# Relation-Driven Query of Multiple Time Series

Shuhan Liu, Yuan Tian, Zikun Deng, Weiwei Cui, Haidong Zhang, Di Weng, *Member, IEEE*, and Yingcai Wu, *Senior Member, IEEE*

***Abstract*—Querying time series based on their relations is a crucial part of multiple time series analysis. By retrieving and understanding time series relations, analysts can easily detect anomalies and validate hypotheses in complex time series datasets. However, current relation extraction approaches, including knowledge- and data-driven ones, tend to be laborious and do not support heterogeneous relations. By conducting a formative study with 11 experts, we concluded six time series relations, including correlation, causality, similarity, lag, arithmetic, and meta, and summarized three pain points in querying time series involving these relations. We proposed RelaQ, an interactive system that supports the time series query via relation specifications. RelaQ allows users to intuitively specify heterogeneous relations when querying multiple time series, understand the query results based on a scalable, multi-level visualization, and explore possible relations beyond the existing queries. RelaQ is evaluated with two cases and a user study with 12 participants, showing promising effectiveness and usability.**

***Index Terms*—Multiple time series query, time series relations, interactive visual query system, time series analysis.**

## I. Introcution

MANY research efforts [1], [2], [3] have been devoted to the interactive query of multiple time series, empowering users to find patterns from large-scale time series data quickly. Among the constraints used in such queries, time series relations, such as correlation [4], [5], causality [6], [7], and similarity [2], [8], are frequently employed to describe the patterns that span across multiple time series. For instance, when analyzing a stock dataset with multiple time series, an analyst may query two time fragments (e.g., one month) that have a local negative correlation while the two time series have a positive correlation overall (e.g., one year), as shown in the Fig. 1

Most of the current approaches for the interactive query of multiple time series identify the desired time series fragments based on the given thresholds [9], [10] and trend patterns [11], [12], [13], such as the *head and shoulders* patterns [14]. Nonetheless, while these approaches primarily measure the similarity between query specifications and the resulting time series, they unfortunately do not facilitate the specification of relationships amongst these time series. Due to the lack of interactive approaches for relation-driven time series queries, analysts often resort to writing lengthy scripts with statistical libraries (e.g., Pandas [15]) or tools (e.g., tradingview [16]), which can be laborious and error-prone. Moreover, analyzing the relations based on the queried time series fragments is also a difficult task: *How do these fragments distribute? Are the strengths of these relations consistent over time? Are there more relations between these fragments and the rest of the data?*

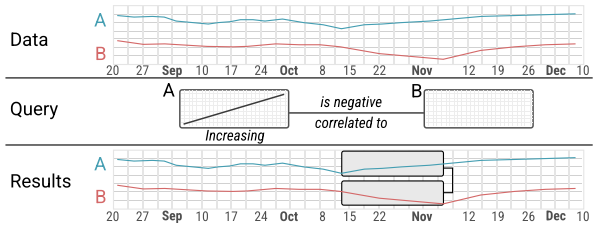


Fig. 1. An example of a relation-driven query. An analyst observed a global positive-correlated pair of time series (A and B) and sought local negative-correlated fragments. Finally, the result time fragments were highlighted.

To better understand the practice, challenges, and requirements of the relation-driven query of multiple time series, we conducted a formative study with eleven time-series analysts from different domains. This study helps us summarize a) in *what* time series relations the analysts are interested; b) *why* these relations are employed in multiple time series analysis; and c) *how* these relations are queried and support the workflows of the analysts. We conclude the following three challenges in designing a new relation-driven time series query approach from our observations.

*Diverse and heterogeneous time series relations:* In the formative study, we have identified six types of time series relations. Analysts must retrieve the multiple time series satisfying multiple types of relation constraints. Not only are new interactions required to enable the easy specification of the relations among time series, but a new retrieval algorithm is also needed to efficiently find the matches satisfying the different relations in one query.

*Intuitive interpretation of complex query results:* The results of relation-driven time series queries can be difficult to comprehend, given that many matches can be produced from a large-scale time series dataset. For example, analysts need to

Manuscript received 6 February 2024; revised 17 April 2024; accepted 25 April 2024. Date of publication 7 May 2024; date of current version 3 July 2025. This work was supported by the National Key R&D Program of China under Grant 2022YFE0137800, in part by the Key "Pioneer" R&D Projects of Zhejiang Province under Grant 2023C01120, in part by NSFC under Grant U22A2032, in part by the the Collaborative Innovation Center of Artificial Intelligence by MOE and Zhejiang Provincial Government (ZJU), and in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2023B1515120078. Recommended for acceptance by J. Yang. *(Corresponding author: Di Weng.)*

Shuhan Liu, Yuan Tian, Di Weng, and Yingcai Wu are with the Zhejiang University, Hangzhou 310027, China (e-mail: shliu@zju.edu.cn; yuantian@zju.edu.cn; dweng@zju.edu.cn; ycwu@zju.edu.cn).

Zikun Deng is with the School of Software Engineering, South China University of Technology, Guangzhou 510641, China (e-mail: zkdeng@scut.edu.cn).

Weiwei Cui and Haidong Zhang are with Microsoft, Beijing 100086, China (e-mail: weiwei.cui@microsoft.com; haidong.zhang@microsoft.com).

This article has supplementary downloadable material available at https://doi.org/10.1109/TVCG.2024.3397554, provided by the authors.

Digital Object Identifier 10.1109/TVCG.2024.3397554

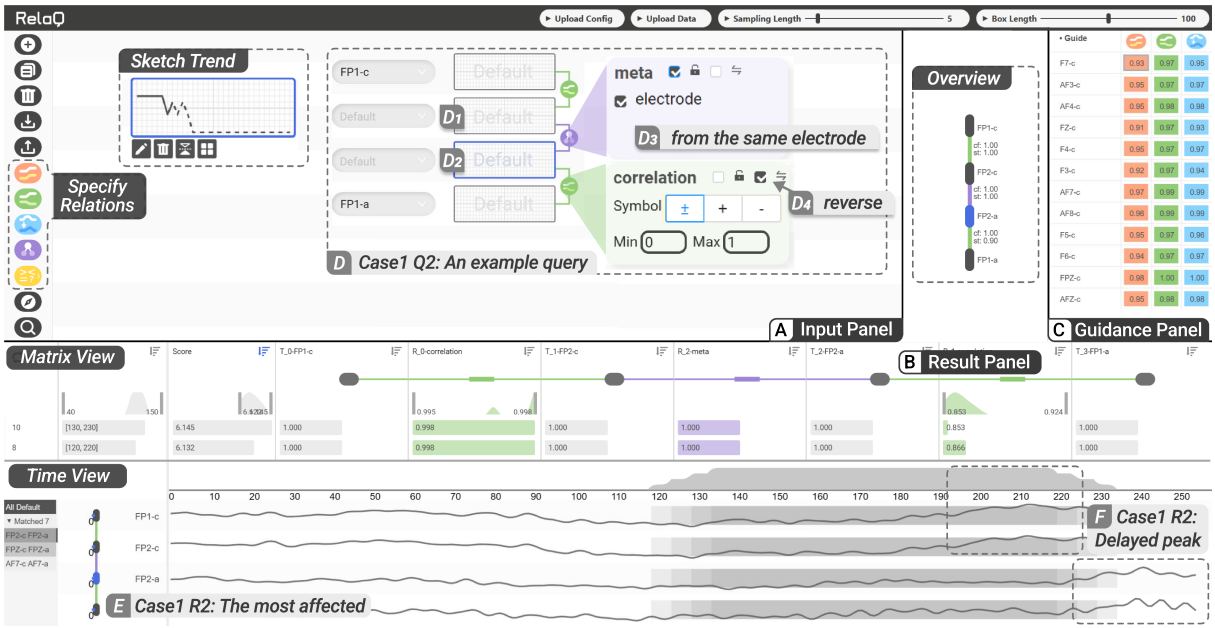


Fig. 2. The user interface of RelaQ. It is composed of three main parts: In the (A) *input* panel, users can sketch trends and specify relations. In the (B) *result* panel, users can obtain an overview, compare results in the matrix view, and inspect details in the time view. In the (C) *guidance* panel, there are recommended timeboxes. Besides, this figure also displays part of the first case in Section V-A: (D) an example query and (e-f) some patterns in its results.

check the fluctuation in the strengths of the queried relations to determine whether the relations become more consistent over time. Moreover, the incorporation of relaxed retrieval, which allows partial matching to increase the diversity of query results, also introduces the difficulty in understanding the structural changes in the resulting time series relations.

*Efficient generation of reliable relation suggestions:* Inspired by the query suggestions provided by modern search engines, the proposed approach should support the exploration of the potential relations among the time series of interest beyond what have been specified in the queries. However, given the sheer volume of time series and the diverse types of time series relations, it is challenging to efficiently generate and provide useful relation suggestions to complement the existing queries.

We propose RelaQ, an interactive approach that retrieves multiple time series based on their relations. To address the above challenges, RelaQ comprises a query interface (Fig. 2) that allows users to visually specify various relations among time series and processes the queries with a new search algorithm that supports matching time series with heterogeneous relation constraints. The query interface is composed of three parts (Fig. 2(A)–(C)): an input panel, a result panel, and a guidance panel. A scalable visual design of the result panel, depicting the topologies and characteristics of the matched results, is incorporated to visualize the resulting time series at different levels of detail, allowing users to perform multiscale analysis. Moreover, the guidance panel displays an overview of the suggestions that extend the existing queries with more relations and time series.

Our contributions are summarized as follows:

- A formative study that discusses the scope of time series relations and their applications and summarizes the challenges and requirements in the relation-driven query of multiple time series;
- A novel approach that combines a fuzzy query model and an interactive interface to support the easy formulation of heterogeneous relation constraints, the flexible query of multiple time series based on specified relations, and the intuitive interpretation of the query results.

## II. Related Work

We mainly reviewed time series studies on query specification, query matching, and query results visualization.

### A. Time Series Query Specification

Specifying a query is the first step of time series retrieval. Intuitive specification approaches facilitate users to formulate queries efficiently. According to different input forms, existing methods can be divided into four categories: text-, value-, example-, and sketch-based query.

The **text-based** query allows expressing desired patterns through natural language or regex. Agrawal et al. [11] were the first to propose a shape definition language (SDL) that describes time series patterns. SDL assisted in formulating a time series into trend shape symbol sequences (e.g., up, down, and stable). In recent years, there have also been many methods to support regex [17] and constrained natural languages [18]. However, text-based methods suffer from the concern of learnability and intuitiveness because users must convert imagined patterns into a certain abstract language.

The **value-based** query requires to set value bounds to query time series passing through. TimeSearcher [9], [19] was the first to adopt timeboxes to identify value constraints. A timebox is a rectangle, where the horizontal axis represents the length constraints of a time range, while the vertical axis represents the value bounds of time series data.

Later, the **example-based** methods extensively consider time series within a timebox as examples for querying similar patterns. Many models built based on examples make the real-time performance of querying time series significantly improved, such as PSEUDo [2], PEAX [1], and KD-Box [3]. However, value- and example-based methods are not flexible to describe various trend shapes like *"head-and-shoulder"* , especially when the desired example is hard to locate.

In the **sketch-based** query methods, users can sketch desired trend shapes to retrieve similar time series. We followed Mannino et al. [14] and divided existing sketch-based approaches into three categories: overlays, annotated sketches, and shape-restricted sketches. Overlaying approaches [12], [13], [20] leveraged the predefined time and amplitude axes to make the scale clear but were highly dependent on the selected reference time series. Annotated sketches approaches (e.g., Qetch [14], Holz et al. [21]) allowed users to freely hand-draw desired patterns on blank canvas but required providing scale or pattern descriptions. Many visual query systems allowed users not to provide scale annotations but to adaptively match patterns based on restricted shapes [22], [23]. These visual query systems had good robustness, but slacked scale constraints led to a high recall rate. While sketching is useful and expressive, none of the existing methods supports specifying multiple time series and their relations. Users cannot always sketch what they want, such as two correlated time series.

In addition, existing methods prioritize similarity between the input query and output results but ignore relations among different time series in one query. Thus, we are inspired to propose RelaQ, which supports querying multiple time series with heterogeneous relations. RelaQ is based on sketching because it describes temporal features expressively.

### B. Time Series Query Matching

Most **general time series query matching** approaches focus on retrieving similar time series, so many methods have been proposed to evaluate the similarity between the query and the results. Euclidean distance (ED) [24] and dynamic time warping (DTW) [25] are popular for measuring distances between time series. According to the survey of Ding et al. [26], DTW is the best among all distance measures, but ED is faster so can be used when the scale of data increases. Both ED and DTW require a sliding window technique. Some researchers also transform time series into symbolic sequences and exploit string-matching techniques. Symbolic Aggregate approXimation (SAX) [27] is a proven efficient tool. SAX compresses the length by segmenting the time series and reduces the cardinality by discretizing and symbolizing the values. There are also many approaches based on machine learning techniques [13], [28] or designed for special situations [3]. For large-scale time series data, there are mature commercial database query tools available, including time series databases (e.g., InfluxDB [29], Timestream [30]), relational time series databases (e.g., TimescaleDB [31]), and general search databases (e.g., Elasticsearch [32]).

As for **relation-driven query matching**, despite many studies devoted to helping query, extract, and analyze relations between time series, most are designed for certain relations, such as correlation [33], [34], causality [35], [36], co-occurrence [37], [38], [39]). In other words, algorithms and tools that support matching heterogeneous relations have still not been sufficiently studied. If users want to obtain complex relations in real analysis scenarios, they can only return to scripts or commercial tools, such as tradingview [16]. Thus, we propose a flexible algorithm that simultaneously matches trends and heterogeneous relations, enabling RelaQ to support complex relation-driven queries.

### C. Dynamic Graph Visualization

Intuitively visualizing query results is beneficial for users to refine queries or perform further exploration. Query results are dynamic graphs of time fragments (nodes) with changing relations (links), like two stocks that may shift from positive to negative correlation and become unrelated over time. Beck et al. [40] developed a three-level taxonomy for dynamic graph visualization techniques. The first level includes three types: animation, timeline, and hybrid.

As relations are integrated closely with time lags and trends, a timeline-based visualization could enable time-oriented exploration of query results. Hence, we mainly reviewed linear timeline techniques [41]. Burch et al. [42] embedded a timeline into each cell of the matrix design, while Bach et al. [43] stacked matrices to a 3D cube by timeline. Besides, Hlawatsch et al. [44] introduced a visualization based on adjacent lists. However, neither matrix-based nor list-based design is intuitive for depicting time lags. The node-link-based designs mapped time to space and created lag-aware static images, which assist users in understanding the temporal graph evolution. Beck et al. [40] categorized these techniques into juxtaposed [45], [46], superimposed [47], or integrated [48] layouts.

Moreover, since the node-link-based visualization techniques underperform on the comparison task, we consider integrating matrix-based methods. Inspired by LineUp [49], we design an exceptional matrix-node-integrated visualization to balance the data nature and user experience. It also supports flexible decision-making on dynamic temporal graphs.

## III. Formative Study

This section presents a formative study that aims to discover the pain points and needs in the time series query from the relation perspective and further compile the user requirements that guide the development of a relation-driven time series retrieval tool. By interviewing time series analysts, we attempted to gather insights on three research questions: a) *what* time series relations were considered important to their daily analysis tasks; b) *why* their analysis tasks required these time series relations; and c) *how* they retrieved and leveraged these time series relations to support their analysis tasks.

### A. Method

To answer the above questions, we first conducted a literature survey to search for the types of time series analysis tasks and narrow down the scope of time series relations. Then, we prepared seven interview questions and conducted semi-structured interviews individually with eleven time series analysts. Finally, we analyzed their responses with the thematic coding approach [50] and compiled three major user requirements that supported the design of the proposed tool.

*Literature survey:* To produce an initial list of time series relations, we searched for the papers published in the prominent data mining, visualization, and human-computer interaction conferences and journals that contained keywords including *multiple time series* and *time series relation*. The surveyed conferences and journals include but are not limited to ACM KDD, IEEE VIS, IEEE TVCG, ACM CHI, etc. Two co-authors with years of experience in time series analysis went through 112 papers separately and tagged the keywords that described the time series relations used in the papers. After reconciling and merging the keywords, we summarized ten relations: *similarity*, *correlation*, *causality*, *co-occurrence*, *precedence*, *cascade*, *lag*, *cycle*, *hierarchy*, and *group*.

*Semi-structured interviews:* To refine the preliminary scope of time series relations and understand how these relations were involved in the time series analysis tasks, we recruited eleven time series analysts (five male and six female) as participants and conducted semi-structured interviews. The participants had diverse backgrounds such as energy 1), robot simulation 2), urban computing 1), stock trading 2), cloud services 3), and sports computing 2), and each participant had a minimum of two years of experience in analyzing time series data. They did not receive monetary compensation for participating in this formative study. The interviews were conducted via video telephony due to pandemic restrictions and followed the questions listed below (`M` denotes multiple-choice questions, `S` denotes single-choice questions, and `O` denotes open-ended questions):

- Q1. `[M]` Which time series analysis tasks do you usually work on?
- Q2. `[S]` How many times per week do you work on time series tasks?
- Q3. `[O]` What is the scale of the time series data (in the order of magnitude) you are dealing with?
- Q4. `[M,O]` Which relations do you want to retrieve during the analysis process?
- Q5. `[O]` Which insights do you gain from these relations, and how important are they?
- Q6. `[O]` How do you typically retrieve these time series relations in your data? Is there any limitation?
- Q7. `[O]` In existing approaches, are there any advantages or functions you prefer? After retrieving time series relations, what else will you explore?

Q1–3 were designed to learn the participants' expertise and routines, including their familiar analysis tasks (Q1), experience profile (Q2), and data scale (Q3). Q1 has 9 choices (Fig. 3(A)) based on the prior surveys of time series analysis tasks [51], [52]. For Q3, we asked the participants to estimate the average length and the number of time series. Q4–7 were designed to solicit the participants' opinions on the scope of time series relations, including the desired relations in time series query (Q4), the usages and importance of the relations (Q5), the prior approaches used in the query (Q6), and analysis preferences and requirements (Q7). Q4 is a multiple-choice and open-ended question, where we first asked the participants to choose from our initial list of time series relations and then asked the participants if some relations should be removed and/or more relations should be added. Each interview lasted between 30–50 minutes. The participants' responses were recorded and coded with the thematic coding approach [50].

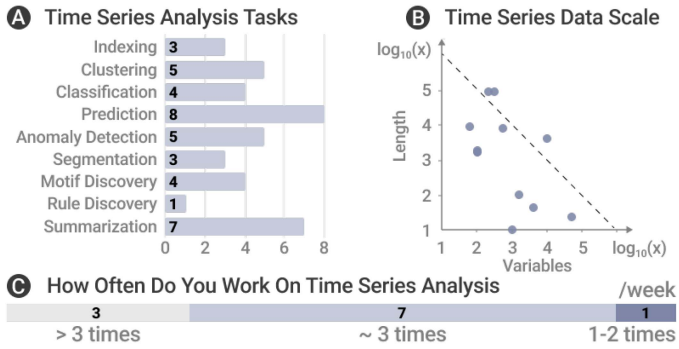


Fig. 3. Investigation responses from time series analysts. (A) Typical time series analysis tasks that participants work on, all nine types of tasks are covered. (B) The scale of time series data that participants usually deal with, the number of **variables** × the **length** of a single time series. (C) Frequency that analysts analyze time series in a week, reflecting all participants are very familiar with time series data.

### B. Results

*1) Participants' Backgrounds (Q1–3):* Fig. 3(A) displays experts' familiar time series analysis tasks (Q1). Most of the participants are familiar with more than two types of analysis tasks, and prediction and summarization are the most common. Though only P5 has worked on rule discovery, all other analysis tasks are mentioned by at least three participants.

We investigate participants' experience profiles via their working frequency (Q2). Fig. 3(C) reveals that most participants maintained a frequency of three or more times a week.

To paint a clearer picture of the participants' analysis work, we survey the scale of the time series data (Q3). Since multiple time series have two dimensions (the number of series and the length of each time series), we asked participants to describe both. According to Fig. 3(B), the number of time series ranges in order of magnitude from 10 to $10^4$, while the lengths range in order of magnitude from 10 to $10^5$.

Thus, the diverse backgrounds and high average proficiency of our participants make us confident that various domains and tasks are covered in our formative research.

*2) Time Series Relations Scope (Q4-5):* We summarized six categories of relations, including *similarity*, *correlation*, *causality*, *lag*, *meta*, and *arithmetic*, based on the participants' responses. *Co-occurrence*, *precedence* and *cascade* were removed because they can be represented by the combination of *lag* and

TABLE I
THE GRANULARITY AND ADOPTED CALCULATION METHODS OF THE TIME SERIES RELATIONS

| Relations | Granularity | | | Calculation Methods | |
|---|---|---|---|---|---|
| Names | P | F | S | RelaQ Adopted | Domains |
| Correlation | × | ✓ | ✓ | Pearson Coefficient [53] | [-1.0,+1.0] |
| Similarity | ✓ | ✓ | ✓ | Euclidean Distance [24] | [+0.0,+1.0] |
| Causality | ✓ | ✓ | ✓ | Granger Test [54] | [+0.0,+1.0] |
| Lag | ✓ | ✓ | ✓ | Attached to other relations | |
| Meta | ✓ | | | Predefined by users | 0-N, 1-Y |
| Arithmetic | ✓ | ✓ | ✓ | $\{\sum, Avg, Var, Min, Max\} \times \{\geq, \leq, =\}$ | |

All relation strengths are normalized.

other relations; *cycle* was removed because it mainly focuses on the temporal characteristics of individual time series; and *hierarchy* and *group* were aggregated into *meta*, which captures the latent semantic relations between time series. Besides, we added *arithmetic* according to participants' analysis requirements.

We started the introduction of the scope from time series' different granularity, from high to low: time **S**eries, **F**ragments, and **P**oints. The strengths of a relation vary on different granularity (see Table I). This section will discuss the definition, usage, and calculation of time series relations.

*Similarity:* Similarity between two time series refers to the degree to which they exhibit comparable patterns and behavior. The similarity relation is widely used in many analysis tasks, especially those that depend on extracting commonalities or differences. For tasks including clustering (P1, P3, P6, P8, P10), indexing (P2, P4), and prediction (P1, P11), time series with high similarity are important. For similarity-aware anomaly detection (P4-5, P8), participants are interested in time series with low similarity. According to the interviews, Euclidean distance [24], dynamic time warping [25], and MAPE [55] are commonly used similarity measurements.

*Correlation:* Correlation between two time series refers to the degree to which they exhibit an associated relation with each other. Correlation is one of the most important relations serving various time series analysis tasks, such as clustering (P1, P3, P6), classification (P6-7), prediction (P3, P6), anomaly detection (P6), and motif/rules discovery (P4-5). There are two categories of correlations: linear and non-linear. The most commonly used way to calculate the linear correlation is Pearson coefficient [53]. Also, there are approaches for non-linear correlation (SROCC [56] and Copula [57]) and specific domain measures (P7:PLV [58]).

*Causality:* Causality between two time series refers to the relation where one time series, known as the cause, has a direct influence on the other time series, known as the effect. The causality is critical in the prediction task (P1, P5-6, P11). Participants often query time series with causal relations in history to validate predicted results. Besides, it is sometimes adopted to help mine motifs or rules as well as detect anomalies (P5-6). The causality is usually established by causal inference techniques, such as Granger causality [54] or Bayesian networks [59]. The strength can be computed via p-value and Bayesian probability, respectively.

*Lag:* Lag relation between two time series refers to the temporal delay or phase shift between them. The lag relation reflects the time dimension independently and can be combined with other five relations. Participants usually query lag relation in analysis tasks to obtain temporal features, for example, prediction (P3, P6-7, P9-10) and periodical motif discovery (P4, P7). The time lag is usually calculated by cross-correlation [60], self-correlation (P7), n-derivation (P3), or inferred from domain knowledge (P7, P9-10).

*Meta:* The meta relation between two time series refers to the semantic or contextual attributes that they share, usually about hierarchies and categories. For example, two time series representing daily births in LA and California state may have a meta relation due to their hierarchical structure. The meta relation provides contextual information and helps understand time series in context. Participants usually adopt it as a kind of filter. It is widely used in time series analysis tasks, such as clustering (P1, P3), classification (P3), and semantic anomaly detection (P8). Participants usually measure it through predefined data configuration.

*Arithmetic:* Arithmetic relation between two time series refers to the statistical features. Specifically, the arithmetic relation is usually composed of an operator and a comparator. Participants usually query arithmetic relations to locate time series with specific statistical features. For example, the average (operator) value of time series A is greater (comparator) than that of B. According to the interview, participants query for arithmetic relations in many analysis tasks: classification (P7), prediction (P9), and anomaly detection (P8, P10). Commonly used operators and comparators are listed in Table I.

*3) User Requirements Analysis (Q6-7):* Based on the participants' responses, we summarized two types of queries (target- and breadth-oriented) and three user requirements.

*R1. Straightforward query of desired time series:* According to the participants' responses, a straightforward approach is required to support intuitive query formulation and flexible query processing since there were many limitations in their daily-used approaches.

In terms of query formulation, most participants found describing the relation among time series to be the most challenging part. Currently, the method for specifying relations involves writing scripts and rules, which participants described as *"an empirical work"* and a *"heavy mental burden"*. They must transform abstract relations into concrete rules (e.g., if-else statements, P2, P7, P11) or accurate parameters (P3, P4-6, P8). Many participants struggled with this transformation and expressed a desire to describe relations directly. Additionally, participants discussed constraints, such as values and trends, that describe the inner features of time series. While many participants agreed that sketching trends to query time series was expressive (P1-2, P5−7, P11), they found it challenging to adopt this approach in their analysis workflow. P5 explained that she often needed to search for correlated time series with partly unknown trends, which she could not sketch entirely. Nevertheless, participants still found sketching trends to be an intuitive approach and expressed a desire to use it.

In terms of query processing, participants expressed concern about processing strategy, highlighting the importance of flexible processing in reducing tedious query constraint refinement. They were not always confident that their queries would be entirely correct on the first attempt (P2, P5). Some participants suggested a fuzzy-match approach to processing queries, as it could include more potential results (P1, P3).

*R2. In-depth understanding of query results:* Participants reported that the in-depth understanding of query results primarily involves evaluating and identifying queried patterns within time series. Thus, a comprehensive and interactive display of results is essential for supporting such a requirement.

Nearly all participants expressed a desire to prioritize query results based on diverse metrics, such as confidence (P8), similarity (P9–11), correlation (P1, P11), and matching score (P3, P7). However, they often had to manually calculate and balance between multiple metrics. P1, a stock analyst, reported that homogeneous evaluation criteria can affect feature selection and thus prediction results, prompting a desire to identify evaluation metrics dynamically. Overall, an interactive approach is needed to support flexible evaluation, allowing customizing metrics based on specific needs.

Moreover, some participants found it cumbersome to locate desired patterns accurately in query results. For example, P9, a software analyst, told us that she usually visualized results manually using matplotlib, then inspected the beginning timestamp of anomaly increasing by naked eyes. To overcome the limitations of such a tedious and inaccurate inspection process, an intuitive display of query results is needed to support the analysis of detailed time series and relations.

*R3. Exploration based on reliable guidance:* The majority of participants reported that the current query process was time-consuming. Upon further investigation into participants' workflow, we identified the most tedious stage: exploring desired queries. This often occurs when analysts did not have a specific target in mind, and therefore needed to explore desired queries by iteratively finding related relations and extending existing queries. It differs from the situations previously discussed in **R1**: participants could describe a clear target, such as querying two correlated stocks. Taking inspiration from Heer et al. [61], we categorized these two types of queries as breadth-oriented and target-oriented, respectively, with participants needing to perform both types for complex time series analysis tasks.

Breath-oriented queries required participants to find new related relations and extend existing queries in large-scale time series data. Due to complexity, participants reported getting lost in the exploration process (P2). Therefore, many participants (P1-2, P4, P6, P11) expected reliable guidance. Specifically, step-by-step guidance is preferred (P2, P4) as it gives them full control over the query.

## IV. RELAQ

This section presents RelaQ, an interactive time series retrieval tool based on relation queries, which was carefully designed to support the requirements summarized in Section II-I-B3. The workflow of using RelaQ is illustrated in Fig. 4. First, users can upload a dataset consisting of multiple time series, along with a configuration file describing the semantic labels (e.g., group of belonging) for each time series. RelaQ preprocesses the uploaded data into compressed time series, relation indexes, and trend indexes. Second, users can specify query constraints among interested time series with the input panel, and RelaQ finds and returns matches for the input query with a search model. Third, RelaQ can provide guidance in query specification by offering additional time series relations within the dataset. RelaQ is implemented based on React-Redux-TS [62] and Flask-Python [63].

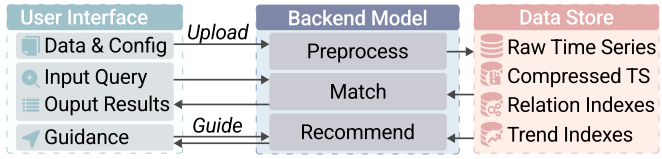


Fig. 4. The overview of RelaQ's workflow. Users upload a multiple time series dataset and a configuration file to RelaQ, which preprocesses the data, builds various indexes, and allows users to specify queries. The system can also provide guidance by recommending query constraints.

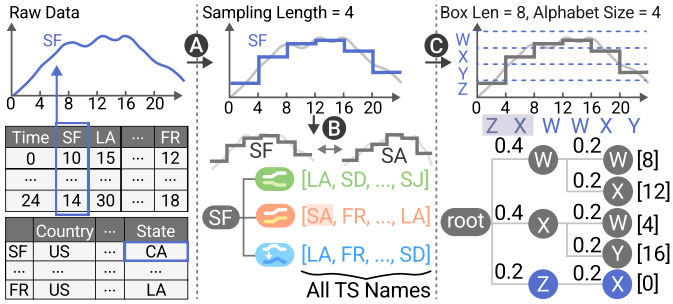


Fig. 5. This shows a three-step data preprocessing method using an example. The raw data contains multiple time series (*SF, LA,..., FR*) and label descriptions (*SF's State being CA*). (A) First, time series are compressed by taking the average of segments with Sampling Length=4 so the blue line shows compressed data. (B) Second, relation indexes are computed. We compute the relation strength between each pair of time series and record all time series names in the descending order of relation strength on the whole length. e.g., *SA is the city with the highest similarity strength with SF*. (C) Third, trend indexes are computed. We transformed compressed data into symbolic sequences using SAX [27] (alphabet size = 4). The sequence (*ZXWWXY*) is built as a trie (*depth = 8/4 = 2, box length = 8, sampling length = 4*). e.g., *The starting points of ZX contain [0], and the ratio is 0.2*.

### A. Preprocessing Data

Before querying, users should upload two CSV files - one dataset and one configuration file. The dataset file consists of rows representing time points, with the first column indicating timestamp and each of other columns containing data for time series. The configuration file describes labels (e.g., SF's State is CA) for each time series in each row. RelaQ preprocesses the uploaded dataset with the following three steps (see Fig. 5).

First, we normalize and compress time series (Fig. 5(A)). Users should specify Sampling Length, the length of each segment of the time series divided for compression. RelaQ uses PAA [64], which takes the average of segments to compress

time series. Second, we calculate relation indexes (Fig. 5(B)). We enumerate pairs of compressed time series and calculate their correlation, similarity, and causality strength via methods listed in Table I on the whole length. All time series' names are recorded in the descending order of relation strength. Third, we calculate trend indexes (Fig. 5(C)). We repeat the first and second steps but using Z-normalization instead of min-max-normalization and transform data into symbolic sequences via SAX [27] (alphabet size $= 4$, the size of symbols' set). Users should specify Box Length, the desired length of time segments the pattern should continue. Symbolic sequences are then divided into many segments. The division adopts a sliding window technique: the sliding step is one symbol, while the window size depends on the box and sampling length (e.g., $8/4 = 2$, round when indivisible). Symbolic segments are built into a trie, where each node records possible next symbols and their occurrence ratio. The segment starting is also recorded at the leaf node.

If the preprocessing takes longer than 2 minutes, it will be moved to the background for continued computation, allowing the user to begin querying. When a query involves uncomputed indexes, the computation of those indexes will be prioritized.

*Justification:* There are two types of parameters in RelaQ, pre-defined (alphabet size) and user-specified (sampling length and box length). The selection of alphabet size considered the user's mental burden and the refinement. We choose $\alpha = 4$, trying to distinguish not only increasing/decreasing trends but also distinguish steep/gentle slopes. As for user-specified parameters, manual clarification or automated adaptive matching would be a trade-off. On the one hand, when the time step unit of the dataset has a clear semantic meaning, specifying a definite length can make the query clear. For example, if the time step unit of the dataset is "days", querying the daily air pollution index for a week would require a box length of "7 days" and a sampling length of "1 d". In this case, the user-specified parameters have concrete semantics, and a slight variation to 5 days may lead to ambiguity since there is a difference between weekdays and a whole week. On the other hand, when the semantic impact of the time step unit is not significant, especially when precision is crucial for queries, such as in milliseconds, the length of the timebox may affect the final query results. As the overall design of RelaQ aims to minimize vague constraints in query specification and enable users to describe a query more directly, we have adopted a design that allows users to choose the length by themselves.

### B. Formulating Queries

The first step to serve easy query is enabling intuitive specification of query constraints (**R1**). Constraints in RelaQ consist of two parts: inner-constraints (trends: *the variation of a time series*, names: *each name maps a specific time series*, values: *the value domain of a time series*) that reveal inner features of a single time series and inter-constraints (*relations*) that indicate relation features among multiple time series.

*Basic settings:* RelaQ uses two axes settings for organizing queries: the horizontal axis encodes a timeline, and the vertical axis is divided into multiple time series tracks. Relations are

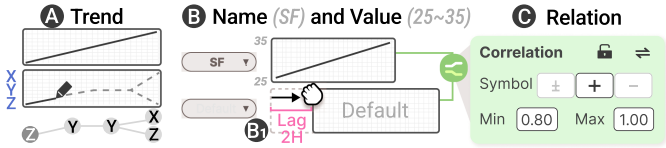


Fig. 6. An example query illustrates inner- and inter-constraints. "Which city's air quality index (AQI) will rise two hours later, following an upward trend of SF's AQI ranging in [25,35], and have a correlation strength greater than 0.8? " The inner-ones include (A) rising (trend), (B) SF (name), and [25,35] (value). The inter-ones include (C) "2 hours later" (lag) and greater than 0.8 (correlation). The phrase "which city" implies a default timebox.

encoded using color hue and stroke width to encode the type and strength, respectively. RelaQ supports two-mode searches to persist flexibility. The fuzzy search allows partial matching of constraints and slight differences, while the strict search requires all constraints to be satisfied. RelaQ commonly matches strictly unless users check the fuzzy option.

*Inner-constraints:* RelaQ employs timeboxes, following the approach in [19], to encode inner-constraints including trends, names, and values. Users can sketch trends as straight line segments on meshed timeboxes (Fig. 6(A)) and select names in the interface (Fig. 6(B)). Once the name is selected, the value range is determined by the minimal and maximal value of the name's corresponding time series. The aspect ratio of sketched lines is fixed under the strict match mode, while it only indicates trends (e.g., increasing) under the fuzzy match mode. To reduce users' mental burden, RelaQ offers two features. First, it recommends frequent trends via trend indexes and displays them as dashed lines in the timebox, which change dynamically when sketching (Fig. 6(A)). Second, all inner-constraints can be empty and will be matched automatically. Empty timeboxes with no inner-constraint are called default timeboxes. A default timebox is useful when a user wants to emphasize the relation. For example, *"querying two positively correlated time fragments"* only emphasizes their relation should be the correlation but ignores inner-constraints.

*Inter-constraints:* RelaQ facilitates the explicit specification of relations as outlined in Section III-B2, with relation icons listed in the input panel (Fig. 2(A)) for easy selection and addition of relation constraints. Colored lines with relation icons are the visual representation of inter-constraints, linking timeboxes. Since these lines display the relations between time series, we named them relalinks. The relation interactions cover six types summarized above and support free combinations. Being different from the other five relations, the lag relation pertains to the time dimension, so we allow users to set lags by dragging timeboxes horizontally (Fig. 6($B_1$)) instead of clicking a certain icon. Users can also adjust thresholds and customize options (Fig. 6(C)) to allow fuzzy matching and reverse matching (e.g., minimum similarity).

*Justification:* We mainly justify the design alternatives of constraint specification interactions. As discussed in Section II, Mannino et al. [14] identified that there were three types of sketch approaches to query time series: a) overlay sketches, b) annotated sketches, and c) restricted sketches. Overlay sketches

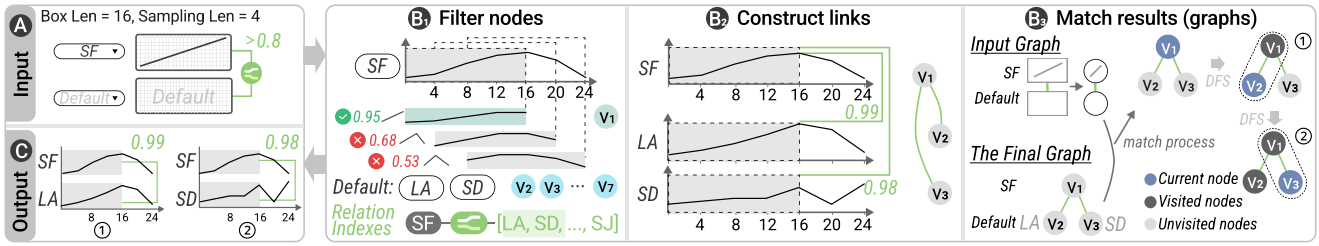


Fig. 7. This figure displays a simplified example query: *"Which city has a temperature that rises together with San Francisco and has a correlation strength above 0.8"* (A) The user input includes two timeboxes, [name (SF), trend (rising)] and [default (which city)], as well as relations, [correlation (above 0.8)] between SF and the default timebox. ($B_1$) RelaQ filters time fragments based on trends and records them as nodes. $v_1$ is the only node that meets the rising trend with a degree of 0.95 in SF, while $v_{2-7}$ are all valid for the default timebox. ($B_2$) For all pairs (SF-default) of nodes, relations are checked. The correlation strength between $v_1$ (SF) and $v_2$ (default) is more than 0.8, so we construct a link. The same is between $v_1$ and $v_3$. ($B_3$) RelaQ uses a depth-first search on the final graph and obtains two results. (C) Two results reveal that SF will affect LA and SD with a strength of 0.99 and 0.98, respectively.

rely on a specific reference time series, so the name must be specified, limiting the flexibility. Annotated methods can lead to visual clutter when multiple time series with labels are not placed clearly and organized. In contrast, restricted sketches [22] focus on capturing the essential features. Restricted sketches can make the sketch more visually appealing and easier to understand for users at a glance. We take clarity and simplicity as key considerations when designing the specification approach, so we finally adopt restricted sketches.

### C. Processing Queries

Besides the aforementioned specification approaches, an effective model that allows matching heterogeneous constraints flexibly is also necessary to support easy query (**R1**). The query process is end-to-end: Every time a user edits a query, RelaQ searches for all matched results via a three-step matching algorithm. An example input and output are displayed in Fig. 7(A)(C), while Fig. 7($B_{1-3}$) reveal the matching process.

Since dynamic relations that change over time are often modeled as graph problems, we are inspired to leverage the node-link graph as a proxy to model this matching problem. For each input graph, nodes are timeboxes, and links are relalinks. To distinguish, in the output graph, nodes are fragments, and links are relations. Thus, this problem is to match the input graph on the constructed dataset graph.

The construction of the dataset graph and matching are integrated and progressive. The method is three-step (Fig. 7(B): 1) filter valid fragments as nodes, 2) construct links based on relations, and 3) match final results. To simplify the narration, we illustrate the method with an example of a single non-dynamic relation. For more detailed explanations of dynamic-relation queries, please refer to the supplementary material.

First, RelaQ filters fragments based on timeboxes $x_i$. For each timebox, we obtain the compressed time series according to its *name* and use a sliding window technique to divide it into several fragments. The window size is the box length, while the sliding step is the sampling length. Then, we measure the *trend* matching degree based on ED [24]. If the degree exceeds threshold $s = 0.7$, we record the fragment as a valid node in the set $X_i$. For example, regarding timebox $x_1$ belonging to SF, only $v_1 \in X_1$ matches the rising trend with the degree 0.95 in Fig. 7($B_1$). When the trend constraint is not declared, the matching degree will be set as 1. For the default timebox with no name, RelaQ enumerates the first 20 time series in the relation indexes. For instance, in Fig. 7($B_1$), LA and SD are the first two cities with the highest correlation to SF.

Second, RelaQ builds links based on relalinks $y_k$, which links $x_i$ and another timebox $x_j$. We obtain $X_i$ and $X_j$ valid nodes computed in the first step. For each relation $y_k$, we validate each pair of compliant fragments (nodes) in $X_i \times X_j$, checking lag constraints and computing relation strength on the fragment length via discussed formulas in Table I. If the relation strength of a pair is over user-given threshold of $y_k$, we reserve it as a valid link in the set $Y_k$. For instance, the relation strength of link $e_1 \in Y_1$ between $v_1 \in X_1$ and $v_2 \in X_2$ is 0.99 in Fig. 7($B_2$). The final dataset graph is $G = (X, Y)$.

Third, RelaQ uses depth-first search [65] with memoization pruning techniques to form results. The search process is to match the input graph in the built dataset graph. The search order is based on the temporal order of timeboxes. RelaQ starts the search from the earliest timebox in a query. Each result is a connected subgraph $g = (v, e)$ in $G$. In Fig. 7($B_3$), the search starts from $v_1$, then transfers to $v_2$, forming the first result, and ends on $v_3$, forming the second result.

Besides, the matching score $r(g)$ is defined as (1), here $d(v)$ is the trend matching degree of node $v$, and $l(e)$ is the relation strength of link $e$. In Fig. 7(C), the matching score of the first result is $0.95 + 1 + |0.99| = 2.94$.

$$r(g) = \sum_{v \in X} d(v) + \sum_{e \in Y} |l(e)| \tag{1}$$

### D. Interpreting Results

Intuitive visualizations of queried results are critical to helping users understand retrieved time series and underlying patterns. To satisfy **R2**, RelaQ should 1) visualize queried time series with relations and 2) enable comparing different results flexibly. RelaQ adopts a matrix view that compactly organizes results in a matrix for comparison and a time view that depicts results along the timeline in a temporal context.

*The matrix view (Fig. 8)* is inspired by LineUp [49], a widely used technique for comparing multidimensional items. In the matrix view, each row is a result and each column is a fragment

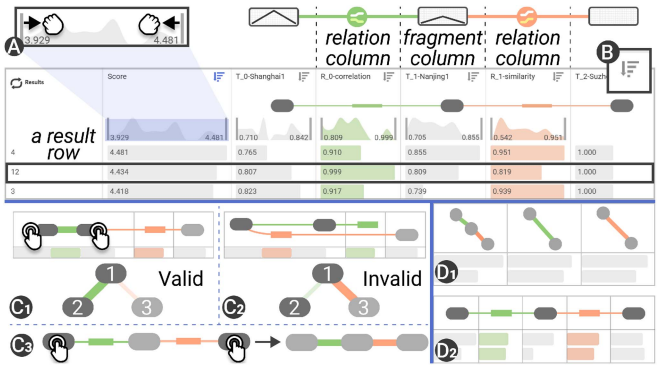


Fig. 8. The visual design of the matrix view. Each row of the matrix is a result and each column matches a relation or a fragment. There are three interactions: (A) filter the range of distribution, (B) rank the results by a specific column, and ($C_{1-3}$) combine some columns. ($C_1$) shows a valid combination, a green relation links fragment1 and fragment2, while ($C_2$) is invalid, where the orange relation links these two fragments incorrectly. ($C_3$) reveals the auto-combination feature of RelaQ. ($D_1$) is a design alternative based on juxtaposed graphs and ($D_2$) is the dismantled graph we adopt.

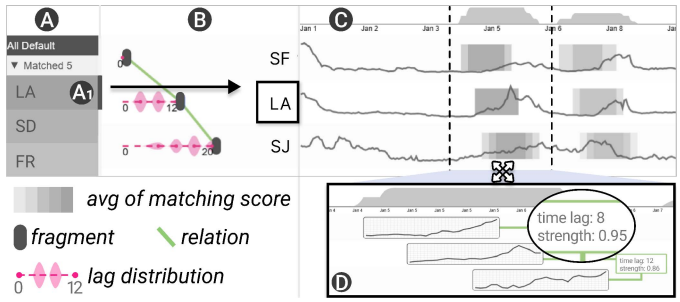


Fig. 9. The visual design of the time view. (A) is the list of alternative time series matching the default timebox in the decreasing order of its average matching score. (B) is an overview of all results. (C) is the distribution of results and raw time series. (D) shows the detail level of a result.

(the result of a timebox) or a relation (the result of a relalink). Since fragments and relations together display a pattern, we connect the columns of fragments and relations with links to keep consistency with the input query. The column header displays the distribution of strengths for relations or trend-matching rates for fragments. Users can adjust the range of distribution to filter results by moving the slider (Fig. 8(A)). Each column has a sorting icon that (Fig. 8(B)), when clicked, will arrange the results in descending order based on the matching score of that column. Clicking the icon again will switch to ascending order. To further assist users in comparing patterns that have multiple relations and fragments, we allow them to combine multiple columns. RelaQ enables users to combine columns by simply clicking on them. For instance, see Fig. 8($C_1$), users can click the first three columns (two fragment columns and a correlation column) to view a sub-pattern, such as the correlated portion of a query. However, some combinations may not be valid (Fig. 8($C_2$)), that is, when relations and fragments are linked incorrectly, patterns may be meaningless. Therefore, RelaQ employs an auto-combination mechanism to ensure meaningful sub-patterns are grouped. When users combine any two fragments, RelaQ will find all relations and fragments between these two fragments and combine them all. As shown in Fig. 8($C_3$), when the user clicks on the two end fragments, all the fragments and relations that connect them in between will be combined.

*The time view (Fig. 9)* consists of three parts arranged from left to right. First, as the default timebox has not been specified with a name, RelaQ presents a list of time series (Fig. 9(A)) that can be used to complete the default timebox. For example, SF-default can be completed with either SF-LA or SF-SD, and both LA and SD will appear in the list. The opacity of each item in the list encodes the average matching score (divided by the highest one) when completing the default timebox with that particular time series. If a time series is selected to complete the default timebox, it will be highlighted like Fig. 9($A_1$). Second, the structure overview of all results is visualized as a node-link graph (Fig. 9(B)). The thickness of lines encodes the average strength of such relations. We also visualize the time lag distribution between two fragments. Third, the time series involved in the queried results are visualized with line charts in different rows (Fig. 9(C)). An area chart on the top shows the temporal occurrence of the results. Matched time fragments are highlighted in the line chart to make the matched pattern clear. Users can zoom in on each result to obtain details (Fig. 9(D)). RelaQ adopts the same timebox visualization as the input panel for visual consistency and easy comparison.

*Justification:* There are many approaches supporting comparison of node-link graphs according to Beck's survey [40]. We evaluate two types of visualization techniques, namely juxtaposed and dismantled graphs. In juxtaposed graphs (Fig. 8($D_1$)), each sub-pattern is presented as a column, while in dismantled graphs (Fig. 8($D_2$)), nodes and relations are split into separate columns. Many visual analytics applications employ juxtaposition techniques (e.g., [35], [66], [67]), which is useful for comparing subgraphs. However, juxtaposition techniques suffer from scalable problems and do not allow adjusting the weight of nodes and links. Thus, we adopt dismantled graphs.

### E. Providing Recommendations

The guidance panel instructs users when they want to perform a breadth-oriented query (**R3**). It provides alternative timeboxes that can be added to an existing query, just like the "related searches" module in the bottom of a search engine results page. The guidance panel supports displaying relevant timeboxes that can be added to the existing query and comparing alternative timeboxes.

*Visual design:* Presenting all recommended timeboxes simultaneously may not be feasible due to the large number of potential recommendations. To address this issue, we propose a solution that involves the provision of recommendations on demand. First, we require users to specify their "focus timebox" (Fig. 10(A)) by hovering. We exclusively search for time series that are relevant to the "focus timebox". Second, we employ a matrix-based design (Fig. 10(B)) to display all recommendations. Each cell in the matrix represents an alternative timebox that can be added to the existing query. Cells within the same row pertain to a single time series, while cells within the same

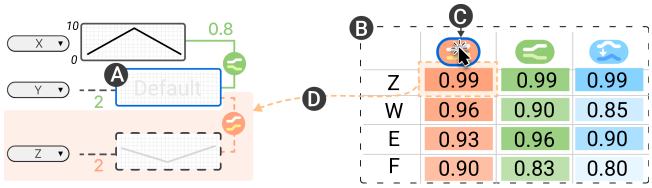


Fig. 10. The guidance panel's design. (A) The focus timebox. (B) The recommended timeboxes that can be added to the query. (C) The relation icon that can be clicked to sort recommendations. (D) The preview of a new query after hovering.

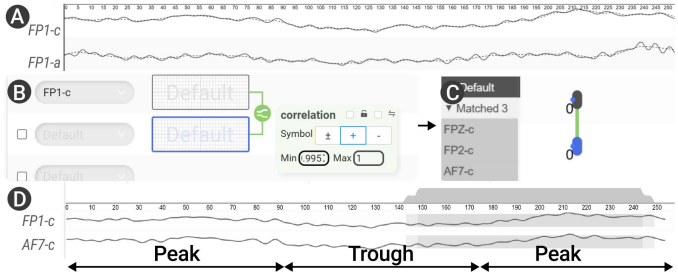


Fig. 11. This displays Q1 and R1 for the first case. (A) The time series of Fp1-c and -a. (B) Query time series correlated to Fp1-c. (C) Results include FPZ-c, FP2-c, and AF7-c. (D) Results mainly distribute in the second peak.

column denote the same type of relation. Users can click the icon in the column header to sort timeboxes (Fig. 10(C)). We utilize a confidence-based approach to determine the order of cells. Specifically, the confidence is calculated based on the percentage of the relation within the total recommended relations. This approach ensures that recommendations are sorted based on their relevance and usefulness to the user. The opacity of each cell encodes the confidence, while the text means the strength. We allow users to hover over a cell and check its preview (Fig. 10(D)).

*The recommendation algorithm:* This explicitly features three relation types: similarity, correlation, and causality, which we selected based on their complexity and dynamic nature. Besides, the relation lag is also recommended implicitly with these three relations. We adopt a statistical recommendation algorithm. A group of time series related to the "focus timebox" were selected from pre-calculated relation indexes. RelaQ then evaluates each time series by enumerating lags and summarizes the average strength and confidence. Finally, the top 20 time series with the highest confidence are displayed. Relations, including meta and arithmetic, were not included in the analysis due to their complex possible scenarios and large search space, which can significantly impact the algorithm's time performance. Moreover, compared to other relation types, they are not commonly queried. Thus, RelaQ does not support recommendations for meta and arithmetic relations.

## V. Case Study

This session is divided into three stages: **[20 min]** a 15-minute tutorial and 5 minutes free exploration, **[50 min]** Free relation-driven retrieve of time series based on real-world dataset (Q-Queries, R-Results), following the think-aloud protocol [68], **[10 min]** a final one-to-one expert interview. The aim of the case study is to 1) evaluate the effectiveness and usability of RelaQ and 2) collect feedback from experts.

### A. Case 1: Analyze the Relation Between Brain Regions

We invited Expert A (female, EA), who has four years of experience in analyzing EEG data. EA aimed at figuring out how the correlation between two brain regions changed after drinking when humans were exposed to two matched stimuli.

*Dataset:* This case study is based on an EEG Database data set, which includes the control group data and the alcoholic group data. Each group tests eight subjects under two matched stimuli and collects signals under 256 Hz in a second. To reduce noises, we average multiple results from the same subject under the same condition. The final data set has 122 * 256 = 31,232 samples and a size of 275 KB. Fp1, Fp2, Fpz, and AF7 are electrodes. (c: common, a: alcoholic).

*Q1. Query which electrodes correlated with Fp1 before drinking:* EA first loaded the data and adjusted Sampling Length (5) and Box Length (100). The difference between time series of Fp1-c and Fp1-a (Fig. 11(A)) attracted EA as it indicated that alcohol affected Fp1. Thus, EA began to query the electrodes correlated to Fp1 before drinking. She created two timeboxes, identified one as Fp1-c, and kept the other as default. Then, she selected correlation and clicked two timeboxes to link them. EA set the threshold as [0.995,1] according to domain knowledge, as shown in Fig. 11(B).

*R1. Explore how electrodes correlated with Fp1 before drinking:* EA found there were three time series in the results (Fig. 11(C)), including Fp2-c, Fpz-c, and AF7-c. She said that *it was reasonable because Fp2 and Fpz were in the same region with Fp1, while AF7 was close to Fp.* The result was in line with her expectations. Furthermore, when EA inspected details, she found that results with a high matching score of AF7 were most in the right peak segment (Fig. 11(D)). This phenomenon inspired her that the effect might vary in the peaks and troughs. Thus, EA filtered the results and found that results with high correlation were mainly distributed in the peaks for all three pairs of electrodes. She said it indicated that these three pairs of electrodes show synergy to this stimulus and the peak might represent the response.

*Q2. Query the pattern of correlation changes after drinking:* Next, EA continued to query how the correlation strength between Fp1 and other electrodes decreased after drinking. She created a timebox labeled Fp1-a, linked it to a new default one, and set the correlation relation constraint. Then, EA added the meta relation constraint of two default timeboxes. This requires that two timeboxes (Fig. 2($D_1$) and ($d_2$)) be from the same electrode (Fig. 2($D_3$)). $D_1$ has a high correlation with Fp1-c, while $D_2$ has a low correlation with Fp1-a (Fig. 2($D_4$) revert matching finds minimum value). Thus, the query selects the electrode whose correlation with Fp1 decreases the most. The query is shown in Fig. 2(D). After a few seconds, EA found that Fp2, Fpz, and AF7 were still in the results, and Fp2 was affected most (Fig. 2(E)). EA scrolled to inspect several results. EA said, *"The seriously decreased correlation strength between Fp1 and Fp2 may indicate some some asymmetrical impacts."*

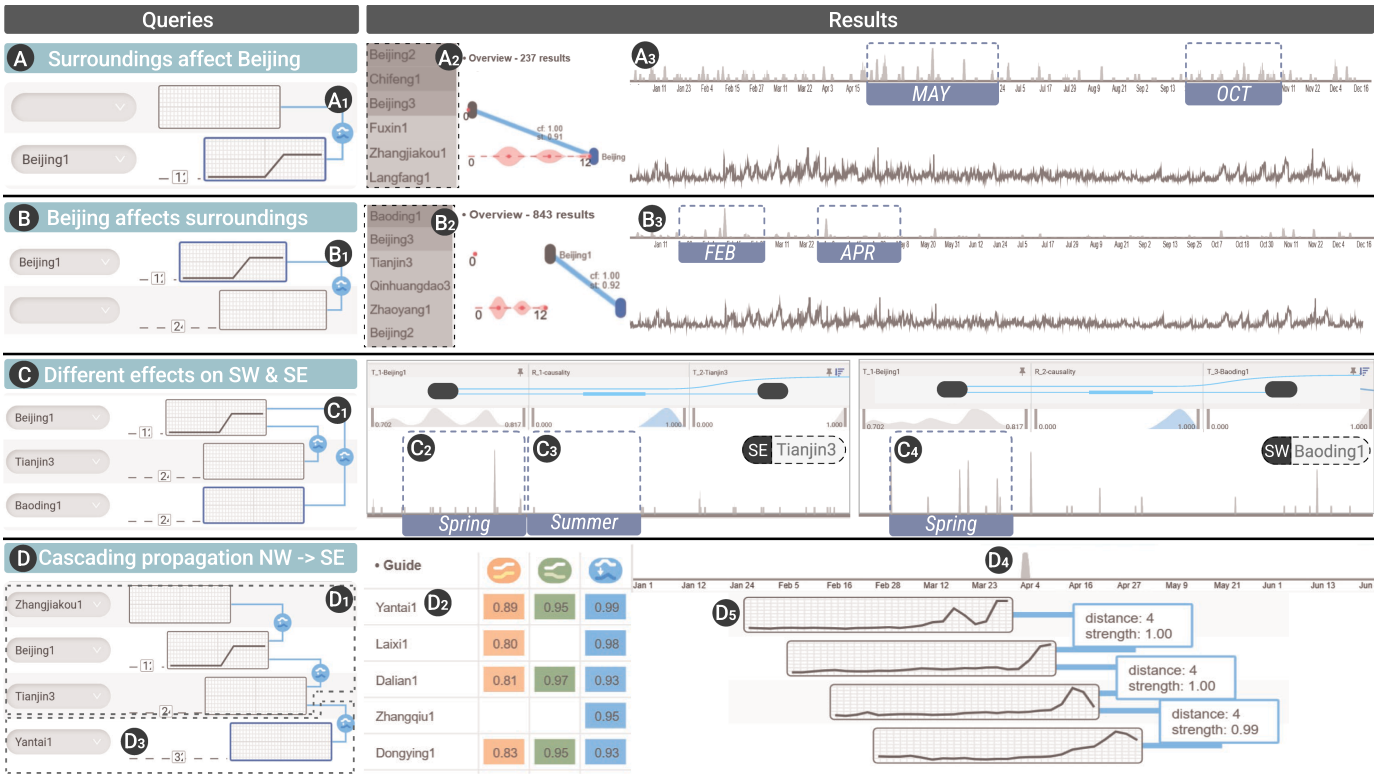


Fig. 12. 4 queries and results in Case 2. (A) How does Beijing's surroundings affect it? (B) How does Beijing affect its surroundings? (C) What are the differences in pollution transmission from Beijing to various directions? (D) What is the pattern of cascading propagation from Zhangjiakou to Beijing?.

*R2. Explore differences between the peak and the trough:* Inspired by the observation in the second step, EA further analyzed whether the difference between the peak and the trough still existed. Thus, she checked the results' distribution in the Fp1-Fp2 pair. EA filtered results with higher matching scores and found the time range of the peak was highlighted obviously (Fig. 2(F)). The Fp1-Fpz and Fp1-AF7 pair had the same visual pattern, which EA thought was interesting. It revealed that the peak region was affected the most. To investigate details, EA observed time series in the analysis panel. She found that the peak appeared delayed in the alcoholic group (Fig. 2(F)). Till now, EA could summarize some preliminary results. She added, *"The correlation between Fp1-Fp2 decreases most and the peak delays reveal that alcohol may affect balance ability and reaction speed, respectively."*

### B. Case 2: Explore the Relation of Air Pollution Between Cities in the North China Plain

Analyzing air quality time series is a popular and important topic [35], so we invite Expert B (male, EB), who has five years of experience in analyzing urban air data. Many efforts have been devoted to protecting the environment in the North China Plain, which is a densely populated area and constantly affected by air pollution. Since Beijing was one of the most important monitoring sites located there, EB explored the air pollution patterns involving Beijing.

*Dataset:* The dataset collects the PM2.5 concentration from 448 air quality monitoring sites in China. Each site records hourly between Jan 1st and Dec $20^{th}$, 2018. This dataset has $448 * 8,472 = 3,795,456$ records and a size of 11.4 MB.

*Q1R1. Query how surroundings affected Beijing:* The Sampling Length (4H) and Box Length (24H) were adjusted at the beginning. First, EB was curious about how air pollution spread from surrounding cities to Beijing. EB added a timebox A (Default) and a timebox B (Beijing1), and clicked the causality icon to connect them. He then dragged B nearly to the half of A because the delay in propagation might be up to 12 hours depending on the wind speed. Since EB was interested in when the air pollution index is rising, he sketched a trend in timebox B referencing the trend guidance. The query is shown in Fig. 12($A_1$). After a few seconds, 237 results in total have been displayed on the screen. EB found that Beijing was affected throughout the year and the peak occurred in May and October (Fig. 12($A_3$)). He explored which cities affected Beijing most. The order of alternatives (Fig. 12($A_2$)) revealed that Beijing has been affected by air pollution from the north (Zhangjiakou) more seriously than the south (only Langfang).

*Q2R2. Query how Beijing affected surroundings:* EB then wondered how Beijing's air pollution had spread, so he adjusted the query (Fig. 12($B_1$)). There were 843 results, which was nearly four times more than the last one. Peaks occurred in February and April (Fig. 12($B_3$)). EB explained that with some effective environmental governance measures like the use of new

energy, the current source of air pollution has mainly become urban traffic pollution. Through the alternative list (Fig. 2($B_2$)), EB further found that the air pollution transmitted from Beijing mainly to the south (Tianjin, Baoding) and the east (Qinghuangdao) but hardly to the west. EB explained that the western mountains blocked pollution from spreading from Beijing to the northwest regions with higher elevations.

*Q3R3. Compare the spread of air pollution in different directions:* Inspired by the observations in Q2, EB was interested in the differences between patterns of air pollution propagation from Beijing to different directions. He configured Timebox A as Tianjin (southeast) and Timebox B as Baoding (southwest) (Fig. 12($C_1$)). To compare these two propagation patterns, EB grouped two subgraphs involving Tianjin and Baoding, respectively. He found that the results were mainly distributed in spring for both cities (Fig. 12($C_{2,4}$)). Moreover, Tianjin was hardly affected in summer (Fig. 12($C_3$)) but Baoding was affected nearly throughout the whole year. EB said that the results were in line with climatic characteristics, *"Because of monsoon factors, the wind blows from Tianjin to Beijing in summer, so there is less pollution coming to Tianjin."*

*Q4R4. Query cascading propagation involving Beijing:* EB investigated the chain of air pollution propagation from northwest to Beijing and its effects. He removed Baoding and added Zhangjiakou (Fig. 12($D_1$)). EB clicked the guidance icon to find cascading effects. After the recommended queries were shown, EB triggered the sorting in the guidance panel and found Yantai had the highest average strength with the highest confidence (Fig. 12($D_2$)). Thus, EB added Yantai to the existing query (Fig. 12($D_3$)). In the results, the distribution showed that the cascading air pollution patterns mainly occurred in April (Fig. 12($D_4$)). EB inspected several results with the highest matching score (Fig. 12($D_5$)) and found that they aligned with his expectations. These findings suggest the possibility of long-range air pollution transmission from Zhangjiakou to Yantai in spring, emphasizing the need for preventative measures.

### C. Expert Interview

After the experts finished analyzing the real-world dataset, we followed a semi-structured questionnaire to collect their feedback on effectiveness, usability, and improvements.

*Effectiveness and visual designs:* Both EA and EB spoke highly of RelaQ and confirmed its effectiveness and intuitiveness. EA commented it was intuitive to formulate her queries through RelaQ. *"Functions including relation interactions, sketches, and default constraints support complex queries, which can cover massive analysis scenarios that previously had to be done by writing scripts,"* she said. EB noted RelaQ as *"useful"* since existing important patterns, such as cascading propagation or ego-network effects, can both be queried easily. Moreover, EA and EB praised the matrix view in filtering, grouping, and comparing desired patterns, saying it was *"flexible"* and *"powerful"*. In addition, EB commented the guidance was helpful, saying *"It helps me quickly narrow down my choices from a huge number of sites."*

*Usability and improvements:* Though EA and EB agreed that RelaQ had good usability, noting that interactions and designs were *"fluency"* and *"expressive"*, they also gave many suggestions on improving RelaQ. EA recommended adding distribution visualization of parameters in the query panel to guide users in setting initial queries. EB suggested that we should highlight how the trend matches and support adding time series in the result view on demand. We have optimized RelaQ based on their suggestions.

## VI. User Study

We conducted a task-based user study to evaluate the usability of RelaQ and collect usability feedback.

### A. Experiment Settings

*Participants and Data:* We recruited twelve participants (S1-S12, six males and six females) from different departments, including Computer Science 6), Traffic Engineering 1), Electronic Engineering 1), Mathematics 1), Industrial Design 1), Media 1) and Sports Science 1). All subjects were familiar with time series data, with an average self-reported score of 3.5 on a 5-point Likert Scale. Among them, 7 subjects had experience using tools to query or explore time series data, including Python, Microsoft Excel, MATLAB, and Unity. None of the subjects have been involved in the development of RelaQ. In this study, we use a dataset (142 time series * the length 5,000) collected from a real factory.

*Procedure and Tasks:* The experiment lasts about 50 minutes and includes the following steps. First, subjects were presented with a 15-minute tutorial on the background of time series query and the visual encodings and interactions of RelaQ. Then, they freely used the system for 5 minutes. Afterward, subjects were required to complete nine tasks. All tasks were designed to evaluate RelaQ's different functions, including specifying queries (**R1**), understanding results (**R2**), and exploring possible relations (**R3**). We ensured that all interactions and visual encodings were covered during the process. We divided these tasks into the following two stages according to the type of queries summarized in Section III-B3.

- *Stage 1. Query target relations (R1, R2):* Subjects are required to specify queries (**R1**) and answer questions (**R2**). We asked subjects to design three queries and answer questions (T1-3) with different levels of difficulty.
- *Stage 2. Explore possible relations (R2, R3):* Subjects are required to explore possible relations (**R3**) and answer questions (**R2**). We asked subjects to explore based on recommended series (T4, T7) and trends (T5-6) and understand results (T8-9).

Subjects are asked to follow the think-aloud protocol [68]. Their completion time, behavior, and answers were all recorded. After completing tasks, subjects were required to fill in a post-test questionnaire based on System Usability Scale (SUS) [69] with a 5-point Likert scale. Finally, we conducted a semi-structured interview about their experience of using RelaQ to collect feedback.

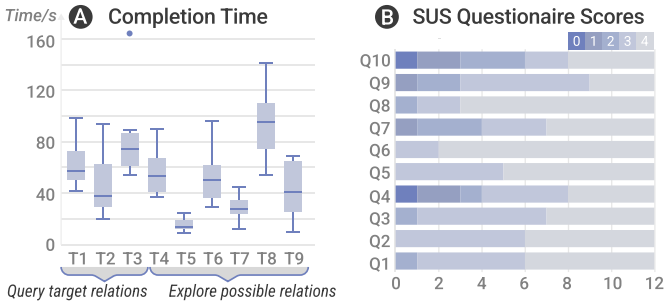


Fig. 13. The quantitative results of our task-based user study. (A) The record of completion time reveals a high task pass rate and varied difficulty. (B) We summarized the distribution of SUS scores given by subjects. In general, subjects commented positively on the usability of RelaQ.

### B. Quantitative Results

To faithfully reflect how users completed tasks using RelaQ, we counted the completion time and the pass rate. We considered a subject as passed when he/she fully understood the task, performed reasonable operations, and answered correctly.

Most subjects passed tasks successfully, with an average passing rate for all tasks of 0.98. There were two failures in total: S4 failed Q2 because he misunderstood the task description and deleted a relation constraint that should have existed; S11 failed Q4 because he confused causality and correlation, resulting in adding a wrong relation. For passed cases, the completion time is summarized in Fig. 13(A). The time for the subjects to complete a task ranged from 9 seconds to 163 seconds, with an average of 53 seconds. Specifically, S4 spent more time in Q3 than others because he misunderstood the encoding of result distribution and browsed the results repeatedly on the matrix panel. Overall, subjects took an average of 7.8 minutes to answer all tasks.

The subjects gave a SUS score of 81.04 on average, and a score above 80.3 is considered as the top 10% of scores [69]. We also computed the detailed factors of SUS [70], with a usability score of 85.16 and a learnability score of 64.58. Considering that the user study is conducted with non-experts, the learnability is still within an acceptable range. Besides, based on expert interviews in the case study, we argue that the system is easy to learn and use for experts in the field. Nonetheless, we plan to address the learning curve by providing more comprehensive tutorials and guidance in the future. Generally, subjects highly evaluated the consistency of system design. Most of them agreed that RelaQ is convenient to use and are willing to use it in the future. The details of the subjects' scores are shown in Fig. 13(B), and we summarize their detailed feedback in the following section.

### C. Qualitative Usability

We summarized subjects' feedback in terms of query, understanding, and guidance and improved RelaQ correspondingly.

*RelaQ enables easy formulation of query constraints (R1):* All subjects edited queries smoothly and confidently without confusion. Specifically, S4 pointed out that sketches are consistent with his intent, enabling him to draw desired trends easily. In terms of relations, S2 thought that the interaction of the relation query was very straightforward, as she could directly pick desired relations. S10 stated that relation interactions require a low mental burden, even when specifying complex queries. S7 appreciated smooth interactions: *"I can check the changes of trends in real-time with animation while editing the query."* While subjects agreed with the intuitiveness, they also noted some limitations and suggested some improvements. S4, S6, S9, and S11 commented that the interactions and visual encodings used in specifying queries cause a little memory burden. S4 suggested hiding unnecessary buttons at the beginning and adding guided labels.

*RelaQ supports comprehensive understanding of query results (R2):* Subjects found that RelaQ facilitated their understanding of query results. S8 appreciated the node-link diagram's ability to show the abstract of the query and quickly convey the distribution of searched results: *"It helps me quickly learn the distribution of searched results, such as whether a certain relation has matched a result."* S6 highly valued the combination feature for filtering subgraphs flexibly. Some subjects, especially those (S3, 6-7, 11) who were familiar with native LineUp [49], raised their concern on the graph-based design. S3 expressed that he was not confident about the results of auto combination: *"When I was not so proficient in using RelaQ, I was not sure how the intermediate paths of node A and node B will be joined after combining them."* They commented that the graph-based design and auto combination were completely unfamiliar interactions for them, resulting in their lack of confidence at the beginning. However, they also agreed that it can be fully mastered after a few minutes of training. S6 commented: *"The auto combination is very convenient. I can combine a subgraph by selecting the start and end nodes only."* S11 also stated that integrating graphs into LineUp is novel and useful in practical cases.

*Guidance effectively reduces the mental burden of exploration (R3):* S5 highly appreciated recommendation: *"Guidance effectively helped me find possible patterns, though I don't know much about domain knowledge."* Other subjects also gave positive comments, such as *"effectively help narrow down search space"* (S9), and *"smart guidance"* (S12). S10 expressed a strong preference for the guidance and successfully used it to query in a non-standard workflow, achieving the correct result. S10 was the only participant to use the guidance during the target-oriented query evaluation stage. However, some limitations were noted. Specifically, S6 and S7 commented that the guidance should be automatically updated as queries are edited rather than requiring the user to click a corresponding button, which was perceived as inconvenient.

## VII. Discussion

In this section, we discuss the implications, limitations, and future work of RelaQ.

*Implications:* This study systematically discusses the scope of relations between time series, taking the first step to explore the heterogeneous relation-driven approach of querying multiple time series. RelaQ models the query problem as a graph-matching problem, not only concentrating on the sequential features of single time series but also on the structural features of multiple time series. We propose a novel approach that combines

a fuzzy query model and an interactive interface to support the direct and flexible specification of relations among time series without ambiguity, the in-depth understanding of queried temporal patterns, and reliable exploration of new queries. Through a series of evaluations, RelaQ was verified as useful for solving real analysis problems in multiple domains.

*Limitations and future work:* We collected valuable suggestions from users and optimized RelaQ. There is still room for improvement in the long run, especially the trade-off between advanced functions and learnability and the concern about the scalability of the matching algorithm.

*Support flexible relation computation:* Although we have chosen commonly used computations for each relation, experts prefer optional algorithms (EB) or even modifiable code blocks (EA) in extremely professional analysis tasks. On one hand, RelaQ will become more powerful with the integration of modifiable computation. On the other hand, such integration can increase the learning curve and mental burden for users. Considering that RelaQ is our initial attempt at the relation-driven query of time series, we try to avoid excessive learning costs to encourage user adoption of this new query framework. Thus, these advanced functions are left as future work.

*Integrating retrieval with intelligent pattern mining:* The time complexity of the algorithm is nearly O($kN^2M$), where $k$ is the number of relations, $N$ is the number of variables, and $M$ is the length of time series. A simple reduction of the complexity is discussed in Appendix 4.1.3, available online. This means the time costs highly depend on the data scale. Unfortunately, when the scale of the time series reaches $10^8$, there is a noticeable lag in use. We tested the time performance of RelaQ, and the results showed that it has good real-time performance in most cases. The average response time for queries of medium size and difficulty is 1.38 s. More details can be found in Appendix 4.1.2, available online. Besides, RelaQ requires users to specify the match parameters: the box length and the sampling length, which means RelaQ cannot perform free-scale matching. A proven effective approach to solving similar search problems is integrating intelligent pattern mining models. Since RelaQ meets users' requirements in most daily analysis situations, we will enhance our model in the future by integrating intelligent approaches.

## VIII. Conclusion

In this article, we conduct formative research to develop a semantic scope of time series relations and propose a novel approach for easy relation-driven time series queries through a prototype named RelaQ. Evaluations include a quantitative user study and qualitative case study, demonstrating that RelaQ enables intuitive query and understanding of multiple time series with heterogeneous relations, allowing users to easily formulate complex queries through explicit relation interactions and sketches. RelaQ is effective in real analysis tasks such as urban air pollution and EEG correlation analysis. As an initial attempt at relation-driven time series queries, our future work will focus on optimizing RelaQ by integrating intelligent modules and adapting to more professional scenarios.

## References


[1] F. Lekschas, B. Peterson, D. Haehn, E. Ma, N. Gehlenborg, and H. Pfister, "PEAX: Interactive visual pattern search in sequential data using unsupervised deep representation learning," *Comput. Graph. Forum*, vol. 39, no. 3, pp. 167–179, 2020.

[2] Y. Yu, D. Kruyff, J. Jiao, T. Becker, and M. Behrisch, "PSEUDo: Interactive pattern search in multivariate time series with locality-sensitive hashing and relevance feedback," *IEEE Trans. Vis. Comput. Graph.*, vol. 29, no. 1, pp. 33–42, Jan. 2023.

[3] Y. Zhao, Y. Wang, J. Zhang, C.-W. Fu, M. Xu, and D. Moritz, "KD-Box: Line-segment-based KD-tree for interactive exploration of large-scale time-series data," *IEEE Trans. Vis. Comput. Graph.*, vol. 28, no. 1, pp. 890–900, Jan. 2022.

[4] V. Peña-Araya, E. Pietriga, and A. Bezerianos, "A comparison of visualizations for identifying correlation over space and time," *IEEE Trans. Vis. Comput. Graph.*, vol. 26, no. 1, pp. 375–385, Jan. 2020.

[5] H. Wang et al., "A visual analytics framework for spatiotemporal trade network analysis," *IEEE Trans. Vis. Comput. Graph.*, vol. 25, no. 1, pp. 331–341, Jan. 2019.

[6] J. Wang and K. Mueller, "DOMINO: Visual causal reasoning with time-dependent phenomena," *IEEE Trans. Vis. Comput. Graph.*, vol. 29, no. 12, pp. 5342–5356, Dec. 2023.

[7] Y. Lu, H. Wang, S. Landis, and R. Maciejewski, "A visual analytics framework for identifying topic drivers in media events," *IEEE Trans. Vis. Comput. Graph.*, vol. 24, no. 9, pp. 2501–2515, Sep. 2018.

[8] A. Gogolou, T. Tsandilas, T. Palpanas, and A. Bezerianos, "Comparing similarity perception in time series visualizations," *IEEE Trans. Vis. Comput. Graph.*, vol. 25, no. 1, pp. 523–533, Jan. 2019.

[9] P. Buono, A. Aris, C. Plaisant, A. Khella, and B. Shneiderman, "Interactive pattern search in time series," in *Proc. Vis. Data Anal.*, 2005, pp. 175–186.

[10] M. C. Hao, U. Dayal, D. A. Keim, D. Morent, and J. Schneidewind, "Intelligent visual analytics queries," in *Proc. IEEE Symp. Vis. Analytics Sci. Technol.*, 2007, pp. 91–98.

[11] R. Agrawal, G. Psaila, E. L. Wimmers, and M. Zaït, "Querying shapes of histories," in *Proc. 21th Int. Conf. Very Large Data Bases*, 1995, pp. 502–514.

[12] M. Wattenberg, "Sketching a graph to query a time-series database," in *Proc. Extended Abstr. Hum. Factors Comput. Syst.*, 2001, pp. 381–382.

[13] C. Fan, K. Matković, and H. Hauser, "Sketch-based fast and accurate querying of time series using parameter-sharing LSTM networks," *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 12, pp. 4495–4506, Dec. 2021.

[14] M. Mannino and A. Abouzied, "Expressive time series querying with hand-drawn scale-free sketches," in *Proc. CHI Conf. Hum. Factors Comput. Syst.*, 2018, pp. 1–13.

[15] Pandas, 2019. [Online]. Available: https://doi.org/10.5281/zenodo.3509134

[16] TradingView, 2011. [Online]. Available: https://www.tradingview.com

[17] T. Siddiqui, P. Luh, Z. Wang, K. Karahalios, and A. G. Parameswaran, "ShapeSearch: A. flexible and efficient system for shape-based exploration of trendlines," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2020, pp. 51–65.

[18] S. Imani, S. Alaee, and E. Keogh, "Qute: Query by text search for time series data," in *Proc. Future Technol. Conf.*, 2021, pp. 412–427.

[19] H. Hochheiser, "Interactive querying of time series data," in *Proc. Extended Abstr. Hum. Factors Comput. Syst.*, 2002, pp. 552–553.

[20] M. Correll and M. Gleicher, "The semantics of sketch: Flexibility in visual query systems for time series data," in *Proc. IEEE Conf. Vis. Analytics Sci. Technol.*, 2016, pp. 131–140.

[21] C. Holz and S. Feiner, "Relaxed selection techniques for querying time-series graphs," in *Proc. 22nd Annu. ACM Symp. User Interface Softw. Technol.*, 2009, pp. 213–222.

[22] E. J. Keogh and P. Smyth, "A probabilistic approach to fast pattern matching in time series databases," in *Proc. Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 1997, pp. 24–30.

[23] 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 *Proc. IEEE Conf. Vis. Analytics Sci. Technol.*, 2016, pp. 121–130.

[24] P.-E. Danielsson, "Euclidean distance mapping," *Comput. Graph. Image Process.*, vol. 14, no. 3, pp. 227–248, 1980.

[25] M. Müller, *Dynamic Time Warping*. Berlin, Germany: Springer, 2007, pp. 69–84.

[26] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh, "Querying and mining of time series data: Experimental comparison of representations and distance measures," *Proc. VLDB Endowment*, vol. 1, pp. 1542–1552, 2008.

[27] J. Lin, E. Keogh, S. Lonardi, and B. Chiu, "A symbolic representation of time series, with implications for streaming algorithms," in *Proc. SIGMOD Workshop*, 2003, pp. 2–11.
[28] Y. Zhou, R. Jiang, H. Qin, and H. Hu, "Representation and analysis of time-series data via deep embedding and visual exploration," *J. Vis.*, vol. 26, no. 3, pp. 593–610, 2023.
[29] InfluxDB, 2013. [Online]. Available: https://docs.influxdata.com
[30] Amazon Timestream, 2020. [Online]. Available: https://docs.aws.amazon.com/timestream/latest/developerguide/what-is-timestream.html
[31] TimescaleDB, 2017. [Online]. Available: https://www.timescale.com
[32] ElasticSearch, 2010. [Online]. Available: https://www.elastic.co
[33] J. Eirich et al., "IRVINE: A design study on analyzing correlation patterns of electrical engines," *IEEE Trans. Vis. Comput. Graph.*, vol. 28, no. 1, pp. 11–21, Jan. 2022.
[34] S. Liu et al., "ECoalVis: Visual analysis of control strategies in coal-fired power plants," *IEEE Trans. Vis. Comput. Graph.*, vol. 29, no. 1, pp. 1091–1101, Jan. 2023.
[35] Z. Deng et al., "Compass: Towards better causal analysis of urban time series," *IEEE Trans. Vis. Comput. Graph.*, vol. 28, no. 1, pp. 1051–1061, Jan. 2022.
[36] Z. Jin, S. Guo, N. Chen, D. Weiskopf, D. Gotz, and N. Cao, "Visual causality analysis of event sequence data," *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 2, pp. 1343–1352, Feb. 2021.
[37] T.-Y. Lee and H.-W. Shen, "Visualization and exploration of temporal trend relationships in multivariate time-varying data," *IEEE Trans. Vis. Comput. Graph.*, vol. 15, no. 6, pp. 1359–1366, Nov./Dec. 2009.
[38] G. Shirato, N. Andrienko, and G. Andrienko, "Exploring and visualizing temporal relations in multivariate time series," *Vis. Inform.*, vol. 7, no. 4, pp. 57–72, 2023.
[39] G. Shirato, N. Andrienko, and G. Andrienko, "Identifying, exploring, and interpreting time series shapes in multivariate time intervals," *Vis. Inform.*, vol. 7, no. 1, pp. 77–91, 2023.
[40] F. Beck, M. Burch, S. Diehl, and D. Weiskopf, "A taxonomy and survey of dynamic graph visualization," *Comput. Graph. Forum*, vol. 36, no. 1, pp. 133–159, 2017.
[41] V. Filipov, A. Arleo, M. Bögl, and S. Miksch, "On network structural and temporal encodings: A space and time odyssey," *IEEE Trans. Vis. Comput. Graph.*, to be published, doi: 10.1109/TVCG.2023.3310019.
[42] M. Burch, B. Schmidt, and D. Weiskopf, "A matrix-based visualization for exploring dynamic compound digraphs," in *Proc. 17th Int. Conf. Inf. Vis.*, 2013, pp. 66–73.
[43] B. Bach, E. Pietriga, and J. Fekete, "Visualizing dynamic networks with matrix cubes," in *Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, 2014, pp. 877–886.
[44] M. Hlawatsch, M. Burch, and D. Weiskopf, "Visual adjacency lists for dynamic graphs," *IEEE Trans. Vis. Comput. Graph.*, vol. 20, no. 11, pp. 1590–1603, Nov. 2014.
[45] M. Burch, C. Vehlow, F. Beck, S. Diehl, and D. Weiskopf, "Parallel edge splatting for scalable dynamic graph visualization," *IEEE Trans. Vis. Comput. Graph.*, vol. 17, no. 12, pp. 2344–2353, Dec. 2011.
[46] M. Greilich, M. Burch, and S. Diehl, "Visualizing the evolution of compound digraphs with TimeArcTrees," *Comput. Graph. Forum*, vol. 28, no. 3, pp. 975–982, 2009.
[47] U. Brandes and S. R. Corman, "Visual unrolling of network evolution and the analysis of dynamic discourse," in *Proc. IEEE Symp. Inf. Vis.*, 2002, pp. 145–151.
[48] L. Shi, C. Wang, Z. Wen, H. Qu, C. Lin, and Q. Liao, "1.5D egocentric dynamic network visualization," *IEEE Trans. Vis. Comput. Graph.*, vol. 21, no. 5, pp. 624–637, May 2015.
[49] S. Gratzl, A. Lex, N. Gehlenborg, H. Pfister, and M. Streit, "LineUp: Visual analysis of multi-attribute rankings," *IEEE Trans. Vis. Comput. Graph.*, vol. 19, no. 12, pp. 2277–2286, Dec. 2013.
[50] G. R. Gibbs, *Thematic Coding and Categorizing*. London, U.K.: Sage, 2007, pp. 38–56.
[51] A. Fakhrazari and H. Vakilzadian, "A survey on time series data mining," in *Proc. IEEE Int. Conf. Electro Inf. Technol.*, 2017, pp. 476–481.
[52] P. Esling and C. Agón, "Time-series data mining," *ACM Comput. Surv.*, vol. 45, no. 1, pp. 12:1–12:34, 2012.
[53] K. Pearson LIII., "On lines and planes of closest fit to systems of points in space," *Philos. Mag.*, vol. 2, no. 11, pp. 559–572, 1901.
[54] C. W. J. Granger, "Investigating causal relations by econometric models and cross-spectral methods," *Econometrica*, vol. 37, pp. 31–47, 2001.
[55] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *Int. J. Forecasting*, vol. 22, no. 4, pp. 679–688, 2006.
[56] C. Spearman, "'FOOTRULE' for measuring correlation," *Brit. J. Psychol.*, vol. 2, no. 1, pp. 89–108, 1906.
[57] D. X. Li, "On default correlation: A. copula function approach," *J. Fixed Income*, vol. 9, no. 4, pp. 43–54, 2000.
[58] J.-P. Lachaux, E. Rodriguez, J. Martinerie, and F. J. Varela, "Measuring phase synchrony in brain signals," *Hum. Brain Mapp.*, vol. 8, no. 4, pp. 194–208, 1999.
[59] M. Henrion, "Propagating uncertainty in bayesian networks by probabilistic logic sampling," in *Proc. 25th Conf. Uncertainty Artif. Intell.*, 1986, pp. 149–164.
[60] Y. G. Udny, *On Method Investigating Periodicities Disturbed Series with Special Reference to Wolfer's Sunspot Numbers*, *Philos. Trans. R. Soc. Lond.*, vol. 226, 1927, pp. 267–298, doi: 10.1098/rsta.1927.0007.
[61] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer, "Voyager: Exploratory analysis via faceted browsing of visualization recommendations," *IEEE Trans. Vis. Comput. Graph.*, vol. 22, no. 1, pp. 649–658, Jan. 2016.
[62] React-Redux-TypeScript, 2015. [Online]. Available: https://react-redux.js.org/tutorials/typescript-quick-start
[63] A. Ronacher, Flask, 2010. [Online]. Available: https://flask.palletsprojects.com/en/2.3.x
[64] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra, "Dimensionality reduction for fast similarity search in large time series databases," *Knowl. Inf. Syst.*, vol. 3, no. 3, pp. 263–286, 2001.
[65] R. Tarjan, "Depth-first search and linear graph algorithms," in *Proc. 12th Annu. Symp. Switching Automata Theory*, 1971, pp. 114–121.
[66] C. D. G. Linhares, J. R. Ponciano, D. S. Pedro, L. E. C. Rocha, A. J. M. Traina, and J. Poco, "LargeNetVis: Visual exploration of large temporal networks based on community taxonomies," *IEEE Trans. Vis. Comput. Graph.*, vol. 29, no. 1, pp. 203–213, Jan. 2023.
[67] E. Cakmak, U. Schlegel, D. Jäckle, D. Keim, and T. Schreck, "Multiscale snapshots: Visual analysis of temporal summaries in dynamic graphs," *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 2, pp. 517–527, Feb. 2021.
[68] M. E. Fonteyn, B. Kuipers, and S. J. Grobe, "A description of think aloud method and protocol analysis," *Qual. Health Res.*, vol. 3, no. 4, pp. 430–441, 1993.
[69] J. Brooke, *SUS: A Quick and Dirty Usability Scale*. Boca Raton, FL, USA: CRC Press, 1996, pp. 189–194.
[70] J. R. Lewis and J. Sauro, "The factor structure of the system usability scale," in *Proc. Int. Conf. Hum. Centered Des.*, 2009, pp. 94–103.

**Shuhan Liu** received the BS degree in computer science from Zhejiang University, in 2021. She is currently working toward the doctoral degree with the State Key Lab of CAD&CG, Zhejiang University. Her research interests include spatiotemporal data mining, visualization, and industrial data visual analytics.

**Yuan Tian** received the BS degree in computer science from Zhejiang University, in 2022. She is currently working toward the PhD degree with the State Key Lab of CAD&CG, Zhejiang University. Her research interests include machine learning for visualization and visual analytics.

**Zikun Deng** received the PhD degree in computer science from Zhejiang University, in 2023. He is a tenure-track associate professor with the School of Software Engineering, South China University of Technology. His research interests mainly include visual analytics, visualization, data mining, and their application in smart city, industry 4.0, and digital twins. For more information, please visit https://zkdeng.org.

**Weiwei Cui** received the BS degree in computer science and technology from Tsinghua University, China, and the PhD degree in computer science and engineering from the Hong Kong University of Science and Technology, Hong Kong. He is a principal researcher with Microsoft. His primary research interest include visualization, with the focuses on democratizing visualization and AI-assisted design. For more information, please visit https://www.microsoft.com/en-us/research/people/weiweicu/.

**Haidong Zhang** received the PhD degree in computer science from Peking University, China. He is a principal architect with Microsoft. His research interests include visualization and human-computer interaction.

**Di Weng** (Member, IEEE) received the PhD degree in computer science from the State Key Lab of CAD&CG, Zhejiang University. He is a ZJU100 young professor with the School of Software Technology, Zhejiang University. His main research interests include information visualization and visual analytics, focusing on interactive data transformation and spatiotemporal data analysis. Before his current position, he was a researcher with Microsoft Research Asia from 2022 to 2023. For more information, please visit https://dwe.ng.

**Yingcai Wu** (Senior Member, IEEE) received the PhD degree in computer science from the Hong Kong University of Science and Technology. He is a professor with the State Key Lab of CAD&CG, Zhejiang University. His main research interests include information visualization and visual analytics, with focuses on urban computing, sports science, immersive visualization, and social media analysis. Prior to his current position, he was a postdoctoral researcher with the University of California, Davis, from 2010 to 2012, and a researcher with Microsoft Research Asia from 2012 to 2015. For more information, please visit http://www.ycwu.org.