilepsy Society, 2016. ISBN: 978-0-9979756-0-4. eprint: https://www.ncbi.nlm.nih.gov/books/NBK390354/pdf/Bookshelf_NBK390354.pdf. URL: https://www.ncbi.nlm.nih.gov/books/NBK390343/.

[79] Miro Mannino and Azza Abouzied. "Expressive Time Series Querying with Hand-Drawn Scale-Free Sketches." In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. CHI '18. Montreal QC, Canada: ACM, 2018, 388:1–388:13. ISBN: 978-1-4503-5620-6. DOI: 10.1145/3173574.3173962. URL: http://doi.acm.org/10.1145/3173574.3173962.

[80] Peter McLachlan, Tamara Munzner, Eleftherios Koutsofios, and Stephen North. "LiveRAC: Interactive Visual Exploration of System Management Time-series Data." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '08. Florence, Italy: ACM, 2008, pp. 1483–1492. ISBN: 978-1-60558-011-1. DOI: 10.1145/1357054.1357286. URL: http://doi.acm.org/10.1145/1357054.1357286.

[81] Luana Micallef, Hans-Jörg Schulz, Marco Angelini, Michaël Aupetit, Remco Chang, Jörn Kohlhammer, Adam Perer, and Giuseppe Santucci. "The Human User in Progressive Visual Analytics." In: *Short Paper Proceedings of EuroVis'19*. Eurographics Association, 2019, pp. 19–23. DOI: 10.2312/evs.20191164.

[82] Dominik Moritz, Bill Howe, and Jeffrey Heer. "Falcon: Balancing Interactive Latency and Resolution Sensitivity for Scalable Linked Visualizations." In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. CHI '19. Glasgow, Scotland Uk: ACM, 2019, 694:1–694:11.

ISBN: 978-1-4503-5970-2. DOI: 10.1145/3290605.3300924. URL: http://doi.acm.org/10.1145/3290605.3300924.

[83] Dominik Moritz, Danyel Fisher, Bolin Ding, and Chi Wang. "Trust, but Verify: Optimistic Visualizations of Approximate Queries for Exploring Big Data." In: *CHI*. 2017.

[84] Wolfgang Müller and Heidrun Schumann. "Visualization for Modeling and Simulation: Visualization Methods for Time-dependent Data - an Overview." In: *Proceedings of the 35th Conference on Winter Simulation: Driving Innovation*. WSC '03. New Orleans, Louisiana: Winter Simulation Conference, 2003, pp. 737–745. ISBN: 0-7803-8132-7. URL: http://dl.acm.org/citation.cfm?id=1030818.1030916.

[85] P. K. Muthumanickam, K. Vrotsou, M. Cooper, and J. Johansson. "Shape grammar extraction for efficient query-by-sketch pattern matching in long time series." In: *2016 IEEE Conference on Visual Analytics Science and Technology (VAST)*. 2016, pp. 121–130. DOI: 10.1109/VAST.2016.7883518.

[86] Daniele Nadalutti and Luca Chittaro. "Visual analysis of users' performance data in fitness activities." In: *Computers & Graphics* 31.3 (2007), pp. 429 –439. ISSN: 0097-8493. DOI: https://doi.org/10.1016/j.cag.2007.01.032. URL: http://www.sciencedirect.com/science/article/pii/S0097849307000635.

[87] Jakob Nielsen. *Usability Engineering*. Academic Press Limited, 1993. ISBN: 0125184069.

[88] San juan Orta D, Chiappa KH, Quiroz AZ, Costello DJ, and Cole AJ. "Prognostic implications of periodic epileptiform discharges." In: *Archives of Neurology* 66.8 (2009), pp. 985–991. DOI: 10.1001/archneurol.2009.137. eprint: /data/journals/neur/7763/noc90022_985_991.pdf. URL: http://dx.doi.org/10.1001/archneurol.2009.137.

[89] Themis Palpanas. "Data Series Management: The Road to Big Sequence Analytics." In: *SIGMOD Record* 44.2 (2015), pp. 47–52. DOI: 10.1145/2814710.2814719. URL: http://doi.acm.org/10.1145/2814710.2814719.

[90] Themis Palpanas. "Big Sequence Management: A Glimpse on the Past, the Present, and the Future." In: *Lecture Notes on Computer Science (LNCS)* 9587 (2016).

[91] Themis Palpanas and Volker Beckmann. "Report on the First and Second Interdisciplinary Time Series Analysis Workshop (ITISA)." In: *ACM SIGMOD Record, accepted for publication, 2019*. 2019.

[92] Botao Peng, Themis Palpanas, and Panagiota Fatourou. "ParIS: The Next Destination for Fast Data Series Indexing and Query Answering." In: *IEEE BigData* (2018).

[93] Charles Perin, Frédéric Vernier, and Jean-Daniel Fekete. "Interactive Horizon Graphs: Improving the Compact Visualization of Multiple Time Series." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '13. Paris, France: ACM, 2013, pp. 3217–3226. ISBN: 978-1-4503-1899-0. DOI: 10.1145/2470654.2466441. URL: http://doi.acm.org/10.1145/2470654.2466441.

[94] Nathaniel Phillips. *A Companion to the e-Book "YaRrr!: The Pirate's Guide to R"*. https://github.com/ndphillips/yarrr. 2017.

[95] Stuart J. Pocock. "Group Sequential Methods in the Design and Analysis of Clinical Trials." In: *Biometrika* 64.2 (1977), pp. 191–199. ISSN: 00063444. URL: http://www.jstor.org/stable/2335684.

[96] Sajjadur Rahman, Maryam Aliakbarpour, Hidy Kong, Eric Blais, Karrie Karahalios, Aditya G. Parameswaran, and Ronitt Rubinfeld. "I've Seen "Enough": Incrementally Improving Visualizations to Support Rapid Decision Making." In: *PVLDB* 10.11 (2017), pp. 1262–1273.

[97] Thanawin Rakthanmanon, Bilson Campana, Abdullah Mueen, Gustavo Batista, Brandon Westover, Qiang Zhu, Jesin Zakaria, and Eamonn Keogh. "Searching and Mining Trillions of Time Series Subsequences Under Dynamic Time Warping." In: *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '12. Beijing, China: ACM, 2012, pp. 262–270. ISBN: 978-1-4503-1462-6. DOI: 10.1145/2339530.2339576. URL: http://doi.acm.org/10.1145/2339530.2339576.

[98] Chotirat Ann Ratanamahatana and Eamonn Keogh. "Everything you know about dynamic time warping is wrong." In: *Third Workshop on Mining Temporal and Sequential Data*. Citeseer. 2004.

[99] Hannes Reijner. "The development of the horizon graph." available online at http://www.stonesc.com/Vis08_Workshop/DVD/Reijner_submission.pdf. 2008.

[100] Hernan Gonzalo Rey, Carlos Pedreira, and Rodrigo Quian Quiroga. "Past, present and future of spike sorting techniques." In: *Brain Research Bulletin* 119.Pt B (2015), pp. 106–117. DOI: 10.1016/j.brainresbull.2015.04.007.

[101] Kathy Ryall, Neal Lesh, Tom Lanning, Darren Leigh, Hiroaki Miyashita, and Shigeru Makino. "QueryLines: Approximate Query for Visual Browsing." In: *CHI '05 Extended Abstracts on Human Factors in Computing Systems*. CHI EA '05. Portland, OR, USA: ACM, 2005, pp. 1765–1768. ISBN: 1-59593-002-7. DOI: 10.1145/1056808.1057017. URL: http://doi.acm.org/10.1145/1056808.1057017.

[102] Takafumi Saito, Hiroko Nakamura Miyamura, Mitsuyoshi Yamamoto, Hiroki Saito, Yuka Hoshiya, and Takumi Kaseda. "Two-Tone Pseudo Coloring: Compact Visualization for One-Dimensional Data." In: *Proceedings of the Proceedings of the 2005 IEEE Symposium on Information Visualization*. INFOVIS '05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 23–. ISBN: 0-7803-9464-x. DOI: 10.1109/INFOVIS.2005.35. URL: https://doi.org/10.1109/INFOVIS.2005.35.

[103] Hans-Jörg Schulz, Marco Angelini, Giuseppe Santucci, and H Schumann. "An Enhanced Visualization Process Model for Incremental Visualization." In: *IEEE Transactions on Visualization and Computer Graphics* 22 (July 2016), pp. 1830–1842. DOI: 10.1109/TVCG.2015.2462356.

[104] Incorporated Research Institutions for Seismology. *IRIS Seismic Data Access*. http://ds.iris.edu/data/access/. 2014.

[105] Tarique Siddiqui, Albert Kim, John Lee, Karrie Karahalios, and Aditya Parameswaran. "Effortless Data Exploration with Zenvisage: An Expressive and Interactive Visual Analytics System." In: *Proc. VLDB Endow.* 10.4 (2016), pp. 457–468. ISSN: 2150-8097. DOI: 10.14778/3025111.3025126. URL: https://doi.org/10.14778/3025111.3025126.

[106] Robert L. Spitzer and Joseph L. Fleiss. "A Re-analysis of the Reliability of Psychiatric Diagnosis." In: *The British Journal of Psychiatry* 125.587 (1974), pp. 341–347. ISSN: 0007-1250. DOI: 10.1192/bjp.125.4.341. eprint: http://bjp.rcpsych.org/content/125/587/341.full.pdf. URL: http://bjp.rcpsych.org/content/125/587/341.

[107] Kevin J Staley and F Edward Dudek. "Interictal Spikes and Epileptogenesis." In: *Epilepsy Currents 6.6* (2006), pp. 199–202. DOI: 10.1111/j.1535-7511.2006.00145.x. eprint: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1783497/pdf/epc0006-0199.pdf. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1783497/.

[108] Kevin J Staley, Andrew White, and F Edward Dudek. "Interictal Spikes: Harbingers or Causes of Epilepsy?" In: *Neuroscience letters 497.3* (2011), pp. 247–250. DOI: 10.1016/j.neulet.2011.03.070. eprint: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3124147/pdf/nihms285730.pdf. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3124147/.

[109] Charles D. Stolper, Adam Perer, and David Gotz. "Progressive Visual Analytics: User-Driven Visual Exploration of In-Progress Analytics." In: *IEEE TVCG* 20 (2014).

[110] M. Stone. "In Color Perception, Size Matters." In: *IEEE Computer Graphics and Applications* 32.2 (2012), pp. 8–13. ISSN: 0272-1716. DOI: 10.1109/MCG.2012.37.

[111] Bruce Swihart, Brian Caffo, Bryan James, Matthew Strand, Brian Schwartz, and Naresh Punjabi. "Lasagna plots: a saucy alternative to spaghetti plots." In: *Epidemiology (Cambridge, Mass.)* 21.5 (2010), pp. 621–625. DOI: http://doi.org/10.1097/EDE.0b013e3181e5b06a.

[112] Iván Sánchez Fernández, Tobias Loddenkemper, Aristea S. Galanopoulou, and Solomon L. Moshé. "Should epileptiform discharges be treated?" In: *Epilepsia* 56.10 (2015), pp. 1492–1504. ISSN: 1528-1167. DOI: 10.1111/epi.13108. URL: http://dx.doi.org/10.1111/epi.13108.

[113] J. Talbot, J. Gerth, and P. Hanrahan. "An Empirical Model of Slope Ratio Comparisons." In: *IEEE Transactions on Visualization and Computer Graphics* 18.12 (2012), pp. 2613–2620. ISSN: 1077-2626. DOI: 10.1109/TVCG.2012.196.

[114] *The ICM Brain and Spine Institute*. https://icm-institute.org/en/.

[115] Christian Tominski and Heidrun Schumann. "Enhanced Interactive Spiral Display." In: 2008.

[116] Theophanis Tsandilas. "Fallacies of Agreement: A Critical Review of Consensus Assessment Methods for Gesture Elicitation." In: *ACM Transactions on Computer-Human Interaction (TOCHI)* 25.3 (June 2018), 18:1–18:49. ISSN: 1073-0516. DOI: 10.1145/3182168. URL: http://doi.acm.org/10.1145/3182168.

[117] Edward R. Tufte. *The Visual Display of Quantitative Information*. 1986.

[118] Cagatay Turkay, Erdem Kaya, Selim Balcisoy, and Helwig Hauser. "Designing Progressive and Interactive Analytics Processes for High-Dimensional Data Analysis." In: *IEEE TVCG* 23.1 (2017).

[119] Southwest University. *Southwest University Adult Lifespan Dataset (SALD)*. http://fcon_1000.projects.nitrc.org/indi/retro/sald.html. 2017.

[120] Jarke J. Van Wijk and Edward R. Van Selow. "Cluster and Calendar Based Visualization of Time Series Data." In: *Proceedings of the 1999 IEEE Symposium on Information Visualization*. INFOVIS '99. Washington, DC, USA: IEEE Computer Society, 1999, pp. 4–. ISBN: 0-7695-0431-0. DOI: 10.1109/INFVIS.1999.801851. URL: http://dl.acm.org/citation.cfm?id=857189.857665.

[121] Skoltech Computer Vision. *Deep billion-scale indexing*. http://sites.skoltech.ru/compvision/noimi. 2018.

[122] Abraham Wald. "Sequential Tests of Statistical Hypotheses." In: *The Annals of Mathematical Statistics* 16.2 (June 1945), pp. 117–186. DOI: 10.1214/aoms/1177731118. URL: https://doi.org/10.1214/aoms/1177731118.

[123] Matt P. Wand and Michael C. Jones. "Comparison of Smoothing Parameterizations in Bivariate Kernel Density Estimation." In: *Journal of the American Statistical Association* 88.422 (1993), pp. 520–528. DOI: 10.1080/01621459.1993.10476303.

[124] Matt P. Wand and Michael C. Jones. "Multivariate Plug-in Bandwidth Selection." In: *Computational Statistics* 9.2 (1994), pp. 97–116. URL: http://oro.open.ac.uk/28244/.

[125] Yang Wang, Peng Wang, Jian Pei, Wei Wang, and Sheng Huang. "A Data-adaptive and Dynamic Segmentation Index for Whole Matching on Time Series." In: *Proc. VLDB Endow.* 6.10 (2013).

[126] Martin Wattenberg. "Sketching a Graph to Query a Time-series Database." In: *CHI '01 Extended Abstracts on Human Factors in Computing Systems*. CHI EA '01. Seattle, Washington: ACM, 2001, pp. 381–382. ISBN: 1-58113-340-5. DOI: 10.1145/634067.634292. URL: http://doi.acm.org/10.1145/634067.634292.

[127] Marc Weber, Marc Alexa, and Wolfgang Müller. "Visualizing Time-Series on Spirals." In: *Proceedings of the IEEE Symposium on Information Visualization 2001 (INFOVIS'01)*. INFOVIS '01. Washington, DC, USA: IEEE Computer Society, 2001, pp. 7–. ISBN: 0-7695-1342-5. DOI: 10.1109/INFVIS.2001.963273. URL: http://dl.acm.org/citation.cfm?id=580582.857719.

[128] Sai Wu, Beng Chin Ooi, and Kian-Lee Tan. "Online Aggregation." In: *Advanced Query Processing, Volume 1: Issues and Trends*. 2013, pp. 187–210.

[129] Djamel-Edine Yagoubi, Reza Akbarinia, Florent Masseglia, and Themis Palpanas. "Massively Distributed Time Series Indexing and Querying." In: *TKDE* (2018).

[130] Emanuel Zgraggen, Alex Galakatos, Andrew Crotty, Jean-Daniel Fekete, and Tim Kraska. "How Progressive Visualizations Affect Exploratory Analysis." In: *IEEE TVCG* 23 (2017).

[131] Emanuel Zgraggen, Zheguang Zhao, Robert C. Zeleznik, and Tim Kraska. "Investigating the Effect of the Multiple Comparisons Problem in Visual Analysis." In: *CHI*. 2018.

[132] Jian Zhao, Fanny Chevalier, and Ravin Balakrishnan. "KronoMiner: Using Multi-foci Navigation for the Visual Exploration of Time-series Data." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '11. Vancouver, BC, Canada: ACM, 2011, pp. 1737–1746. ISBN: 978-1-4503-0228-9. DOI: 10.1145/1978942.1979195. URL: http://doi.acm.org/10.1145/1978942.1979195.

[133] Jian Zhao, Fanny Chevalier, Emmanuel Pietriga, and Ravin Balakrishnan. "Exploratory Analysis of Time-Series with ChronoLenses." In: *IEEE Transactions on Visualization and Computer Graphics* 17.12 (Dec. 2011), pp. 2422–2431. ISSN: 1077-2626. DOI: 10.1109/TVCG.2011.195. URL: http://dx.doi.org/10.1109/TVCG.2011.195.

[134] Kostas Zoumpatianos, Stratos Idreos, and Themis Palpanas. "RINSE: Interactive Data Series Exploration with ADS+." In: *PVLDB* 8.12 (2015), pp. 1912–1915. DOI: 10.14778/2824032.2824099. URL: http://www.vldb.org/pvldb/vol8/p1912-zoumpatianos.pdf.

[135] Kostas Zoumpatianos, Stratos Idreos, and Themis Palpanas. "ADS: the adaptive data series index." In: *VLDB J.* 25.6 (2016), pp. 843–866. DOI: 10.1007/s00778-016-0442-5. URL: https://doi.org/10.1007/s00778-016-0442-5.

[136] Kostas Zoumpatianos, Yin Lou, Themis Palpanas, and Johannes Gehrke. "Query Workloads for Data Series Indexes." In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*. 2015, pp. 1603–1612. DOI: 10.1145/2783258.2783382. URL: http://doi.acm.org/10.1145/2783258.2783382.

A

APPENDIX

background information Participant Code:

1. How often do analyze data series as part of your work?

[ ] Never

[ ] A few times per year

[ ] Monthly (1 - 2 times per month)

[ ] Weekly (1 - 2 times per week)

[X] Daily

2. What do you use data series for (e.g., for what kind of tasks)?

Gravitational-wave detection and analysis. Including analysis of spurious, noisy artifacts in the data series. I work as part of an international collaboration: LIGO-Virgo Collaboration.

3. Do you ever visually explore your data series data? If yes, explain why.

Yes, however most explorations use Fourier domain analysis, spectrograms and periodograms, and filtering techniques developed specifically for gravitational-wave signals. The "raw" data series are usually not very useful, with a noise floor several orders of magnitude higher than the signal(s) of interest.

4. Can you briefly describe any limitations in your current visualization tools?

Lack of interactivity. We are trying to develop better tools (based on D3 for instance), but this work is rarely our top priority. We tend to either look at pre-generated, static representation of the data, or use off-the-shelf visualisation included in scientific softwares (matplotlib, ipython, matematica, mathlab, ...).

5. Would you be interested in trying new visual exploration tools?

Absolutely !

Part 1: scenarios Participant Code :

Scenario 1:

1.1 Briefly describe your data (what they represent, their dimensions, their size, etc.)

The unprocessed data represents deformation of the fabric of space-time, or "strain". The units are ratio of length (dimensionless) over units of time. The raw data out of the detector is in photon count as a function of time, and gets calibrated into dimensionless "strain data". It is usually sampled at 16kHz, from a few detectors, and is separated in several month-long "observing runs." We also analyse the output of noise-free simulations: in this case, the data is just the pre-computed, predicted deformation of space-time.

1.2 What is your specific goal? (e.g., what are you looking for in these data?)

We are looking for Gravitational-wave signals. Specific patterns in the strain created by the motion of very dense (for instance, black-holes) astrophysical objects. In the case of simulation analysis, we are looking to test our tools and asses their performances.

1.3 What tools do you use to accomplish this goal?

Matched filtering, Fourier transforms, wavelets transforms and transformations from time-domain or frequency-domain to manifolds of expected signals. Those transformations are specific to gravitational-waves, and can be computationally expensive, requiring other tools to optimise the analysis.

1.4 How long does this typically take?

From raw data to full analysis can take minutes (for the fastest algorithms making strong simplifying assumptions for the sake of speed), to months (for the slowest, most complex analyses).

Scenario 2:

2.1 Briefly describe your data (what they represent, their dimensions, their size, etc.)

In addition to the "strain data" mentioned in scenario 1, various "environmental channels" record data series on the behaviour of the detectors and the environment. For instance seismometers, magnetometers, microphones, etc … Some are sampled at 16kHz, other at lower rates, and record seismic vibrations, magnetic field fluctuations, noise levels, … They span the same range in time as the "strain data" from scenario 1.

2.2 What is your specific goal? (e.g., what are you looking for in these data?)

The goal is to find correlations between those "environmental channels" and the main, "strain channel". Any such correlation is a hint that the environment is contaminating the data, and the data should not be trusted.

Part 1: scenarios Participant Code :

2.3 What tools do you use to accomplish this goal?

We use software developed in-house to compute cross-correlations between all pairs of channels of interest. We then flag time-frequency spaces of high correlation values for further analysis with all the tools of scenario 1. Classification algorithms are used to build a database of noisy environmental features, which is then use to improve the detector and the data analysis.

2.4 How long does this typically take?

We limit the number of computation so that the results can appear in a few hours. This allows for daily analysis of on-going observations.

Part 2: detailed example Participant Code:

A. YOUR DATA

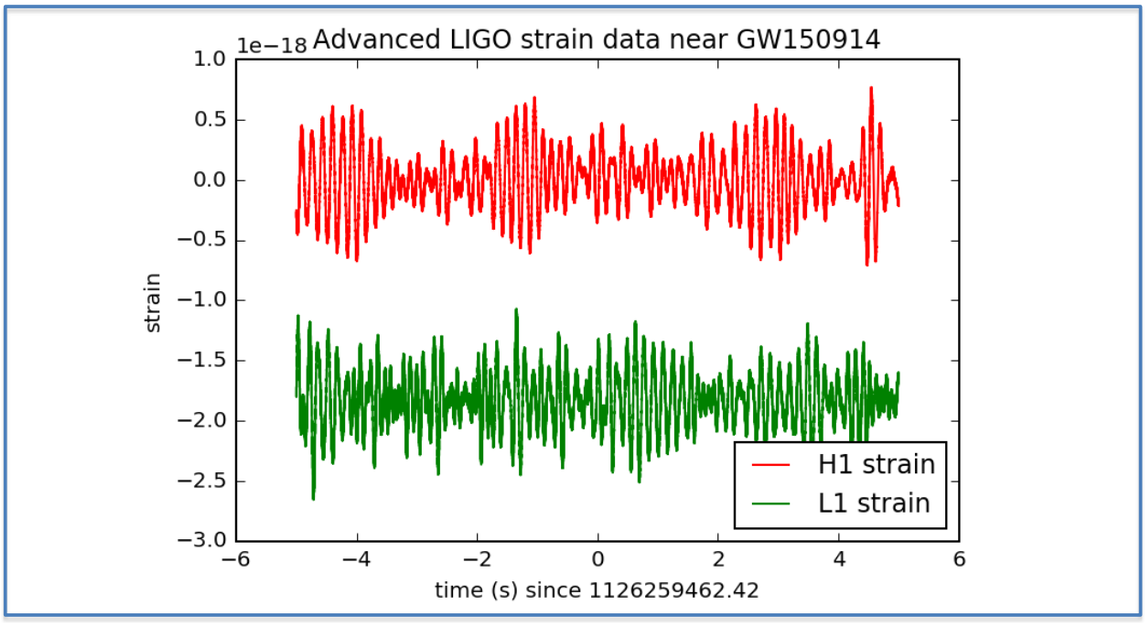
Present one or two examples of data series from your work:

A1. Sketch your data (dimensions, scale, representative patterns and values).

A2. Annotate interesting aspects, e.g., patterns of particular interest.

A3. Explain why they are interesting or important for you.

Example 1

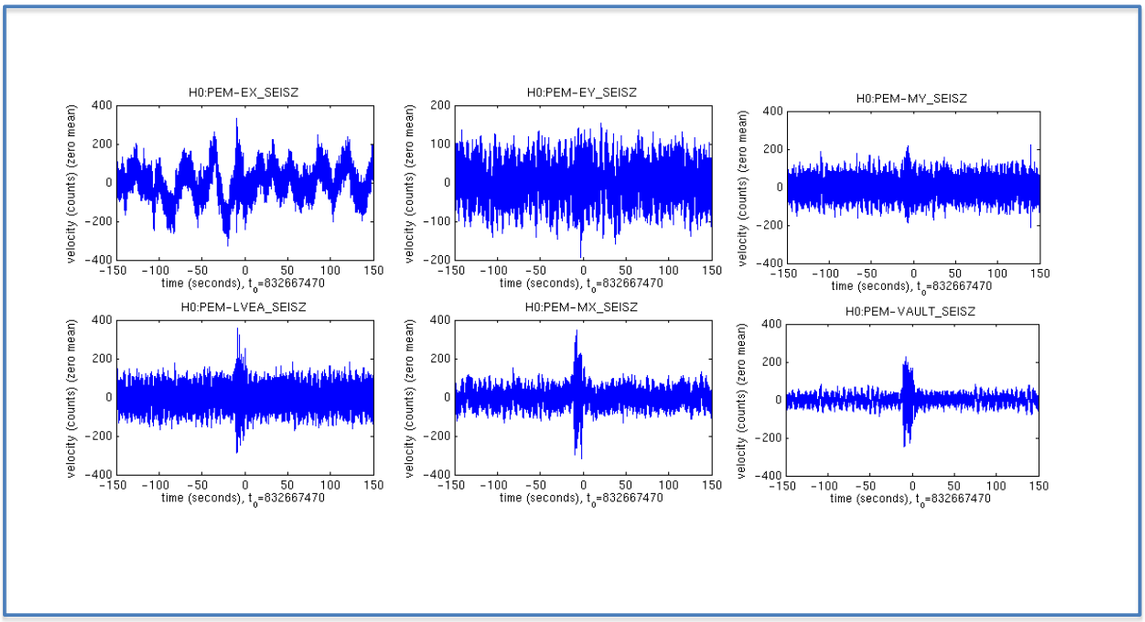


Annotations (or explanations):

Taken from losc.ligo.org. "H1" and "L1" two different detectors, the dimensionless "strain" data from both is plotted as a function of time in seconds. This particular time-series has been filtered to remove the noisy low and high frequency content. The pattern of interest (the signal in this case) is still not visible on the plot, being ~3 orders of magnitude smaller that either traces.

Part 2: detailed example Participant Code:

Example 2



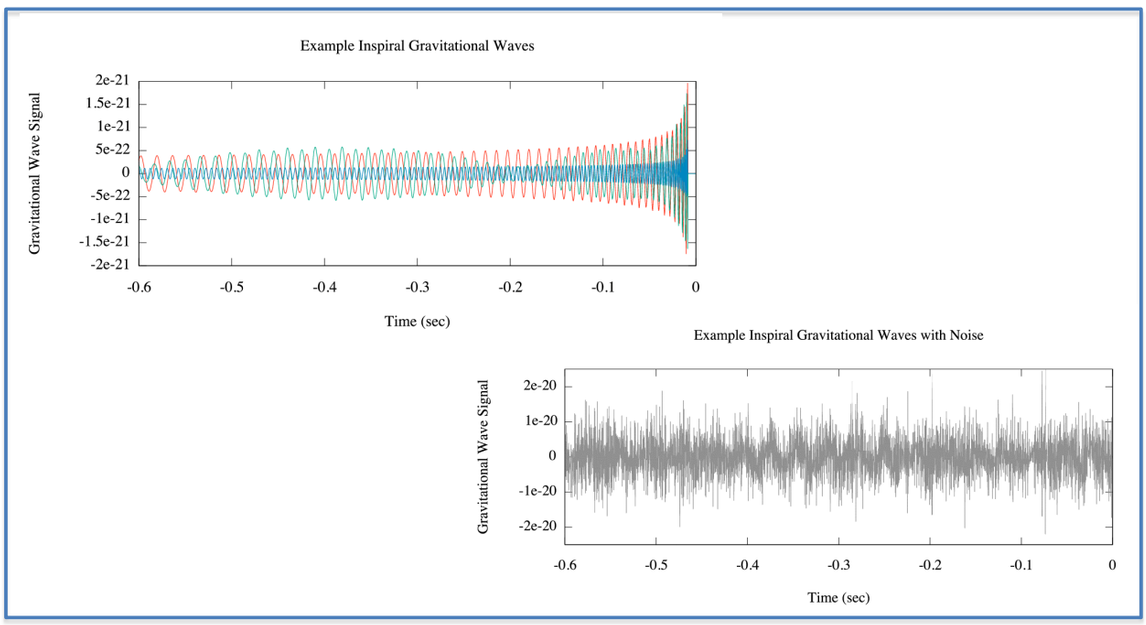
Annotations (or explanations):

Taken from http://www.pas.rochester.edu/. The time series output of seismometer channels are shown here. There is at time 0 an excitation of several channels. If the main "gravitational-wave strain" data from the detector shows some signal at the same time, we would suspect a contamination of the data due to seismic motion.

Part 2: detailed example Participant Code:

B. YOUR QUESTION

Describe a specific question that you would like to ask. Can you express it visually, by drawing it? (If not, please explain.)



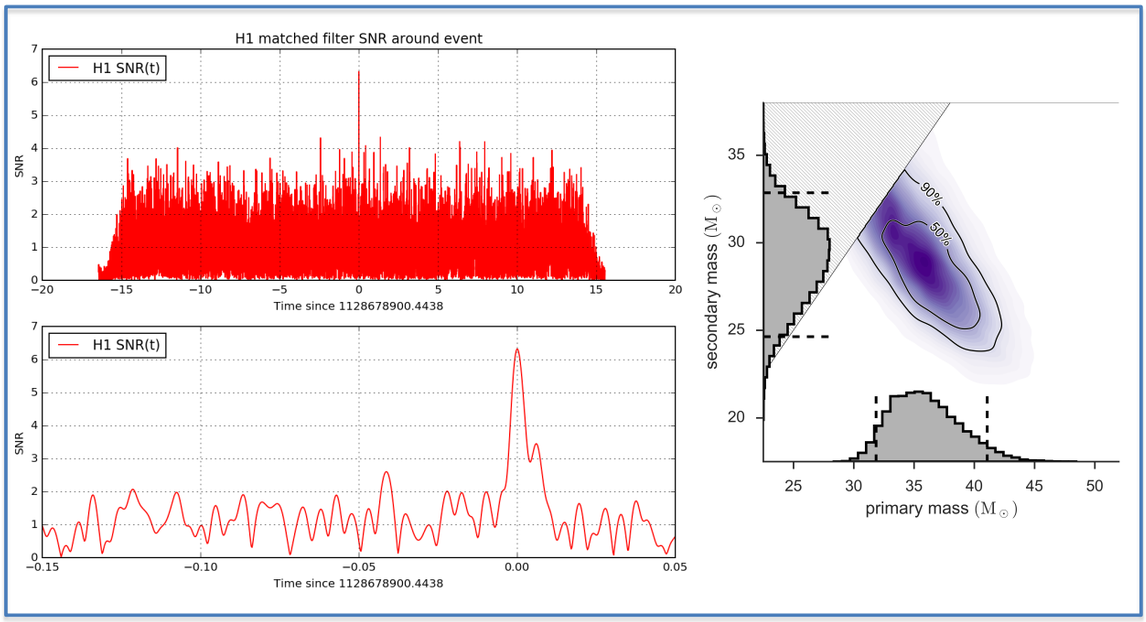
Annotations (or explanations):

Taken from ligo.org. The top-left plot shows 3 models of simulated gravitational-wave signals. The bottom-right plot shows a stretch of data from the detector. We want to know if, (and which) signal is present in the data, with which probability, and for which parameters.

Part 2: detailed example Participant Code:

C. YOUR EXPECTED RESULTS

Give an example of representative results to your question.



Annotations (or explanations):

Taken from losc.ligo.org (left) and ligo.org (right). The left plot shows the correlation between a model signal and a stretch of data. At time 0, the correlation function peaks, showing that we have a signal, with the height a function of the strength of the signal. The right plot shows some of the recovered parameters (here the masses of the 2 objects which created the signal). By cross-correlating many simulated signals with the data, we obtain the probability of each parameter.

Part 2: detailed example Participant Code:

D. WAITING FOR YOUR RESULTS

D1. How long would you be willing to wait for such results?

While faster is almost always better, we can wait up to several months for full results.

D2. Would you be interested in working with quick but rough results (e.g., approximations or incomplete answers), while you wait for the complete, more accurate ones? Explain.

Yes. There are use-cases for approximate results in minutes to inform other observatories of the presence of a signal. The eventual full analysis can happen without such time-constraint.

D3. If you answered yes (in D2), what does *"roughness"* mean for you? (e.g., error bounds, uncertainty measure, portion of results?) Could you express this roughness visually with an example?

Some parameters can be uncertain in the "*rough*" results, while others, such as the position in the sky of the signal, or the peak time of the cross-correlation (which is used to measure the time of arrival) need to be within certain bounds. Those bounds in turn depend on the type of observatory which will try to follow-up on the signal.

COLOPHON

This document was typeset in LaTeX using the typographical look-and-feel `classicthesis`. The graphics were generated using the `R` statistical computing environment. The bibliography is typeset using `biblatex`.

université PARIS-SACLAY
ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre : Requêtes itératives et expressives pour l'analyse de grandes séries de données

Mots clés : séries temporelles, perception de similarité, recherche de similarité progressive

Résumé : Les séries temporelles deviennent omniprésentes dans la vie moderne et leur analyse de plus en plus difficile compte tenu de leur taille. L'analyse des grandes séries de données implique des tâches telles que l'appariement de modèles (motifs), la détection d'anomalies, l'identification de modèles fréquents, et la classification ou le regroupement (clustering). Ces tâches reposent sur la notion de similarité. La communauté scientifique a proposé de plusieurs techniques, y compris de nombreuses mesures de similarité pour calculer la distance entre deux séries temporelles, ainsi que des techniques et des algorithmes d'indexation, afin de relever les défis de l'évolutivité lors de la recherche de similarité.

Les analystes, afin de s'acquitter efficacement de leurs tâches, ont besoin de systèmes d'analyse visuelle interactifs, extrêmement rapides, et puissants. Lors de la création de tels systèmes, nous avons identifié deux principaux défis: (1) la perception de similarité et (2) la recherche progressive de similarité. Le premier traite de la façon dont les gens perçoivent des modèles similaires et du rôle de la visualisation dans la perception de similarité. Le dernier point concerne la rapidité avec laquelle nous pouvons redonner aux utilisateurs des mises à jour des résultats progressifs, lorsque les temps de réponse du système sont longs et non interactifs. Le but de cette thèse est de répondre et de donner des solutions aux défis ci-dessus.

Dans la première partie, nous avons étudié si différentes représentations visuelles (Graphiques en courbes, Graphiques d'horizon et Champs de couleur) modifiaient la perception de similarité des séries temporelles. Nous avons essayé de comprendre si les résultats de recherche automatique de similarité sont perçus de manière similaire, quelle que soit la technique de visualisation; et si ce que les gens perçoivent comme similaire avec chaque visualisation s'aligne avec différentes mesures de similarité. Nos résultats indiquent que les Graphes d'horizon s'alignent sur des mesures qui permettent des variations de décalage temporel ou d'échelle (i.e., ils promeuvent la déformation temporelle dynamique). En revanche, ils ne s'alignent pas sur des mesures autorisant des variations d'amplitude et de décalage vertical (ils ne promeuvent pas des mesures basées sur la z-normalisation). L'inverse semble être le cas pour les Graphiques en courbes et les Champs de couleur. Dans l'ensemble, nos travaux indiquent que le choix de la visualisation affecte les schémas temporels que l'homme considère comme similaires. Donc, la notion de similarité dans les séries temporelles est dépendante de la technique de visualisation.

Dans la deuxième partie, nous nous sommes concentrés sur la recherche progressive de similarité dans de grandes séries de données. Nous avons étudié la rapidité avec laquelle les premières réponses approximatives et puis des mises à jour des résultats progressifs sont détectées lors de l'exécuton des requêtes progressives. Nos résultats indiquent qu'il existe un écart entre le moment où la réponse finale s'est trouvée et le moment où l'algorithme de recherche se termine, ce qui entraîne des temps d'attente gonflés sans amélioration. Des estimations probabilistes pourraient aider les utilisateurs à décider quand arrêter le processus de recherche, i.e., quand l'amélioration de la réponse finale est improbable. Nous avons développé et évalué expérimentalement une nouvelle méthode probabiliste qui calcule les garanties de qualité des résultats progressifs de k-plus proches voisins (k-NN). Notre approche apprend d'un ensemble de requêtes et construit des modèles de prédiction basés sur deux observations: (i) des requêtes similaires ont des réponses similaires; et (ii) des réponses progressives renvoyées par les indices de séries de données sont de bons prédicteurs de la réponse finale. Nous fournissons des estimations initiales et progressives de la réponse finale.

Title : Iterative and Expressive Querying for Big Data Series

Keywords : time series, similarity perception, progressive similarity search

Abstract : Time series are becoming ubiquitous in modern life, and given their sizes, their analysis is becoming increasingly challenging. Time series analysis involves tasks such as pattern matching, anomaly detection, frequent pattern identification, and time series clustering or classification. These tasks rely on the notion of time series similarity. The data-mining community has proposed several techniques, including many similarity measures (or distance measure algorithms), for calculating the distance between two time series, as well as corresponding indexing techniques and algorithms, in order to address the scalability challenges during similarity search.

To effectively support their tasks, analysts need interactive visual analytics systems that combine extremely fast computation, expressive querying interfaces, and powerful visualization tools. We identified two main challenges when considering the creation of such systems: (1) similarity perception and (2) progressive similarity search. The former deals with how people perceive similar patterns and what the role of visualization is in time series similarity perception. The latter is about how fast we can give back to users updates of progressive similarity search results and how good they are, when system response times are long and do not support real-time analytics in large data series collections. The goal of this thesis, that lies at the intersection of Databases and Visualization/Human-Computer Interaction, is to answer and give solutions to the above challenges.

In the first part of the thesis, we studied whether different visual representations (Line Charts, Horizon Graphs, and Color Fields) alter time series similarity perception. We tried to understand if automatic similarity search results are perceived in a similar manner, irrespective of the visualization technique; and if what people perceive as similar with each visualization aligns with different automatic similarity measures and their similarity constraints. Our findings indicate that Horizon Graphs promote as invariant local variations in temporal position or speed, and as a result they align with measures that allow variations in temporal shifting or scaling (i.e., dynamic time warping). On the other hand, Horizon Graphs do not align with measures that allow amplitude and y-offset variations (i.e., measures based on z-normalization), because they exaggerate these differences, while the inverse seems to be the case for Line Charts and Color Fields. Overall, our work indicates that the choice of visualization affects what temporal patterns humans consider as similar, i.e., the notion of similarity in time series is visualization-dependent.

In the second part of the thesis, we focused on progressive similarity search in large data series collections. We investigated how fast first approximate and then updates of progressive answers are detected, while we execute similarity search queries. Our findings indicate that there is a gap between the time the final answer (best answer) is found, and the time when the search algorithm terminates, resulting in inflated waiting times without any improvement. Computing probabilistic estimates of the final answer could help users decide when to stop the search process. We developed and experimentally evaluated using benchmarks, a new probabilistic learning-based method that computes quality guarantees (error bounds) for progressive k-Nearest Neighbour (k-NN) similarity search results. Our approach learns from a set of queries and builds prediction models based on two observations: (i) similar queries have similar answers; and (ii) progressive best-so-far (bsf) answers returned by the state-of-the-art data series indexes are good predictors of the final k-NN answer. We provide both initial and incrementally improved estimates of the final answer.

Université Paris-Saclay
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France