Journal Title
XX(X):1–13

Reprints and permission:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Scalable Visual Exploration of Time Series and Anomalies With Adaptive Level of Detail

Julian Rakuschek[1], Josef Suschnigg[2], Patrick Louis[2], Belgin Mutlu[2] and Tobias Schreck[1]

Abstract
Time series datasets often contain numerous time series or segments that cannot be effectively displayed on a single screen, leading to information overload. This work addresses this challenge by integrating hierarchical navigation and semantic zooming techniques to enhance dataset exploration without overwhelming users. Our approach emphasizes scalability through hierarchical clustering, which allows efficient navigation of pattern groups. To inspect a specific pattern group, we employ a semantic zooming technique that combines three complementary visualizations: line charts, horizon graphs, and heatmaps. While our approach is generic, we also tailor it to support visual analysis of anomalies in time series, an important domain benefiting in particular from semantic zoom. We provide a visual anomaly score indicator that summarizes anomaly locations within clusters of time series. Additionally, a glyph visualizes each algorithm's sensitivity for each cluster and time series. The practical value of our unique approach is validated through an ICE-T user study and several case studies, demonstrating its effectiveness for time series data exploration and analysis.

Keywords
Visual Analytics, Time Series, Clustering, Semantic Zoom, Anomaly Detection

Introduction

Datasets in the form of time series are ubiquitous and arise in many research domains and industrial settings, with line charts being the most common and traditional visualization technique since the dawn of information visualization.[2] Although in simple cases, users tend to be satisfied with a single line chart generated through spreadsheet tools, the visualization task becomes increasingly difficult as the dataset size increases.[25] Large datasets are prevalent across various domains, including finance, sensor measurements, manufacturing, and health monitoring. It can be cumbersome for data science practitioners to explore each dataset manually in order to gain an overview of the patterns within the archive, especially since visualizing over one thousand time series is infeasible on a single screen without compromising a significant amount of detail. The well-known University of California Riverside (UCR) archive[9] is an example of a dataset particularly concerned with the task of anomaly detection in time series, where each time series contains exactly one non-trivial anomaly, which is a necessary design criterion.[31] It is of particular interest for users to explore not only the time series themselves but also the results of anomaly detection algorithms. This necessitates a visual exploration workflow to assist not only data scientists but also domain experts in gaining an overview of patterns in time series datasets, together with the corresponding algorithm performance.

To address this challenge, we propose a visualization tool that provides a comprehensive overview of a large time series dataset by arranging groups of similar time series in a grid pattern. In large grids, we offer users a semantic zooming feature, such that more details are revealed as users zoom in on clusters and individual time series. Users may switch between different visualization techniques (introduced in Section Proposed Visualization Design) at any point, enabling the consideration of multiple perspectives. Finally, each cluster and time series includes an aggregated anomaly score to highlight the results of anomaly detection algorithms. The resulting visualization tool follows several requirements defined as follows:

R1: **Time series visualization:** Given a large number of time series, the tool should visualize all of them and ensure they are easily perceivable by the user. It should be possible to view all time series simultaneously at a low level of detail, and to examine individual time series at a high level of detail.

R2: **Multiple perspectives:** Given the adaptive level of detail from the previous requirement, users should have the option to switch between visualization types, thereby providing alternatives to line charts.

R3: **Clustering similar time series:** To save valuable screen space, the tool should group similar time series together, allowing users to inspect individual clusters on demand.

R4: **Anomaly detection:** The tool should indicate where anomalies occur and provide the user with an indicator

[1]Institute of Visual Computing, Graz University of Technology, Austria
[2]Pro2Future GmbH, Graz, Austria

Corresponding author:
Julian Rakuschek, Institute of Visual Computing, Graz University of Technology, Graz, Austria
Email: julian.rakuschek@tugraz.at

summarizing the contribution of each algorithm to the anomaly scoring.

These requirements are derived and extended from the work of Louis et al.[12] Additionally, **R3** is inspired by the VisInfo system,[7] while **R4** draws from the work of Suschnigg et al.,[24] where the authors introduce a glyph-based approach to visualize algorithm contributions within an ensemble score. Each requirement is addressed by tailored visual components presented in Section Proposed Visualization Design. Applications of the resulting workflow are discussed further in Section Applications and evaluated by users in Section User Study. Finally, the Conclusion discusses the hypothesis mentioned above in a concise manner, thereby concluding our study.

Background and Related Work

This section provides an overview of related work, including time series clustering and adaptive level of detail techniques. In the concluding subsection, we highlight our work's novelties by comparing it to prior studies.

Visual Clustering and Anomaly Detection

The survey by Warren Liao[29] provides a comprehensive overview of time series clustering methods, categorizing them into raw-data-based, feature-based, and model-based approaches. While the focus is on automated analysis, the survey highlights clustering as a fundamental technique for uncovering structure in temporal data. Its comprehensive scope remains relevant for our work, offering valuable context for developing visual analytics approaches to time series clustering. A decade later, Aghabozorgi et al.[1] provide a review of time-series clustering methods, formalizing whole time-series clustering as a core approach, where each time series is treated as a single instance. The authors emphasize that visualizing cluster structures as images significantly aids in identifying patterns, outliers, and regularities within large time-series datasets. This reinforces the importance of integrating visual representations into clustering workflows, particularly when users must interpret structure and behavior at scale.

More recently, two extensive surveys for interactive clustering and visual analytics have been published: (1) Ali et al.[4] provide a structured study that specifically addresses the integration of clustering and classification methods within visual analytics systems. Unlike earlier surveys focusing primarily on algorithms, this work emphasizes the importance of interactive exploration, parameter tuning, and visualization techniques for understanding cluster structures. It highlights the growing need to combine machine learning with interactive visual interfaces to support user-centered analysis. (2) The interactive clustering survey by Bae et al.[6] comprehensively reviews interactive clustering, analyzing 105 papers based on different aspects, such as user feedback, visual representations, user interactions, or hyperparameter selection. This work is relevant to our work since it highlights how visual analytics enables an effective exploration and understanding of datasets through clustering.

Focusing on time-dependent data, Wijk et al.[26] introduced one of the pioneering visual analytics approaches to visualize time series clustering with a calendar-based view, enabling intuitive analysis of temporal patterns. Their method utilizes clustering to group daily time series based on similarity and maps them onto a calendar, visually revealing reoccurring patterns and daily deviations within different periods. The system allows for selecting days, showing its raw time series, and allowing users to instantly highlight all similar days via clustering with a single click. This approach laid the groundwork for visual time series cluster exploration. More recently, the TimeCluster system[5] is an interactive visual analytics system that combines dimensionality reduction with sliding window techniques to support clustering and anomaly detection in time series data. Their method projects time series into a 2D space using dimensionality reduction, allowing users to explore patterns and outliers visually via scatter plots. Clusters corresponding to reoccurring temporal patterns appear as dense regions, whereas anomalies appear as isolated regions. The linking of the system between time series plots and scatter visualizations facilitates intuitive exploration of temporal structure and transitions. The SAX Navigator[15] is another example, a visual analytics system for large-scale time series exploration using hierarchical clustering and symbolic aggregate approximation (SAX). The tool enables users to navigate global time-series structures via a dendrogram, supported by cluster heat maps and an interactive sketch-based query interface. Local exploration is enhanced through linked views, allowing users to inspect and compare raw time series within and across clusters using 1:1, 1:n, or n:m comparisons. SAX Navigator is a system that effectively bridges global clustering structure and local anomaly inspection while remaining visually interpretable.

Adaptive Level of Detail Visualization

The simultaneous visualization of multiple time series in a grid can make the exploration challenging, as individual line plots can be difficult to perceive. Simple geometric zooming often fails to resolve this issue, as it only scales visual elements without adjusting the level of abstraction. Therefore, adaptive level of detail techniques[19] can dynamically change the visual representation based on zoom level or the number of displayed elements, which enables a detailed exploration across different scales. Several research papers have explored the concept of adaptive level of detail in time series visualization, aiming to improve scalability. For example, Yoon et al.[33] introduce an adaptive level of detail via semantic zooming to visualize code change histories, where edits are collapsed into higher-level abstractions depending on zoom level. This approach shows how adaptive levels of detail can support a more interpretable exploration. Another example of implementing an adaptive level of detail for time series data is the work by Aigner et al.,[3] which introduces an interactive visualization technique that uses an integrated semantic zoom along the value axis to adapt visual detail based on display space. Their study demonstrates that such adaptive representations can facilitate comparison tasks by reducing the cognitive load. More recently, Suschnigg et al.[23] present a system for the detection and exploration of anomalies in multivariate time series data. The system integrates an adaptive level of detail of time series data within a scatterplot matrix, dynamically adjusting

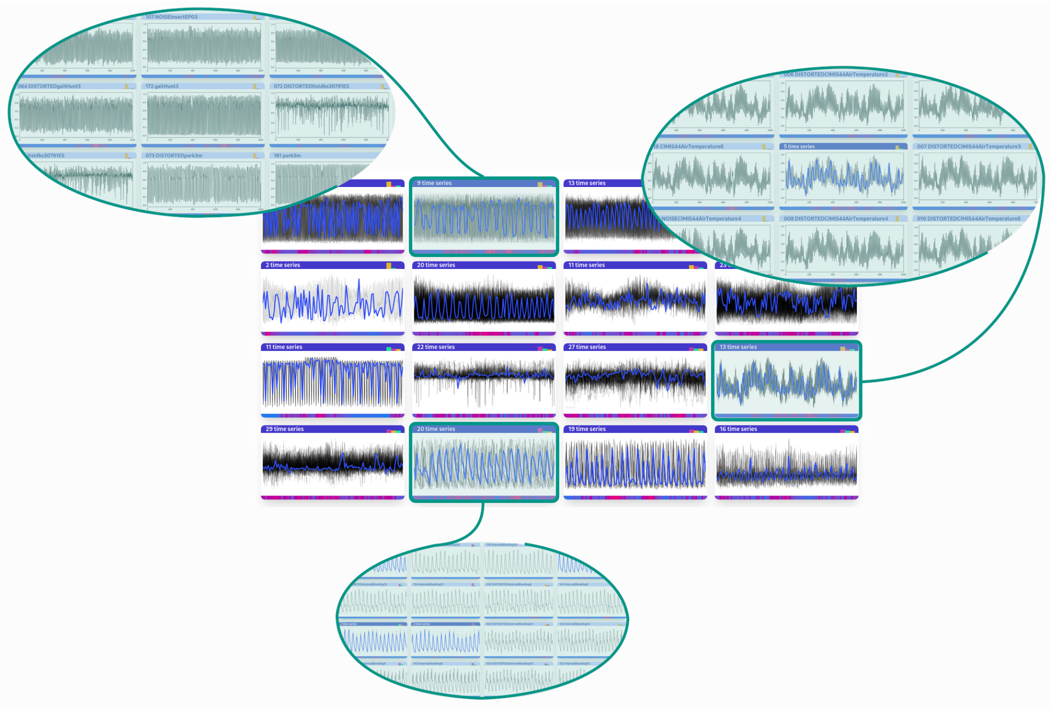

Figure 1. The resulting tree structure after applying hierarchical clustering to the dataset enables the user to explore the dataset through layers of clusters. The figure illustrates this concept in a schematic manner: The initial layer is depicted in the middle consisting of 16 clusters. Users then select clusters to traverse to the next layer, schematically shown through the ”magnifying glasses”.

hexagonal binning granularity, axis detail, and point size based on zoom level for enhanced scalability.

Delineation and Our Contribution

In the reviewed literature, anomaly detection and time series clustering are often combined, as clustering reveals typical patterns while anomaly detection supports finding deviating instances. Many existing approaches utilize dimensionality reduction techniques, such as symbolic encodings or principal component analysis, to make time series clustering and visualization scalable, often at the cost of interpretability. In contrast, our approach deliberately avoids reducing the data to lower-dimensional representations. Instead, we aim to preserve the raw time series data during automated analysis and visualization, ensuring interpretability throughout the analysis. While dendrograms are commonly used for visualizing hierarchical clustering, we emphasize comparing clusters directly through their raw time series, supported with overview dendrogram visualizations. This enables users to understand clusters and anomalies visually and contextually. Additionally, we utilize alternative visualization types, such as horizon graphs and heatmaps, combined with adaptive levels of detail, depending on the zoom level or user-selected grid sizes, providing a novel way to explore multiple time series clusters simultaneously. In comparison to the related work by Ruta et al.,[15] we address some scalability issues with our design and its user-selected grid size: (1) Our system is not limited to a minimum cluster size, whereas our drill-down capabilities allow the exploration down to individual time series. (2) Our design does not rely on panning and zooming, which might introduce scalability issues when many clusters are formed. Instead, our proposed design utilizes dendrograms as a contextual overview.

Proposed Visualization Design

In this section, we derive our proposed interactive visualization design, aiming at effective exploration of large time series data. It will be based on the key principles of (1) hierarchical data aggregation and (2) adaptive time series visualization, in combination supporting the visual analysis. We first describe the hierarchical navigation through the dataset (**R3**), followed by the visualization types in combination with semantic zooming (**R1**, **R2**), and conclude with an application to anomaly detection (**R4**).

Creating Overview Visualizations and Navigation Structure

As mentioned in the Introduction, visualizing a large volume of time series is infeasible on a single screen. This necessitates an overview visualization, presenting the user with the most common pattern groups in the dataset, following the information visualization mantra.[20] Computing such an overview is commonly realized by finding clusters within the time series, which represent common patterns, as, for example, seen in the VisInfo system.[7] We further not only implement clustering as an overview technique, but also to allow for hierarchical navigation, that is, users explore sub-clusters upon selecting a cluster from the overview. Figure 1 illustrates this concept. Users therefore navigate through a tree structure of patterns, with the difference in patterns gradually decreasing as users approach lower levels. As we will show, this concept avoids

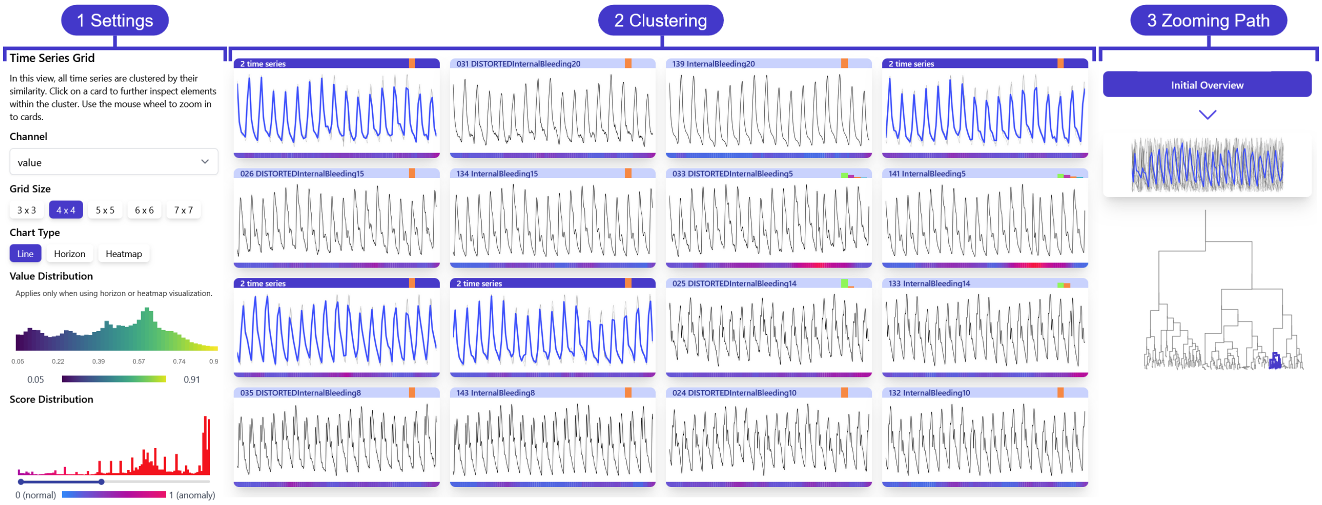

Figure 2. The interface is divided into three areas: (1) The settings column enables the user to adjust the grid size and the visualization type, which can be changed on any zooming level. The majority of the available space (2) is devoted to the clustering visualization, which arranges groups of time series in a grid pattern and allows semantic zooming on the current level. Finally, the zooming path (3) shows the level on the hierarchical navigation utilizing a card stack for navigation and a dendrogram as the bird's eye perspective.

overwhelming users with the entire dataset by summarizing core patterns and providing detail on demand. The choice of the clustering method, together with an appropriate distance method, is critical and will be explained in the subsequent paragraphs.

Although a wide variety of clustering algorithms exist,[4] every method is dependent on the comparison between time series, which is commonly implemented using either the Euclidean distance (ED) or Dynamic Time Warping (DTW).[8] The ED between two time series is generally fast to compute, since an element-wise difference is considered, which can be achieved in $\mathcal{O}(n)$. However, the ED does not account for phase shifts within the time series and expects patterns to align. DTW overcomes this issue by computing an alignment path between two time series, which optimally assigns patterns between them. Although the resulting distance is more accurate, the computation time increases drastically with an $\mathcal{O}(nm)$ time complexity.

This background information is necessary to comprehend the following design decision regarding the chosen clustering method for our tool. The ultimate goal of the tool is to facilitate a fluent exploration experience for the user without interruptions lasting several minutes. Stolper et al.[21] demonstrate the importance of fluent visual analytics workflows in an extensive study, coining the field of *Progressive Visual Analytics*. As a first experiment, a Self-Organizing Map (SOM),[11] inspired by the VisInfo system, was implemented, which arranges the time series in a grid, where neighboring cells share similar elements to facilitate a gradual change across the grid.

However, in our use case, the SOM method is not an appropriate choice due to several limiting factors. First, computing the SOM with the ED does not yield accurate results in many cases; therefore, employing the DTW similarity measure is essential. However, it is not possible to supply the SOM algorithm with a precomputed DTW similarity matrix due to each grid cell having a prototype time series. Further, it is infeasible to compute the SOM for every layer in advance since users should be able to change the number of clusters per layer. Since DTW has a time complexity of $\mathcal{O}(nm)$, it would take several minutes for a larger dataset to compute the next layer clustering as users explore the tree, which is unacceptable for an exploration workflow and would not scale well. The final disadvantage is the numerous parameters the SOM exposes, making it difficult to find suitable hyperparameters without a rigorous grid search procedure.

Based on these considerations, we chose to use hierarchical clustering instead, which recursively merges similar time series or clusters to form a tree-like structure (dendrogram) that reveals nested relationships at multiple scales without requiring a predefined number of clusters. Due to the recursive nature and direct comparison of merged clusters, no prototypes of clusters are computed. Consequently, a precomputed similarity matrix can be supplied to the algorithm, which means that the matrix can be cached and no computationally expensive DTW calls are necessary during clustering. Furthermore, extracting a given number of clusters is an efficient operation, since only a distance value needs to be specified such that all sub-dendrograms below a given distance value are retrieved. The distance value is chosen based on the distances at which two clusters merge. Unlike in the case of the SOM, no cluster needs to be computed again when selecting a subgroup, since users effectively select a branch of the dendrogram when inspecting a cluster. Therefore, computing the dendrogram for the entire dataset once is sufficient and can be efficiently cached. We emphasize that reasonable performance is critical to ensure a fluent workflow experience for users. The dendrogram consequently enables an instantaneous switching between layers with the cluster representations, discussed in the following subsection, being computed asynchronously afterwards.

The extracted clusters can still be arranged in a grid pattern, following a row-major ordering. In our implementation, the dendrogram is traversed in post-order (comparable to depth-first search in graphs); as a consequence, horizontal neighbors in the grid are similar to each other. Finally, the only parameter required for hierarchical clustering is the choice of the linkage method, which determines how merged clusters are compared regarding their distance. In our use case, we chose Ward's linkage method,[28] since it produces clusters similar in size and variance. This is beneficial for the aim of providing an overview, since the clusters tend to be balanced. However, no linkage method can prevent imbalance when outliers heavily deviate from the distribution.

Figure 2 presents the proposed interface for the visualization tool. Users may adjust the grid size and explore the result in the clustering view, which arranges the clusters in a grid pattern with row-major order, as discussed before. Each cell contains a cluster card, whose design is depicted in Figure 9. If the card represents a cluster with more than one element, users may click on it and thus explore the sub-cluster in the next layer. Each layer in the current hierarchical navigation is depicted in a card stack on the right-hand side of the interface. Since selecting a sub-cluster is equivalent to selecting a branch in the dendrogram, we provide a dendrogram visualization with an indication of which part of the dendrogram is visualized on the current layer.

However, while the line chart shown in Figure 2 is reasonable for a small grid size, it may become impractical for larger grids. In the next section, we propose semantic zooming with several visualization types as a solution.

Cluster Representatives

Depending on the dataset, clusters may contain hundreds of elements. Therefore, it is crucial to convey a reasonable cluster representation to users such that an estimation of the underlying pattern throughout a given cluster is achieved. The line chart is the most well-known visual representation of time series as it clearly conveys the shape of the data. It is common practice to plot all elements in a cluster in the same line chart[4] and thus the overplotting effect is beneficial when inspecting matching patterns. However, patterns may not properly align and therefore create a plot with much noise. Figure 3 demonstrates that the point-wise average representative of such a cluster may not properly summarize the grouped patterns. As an alternative, *Dynamic Barycentric Averaging* (DBA)[14] accounts for phase shifts when computing the average of several time series. Despite the results being more accurate, the computation time increases drastically due to the $\mathcal{O}(nm)$ time complexity of DTW, which poses a hindrance for exploring datasets. We propose to use an approximation instead to combine the best of both worlds: fast computation time and accounting for phase shifts. The approximation is achieved by resampling all elements to a smaller size prior to applying DBA.

As we show in Figure 3, the approximation still conveys an idea of the underlying pattern, yet is orders of magnitude faster to compute, which makes it an appropriate cluster representative for an exploration workflow. Figure 4 shows an experiment to justify the approximation. For a small time series length, the computation of a cluster representative does not exceed 5 seconds and is still suitable to gauge the summarizing pattern of a cluster during exploration,

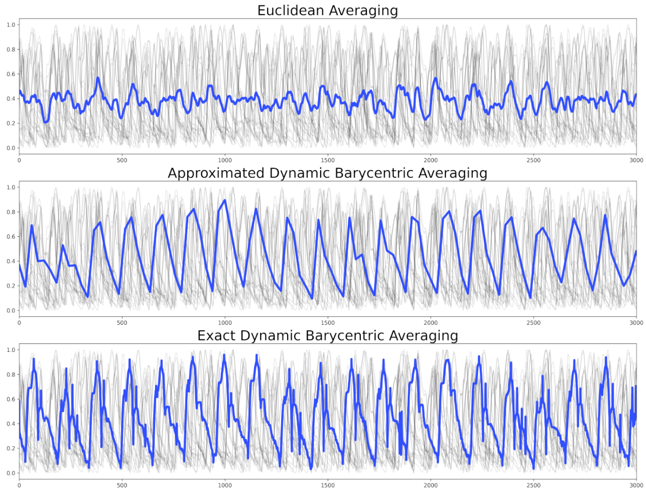

Figure 3. Demonstration of different time series averaging methods using the "Internal Bleeding" time series from the UCR archive. The experiment shows that an approximated version of dynamic barycentric averaging yields a reasonable trade-off between accuracy and computation time.

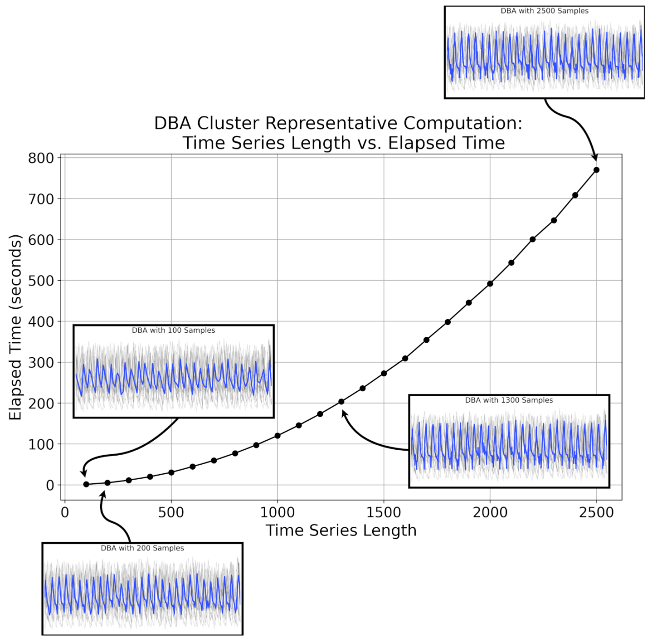

Figure 4. The computation time for the approximated DBA increases polynomially as the time series size increases in length, however, the DBA representation with 100 samples is already sufficient for the exploration workflow with only 1.25 seconds computation time. The experiment was conducted on an Intel i5-12600K processor with the "Internal Bleeding" time series from the UCR archive, which amount to 28 time series.

however, the computation with original length requires several minutes, which is inappropriate for a visual analytics exploration workflow. Finally, the representation of the cluster by summarizing a large number of time series through a single time series is essential for heatmaps and horizon graphs, which we offer as an alternative perspective to the line chart. These visualization types are not capable of rendering multiple time series at once; therefore DBA is essential.

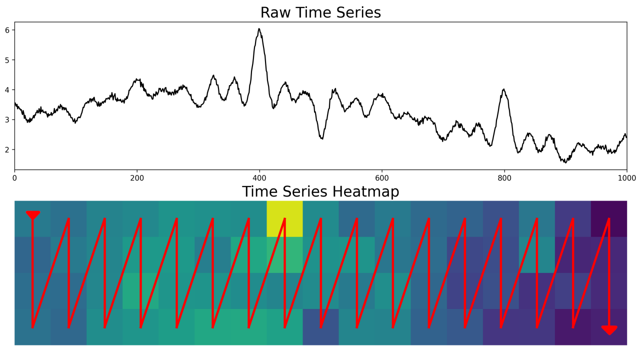

Figure 5. The heatmap is constructed by segmenting the time series into equal-sized non-overlapping windows and computing the average of each segment. The resulting values are arranged in a column-major order, schematically illustrated by the red arrow. Each cell is colored according to the Viridis color palette.

Communicating Value Ranges Through Heatmaps and Horizon Charts

Although the line chart is able to convey an impression of patterns within the time series, it becomes increasingly difficult to retrieve domain information when inspecting many line charts in a small multiples visualization. Therefore, we propose the integration of heatmaps and horizon charts as additional perspectives to assist users in gaining an overview of the value range distribution across the time series clusters. The heatmap is a common visualization technique to communicate value ranges by coloring cells according to average values for corresponding non-overlapping segments in the time series. The cells of the heatmap are ordered in a column-major schema such that the x-axis corresponds to the time domain. Figure 5 provides an example to illustrate the construction of the heatmap. Although the heatmap is able to depict trends within the time series, precise patterns are lost due to the piecewise aggregation of segments.

While heatmaps depict value ranges and overall trends, the horizon chart[16] serves as the middle ground between the two aforementioned visualizations, since this visualization technique is a combination of line charts and heatmap coloring. The construction of the horizon chart is illustrated in Figure 6. First, the time series is segmented along the y-axis, thus segmenting the domain of the series. Next, each band is assigned a color according to the segment index. In the final step, all bands are shifted to a common baseline such that the band with the lowest domain range is in the background while the band with the highest values is in the foreground. The number of bands determines the amount of visible detail in the visualization. For example, a horizon chart with only one single band corresponds to an area chart; thus, the same amount of detail is visible as in a line chart. On the other hand, if the number of bands is large, the horizon chart becomes similar to a heatmap with one row, thereby focusing on the trend within the series. Consequently, users may realize during exploration which visualization type fits their needs and may therefore switch between the three visualization types at any point as shown in Figure 7.

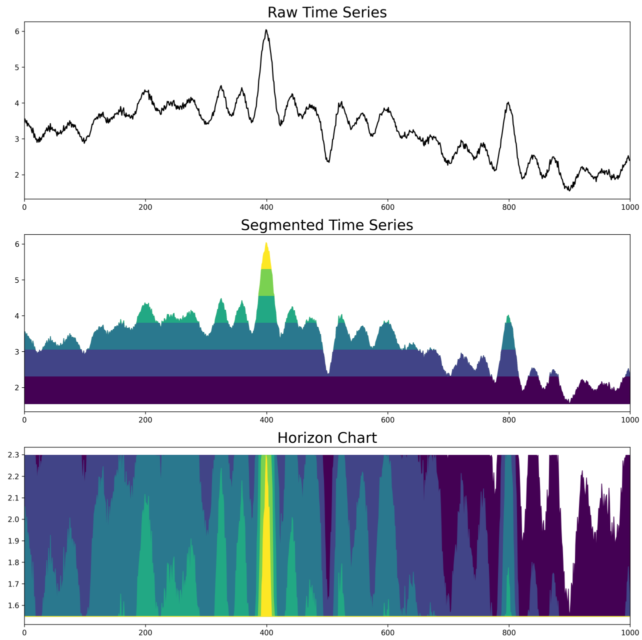

Figure 6. The horizon graph is constructed by first segmenting the time series along the y-axis into equally sized bands and afterwards shifting each band to a common baseline. The segment index determines the color, for which the Viridis color palette was chosen.

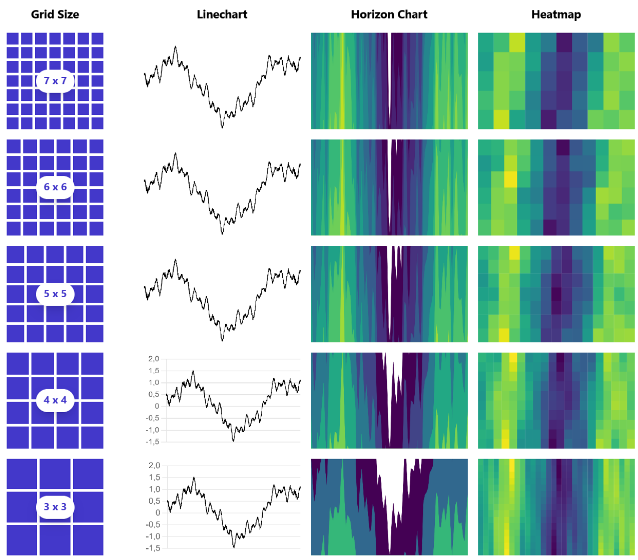

Figure 8. The level of detail is bound to the perceived grid size when zooming in. While the number of bands in the horizon chart increases with the grid size, the number of cells in the heatmap decreases.

Adaptive Level of Detail

Since users may switch between several available grid sizes, the visualization types need to adapt according to the number of visible cells. Given a large cluster size, we depict fewer details such that overall trends and value ranges are emphasized. Users may gradually increase the level of detail by zooming into the grid and the visualization types adapt themselves according to the perceived number of grid cells, that is, the grid size visible on the user's screen as illustrated in Figure 8. While for the line chart the only level of detail is the display of the axis, the heatmap and the horizon chart

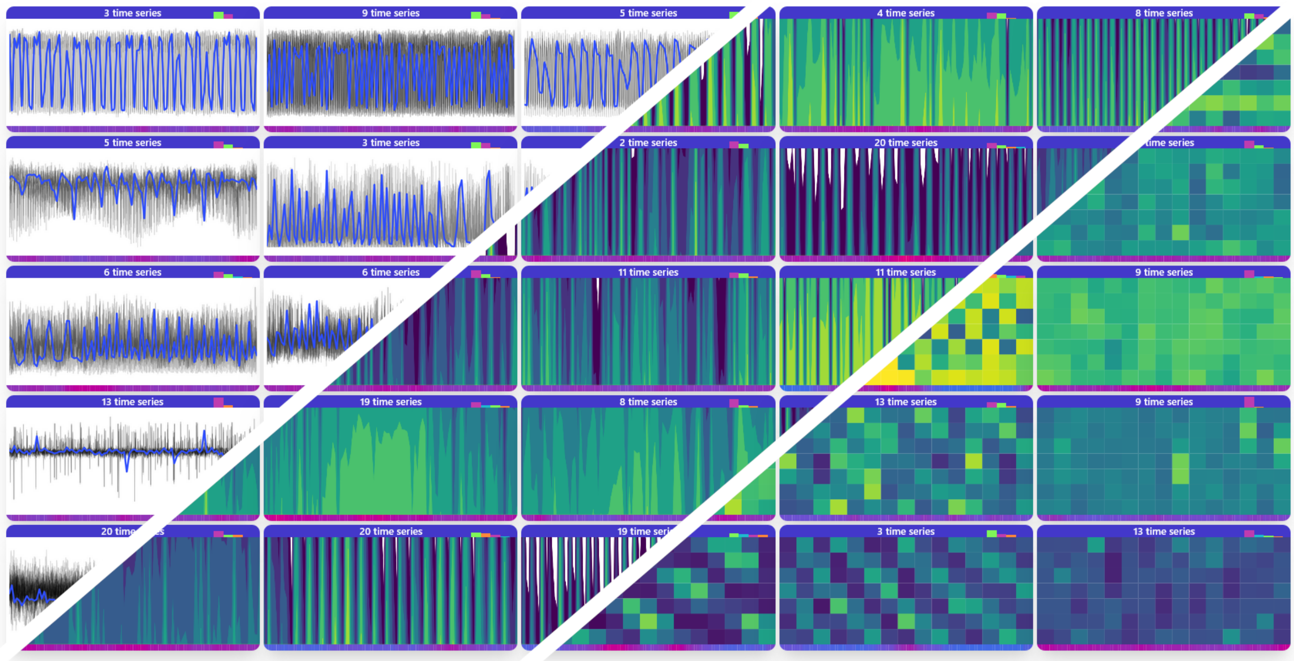

Figure 7. Users may choose between three visualization types: The top left corner shows the classic line chart with the dynamic barycentric average of all time series within the cluster in blue. The middle stripe depicts a horizon chart and the bottom right corner presents a heatmap visualization. Both the horizon chart and heatmap use the Viridis colormap and only show the dynamic barycentric average.

expose parameters with a discrete value range. As shown in previous work,[12] it is a reasonable choice to decrease the number of bands in the horizon chart as the perceived grid size decreases. Consequently, inferring details is an effortless task when the number of bands is kept low, while observing trends and value ranges is convenient at a low level of detail with a higher number of bands. However, the opposite is true for the heatmap, for which the number of cells increases as the user zooms in. As an analogy, one may observe how images are rendered in lower resolution when displayed in a small portion of the screen and increase in resolution when fully filling the screen. Overall, the seamless transition between several levels of detail, in combination with a hierarchical navigation between layers, fully implements the information visualization mantra, focusing on overview first and providing details on demand.[20]

Integration with Anomaly Detection

Although the tool provides an initial overview of patterns, not every pattern is expected. For example, a domain expert may identify the peak shown in Figure 5 and Figure 6 as an anomaly since it may deviate from the normal expected behavior as defined by the user. The visualizations may point toward outliers; however, subtle anomalies often only become apparent through close inspection, which may be infeasible for a large dataset such as the UCR archive mentioned in the introduction.

Therefore, we propose integrating unsupervised anomaly detection algorithms, which output a scoring indicating anomalous regions within the time series. Note we added several methods based on available implementations and aiming at complementarity of methods, the specific choice of methods is interchangeable. Any unsupervised algorithm with a scoring as the output format can be added to the algorithm repertoire of this tool. The scoring is a time series itself, defined as follows:[18]

Definition Scoring: A time series *scoring* $S = \{s_1, s_2, \dots, s_m\}$ with $s_i \in \mathbb{R}$ is the result of a time series anomaly detection algorithm that assigns each data point $T_i \in T$ an anomaly score $s_i \in S$. For any two scores s_i and s_j, it must be true that if $s_i > s_j$, then T_i is more anomalous than T_j (in their respective contexts).

Figure 9 illustrates the visualization of the anomaly scoring using a one-dimensional heatmap. The color palette ranges from blue (low scoring, indicating normal behavior) to red (high scoring, indicating unusual events). This contrasts with the Viridis colormap used in time series visualizations and is intuitive due to the common perception of blue as cool and red as heated.

The final anomaly score visualization is derived from a two-step aggregation process. Each anomaly detection algorithm has its strengths and weaknesses,[17] making it advisable to use multiple algorithms. The authors suggest computing an ensemble score by calculating the element-wise average of all algorithm scorings for a time series. Subsequently, in our tool, the representative anomaly score for each cluster is determined by taking the element-wise maximum of each ensemble score. We use the maximum to ensure that anomalies remain visible, even in large clusters.

Although the anomaly scoring provides an indication of anomalous regions within clusters and time series, it does not convey which algorithms contributed to the ensemble scoring. Therefore, we propose an algorithm sensitivity indicator, as seen in Figure 9. The glyph is composed of a bar chart, where each bar corresponds to one algorithm. The

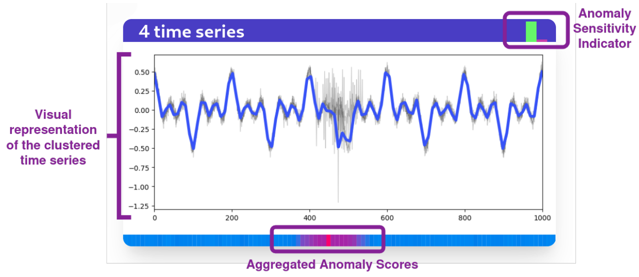

Figure 9. Structure of a cluster card: The time series cluster is represented through one of the three visualization types in the center. The algorithm sensitivity indicator is located in the top right corner and provides information about which algorithms reported findings for this cluster. Finally, the aggregated anomaly score is located in the bottom row, which is an ensemble of all algorithms across all time series within the cluster and is intended to communicate common locations of anomalies.

height of the bar is the ratio of time series within the cluster for which any of the algorithm's scoring elements is greater than a user-defined threshold. Naturally, this is a binary decision in the case of a single time series. We emphasize that this glyph does not provide information about the algorithm's performance; it only conveys information about the sensitivity. However, this information can be valuable to steer algorithms, for instance when certain algorithms react in an unfavorable way to normal behavior.

Summary of our Design

The proposed visualization tool is designed to facilitate the exploration of time series data and anomaly detection through a hierarchical navigation system. The tool employs clustering techniques to manage large datasets, allowing users to navigate through layers of clusters to identify common patterns without being overwhelmed by data volume. It utilizes Dynamic Time Warping (DTW) to accurately match patterns and supports efficient exploration through precomputed similarity matrices. To represent clusters effectively, the tool uses an approximated version of Dynamic Barycentric Averaging (DBA), which accounts for phase shifts and provides a fast, accurate summary of time series patterns. The tool offers semantic zooming with multiple visualization types, including line charts, heatmaps, and horizon graphs, each providing unique insights into the data. The adaptive level of detail feature adjusts visualization complexity based on grid size, optimizing screen space and enhancing user experience. Additionally, the tool integrates unsupervised anomaly detection algorithms to highlight unusual patterns with visual indicators for algorithm sensitivity. This combination of features ensures a balanced approach to data exploration, providing both overview and detailed views to support effective analysis and anomaly detection in time series data.

Applications

This section presents several applications of the approach described in the previous section, demonstrating its usefulness and justifying the previously introduced design decisions within the tool.

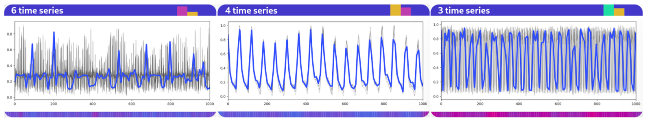

Figure 10. The anomaly sensitivity indicator in the top right of each card provides insights into the sensitivity of each algorithm. In this example, three patterns from the UCR archive are shown with each pattern having a different dominant algorithm.

Exploring the UCR Archive

The UCR archive has been introduced in the Introduction as it presents the challenge of having a substantial amount of diverse patterns, which necessitates an exploration workflow to gain an overview. To focus on the anomalous regions, we specifically removed the training data from the time series. Figure 1 depicts the initial cluster grid upon loading the UCR archive. Although a portion of clusters may not directly reveal patterns, several patterns emerge on the first layer of the hierarchical navigation, which are highlighted in Figure 1. For example, the highlighted cluster in the bottom row corresponds to the "Internal Bleeding" pattern in the UCR archive. Since the time series have different lengths and do not align in phase shifts, this highlights the need for the cluster representation shown in Figure 3, which demonstrates the usefulness of the Dynamic Barycentric Averaging technique on the "Internal Bleeding" cluster. Although Figure 1 shows the plausibility of the exploration use case scenario, we further support our claim with a user study by assigning users the task of exploring the UCR archive using our proposed workflow (shown in the next section).

However, the UCR archive[31] is specifically designed to be an anomaly detection benchmark, such that algorithms can be properly evaluated and compared. The main features of the dataset include non-trivial anomalies, diverse patterns, sound ground truth labels and unbiased occurrences of anomalies regarding their location within the time series. Each anomaly detection algorithm has strengths and weaknesses,[17] which motivates an ensemble technique and a comparison between algorithms. Figure 10 shows three clusters of the UCR archive with three anomaly detection algorithms. The sensitivity glyph in the top right of each figure provides an indication to the data scientist of which algorithms react to the corresponding pattern. Finally, the anomaly score heatmap at the bottom of each card confirmed in our experiments that the anomalies in the UCR archive are randomly placed, thus contributing to an unbiased dataset. However, the anomaly score heatmap can also highlight the opposite case, which is demonstrated in the subsequent use case scenario.

Detecting Biases in Datasets

Synthetic time series datasets with artificial anomalies are valuable tools for testing and validating anomaly detection algorithms, as they provide controlled environments where the nature and timing of anomalies are known. These datasets allow researchers to benchmark algorithm performance and robustness without the variability and unpredictability of real-world data. However, care must be taken to avoid

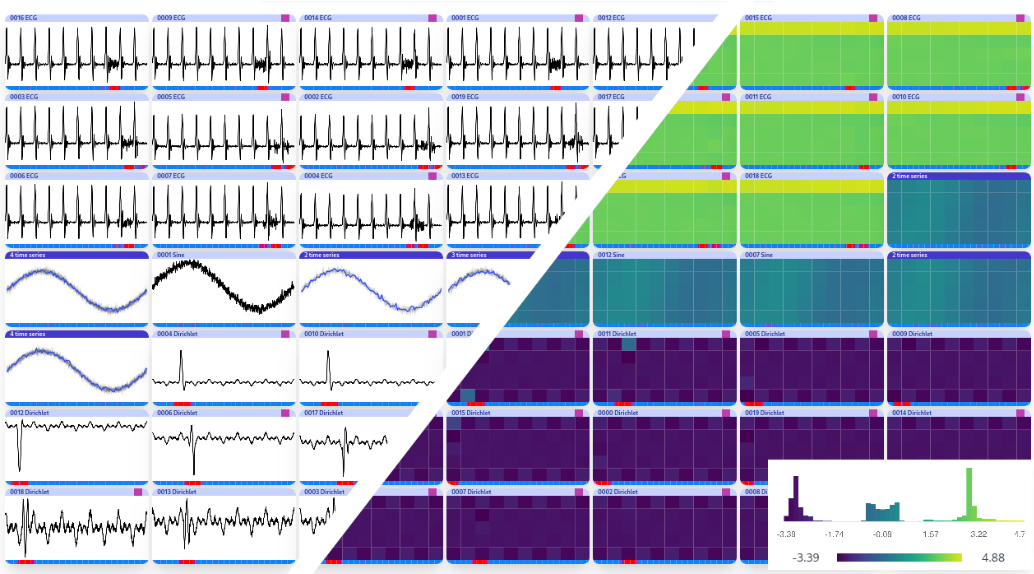

Figure 11. Our synthetic dataset generated via GutenTAG contains three distinct pattern groups. The figure demonstrates the usefulness of the heatmap visualization to estimate value ranges. The histogram on the bottom right shows the distribution of the values.

potential biases in the generated dataset, such as anomalies only appearing in specific parts of the time series. Figure 11 presents such a dataset, which has been generated through the GutenTAG tool,[30] which facilitates the generation of synthetic time series and anomalies and is intended to generate datasets for the evaluation of anomaly detection algorithms.[18] The dataset shown in Figure 11 contains three distinct pattern groups: a pseudo ECG, a simple sine wave and a Dirichlet oscillation. Through the anomaly score heatmap, our tool provides an indicator of common anomaly locations within each pattern group, given a suitable anomaly detection algorithm. As can be seen in Figure 11, anomalies in the pseudo ECGs tend to occur at the end, while they tend to arise at the beginning of the Dirichlet oscillation. This indicates a bias within the pattern groups. The second bias becomes evident when switching to either the heatmap or the horizon graph, which show that each time series tends to be biased toward a certain value range depending on the pattern group. Although this may not be an issue for anomaly detection algorithms, it highlights the importance of having multiple perspectives, to create awareness of diverse domains within the dataset. Therefore, with this example, we demonstrate the usefulness of the tool in assessing the soundness of a synthetic dataset, which will be confirmed in the user study.

Finding Anomaly Pattern Groups

The final use case scenario is based on a dataset introduced in previous work by Suschnigg et al.,[22] which is concerned with time-series process data from Gas Metal Arc Welding operations. The data comprises process measurements of the power supply's current during welding. This time series can be informative, offering insight into individual welds or revealing patterns of collective deviations across multiple welds. In this use case, we examine current measurements from multiple welds, each representing repetitions of the same welding process, which lasts 1.2 seconds and is recorded at 10 Hz. When visualizing this data set in our tool using the 4x4 grid (see Figure 12), two larger prominent clusters emerge without anomaly indication, containing 16 and 37 time series, respectively, reflecting the most common pattern among all time series. Further exploration of clusters and individual welds in this view reveals that other clusters are primarily formed because of a peak towards the end of the time series, with slight differences in the exact timing of these peaks contributing to the formation of distinct clusters. The anomaly indicators of these clusters help advise domain experts that these groups contain potential abnormal behavior, helping to direct their attention to those instances and groups of suspicious time series. We explored this dataset with five domain experts, three directly involved with producing welding machines and their software, and two end users of the welding equipment. All of them agreed that this pattern is interesting and may hint at issues in the welding process, specifically in the end phase of the weld. The welding process is expected to record identical data for each weld, and anomalies or clusters deviating from the majority of data with such distinct patterns are noteworthy. However, such occurrences need further investigation, not only within the data but also in the welding configuration or the welding machine, to derive actionable insights from this exploration. Enabling domain experts to reach this level of insight efficiently highlights the tool's practical value and establishes a strong foundation for further research.

User Study

To demonstrate the practical value of our tool, we performed a user study with the ICE-T method,[27] which is designed to assess the value of a visualization, through the combination of four factors: Time, Insight, Essence, and Confidence. According to the authors of the evaluation method, it suffices to include five users; however, the method is specifically designed for users with a background in visualization. Each

Figure 12. In this example from the welding use case scenario, the majority of data comprise welds that visualize expected behavior (A and B). Other, smaller clusters visualize patterns with a peak toward the end, with differences in the timing of these peaks (C and D), indicating differences in the end phase of the welding process. Our tool can also easily detect individual welds that are anomalous, such as depicted in E.

evaluation was conducted as follows (adopted strategy from Schmidt et al.):

1. Introduction of the system to the participant.
2. The participant solves several tasks, which demonstrate all features of the system. While executing the tasks, the participant is instructed to think aloud.
3. Afterwards, the ICE-T questionnaire is completed by the participant.
4. Finally, the evaluator discusses with the participant the problems they faced during the tasks.

For this user study, we recruited five participants, designated as P1 through P5. P1 is a Master's student with expertise in data science and visualization. Participants P2, P3, and P4 are PhD students conducting research in visualization and visual analytics. Lastly, P5 is a postdoctoral researcher specializing in information visualization. The participants were instructed with the following tasks:

Task 1: The participant should leverage the hierarchical navigation to gain an impression of patterns in the UCR archive, as seen in Figure 1. After an initial exploration, participants were asked to roughly sketch and name three patterns within the dataset on paper.

Task 2: Next the participants were given a synthetic dataset generated by GutenTAG, as seen in Figure 11. The dataset included three pattern groups and participants were asked to determine which of the patterns were located in high, medium, or low value ranges. This encouraged participants to exploit the alternative visualization types.

Task 3: In the same synthetic dataset generated by GutenTAG (Figure 11), participants were asked to identify a bias for anomalies in the dataset through the usage of the anomaly score heatmap.

Task 4: Participants were asked to explore the UCR Archive and interpret the algorithm sensitivity indicators.

Task 5: Finally, the Welding dataset was presented in which participants were asked to determine the normal class together with anomaly patterns (Figure 12).

After completing all tasks, the ICE-T questionnaire was administered to the participants. The resulting aggregated scores are depicted in Figure 13. From observations of users during their task completion, the ICE-T questionnaire and the subsequent discussion with participants, we draw several conclusions about the system.

First, the participants had no issues navigating the dataset and all of them completed the first task. However, due to the large number of diverse patterns in the UCR archive, participants initially felt slightly overwhelmed when exploring the dataset. This highlights the need for further guidance, although all participants eventually committed to three patterns. It was observed that the cluster representation computed by DBA is indeed a valuable visualization and participants changed the grid size to control the clustering granularity. Furthermore, several users utilized the semantic zooming feature to inspect individual cluster representations. During exploration, all participants noted that the cluster representation causes a delay during navigation and thereby interrupting the workflow. This observation highlights that computation time is a critical element in the exploration workflow and underscores the necessity of a flexible clustering method such as hierarchical clustering to reduce delays. Although the approximation of DBA reduced computation time drastically, there is still a need for performance improvement, which is confirmed by the user study.

	P1	P2	P3	P4	P5	Average
Insight	6,38	6,75	6,75	6,63	6,63	6,63
Time	6,00	6,80	6,80	6,40	5,80	6,36
Essence	5,50	7,00	6,75	6,75	6,25	6,45
Confidence	6,25	6,25	5,75	6,25	6,67	6,23

Figure 13. ICE-T evaluation scores of all participants. The scores range from 1 to 7, where 7 is the highest rating. The total average scores per category indicate that the system performs well in all categories.

Second, all participants found the heatmap representation helpful for estimating value ranges in the dataset and were able to complete Task 2. However, several participants noted that the horizon graph may be less intuitive for users less familiar with time series visualizations.

The final three tasks examined the usefulness of the anomaly indication features. First, observations of participants during Task 3 demonstrated that the anomaly score aggregation is an intuitive visualization, as it aligns with the temporal axis of the cluster representation. Furthermore, the choice of coloring for the scoring heatmap is appropriate due to the stark contrast with the Viridis colormap used for the heatmap and horizon graph visualizations. Therefore, all participants were able to identify biases of anomalies within the pattern groups. Next, the algorithm sensitivity indicator was evaluated in Task 4. Participants had no issues comprehending the meaning of the computed indicator, which highlights the benefit of simplicity within the method. Since the bars in the indicator are sorted, all participants quickly identified the most and least sensitive algorithms. Finally, due to the indicator, participants were able to identify normal behavior within the Welding dataset (Task 5) and successfully identified anomaly patterns, highlighting the overall benefit of clustering and the algorithm sensitivity indicator.

The resulting ICE-T scores shown in Figure 13 highlight the beneficial value of the workflow. The average score for *Confidence* received the lowest rating, which aligns with our observation of horizon graphs lacking clarity. Although the DBA representation proved to be helpful during exploration, there may be cases in which the DBA lacks accuracy, leading to misleading representations and thus decreasing confidence. Nevertheless, the evaluation shows that the proposed workflow is helpful for solving the posed tasks.

Future Work

Although the proposed visualization tool solves all posed requirements, several open challenges remain opportunities for future work, which are discussed in the subsequent subsections.

Provenance

The proposed visualization tool aims to enable the exploration of time series data through real-time interaction. It does, however, not have the ability to extract information from previously gained insights. A possible extension in this direction is the use of a provenance interface. Provenance refers to the history of an object or of information. In the visualization context, it describes storing information on how a visualization was used previously. By visualizing which aspects of the data were deemed important previously, new users or users unfamiliar with the data can more easily know where to focus when using the visualization tool. A provenance information interface can be a valuable extension to the tool. Figure 14 shows an illustration of a potential provenance interface layout. The interface uses multiple views to show information on the main view, the three different visualizations and semantic zooming. The first view (Figure 14a) shows all time series arranged in small multiples. The border of each graph is colored with varying thickness, with thicker borders representing time series that were selected more often. The second view (Figure 14b) shows the frequency of the three visualizations (line plots, heatmaps, horizon graphs). Each of the graphs represents one of the time series of the original visualization. The three bar charts show the frequency with which each time series was viewed in each of the three visualizations. In the third view (Figure 14c), the time series are grouped by border colors. Time series with the same border color represent groups of time series that are often selected together or consecutively. The remaining two views show information on the time series which is currently selected in the provenance interface. Frequent Subset-Selection (Figure 14d) shows the time series with colored time regions. These colored regions show time spans within the time series which are often selected. Darker colors represent a higher selection frequency. The current visualization tool lacks a feature to select subsections within a time series. This extension aids in both analysis and provenance interface. In the Level-Of-Detail Distribution view (Figure 14e), the zoom level is plotted on the horizontal axis and the frequency is plotted on the vertical axis. For the selected time series, this view shows how often it was selected at different zoom levels. The tools provenance interface captures and visualizes all data analysis actions, serving as a guide for new users by showcasing what previous users found noteworthy, thus aiding in guiding future analysis.

User-Adaptive Visualization

Tracking user interaction based on provenance principles provides beneficial information for user-adaptive visualizations. Yanez et al.[32] define them as visualizations modifying themselves based on user characteristics and preferences. This approach can help users more quickly identify relevant patterns or insights by presenting them with visualizations tailored to their analytical behavior. Within the context of our proposed workflow, the tool could recommend alternative time series visualizations along with appropriate cluster granularities, informed by prior user interactions. Future work should focus on identifying and evaluating user metrics and indicators that serve as meaningful input features for the recommendation process.

Streaming and Progressive Data Analysis

Our current approach assumes a static dataset, however, time series data often arrives as continuous streams

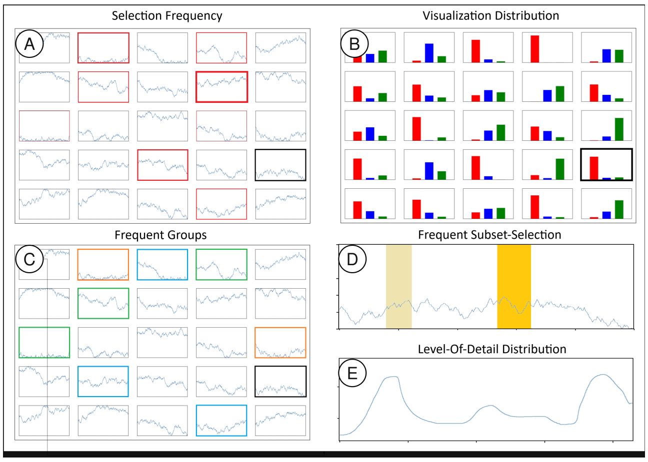

Figure 14. An overview of how different types of information could be shown in a provenance interface. (A) Selection Frequency: Border thickness indicates how often a time series was selected. (B) Visualization Distribution: Shows the frequency of each visualization technique used for selecting time series. (C) Frequent Groups: Highlights time series frequently selected together with border colors. (D) Frequent Subset-Selection: Darker backgrounds denote frequently selected time spans within time series. (E) Level-Of-Detail Distribution: Plots frequency against zoom levels for selected time series.

from sensors. Future work should explore adapting the workflow for streaming computation,[10] including evolving clusters and their visualization over time. A key challenge is efficiently updating cluster representatives without redundant computations. While our DBA approximation improves performance, incremental updating methods should be investigated. This aligns with the principles of Progressive Data Analysis (PDA), which processes large datasets incrementally. Combining PDA with streaming computation could make the tool scalable for both real-time and large-scale time series data.

Labeling Anomalies

Although the proposed tool supports anomaly detection using unsupervised algorithms, future works may extend the algorithm set by including semi-supervised algorithms. This includes methods, which consider predefined normal behavior by users resulting in one-class classification algorithms. The clustering component is especially relevant to facilitate an efficient labeling process, since entire clusters may be marked as normal behavior. For example, users may identify cluster B in Figure 12 as normal behavior such that the system identifies the surrounding clusters as anomalies. Future works should subsequently investigate suitable methods for the one-class classification problem.[13]

Identifying Recurring Patterns

Each time series is naturally linked to time stamps, and the same applies to the resulting clusters. Currently, the tool does not account for the chronological relationships between time series. However, recurring patterns may exist both within individual clusters and across different clusters. Future work should therefore explore visualization methods that help reveal such patterns. This could enable users to make informed predictions about future developments in the time series dataset based on the identified trends.

Conclusion

This work introduces a scalable visual exploration tool for time series and anomalies, addressing the challenge of information overload in large datasets. By integrating hierarchical clustering for efficient navigation and semantic zooming with multiple visualization types (line charts, horizon graphs, and heatmaps), the tool enables users to explore patterns and anomalies at varying levels of detail. The inclusion of anomaly score indicators and algorithm sensitivity glyphs further supports anomaly detection tasks. The effectiveness of this approach is validated through an ICE-T user study and case studies, demonstrating its utility in providing a comprehensive overview and detailed analysis of time series data. This tool successfully combines hierarchical navigation, adaptive detail, and anomaly detection visualization to offer a powerful solution for exploring complex time series datasets.

Acknowledgements

This work has been supported by the BMIMI, BMWET, and FFG, Contract No. 911655: ”ProFuture II”. The work by Julian Rakuschek and Tobias Schreck was partially funded by the Austrian Research Promotion Agency (FFG) within the framework of the flagship project ICT of the Future PRESENT, grant FO999899544.

References

1. Aghabozorgi S, Seyed Shirkhorshidi A and Ying Wah T (2015) Time-series clustering a decade review. *Information Systems* 53: 16–38. DOI:10.1016/j.is.2015.04.007.
2. Aigner W, Miksch S, Schumann H and Tominski C (2023) *Visualization of Time-Oriented Data, Second Edition*. Springer. DOI:10.1007/978-1-4471-7527-8.
3. Aigner W, Rind A and Hoffmann S (2012) Comparative Evaluation of an Interactive Time-Series Visualization that Combines Quantitative Data with Qualitative Abstractions. *Computer Graphics Forum* DOI:10.1111/j.1467-8659.2012.03092.x.
4. Ali M, Alqahtani A, Jones MW and Xie X (2019) Clustering and Classification for Time Series Data in Visual Analytics: A Survey. *IEEE Access* 7: 181314–181338. DOI:10.1109/ACCESS.2019.2958551.
5. Ali M, Jones MW, Xie X and Williams M (2019) TimeCluster: dimension reduction applied to temporal data for visual analytics. *The Visual Computer* 35(6): 1013–1026. DOI:10.1007/s00371-019-01673-y.
6. Bae J, Helldin T, Riveiro M, Nowaczyk S, Bouguelia MR and Falkman G (2020) Interactive Clustering: A Comprehensive Review. *ACM Computing Surveys* 53. DOI:10.1145/3340960.
7. Bernard J, Daberkow D, Fellner D, Fischer K, Kpler O, Kohlhammer J, Runnwerth M, Ruppert T, Schreck T and Sens I (2015) VisInfo: a digital library system for time series research data based on exploratory searcha user-centered design approach. *International Journal on Digital Libraries* 16: 37–59. DOI:10.1007/s00799-014-0134-y.
8. Berndt D and Clifford J (1994) Using Dynamic Time Warping to Find Patterns in Time Series. In: *Proceedings of the 3rd*

International Conference on Knowledge Discovery and Data Mining. AAAI Press, pp. 359–370.

9. Dau HA, Keogh E, Kamgar K, Yeh CCM, Zhu Y, Gharghabi S, Ratanamahatana CA, Yanping, Hu B, Begum N, Bagnall A, Mueen A, Batista G and Hexagon-ML (2018) The UCR Time Series Classification Archive. `https://www.cs.ucr.edu/~eamonn/time_series_data_2018/`.
10. Fekete JD, Fisher D and Sedlmair M (2024) *Progressive Data Analysis*. The Eurographics Association. DOI:10.2312/PDA.20242707.
11. Kohonen T (1988) *Self-organization and associative memory: 3rd edition*. Berlin, Heidelberg: Springer-Verlag. ISBN 0387513876.
12. Louis P, Mutlu B, Suschnigg J and Schreck T (2024) Visual Analysis of Cyclic Time Series with Semantic Zoom. In: *28th International Conference Information Visualisation (IV)*. IEEE, pp. 1–6. DOI:10.1109/iv64223.2024.00019.
13. Mauceri S (2020) *One-Class Time Series Classification*. PhD Thesis, University College Dublin.
14. Petitjean F, Ketterlin A and Ganarski P (2011) A global averaging method for dynamic time warping, with applications to clustering. *Pattern Recognit.* 44: 678–693. DOI:10.1016/j.patcog.2010.09.013.
15. Ruta N, Sawada N, McKeough K, Behrisch M and Beyer J (2019) Sax navigator: Time series exploration through hierarchical clustering. In: *Visualization Conference*. IEEE, pp. 236–240. DOI:10.1109/VISUAL.2019.8933618.
16. Saito T, Miyamura H, Yamamoto M, Saito H, Hoshiya Y and Kaseda T (2005) Two-tone pseudo coloring: compact visualization for one-dimensional data. In: *Symposium on Information Visualization*. IEEE, pp. 173–180. DOI:10.1109/INFVIS.2005.1532144.
17. Schmidl S, Naumann F and Papenbrock T (2024) AutoTSAD: Unsupervised Holistic Anomaly Detection for Time Series Data. *Proc. VLDB Endow.* 17: 2987–3002. DOI:10.14778/3681954.3681978.
18. Schmidl S, Wenig P and Papenbrock T (2022) Anomaly Detection in Time Series: A Comprehensive Evaluation. *Proc. VLDB Endow.* 15: 1779–1797. DOI:10.14778/3538598.3538602.
19. Shao L, Lengauer S, Miri H, Bedek M, Kubicek B, Kupfer C, Zangl M, Dienstbier B, Jeitler K, Krenn C, Semlitsch T, Zipp C, Albert D, Siebenhofer A and Schreck T (2023) Visual Document Exploration with Adaptive Level of Detail: Design, Implementation and Evaluation in the Health Information Domain. In: *Proceedings of the 18th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2023) - IVAPP*. SciTePress, pp. 133–141. DOI:10.5220/0011621800003417.
20. Shneiderman B (1996) The eyes have it: a task by data type taxonomy for information visualizations. In: *Proceedings Symposium on Visual Languages*. IEEE, pp. 336–343. DOI:10.1109/VL.1996.545307.
21. Stolper CD, Perer A and Gotz D (2014) Progressive Visual Analytics: User-Driven Visual Exploration of In-Progress Analytics. *IEEE Transactions on Visualization and Computer Graphics* 20: 1653–1662. DOI:10.1109/TVCG.2014.2346574.
22. Suschnigg J, Mutlu B, Burgholzer M, Bauer M, and Schreck T (2025) Cluster-Based Approach for Visual Anomaly Detection in Multivariate Welding Process Data Supported by User Guidance. In: *Proceedings of the 30th International Conference on Intelligent User Interfaces*. ACM, p. 325340. DOI:10.1145/3708359.3712076.
23. Suschnigg J, Mutlu B, Koutroulis G, Hussain H and Schreck T (2025) MANDALA - Visual Exploration of Anomalies in Multivariate Time Series Data. *Computer Graphics Forum* 44(1): 1 – 17. DOI:10.1111/cgf.70000.
24. Suschnigg J, Mutlu B, Koutroulis G, Sabol V, Thalmann S and Schreck T (2021) Visual Exploration of Anomalies in Cyclic Time Series Data with Matrix and Glyph Representations. *Big Data Research* 26: 100251. DOI:10.1016/j.bdr.2021.100251.
25. Tominski C, Donges JF and Nocke T (2011) Information visualization in climate research. In: *15th International Conference on Information Visualisation*. IEEE, pp. 298–305. DOI:10.1109/IV.2011.12.
26. Van Wijk J and Van Selow E (1999) Cluster and calendar based visualization of time series data. In: *Proceedings Symposium on Information Visualization*. IEEE, pp. 4–9. DOI:10.1109/INFVIS.1999.801851.
27. Wall E, Agnihotri M, Matzen LE, Divis KM, Haass MJ, Endert A and Stasko J (2019) A heuristic approach to value-driven evaluation of visualizations. *IEEE Transactions on Visualization and Computer Graphics* 25: 491–500. DOI:10.1109/tvcg.2018.2865146.
28. Ward JH (1963) Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association* 58: 236–244. DOI:10.1080/01621459.1963.10500845.
29. Warren Liao T (2005) Clustering of time series dataa survey. *Pattern Recognition* 38: 1857–1874. DOI:https://doi.org/10.1016/j.patcog.2005.01.025.
30. Wenig P, Schmidl S and Papenbrock T (2022) Timeeval: A benchmarking toolkit for time series anomaly detection algorithms. *Proc. VLDB Endow.* 15: 3678–3681. DOI:10.14778/3554821.3554873.
31. Wu R and Keogh EJ (2020) Current time series anomaly detection benchmarks are flawed and are creating the illusion of progress. *IEEE Transactions on Knowledge and Data Engineering* 35: 2421–2429. DOI:10.1109/TKDE.2021.3112126.
32. Yanez F, Conati C, Ottley A and Nobre C (2025) The state of the art in user-adaptive visualizations. *Computer Graphics Forum* 44(1): e15271. DOI:10.1111/cgf.15271.
33. Yoon Y and Myers BA (2015) Semantic zooming of code change history. In: *Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. IEEE, pp. 95–99. DOI:10.1109/VLHCC.2015.7357203.