1 2 9 0

UNIVERSIDADE Ð
COIMBRA

António Malta Lopes da Cruz

COORDINATED VISUALIZATION PANELS FOR BIOLOGICAL DATASETS

Tese no âmbito do Programa de Doutoramento em Ciências e Tecnologias da Informação orientada pelo Professor Doutor Fernando Jorge Penousal Martins Machado e pelo Professor Doutor Joel Perdiz Arrais, apresentada ao Departamento de Engenharia Informática da Faculdade de Ciências e Tecnologia da Universidade de Coimbra.

Novembro de 2022

1 2 9 0

UNIVERSIDADE D
COIMBRA

ANTÓNIO MALTA LOPES DA CRUZ

COORDINATED VISUALIZATION PANELS FOR BIOLOGICAL DATASETS

Tese no âmbito do Programa de Doutoramento em Ciências e Tecnologias da Informação orientada pelo Professor Doutor Fernando Jorge Penousal Martins Machado e pelo Professor Doutor Joel Perdiz Arrais, apresentada ao Departamento de Engenharia Informática da Faculdade de Ciências e Tecnologia da Universidade de Coimbra.

Supervised by Fernando Jorge Penousal Martins Machado as Associate Professor of the Department of Informatics Engineering of the Faculty of Sciences and Technology of the University of Coimbra, and Joel Perdiz Arrais as Assistant Professor at the Department of Informatics Engineering of the Faculty of Sciences and Technology of the University of Coimbra. This work is funded by the FCT - Foundation for Science and Technology, I.P./MCTES through national funds (PIDDAC), within the scope of CISUC R&D Unit - UIDB/00326/2020 or project code UIDP/00326/2020, and by FCT - Foundation for Science and Technology, under the grants SFRH/BD/124538/2016 and COVID/BD/152131/2021.

ABSTRACT

Data visualization has been shown to be an important tool in knowledge discovery, being used alongside data analysis to identify and highlight patterns, trends and outliers, aiding users in decision-making. The need for analyzing unstructured and increasingly larger datasets has led to the continued emergence of visualization tools that seek to provide methods that facilitate the exploration and analysis of such datasets. Many fields of study still face the challenges inherent to the analysis of complex multidimensional datasets, such as the field of computational biology, whose research of infectious diseases must contend with large protein-protein interaction networks with thousands of genes that vary in expression values over time.

Throughout this thesis, we explore the visualization of multivariate data through CroP, a data visualization tool with a coordinated multiple views framework that allows users to adapt the workspace to different problems through flexible panels. While CroP is able to process generic relational, temporal and multivariate quantitative data, it also presents methods directed at the analysis of biological data. This data can be represented through various layouts and functionalities that not only highlight relationships between different variables, but also dig-down into discovered patterns in order to better understand their sources and their effects. In particular, we can highlight the exploration of time-series through our dynamic and parameter-based implementation of layouts that bend timelines to visually represent how datasets behave over time.

The implemented models and methods are demonstrated through experiments with diverse multivariate datasets, with a focus on gene expression time-series datasets, and complemented with a discussion on how these contributed to the creation of comprehensible visualizations, facilitated data analysis, and promoted pattern discovery. We also validate CroP through model and interface tests performed with participants from both the fields of information visualization and computational biology.

As we present our research and a discussion of its results, we can highlight the following contributions: an analysis of the available range of visualization models and tools for multivariate datasets, as well as modern data analysis methods that can be used cooperatively to explore such datasets; a coordinated multiple views framework with a modular workspace that can be adapted to the analysis of varied problems; dynamic visualization models that explore the representation of complex multivariate datasets, combined with modern data analysis methods to highlight and analyze significant events and patterns; a visualization tool that incorporates the developed framework, visualization models and data analysis methods into a platform that can be used by different types of users.

RESUMO

A visualização de dados têm sido mostrada como uma ferramenta importante na descoberta de conhecimento, sendo utilizada ao lado da análise de dados para identificar e realçar padrões, tendências e outliers, ajudando os utilizadores na toma de decisões. A necessidade de analisar datasets sem estruturas explicitas e cada vez com maiores volumes tem levado à emergência contínua de ferramentas de visualização que procuram providenciar métodos que facilitam a exploração e análise destes datasets. Muitos campos de estudo ainda enfrentam os desafios inerentes à análise de datasets multidimensionais complexos, como o campo de biologia computacional, cuja pesquisa de doenças infeciosas tem de lidar com redes complexas de interação de proteínas que contém milhares de genes com valores de expressão que variam com o tempo.

No decorrer desta tese, exploramos a visualização de dados multivariados através do CroP, uma ferramenta de visualização de dados com uma framework de múltiplas vistas coordenadas que permite a adaptação do ambiente de trabalho a diferentes problemas através de painéis flexíveis. Enquanto que o CroP é capaz de processar datasets relacionais, temporais e multivariados genéricos, também apresenta métodos direcionados à análise de dados biológicos. Estes dados podem ser representados através de vários layouts dinâmicos e funcionalidades que não só realçam relacionamentos entre variáveis diferentes, mas também aprofundam a procura em padrões descobertos de modo a melhor compreender as suas causas e efeitos. Em particular, podemos realçar a exploração de séries temporais através das nossas implementações dinâmicas e parametrizáveis de layouts que deformam linhas do tempo para representarem visualmente os comportamentos temporais de datasets.

A validade dos modelos e métodos implementados é demonstrada através de experiências com datasets multivariados diversos, com um foco em datasets de expressão temporal de genes, complementado com uma discussão sobre como estes contribuíram para a criação de visualizações compreensíveis, como facilitaram a análise de dados, e como promovem a descoberta de padrões. Adicionalmente, validamos o CroP através de testes de modelo e interface realizados com participantes dos campos de visualização de informação e de biologia computacional.

Na apresentação da nossa pesquisa e discussão dos seus resultados, podemos realçar as seguintes contribuições: uma análise dos modelos e ferramentas de visualização disponíveis para datasets multivariados, bem como métodos de análise de dados modernos que podem ser utilizados cooperativamente para explorar estes datasets; uma framework de múltiplas vistas coordenadas com um ambiente de trabalho modular que pode ser adaptado à análise de uma variedade de problemas; modelos de visualização dinâmicos que exploram a representação de datasets multivariados complexos, combinados com métodos de análise de dados modernos para realçar e analisar eventos e padrões significativos, uma ferramenta de visualização que incorpora a framework desenvolvida, os modelos de visualização e métodos de análise de dados em uma plataforma que pode ser utilizada por diferentes tipos de utilizador.

ACKNOWLEDGMENTS

I'm extremely grateful to my supervisors, Professor Penousal Machado and Professor Joel Perdiz Arrais who have not only guided and assisted me with my thesis, but also supported and encouraged me throughout my academic path. Without them I would not have been able to come this far.

I am also grateful to the Department of Informatics Engineering of the University of Coimbra and the Cognitive and Media Systems Group of the Centre for Informatics and System, which provided me with the conditions to conduct my research and development work.

I would also give a big thank you to all of my colleagues and, more importantly, friends that I have met at the Computational Design and Visualization Lab and during my studies at the university. From those who I have known since the earlier days, like Tiago Martins, Catarina Maçãs, Evgheni Polisciuc, João Correia, João Cunha and Filipe Assunção, to those who I met during my PhD, like Daniel Lopes, Pedro Silva, Sérgio Rebelo, Jéssica Parente and Luis Gonçalo, they have helped me with my work while good and supporting friends.

Um agradecimento muito especial ao meu pai, que sempre cuidou de mim e me ajudou ao longo de todos estes anos. Agradeço também ao meu irmão Gonçalo, que esteve ao meu lado durante muitos anos, bem como a Andreia, sendo que os dois sempre me encorajaram durante este percurso. Um agradecimento final ao resto da minha familia, em especial à minha tia Helena que nos ajudou durante tempos dificeis.

Dedicado à minha mãe.

CONTENTS

LIST OF FIGURES

LIST OF ACRONYMS

CMV	Coordinated Multiple Views
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
GO	Gene Ontology
HCI	Human-Computer Interaction
HSV	Hue, Saturation, Value
OPTICS	Ordering Points to Identify the Clustering Structure
PPI	Protein-Protein Interaction
t-SNE	t-Distributed Stochastic Neighbor Embedding

1 INTRODUCTION

The graphical representation of information has long been used by many fields of study to record their research, usually through the use of abstract representations or metaphors that help portray the relationships between data and the real world. As such, Information Visualization has been and continues to be applied within these domains to document and organize large quantities of data, in addition to providing the means to explore, analyze and extract new information from it. Through the proper use of visual elements it is possible to highlight meaningful points and patterns that would otherwise not be perceptible across large and complex sets of data.

The field of data visualization has been subjected to continuous and significant advancements over the years, and continues to evolve alongside the current technological advances. The origins of the field can be dated back as far as the 200 BC [49], when it was used in Egypt for astrology and navigation. However, it was only around the seventeenth century that the representation of quantitative information on a two-dimensional plane started to develop, particularly due to areas such as astronomy, map creation and navigation. Developments in the field continue through the eighteenth century, which was characterized by a focus on abstract graphs and graphs of functions due to a rising interest in visualizing data from politics and economy. However, some of the most significant developments were a result of the fast technological advancements of the twentieth century. The processing speed and memory capacity of computers increased, allowing for larger amounts of data to be processed faster which led to creation of high-density visualizations. As computers started becoming a more common tool, data and development tools became more widely accessible, leading to the developments in the field of Human-Computer Interaction (HCI) which lead to new interactive visualization tools.

Such developments in data visualization are promoted by other fields of research that must also contend with increasingly larger volumes of data that can now be gathered with the aid of new technologies. Moreover, multiple disciplines must contend with the study of subjects which contain large networks of relationships, as well as temporal variables, requiring the analysis of complex datasets that describe diverse processes changing over time simultaneously [81, 114]. Such challenges are prominent in the field of Biology, where data from measuring various biological systems is being gathered increasingly faster [68, 130] due to biological related technologies and data mining techniques [112]. These datasets include protein-protein interaction (PPI) networks, metabolic pathways and regulatory networks, commonly represented through graph visualizations. However, these networks can be considered as complex due to containing hundreds of thousands of nodes and edges, in addition to integrating processes that present changes over time, such as the gene expression values of infected cells. Proper analysis of such data may lead to new knowledge regarding basic molecular mechanisms in cells and the behaviors of infections, as well as a better understanding of the underlying biology and therefore to the development of new treatments [76].

Researchers are faced with the daunting task of exploring increasingly larger and more complex datasets in order to discover any meaningful relationships that could lead to the extraction of new information, and so they turn to visualization tools that allow them to model and study the relationships of various processes [11]. However, the ability to present complete sets of data to the user in a comprehensive manner still presents a significant bottleneck. Many tools are still limited by static visualization models and outdated methods that are unable to properly handle this increase in the complexity of datasets, creating noisy visualizations that are difficult to interpret, characterized by overlapping elements that not only obscure potentially meaningful patterns but may simply just overwhelm viewers. Moreover, despite the advancements of the past decades, the representation and analysis of relational and temporal data continue to be pertinent topics of research within the field of data visualization, being relevant to various problems from different fields.

This was our main motivation to explore and develop novel paradigms and visualization techniques. More specifically, we have focused our research on the development of dynamic visualization models for networks and time-series data, as well as approaches that can facilitate their exploration, analysis and pattern discovery. To this end we developed CroP, a visualization tool that utilizes a multiple coordinated views layout, designed as a platform that can receive external datasets and represent them through comprehensible visualizations, while providing tools to facilitate their navigation and analysis. Working in an interactive environment enables the amount of information on screen to be controlled by the users, allowing them to switch the most appropriate visualization model, navigate between different levels of detail, and even filter less relevant data points while highlighting those that are more significant.

Moreover, such an environment also supports the further exploration of existing visualization models, such as the integration of dynamic layouts that promote self-organization between points in network models, revealing relationships between variables that would otherwise remain hidden. In particular, we are interested in applying such methods to time-series visualization to represent how the data behaves over time and highlight moments or periods that denote significant events. For instance, the Time Curves layout [10] is a relevant technique that warrants further exploration as it distorts timelines to represent temporal behaviors by using multi-dimensional scaling to position time points relatively to their similarity. As noted before, challenges related to the representation and analysis of relational and temporal data continue to be pertinent within the field of Biology, mainly due to the characteristics of such datasets. In this regard, we are also particularly interested in biological datasets that fit our target research objectives, such as protein-protein interaction (PPI) networks and gene expression time-series data, whose analysis still represents a challenge in molecular biology [35, 138]. It is through such research that it may be possible to obtain a deeper understanding of observed behaviors in datasets, identify their sources and potentially predict future events.

1.1 RESEARCH QUESTIONS

Our research hypothesis is that interactive visualization can be a powerful tool for data analysis, capable of producing comprehensible representations of complex, high-dimensional datasets and providing the methods to navigate, dig-down, analyze and ultimately discover relationships and patterns that can be used to extract new knowledge. In this regard, the main goal of this thesis is the development of a visualization tool that integrates different visualization models in a coordinated multiple views (CMV) framework, through which we explore diverse visual and interactive approaches to represent and analyze multivariate datasets, mainly those with relational and temporal attributes, allowing users to identify significant behaviors in the data as well as their sources and impact. However, as this is a broad subject that can be applied to a vast array of problems, we give particular focus to network and time-series datasets from the biological domain, which are often characterized by their complexity. Considering our goals, we consider the following to be our most prominent research questions:

- *How can we comprehensibly represent both relational data and values changing over time from datasets that can be characterized as complex?* — This question represents one of our main visualization requirements, which is the ability of the developed models to present multivariate datasets accurately and clearly to the viewer. There are several factors that can contribute to the complexity in these types of datasets, such as the number of data points, number of edges, number of time points and diversity of temporal profiles, in addition to the existence of other variables. Representing such proprieties simultaneously for large volumes of data may not be feasible through a single visualization model, which is why we must not only consider the employment of CMVs, but also the implementation of scalable approaches that can facilitate pattern discovery and manage visual complexity.

- *Which visualization and data analysis approaches promote the discovery of meaningful relationships and patterns in high-dimensional data?* — As data increases in dimensionality, so does the difficulty not just in representing all of its attributes, but also in the discovery of relationships between such proprieties, especially through basic visualization models. Data analysis methods, such as dimensionality reduction and clustering algorithms, can process, categorize and subsequently manage large quantities of data, allowing its representation to also be simplified. Moreover, visualization layouts and data aggregation can be used to organize complex relational visualizations, highlighting similarities and differences between groups data points through position and other visual variable. Thus, it is necessary to understand which approaches can help process and represent high-dimensional data so that potentially meaningful relationships and patterns can be more easily identified. Such research can lead to further questions, such as how these approaches can be used together to increase their efficacy.

- *Can the visual complexity that is inherent to the representation of large volumes of data be managed through visual abstractions?* — As we are interested in representing large volumes of multivariate data, it is necessary to consider approaches that can manage the expected visual complexity. Visual abstractions of data are often created as attempts to increase the visibility of specific characteristics, but this comes at the cost of data fidelity. For instance, data aggregation methods may combine groups of data points that have similar characteristics into a single visual element that highlights their common traits. This decreases the number of elements that must be drawn, which improves computational speed while reducing visual noise, making the resulting visualization easier to read and even more aesthetically pleasing. However, combining or distorting elements in favor of emphasizing certain proprieties or portraying general values and trends may result in the obfuscation of smaller events that could have been relevant to the problem being solved. Moreover, the less accurate that the representations are to the original data, the more likely viewers are to misinterpret them. As such, we must consider the types of visual abstraction approaches that can be applied, in addition to how and when such approaches should be used.

- *Can coordinated interaction between multiple views facilitate the exploration of multivariate datasets and provide new insights?* — By using multiple views to focus on different proprieties of the same dataset, the visual variables necessary to represent high-dimensional data can be more easily managed. This allows for the most appropriate model to be used for each specific propriety, such as representing relational data with networks and temporal data with timelines. While this may come at the cost of increasing the complexity of interpreting and managing multiple models, coordinated interaction can be used to more easily keep track of significant data points and proprieties across multiple views, even when they feature different visualization models. Thus, we intend to explore the efficacy of a CMV framework in navigating multi-dimensional data and how it can be used to dig-down and discover new information, while keeping in mind the principles of fluid interaction and how viewers perceive data. Moreover, such methods may also be applied to explore and compare between multiple datasets.

- *What considerations must be taken to ensure that a tool containing multiple visualization models and data analysis methods can be used by users with varying levels of experience?* — There are several pertinent challenges inherent to the development of an interactive visualization tool from the perspective of its potential users. On one hand, the visualization models must be encoded with visual variables that appropriately represent their respective data variables while taking into account a person's limitations, such as visual overload or colorblindness. On the other, navigation through the tool itself should be intuitive whenever possible, and actions that would result in errors should be anticipated and prevented, usually while providing additional help. This requires research into both of these areas, as well as user tests throughout the development of our tool. Furthermore, as we intend to give a particular focus to the representation of molecular biology datasets, we must also consider how researchers from this field of study would interact with the developed tool, as these may include users with a low level of experience with data visualization tools.

In summary, we intend to explore the visualization and analysis of multivariate datasets through the development of a tool with a CMV framework. This requires the implementation of methods that can receive, parse, filter and represent data, while providing users with the ability to interact, mine and refine in order to discover information that could lead to new insights. Moreover, we must contend with the challenges of representing relational and time-series data, particularly visual complexity management and the comprehensible encoding of temporal behaviors exhibited by large groups of data. Such must preceded by a thorough analysis of the state of the art as it is necessary to not only understand the available visualization method at our disposal, but also the limitations of current visualization tools.

1.2 METHODOLOGY

For the design and development of the visualization tool, we adopted Ben Fry's methodology [51], which consists of several steps that establish a path from the collection of raw data to its representation and interaction with users. This is a flexible methodology where various steps can be iterated through successive refinements and validations as to progressively improve data mining, visual encoding and user interaction. In regards to visual encoding, we followed the principles proposed by Jacques Bertin [18] and Edward Tufte [152] to help create the initial infrastructure for our visualization models and ensure a higher standard of efficiency in communication.

Furthermore, to aid in the development of interactive functionalities that do not hinder the user, we explored the principles of fluid interaction [42] which center around supporting a user's immersion and involvement. In this sense, fluid interaction refers to continuous or smooth experience, uninterrupted by potential issues originating from uncertainty in their actions or a lack of feedback. These principles include the implementation of animated transitions, providing immediate visual feedback when performing actions, integrating interface components into the visualization, and allowing the user to make changes intuitively. Following the purposed methodology, both the models and the user interface are then refined throughout the development through surveys and usability tests performed with users of varying levels of experience in data visualization, as to obtain a wide range of feedback.

1.3 CONTRIBUTIONS

In this section we will overview our contributions to the representation and analysis of multivariate datasets, in particular time-series data, through interactive visualizations and CMVs. The cumulative result of our research and development is CroP, a tool that can receive multiple types of dataset files and represent them through various visualization models while providing users with the methods to explore and analyze the data, as to discover and extract patterns and key data points. From this work we can highlight the following contributions:

Survey of Visualization Methods and Tools — A survey on the state-of-the-art of data analysis methods, visualization models and existing tools that are relevant to the development and design of CroP. In this survey, we present an overview on the visualization of both relational and temporal data at different levels of detail, the interactive exploration of the data using multiple coordinated views to compare and contrast between diverse data elements, and the discovery of patterns across large quantities of data elements through dimensionality reduction, feature extraction and clustering algorithms. We also present a discussion on current visualization tools oriented towards the analysis of biological datasets. This survey resulted in the following publication:

- António Cruz, Joel P. Arrais, and Penousal Machado. Interactive and coordinated visualization approaches for biological data analysis. Briefings in Bioinformatics, v. 20, no. 4, pp. 1513–1523, 2019. [29]

Visualization of Temporal Networks — The initial version of CroP utilized an interactive network visualization model to represent the behaviors of molecular networks over time. The tool employed various methods to explore and organize data, including different network layouts, clustering, an integrated biological database, and a timeline for navigating through time-series data. To further analyze temporal attributes, the timeline could be distorted into Time Curve using a force-directed layout where time points are spatially positioned according to their similarity. The visualization model was validated with the use of time-series gene expression RNA-Seq data from the HIV-1 infection. This work resulted in the following publication:

- António Cruz, Joel P. Arrais, and Penousal Machado. Interactive Network Visualization of Gene Expression Time-Series Data. In 22nd International Conference Information Visualisation (IV), pp. 574-580. IEEE, 2018. [28]

Modular Workspace — In order to expand CroP's efficacy in analyzing temporal networks and its ability to solve a wider range of problems, its framework and user interface were reworked to be more flexible. We developed a modular workspace where each visualization model and respective interface functionalities are contained within a flexible panel that can be resized and moved, allowing users to adapt the workspace to the current dataset. Moreover, the workspace is divided by a grid and automatically adjusts the size of panels and prevents overlaps as to help maintain its organization. This modular framework accommodated the integration of new visualization models and functionalities, such as the ability to analyze multiple datasets simultaneously by loading them into different panels. This work resulted in the following publication:

- António Cruz, Penousal Machado, and Joel P. Arrais. CroP — Coordinated Panel visualization for biological networks analysis. Bioinformatics, v. 36, no. 4, pp. 1298-1299, 2020. [31]

Multidimensional Analysis — Through the development of CroP's functionalities and visualisation models directed at the exploration of temporal data, we further explored time-series functionally and aesthetically. Firstly, we complemented the Time Curve model with temporal glyphs, a supporting timeline graph and a lens-based approach, directed at aiding in the interactive discovery and analysis of temporal patterns across complex datasets. In this regard, we demonstrated how these methods can be used to analyze and identify the main agents at the source of significant instances in three biological datasets. Secondly, we introduced Time Paths, a force-directed and parameter-based layout that can dynamically transform a Time Curve model to smoothen the visual elements and transitions between time points, while also reducing visual noise in favor of overall patterns. This work, its experimental results and subsequent user tests were disseminated in the following publications:

- António Cruz, Joel P. Arrais, and Penousal Machado. Exploring Time-Series Through Force-Directed Timelines. In 24th International Conference Information Visualisation (IV), pp. 328-335. IEEE, 2020. [30]
- António Cruz, Joel P. Arrais, and Penousal Machado. Force-Directed Timelines: Visualizing & Exploring Temporal Patterns. Big Data Research, v. 27, pp. 100291, 2022. [33]

Coordinated Panels Visualization Tool — Finally, the previous contributions culminated in the development of the visualization tool entitled CroP, which incorporates the modular workspace, the visualization models for multivariate data and multidimensional data analysis methods. The tool was subjected to an iterative validation process of interface and model tests with participants from different areas of expertise, including Information Visualization and Computational Biology. This work resulted in the following publication:

- António Cruz, Joel P. Arrais, and Penousal Machado. Multivariate Data Exploration Through Coordinated Views. IEEE Access, 2022. [**cruz2022access**]

1.4 STRUCTURE

The research and development of CroP involved the implementation of diverse visualization models and data analysis functionalities. The full scope of this work and its contributions are described throughout this thesis document, which is structured as follows:

In the *State of the Art* chapter, we present a review of related work, beginning with an overview of relevant data analysis and visualization models used in the representation of relational and temporal datasets. We explore the advantages and limitations of multiple views, managing the visual complexity of representing large datasets, and interactive systems. The chapter concludes on a discussion of existing tools and their functionalities in regards to the growing necessity for more dynamic visualization tools and methods that can assist in the exploration and analysis of complex datasets.

In the *Overview* chapter, we introduce CroP's functionalities from a user perspective, providing a general overview of its abilities in the representation and analysis of different types of datasets. We present the structure of its user interface, the available data analysis methods for sorting datasets and highlighting patterns, and a description of each of the visualization panels, their layouts and coordinated functionalities.

The *Analysis Workflow* chapter showcases how the implemented models and methods can be navigated and utilized to explore different types of data and identify data points or groups exhibiting significant patterns of values. This chapter focuses on the usability of CroP as a whole, in particular functionalities that can be coordinated across multiple visualization panels and their utility in data analysis and pattern discovery.

In the *Framework* chapter, we describe the implementation of each of the visualization models, diverse methods and data analysis algorithms into CroP. While the previous chapters focused on user interaction, this chapter details the technical aspects of each functionality, as well as the motivation for their inclusion and additional considerations, such as measures for error prevention and to improve performance.

In the *Experimentation* chapter, we present our analysis of various datasets through CroP, along with the corresponding visualizations created by the tool. The experiments include simple datasets used as a baseline to test the models, as well as both biological datasets, such as gene expression time-series, and non-biological datasets of varying sizes and complexity. The results of each experiment are complemented with a discussion.

In the *Validation* chapter, we present all of the model and interface tests performed by groups of users from different fields of study and levels of experience, both with visualization tools and data visualization in general. We describe each of the tests preformed and thoroughly analyze the results, discussing the performance of the users, their difficulties and how this impacted the development of CroP.

Finally, in the *Conclusion* chapter, we summarize our work and discuss its contributions in regard to our research questions, as well as future work in the research and development of CroP.

2 STATE OF THE ART

A vast quantity of visualization tools has emerged over the past decade as a response to the need for analyzing unstructured and increasingly larger datasets, particularly within the field of Biology [82]. The datasets that are often the target of modern research in the field of computational biology are often classified as complex due to possessing multiple characteristics of what is known as 'big data'. While the description of big data may vary, it is typically characterized large volumes of data that include a variety of different formats, data structures and variable types, as well as other characteristics that negatively affect the velocity at which data is retrieved, analyzed, validated and represented [58].

The amount of research data and the speed at which it is gathered have been increasing along with the technological developments that have taken place across multiple fields. The representation of large volumes of data may lead to slow performance, particularly in interactive environments, as well as a large amount of visual noise and no perceptible structure. Beck et al. [14] presented a survey, which identifies visual scalability as one of the main challenges in developing scientific graphs, emphasizing a lack of visualization methods that work along with data reduction methods, particularly when handling time-series data or dynamic structures [7]. Wang et al. [160] also argues that graphs derived from scientific datasets can be significantly large and complex, advocating for graph simplification and data mining to more easily find community structures and track features over time.

The heterogeneity of biological data can be particularly difficult to manage as the datasets are often multivariate and contain varied structures with temporal or spatial attributes [138]. Additionally, these datasets are often complimented with data integrated from external databases. The challenge in representing diverse datasets stems from choosing graphical elements that can properly convey the values, properties and relationships to the user. This may warrant the exploration of new visualization metaphors that are able to better integrate different types of biological data, such as integrating network-based metabolic pathways with gene expression data [127]. Regarding time-series data and multiple experiments performed under different conditions, the development of techniques for visualizing changes in the data is an ongoing challenge, particularly between two or more networks [14].

Due to these characteristics, modern visualization approaches should be able to not only represent various data structures simultaneously but also provide exploratory methods that allow the identification of meaningful relationships that would not be perceptible through data analysis algorithms alone.

2.1 VISUALIZATION BACKGROUND

Visualization is a significant step in knowledge discovery, as it can be used in conjunction with data analysis to highlight and identify patterns, trends and outliers while engaging users through aesthetic graphical representations and helping them make decisions [64]. As such, understanding the available visualization models and how visual elements are perceived is necessary when choosing those that are most appropriate for representing a dataset.

2.1.1 *Data Analysis*

Data analysis is an essential part of the process of extracting knowledge from data, in particular from the large and complex datasets that are characterized as big data. Techniques such as feature selection, dimensionality reduction and clustering algorithms are used to filter and order data, reducing computational loading times by removing or hiding irrelevant data groups, highlighting patterns and leading to new information that can support the user's navigation and queries.

Exploratory data analysis (EDA) in an approach for data analysis where both statistical and visual techniques are employed to maximize insight into a dataset, uncover hidden patterns, detect outliers, and ultimately formulate new hypotheses [102, 153]. Statistical techniques for EDA include dimensionality reduction and cluster analysis, particularly useful when dealing with large datasets, where showing all data simultaneously can be both confusing to the user and computationally intensive. These techniques can be applied during a pre-processing stage in order to restructure the dataset and only maintain the data that should initially be loaded, rather than keeping the entire dataset in memory. Reducing the amount of data on screen can be achieved either by discerning which data is of most significance through the use of feature selection, through dimensionality reduction techniques or by partitioning the data through clustering and then aggregating it into simple elements that represent groups.

— *Dimensionality Reduction*

Dimensionality reduction methods map data to a lower dimensional space, reducing the number of variables with minimal loss of information [59]. These methods find and extract linear relationships that explain the correlation between multiple datasets and between variables, as well as highlight batch effects or outliers. Owing to the high dimensionality in omics datasets, they have been used in exploratory analysis to better understand molecular pathways in cells and their role in diseases [102], as well as in extracting patterns from gene expression data where there is typically a large amount of noise [20, 78, 156].

Singular value decomposition (SVD) and principal component analysis (PCA) are common dimensionality reduction techniques used in the analysis of multivariate data [158], as well as in the projection of data to create visual representations. SVD can detect weak expression patterns by finding and extracting small signals from gene expression data where there is typically a significant amount of noise. PCA searches for a linear projection of data that preserves the variance in data while minimizing noise and redundancy [78]. As such, it is able to lower dimensional representation of data with small reconstruction errors. In the analysis of gene expression data, particularly when it is measured across multiple time points or conditions, PCA can be used to group together genes with similar behavior patterns [118].

Other techniques have been applied in the analysis of biological time-series, such as recurrence quantification analysis, used in the study of non-linear processes to quantify the number and duration of recurrences in time-series data [164], and time warping algorithms, used for mapping corresponding expression states across different time-series from RNA or protein expression data [1].

— *Feature Selection and Pattern Recognition*

Feature selection is used to identify distinguishing characteristics from a set of candidates that, ideally, can be used to extract patterns and differentiate groups of elements while avoiding noise [75, 163]. In bioinformatics, feature selection has been applied to diverse problems such as the identification of genes with particular biological annotations through their expression levels [93]. Similarly, pattern recognition involves finding sets of patterns that occur frequently in the data, which are also useful in creating distinguishable groups and classifier construction. It has been is used in bioinformatics to understand underlying processes in complex biological networks, as well as identify differentially expressed genes, genotype patterns, and network motifs [96]. Motifs are common subgraphs or patterns identified in networks [106] that have been used to predict interaction patterns of proteins in PPI networks [4] and in analyzing gene regulation networks [151].

— *Clustering*

Clustering is the unsupervised classification of data, where a finite amount of unlabeled data is organized into a discrete set of groups. In general, the objective of clustering is to obtain clearly discernible groups composed of elements that have a high degree of similarity between each other, with the intent of revealing hidden data structures or meaningful patterns. Clustering methods can be applied to a wide range of problems, and there are multiple surveys of clustering algorithms directed at the analysis of gene expression data [17, 77] and PPI networks, as well as for time-series [125] and big data [47].

K-means is a common clustering approach which outputs a set number of groups. It achieves this by iteratively calculating group centroids and reassigning points to groups based on the minimum distance to each centroid until stability is achieved. This is useful in determining position-based clusters on visualizations where points have a fixed position, such as scatter maps and certain networks. However, the user needs to define the number of clusters and may have to run it multiple times to find the most appropriate setup.

Hierarchical clustering algorithms produce a nested series of partitions by picking criteria based on similarity to either merge clusters using agglomerative methods or split them using divisive methods [74]. This results in a hierarchical structure than can be represented with a dendrogram, showing where each cluster merged or split [74]. In the analysis of gene expression, hierarchical clustering methods are often used to group genes that exhibit similar expression patterns over time or over diverse experimental conditions [124, 157].

When representing and analyzing various experimental conditions, biclustering algorithms can be used to perform simultaneous clustering on rows and columns, which can be applied to the analysis of gene expression matrices to identify similar subgroups of genes under specific subset of conditions [99]. However, biclustering can attributed data to different groups simultaneously, resulting in overlapping clusters. The visualization tool BicOverlapper [134] provides an example of multiple clustering representation, portraying nodes with symbols that indicate the number of clusters that they belong to.

2.1.2 *Visualization Models*

In this section we will go over prominent visualization techniques found in current and past tools and variations, focusing on models and layouts used in the representation of biological data. Understanding the visualization models available and the variety of existing graphical representations in a necessary step in deciding those which are most appropriate for representing a dataset.

— *Linear Visualization*

Linear representations are among the simplest visualization models, being used to map data points to observe general patterns or trends. These are often used in biological visualization tools as secondary views that portray additional information about sections of the dataset, such as line charts [36, 147], bar charts [38, 94], histograms [124, 139] and scatter plots [8, 67].

— *Parallel Coordinates*

Parallel coordinates have a similar representation to line charts but are able to represent a higher number of variables [94, 134], listing each variable across a separate axis with values ordered based on properties, such as connectivity, density, centrality or quantitative annotation. However, there exists the limitation that each axis can only have two other neighboring axis, with the first and last axis only having one. A notable variation of parallel coordinate plots is Hive Plots [85], where the axis is arranged in a circle and the edges are drawn as Bezier curves, resulting in a more compact visualization that more easily be embedded into other visualization models or used as small additional views.

— *Heatmaps*

Heatmaps are prevalent among biological visualization tools [19, 111], typically used in the representation of time-series gene expression [40]. While variations to heatmaps are not common, a hexagonal grid layout was presented by GATE [97]. Clustering algorithms can be used to sort heatmaps by ordering the rows of genes, so that those with similar temporal patterns of expression are placed close to each other. If a heatmap is hierarchically clustered, it can be accompanied by a tree structure that shows where each cluster either merged or split, known as a dendrogram [56, 67].

— *Networks*

Networks, or node-link diagrams, are particularly proficient at displaying multivariate data and its relationships, being used in biology to represent PPI, gene regulation and biological pathways [118]. Nodes can represent multiple attributes through basic visual properties like color, shape and size, as well as by being labeled with additional information [89, 133]. However, a greater number of variables can be represented simultaneously through complex representations, known as glyphs. Chernoff faces are an early example of glyphs, which were used to represent various aspects of living conditions in Los Angeles through face symbols with variables mapped to different eyes, mouths, face shapes and colors [25].

Relationships in networks are usually portrayed with edges, lines that connect two nodes that can be characterized using visual attributes such as color, direction and weight. Edges with clear directions are characterized as directed, while those with a discernible width can be described as ribbons. Additional correlations can be inferred through the similarity of visual proprieties defined by the attributes of the nodes, such as proximity, style or other elements that convey the formation of groups. A prominent problem with networks is the representation of large quantities of edges, which often results in unintelligible 'hairballs' which may hinder the extraction of information or new knowledge [85, 115]. To resolve this, the positions of nodes can be calculated through clustering algorithms and force-based layouts [50], with objectives such as edge crossing minimization and grouping similar elements [133, 154]. Additionally, edge bundling reduces visual clutter by drawing their paths closer to those with similar directions [140], creating organic bundles of edges with clear directions that are easier to follow.

The node placement on a network is determined by the chosen layout, which can extend to three-dimensional environments. The position of a node can directly reflect its spatial proprieties, such as in maps and cartograms, or its position can be determined by its other proprieties or relationships with other nodes. In the case of rigid layouts like circular or spiral layouts [34, 86], nodes may have fixed positions. In these cases, the position would be determined by their order, making these layouts useful for portraying ordered events like time-series data, recurring processes and periodic behaviors. As an alternative to networks, pathways can be represented orderly through path lists [116], but these can be extensive, and nodes present on multiple paths will be repeated.

The position of data elements can be calculated iteratively through the use of force-based layouts, where nodes are subjected to attraction and repulsion forces to shift their positions based on their attributes and relationships. This is a nature-inspired method where individual elements are given simple behaviors in order to promote self-organization, with objectives such as edge-crossing minimization or clustering similar elements, while reacting to changes in the environment and user inputs in real time. This is useful in portraying changes in the data over time through the use of animation, such as movement and transitions between sizes, colors and shapes, as well as adapting the visualization in real time to data elements that get added or removed, merged or split.

— *Trees*

Trees are networks with a hierarchical structure where nodes belong to clearly defined, ordered levels. The highest element with no parents is considered the root node, while those that have no children are called leaves. In the field of biology, trees are used in phylogenetic analysis [110, 119] and in the visualization of DNA microarray data [132].

When employing hierarchical clustering techniques, dendrograms are commonly portrayed alongside the resulting clustered visualization, often along with heatmaps, in order to show how each group was formed and how it is composed. MLCut [157] and BioJS [60] use circular dendrograms, where each level in the hierarchy extends along the circumference of a circle (Figure 2.1). In the circular layout, the root node is located at the center of the graph, while nodes can be placed in subsequent circles that each represent a new level. The circular shape also lends itself naturally to the typical tree, where each level is larger than the previous one.

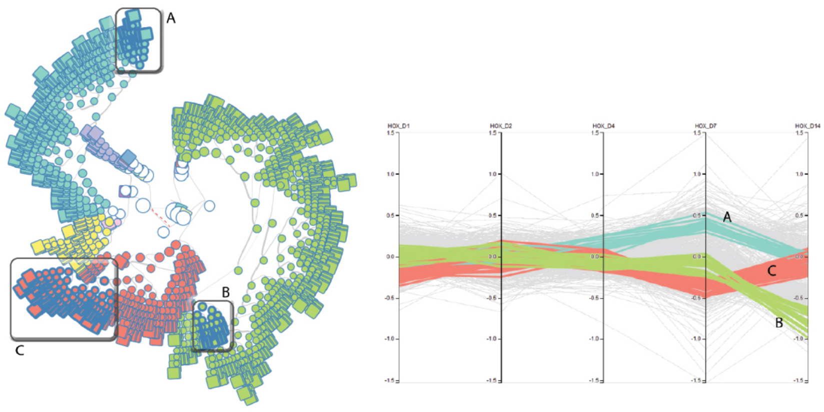


Figure 2.1: MLCut is an interactive tool for visualizing heterogeneous data sets which has been used for pattern discovery in time-series gene expression data, providing the user with the ability to define clustering parameters and dynamically create clusters by preforming cuts on multiple levels on a circular dendrogram.

Layered visualizations can also be used to order data hierarchically. For instance, Arena3D [137] presents a three-dimensional approach to a layout comprised of multiple layers, where two-dimensional networks are displayed parallel to each other, each comprising one level, interconnected through edges. Cerebral [12] has a layered layout option that splits a network into well-defined levels, each representing a different class of proteins (Figure 2.2). Other hierarchical structures include sunburst and icicle visualizations, used by Taylor et al. [146] to visualize gene expression experiments on the developmental mouse.

2.1.3 *Encoding and Perception*

The perceptual phenomena that occur when visualizing graphical elements have been the target of studies that seek to better understand the role of encoding in problem-solving. Choosing the most appropriate visual proprieties to represent specific types of data and how the resulting elements should be presented to the viewer requires some comprehension about the limits of human perception, particularly when the amount of information available is overwhelming. To this end, sets of principles have been established with the aim of providing an objective foundation to a largely subjective field. These principles serve as guidelines to design visualizations that provide a more intuitive experience for the users and relay information more easily, while avoiding nonexistent relationships to be perceived.

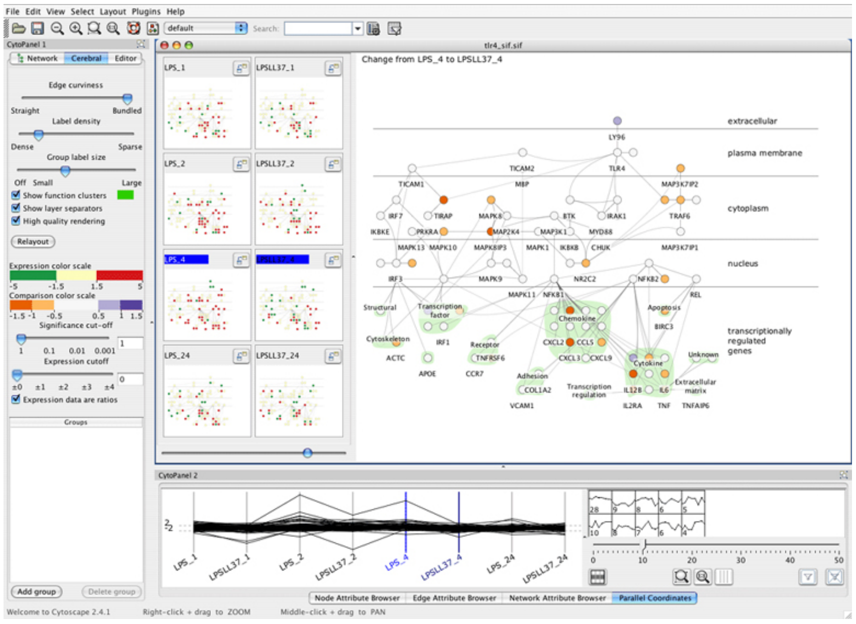


Figure 2.2: Cerebral is a Cytoscape plugin that provides multiple network layouts, including a layered network which organizes groups by cellular level and coordinated small multiples representing different temporal states. Gene expression time-series data can be clustered, using line charts to represent the average values of each group.

Jacques Bertin [18] helped create the foundation for visual encoding by establishing the effectiveness of graphical proprieties in representing diverse variables, such as how a quantitative value should be mapped to an element's position or size but not to its shape. This is complemented by principles that describe the perception of elements based on these properties. For instance, the Gestalt laws explain how elements that are close together or share graphical proprieties, such as color, shape or direction, tend to be associated as being a part of the same group [57]. They also defend the use of uninterrupted, straight or smoothly curved graphical elements to portray relationships, how closed elements are interpreted as wholes. These guidelines are at the base of approaches that group similar elements, such as clustering and edge bundling. Vehlow et al. [155] presented a comprehensive survey that categorizes the representation of groups on graphs based on the visualization model, structure and visual attributes.

While we are interested in how to portray more easily perceptible relationships and patterns, uncertainty should also be minimized. This means that encoding should prioritize data fidelity while avoiding the creation of vague or unintelligible elements. In this regard, Tufte [152] established principles for 'graphical excellence', which favors clarity and precision while avoiding distorting the data for the purpose of aesthetics. Furthermore, Dasgupta et al. [37] proposed a taxonomy that comprehensively describes cases of uncertainty at both an encoding level, which includes handling missing values and graphical limitations of the screen, and at a decoding level, such as indiscernible relationships in cluttered graphs or clusters.

2.2 COMPARATIVE VISUALIZATION

A visualization environment that enables the representation of several multivariate datasets simultaneously can help identify new meaningful relationships, but the number of variables and types of relationships in the data may not fit any one specific visualization model. In such cases, data visualization tools often use comparative views, an exploratory visualization technique that enables the representation of diverse datasets simultaneously by composing multiple visualization models, which can have user interactions coordinated between them [159]. Comparative views can be effective in discovering patterns and unforeseen relationships, identifying and understanding outliers and gaining insight from comparing multiple datasets, or the same dataset using different models [128]. In this section, we review the layouts of visualization models found in across multiple biological visualization tools that use comparative views. Additionally, we identify strategies to explore relationships between large volumes of data, such as their abstraction into manageable elements that can be compared across views.

2.2.1 *Guidelines for Multiple Views*

The amount of information that can be displayed on screen is limited by both the available space and the user's cognitive limits when interpreting large amounts of diverse data representations. It is important to consider how the placement and availability of the views impact the user's navigation and cognitive ability to understand the data to extract new knowledge. In this sense, design decisions benefit from understanding how the workspace environment is perceived. Baldonado et al. [159] presented guidelines on the use of multiple views, noting that they should be used when there are multiple types of attributes, models, user profiles, levels of abstraction or genres. Additionally, multiple views are effective in the discovery of correlations or disparities in the data as they allow for data to be extracted and compared on the screen rather than mentally, which is less straining on the user. While an overview of the data can be helpful, allowing the user to isolate and visualize a particular section of the data, it may also be cognitively overwhelming. As such, a complex visualization should be divided into multiple views that provide more detail and easier management. However, the addition of views should be justified, as every view introduces additional complexity to both the program and the user.

2.2.2 *Composition and Layout*

The composition of the visualization environment determines how diverse visualization models can be compared and how their relationships will be perceived. Previous surveys have categorized the composition of graphical elements or groups of elements into: juxtaposition, superimposition, nesting and encoding [55, 61].

— *Juxtaposition*

Juxtaposition is the most common composition mechanism, where elements are displayed side by side. To compare between views, each can be assigned a space by either dividing the work space [94, 124] or through individual windows [52, 67]. The advantage of windows is that each visualization can be contained even when it is larger than the available space and be navigated through interaction. Additionally, the user can be given control over windows, such as changing their size and position, as well as either hiding or removing them from the workspace. Comparison through juxtaposition can be used to identify patterns and relationships through common graphical properties, or for contextual information. In an overview + detail layout, one view provides an overview of the dataset, while other views focus on specific sections of the visualization with additional details [69, 128]. This encourages navigation, as users can drill-down on different sections without having to roll-up in between.

When the same type of visualization models is displayed multiple times in smaller sizes in a sequence or grid, they are referred to as small multiples [135]. This layout can be used to create static or dynamic overviews of data representations that have various states, such as time-series [69, 97] or experiments performed with different parameters [95]. However, new views have a cognitive impact on users, and smaller changes can go unnoticed, as they shift their attention between views, particularly between small multiples.

— *Superimposition*

Superimposition is an approach for highlighting structural similarities and differences by stacking the same visualization models [27, 44], helping users identify small changes more effectively than juxtaposition. Additional relationships can also be shown by superimposing new elements, such as drawing edges on top of matrices [97, 141]. However, considering that the information is overlapped, it may be difficult to identify individual elements. As such, the scalability of this approach is reliant on the existence of proper interaction techniques to navigate the data [55].

— *Nesting*

Nesting views consists of embedding a visualization model into another one's structure. Unlike in superimposition, the nested model is treated as an element of the parent visualization. Some biological tools use linear visualizations as glyphs, embedding line charts [54, 72] and bar charts [84] into networks, having been used to represent temporal data associated to each gene. Basic node graphical properties can still be altered, such as using different background colors to represent an extra variable [129]. It is also possible to embed more complex visualizations, such as glyphs, to represent relationships between multivariate datasets, such as heatmaps or other networks with edges drawn between each other [94, 145]. However, glyphs add a significant amount of visual complexity, so their use should be justified. Depending on the quantity and type of glyphs, the amount of information presented simultaneously can be cognitively exhausting to the user and result in slow processing speeds, particularly in interactive environments.

— *Encoding*

Differences and similarities can be computed and encoded in a new visualization where regions of interest are explicitly highlighted [55], making them easily identifiable. Encoding time-series through animation is a natural way of conveying changes over time, but it is limited by human perception capabilities [91]. Transitions between successive states can be smoothly animated by interpolating values of properties like color and size [147], but details go unnoticed in short transitions, and it is difficult to compare between time points. Furthermore, the intermediate events between time points may not be accurately portrayed if values are interpolated. Alternatively, a timeline is able to simultaneously represent multiple time points through a variety of scales, shapes and layouts [22], but graphical representations are limited because of space. By using a timeline as a navigation element that accompanies other data visualizations, the user can switch between states in other views and focus on key moments. Choosing and observing temporal states as static visualizations makes it easier to identify details, while differences can be compared by switching between time points.

— *Compound*

When representing diverse datasets, multiple composition mechanisms can be used simultaneously to show different relationships and compare data. For instance, Pathline [103] and MulteeSum [104] represent temporal gene expression profiles through a curvemap, a visualization composed of a grid of area plots. To help analyze these data, the final column and row superimpose their respective plots for an overall comparison. Despite this, the amount of information shown can be overwhelming, and the user would benefit from visual indicators that highlight regions of interest, particularly in Pathline. However, the curvemap is also used to relate time-series to other types of data. Pathline uses a pathway visualization that encodes genes and metabolites, which can be selected to get added to the curvemap (Figure 2.3). MulteeSum implemented the curvemap alongside a plot visualization to relate the time-series gene expression to the spatial location of their respective cells (Figure 2.4).

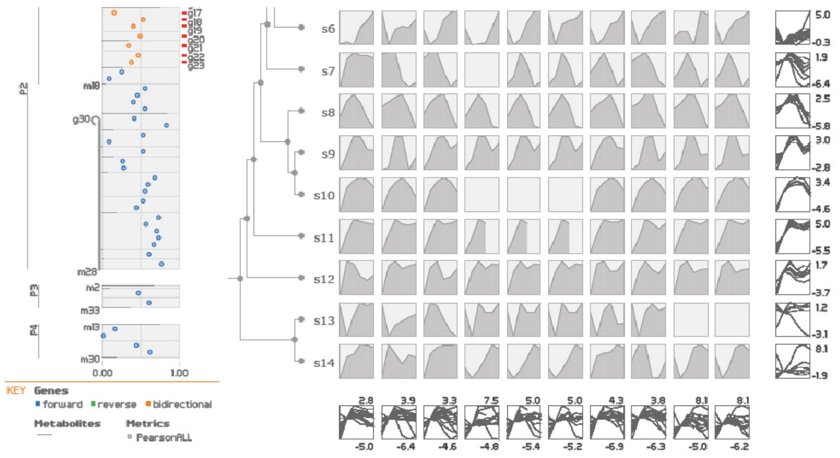


Figure 2.3: Pathline uses a grid of small multiples of line charts to show gene expression time-series data coordinated as a alternative to heatmaps, with a linear pathway visualization that provides an overview of aggregate similarity scores for genes and metabolites.

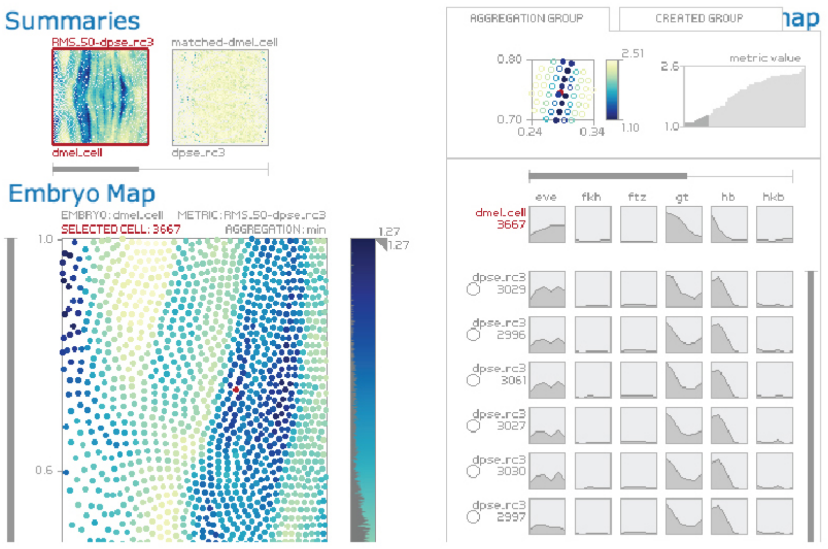


Figure 2.4: Multeesum represents the simultaneous visualization of time-series gene expression data and spatial location of where genes are expressed using a colored point map and a grid of small multiples of line charts, creating visual summaries of the data by aggregating similar measures into groups.

2.2.3 *Managing Visual Complexity*

Reducing the apparent size of the information space is a key strategy in managing complexity in data visualization, particularly in large networks where visual clutter is a frequent obstacle [113]. For instance, Kiwi [154] reduces complex interaction networks by isolating significant gene sets, calculating the shortest path length between each pair and drawing only the best edges.

— *Data Aggregation*

Managing visual complexity can also be achieved through data aggregation, where groups of data points are converted into a single one [91]. These groups are usually formed by clustering algorithms and consist of elements that have one or multiple similar characteristics. They can then be represented by visual proprieties associated with the whole, such as using colors, symbols or size to represent an average of values of a variable. While these methods are more complex than just filtering data, they maintain the presence of all the data, providing overviews of large datasets which may be useful for understanding overall patterns or behaviors.

Several biological visualization tools have used aggregation to represent clusters of time-series gene expression data [12, 44, 56]. By calculating the mean of every value over every time point, a cluster can be represented with a line chart that represent the average variation over time within that cluster. These tools then list the line charts to the user sequentially as small multiples that can be selected to either highlight the respective cluster on another view or open a profile view that superimposes every gene. VisBricks [95] stands out among other tools by representing time-series gene expression clusters

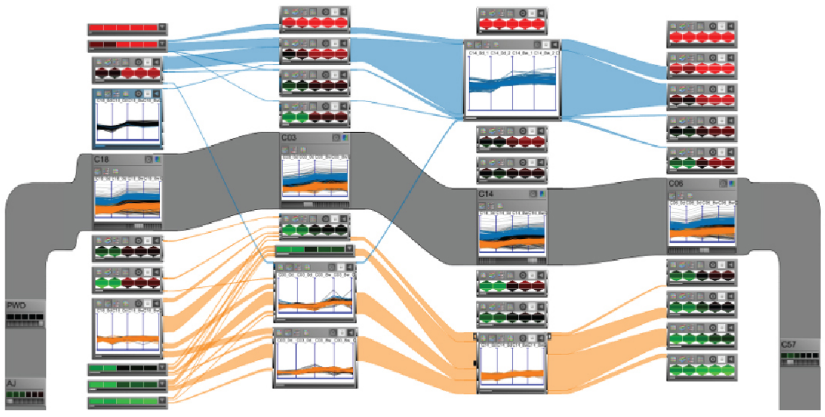


Figure 2.5: VisBricks can represent multiple datasets of time-series gene expression data, where clusters of each set are represented as node glyphs in an axis on a parallel coordinates layout, with edges drawn between sets. The representation of clusters can be switched between a bar-chart, a histogram or a compact view.

using multiform visualizations, where each cluster can be represented as either a line chart, a histogram or a colored compact view of the data. These are displayed as glyphs in a parallel coordinates visualization where each axis corresponds to a different gene expression dataset, connecting shared data between clusters in different datasets with ribbon edges (Figure 2.5). The advantage of this layout is that clusters can be sorted along each axis, but relationships to other axis can only be directly drawn between with those one each side.

— *Simplifying Networks*

In networks, groups can be determined directly from identifiable relationship structures between the data, such as motifs. Dunne and Shneiderman [39] propose motif simplification, where common network sub-graphs are identified and then replaced with a simple symbol that is representative of the layout of each respective motif (Figure 2.6). Maguire et al. [100] also propose simplifying motifs but through glyphs that can be represented using three varying levels of detail, which portray not just the structure of the motif but also attributes of individual nodes. However, the scalability of this approach is limited, as individual attributes can be hard to discern for large motifs, but glyphs can also represent the average attributes of the group.

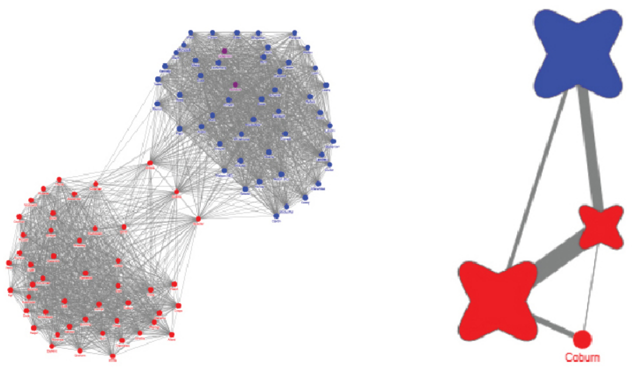


Figure 2.6: Example of motif simplification proposed by Dunne and Shneiderman to improve network visualization readability.

Analyzing the evolution of a network over time and identifying key moments by comparing each state sequentially may prove difficult if the changes are too subtle or if there is a large quantity of states. In this case, aggregation techniques can be used to abstract networks into simple elements that can be compared simultaneously in large amounts by reflecting their current states into graphical proprieties.

Bach et al. presented Time Curves, a visualization model which utilizes multidimensional scaling to position time points in low-dimensional space in such a way that their relative distance reflects the similarity between their attributes [10]. This is achieved by calculating the similarity between time points using data-specific metrics and then applying a force-directed layout on the timeline to attract time points based on their similarity, resulting in a bent timeline whose shape reflects the behaviors of the data, such as significant events, cyclical patterns, regressions and outliers. Elzen et al. presented a similar concept which further showed how this layout is able to represent the overall behaviors of complex systems over time [43]. In their work, the properties of a network at each point in time are abstracted into a point on a two-dimensional plane to build a Time Curve, which will then portray the structural changes in the network over time. Through the Time Curve it is possible to identify the periods in which the network remained with a specific structure, the moments when this structure changed, the intensity of these changes, and when the network returned to similar, previous structures.

Visualizing and analyzing changes in network visualizations has additional challenges involving modifications to the structure in addition to changes in values. This includes direct modifications to spatial variables, as well as the addition, removal or transformation of both nodes and edges. As such, a network may present significant changes between any two states, and the user may need to identify which changes have taken place, in addition to the causes and future consequences of these changes. Structural differences between two networks can be highlighted through colors or labels, or by using animated transitions that draw attention to changes in position or size through the use of movement. ELICIT [32], a visualization for evolutionary algorithm data, portrays individuals evolving along an interactive timeline which controls two other visualizations. Individuals can be represented with tree graphs, and changes throughout their evolution are shown by highlighting the differences between any two subsequent trees, where new branches and nodes are drawn in different colors. Additionally, the trees are drawn using a force-based layout that adapts the position of the nodes as new branches get added or removed, resulting in movements that draw attention to areas that have been altered.

2.3 COORDINATED INTERACTION

Interaction and dynamic visualization environments play a major role in analyzing complex datasets, as the user needs to be able to navigate through diverse datasets, compare data points and identify relationships [123]. In an environment with CMV, user interactions with a visualization model, such as selections and filters, can be dynamically applied to similar data points represented across other views. Coordinating interaction helps users keep track of data and more quickly identify significant relationships and patterns, particularly between diverse visualization structures [142].

Ideally, interaction should be fluid, which mean that any set of actions that the user sets out to perform should be a continuous and smooth process, without interruptions. The concept of fluid interaction is based on a set of general principles, which can be applied to support the user's immersion and involvement [42], such as: using animated transitions, providing immediate visual feedback, integrating interface components into the visualization, and allowing the user to make changes intuitively.

Baldonado et al. [159] described additional guidelines for designing coordinated interaction. These highlight the importance of the time costs of each interaction, such as the computational time necessary to process each change. Time costs also include the time the user takes to understand and switch between visualization models, which can be reduced by using consistent graphical elements to represent the same types of variables between multiple views and different visualization models. Additionally, the user's attention can also be diverted to regions of interest through perceptual cues, such as animation, sounds and highlighting.

2.3.1 *Navigation*

Interactive visualizations, particularly large-scale graphs and maps, are traditionally navigated using panning and zooming techniques where the visualization is moved or transformed. As such, they can be used for position-based filtering, bringing specific elements into focus while moving others off the screen or de-emphasizing them.

— *Levels of Detail*

Given that there is a limited amount of screen space, the size of visual elements and the amount of details shown must be controlled, such as through zooming. Zooming is an action that resizes an area or elements of a visualization and it is characterized as geometric when the affected parts are resized without any changes to their content. Zooming in moves elements off-screen while increasing the distance between them, which also increases the amount of empty space. Semantic zooming methods take advantage of this by adapting the content to the current scale [120], which can involve the addition of labels with details or contextual information [84]. In some cases, the structure of the visualization can be drastically changed to add new points and relationships related to the data in focus which may have been previously hidden [16].

An early implementation of semantic zooming was included in the zoomable user interface [15], consisting of a single large information surface where elements can be placed on at any position and scaled to any size through the use of a graphics engine called Pad++. Rendering is context-sensitive, where the amount and type of details depends on the zoom level and zooming between levels is animated for smooth transitions. This serves as a possible alternative to multiple views, where all the elements can be placed in a single area which the user navigates within, although it lacks organization.

Alternatively, focus + context methods expanding an area in focus without pushing elements off the screen [66]. This is commonly achieved by increasing the size and amount of details on a single element or a group, distorting the position of the surrounding elements and decreasing their size in order to maintain every element on screen for context [150]. TreeJuxtaposer [110] presents trees side by side for comparison, allowing the user to select a rectangular area and then enlarge it freely by dragging the corners, dynamically adapting the size of the rest of the tree and enlarging the same area across other trees through coordination (Figure 2.7). Notably, this tool also implements a draw order where the nodes and branches in focus are drawn first when rendering changes, which maintains the user's attention on regions of interest. Tominski et al. [149] presented a survey on lens-based focus + context approaches, showing how these can be used as interactive objects that can filter data.

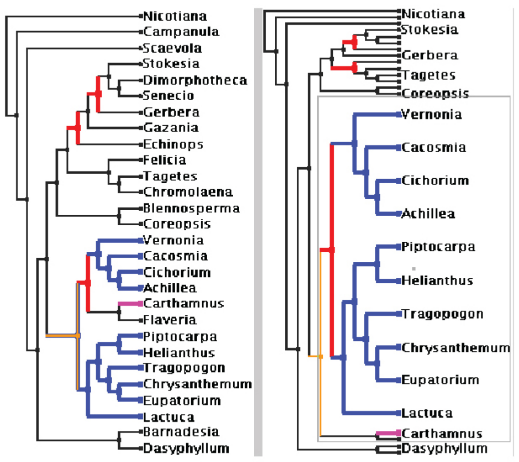


Figure 2.7: TreeJuxtaposer is a tool designed for the comparison of large biological hierarchical networks, such as phylogenetic trees, comparing them through color and providing interactive navigation, such as adjustable and coordinated zooming.

— *Interactive Aggregation*

In a top-down approach to navigation, big data visualizations can initially present the user with an overview of the data created through aggregation and sampling methods, while interactive functions allow the user to drill-down and access the original data. Hierarchical aggregation results in a network with clearly defined levels of detail [41], which can be navigated through recursive expansions or reductions by selecting parent nodes to either add or remove child nodes, as used by VisANT [72] and AVOCADO [144]. iHAT [65] presents a hierarchical table approach that combines the visualization of sequence and expression data, in which the user can aggregate rows and columns of a heatmap interactively. However, the visualization can get cluttered, as the user drills down and expands groups. To prevent this, either a separate view can be created to represent the contents of an expanded set of aggregated data [38] or multiple views can be defined to represent a set of number scale levels [105].

Visual aggregation can also be combined with semantic zooming to balance the number of graphical elements on screen in relation to the scale level. For instance, as the user zooms in on aggregates, the data groups in focus are expanded, while the remainder are pushed off-screen.

— *Visual Guidance*

In general, navigation methods should not just give users freedom to explore but also guide them. This includes establishing limits to prevent users from going out of bounds, as well as visual hints that keep users aware of where information of interest is located, particularly when it is located off-screen. Schulz et al. [136] present a table-based approach for visualizing bipartite biological networks, where the scroll bars contain selection markers that indicate regions of interest, which would otherwise be off-screen because of the vertical length of the tables.

Additionally, one can also consider the degree of interest of each data element in their representation. This was proposed by Furnas in 1986 [53], noting that information items have different levels of importance to different observers. By identifying the user's interests or objectives (such as through their selections) it may be possible to determine which data points are most relevant and then visually highlight them.

CMV can also be used to guide the user and avoid navigation problems. For instance, when visualization is zoomed in enough, panning to another location may require the user to zoom out first before moving and then zooming in again. This breaks the user's workflow, as each step may require a significant amount of time to process in complex visualizations. This can be avoided by using two views to create an overview + detail layout, where one visualization presents a constant overview of the dataset that can allow the user to select a new point which will be viewed in detail in the second visualization window [60, 161].

— *Sorting Multiple Views*

Navigation through environments with multiple views is also reliant on how the views are presented to the user. Several early concepts on window interaction are present in E. Kandogan and B. Shneiderman's elastic windows [80]. This method utilizes a hierarchical organization where the role of each window can be sorted as a nested rectangle tree structure, depicting an easily identifiable hierarchy between them. Furthermore, any operation performed on any level of the hierarchy is propagated to lower level windows inside that group recursively. Organization of the windows is handled by a space-filling tiled layout, where resizing a window will proportionally shrink or stretch the remaining windows dynamically, in accordance to the available space. Instead of overlaying windows and obscuring other views, this method helps the user organize the work environment while maintaining every current view visible. Cerebral [12] also presents an example of coordinated navigation, applying panning and zooming across small multiples representing different states of the same biological network (Figure 2.2). This results in very view focusing on the same region, allowing users to more easily find differences or patterns in the network across several temporal instances.

In cases where multiple views are not shown simultaneously, the ability to switch between them quickly is advantageous for either finding the most fitting model for the same data or comparing patterns between them. This requires some consideration over which options should be readily available in a list or though tabs, and which require navigation through menus.

2.3.2 *Data Queries*

Throughout their session, users may need to perform queries to find and focus on sections of the data that are specific to their current objectives. Queries are requests from the user that involve one or multiple constraints, which can either be categorized as searches, when the objective is to find and emphasize a specific element or group, or as filters, when the user seeks to de-emphasize or remove elements from the view and reduce visual clutter [66]. They are performed through inputs that can be characterized as indirect or direct.

— *Indirect Inputs*

Indirect inputs consist of actions usually preformed through separate menus. Qualitative or discrete data can be listed as options using interface elements like tables, dropdowns and checkboxes that are used to switch between what data are visible [36, 134]. Search bars occupy a small amount of space and are useful for finding elements through partial or full names [88, 89] but have the disadvantage of requiring previous knowledge from the user. Numerical input boxes and sliders are more commonly used for handling queries over continuous data, where the user can pick values to establish thresholds, such as upper and lower limits [157].

— *Direct Manipulation*

Direct inputs involve interacting with the visualization, either by selecting the graphical representations of the data or by manipulating elements through handles or widgets. Handles are sections of graphical elements that the user can select and drag to either move or resize that element [69], change the value of a propriety [105] or establish thresholds on the visualization [27, 103]. When the user is meant to have control over several properties, then widgets can be used instead. These consist of small interface elements embedded into the visualization with multiple handles, buttons or input fields [95]. There are also other types of complex queries, such as the grid-based query technique implemented by PivotSlice [165], which subdivides the visualization into meaningful sections.

— *Brushing*

Selections on a visualization model are usually performed by using the mouse as a brush, hence it is known as brushing. If the brush is a point, then data elements are chosen individually, such as hovering over them to display labels [79, 148] or clicking them so they are added to another view [103]. Alternatively, multiple elements can be brushed simultaneously by using a line or an area, which is commonly drawn either as a rectangle or with a freehand lasso.

TimeSearcher [69], a time-series visualization tool, presents an area brushing method: timeboxes. These are rectangular regions drawn by the user on a two-dimensional display of time-series data to perform queries. After being drawn, timeboxes have handles that allow the user to resize them or move to a new position. The result set from the queries will only consist of patterns that are within the constraints of the active timeboxes. The Hierarchical Clustering Explorer [67] is another visualization tool for exploring multidimensional data, which implements interactive approaches for profile queries directly on line charts. This is achieved by drawing a line pattern to query similar profiles and setting a distance measure to establish how similar other profiles need to be to get added to the result set. Additionally, the user can establish upper and lower thresholds directly on the visualization using the mouse. While both these tools are not specifically designed for exploring biological data, the described methods can be applied in the identification of time-series gene expression profiles.

More recently, we find also similar methods employed by DEVIS [122], a tool for visualizing and analyzing the evolution of relaxed functional dependencies over time. This tool utilizes a line plot and a dependency table that dynamically adapt to continuous discovery processes, allowing users to interact with the results through queries and filters, including the selection of an interval of time through a brush to analyze the changes to the dependencies during that period.

More complex brushing techniques have also been developed, such as the compound brushing system developed by Chen [24], where multiple brushing actions can be combined to include, exclude or reverse selections through logical operations (OR, AND, XOR). Another compound brushing technique is presented by Wright and Roberts [162], named click and brush. This method consists appending brushed elements to a list to then highlight intersections and correlations, while additional information and new discoveries is shown in additional views.

— *Coordinated Brushing*

When brushing is used to concurrently highlight related information across multiple views, then it is known as linked brushing. This type of coordination is advantageous for the user, as it maintains consistency through the use of visual characteristics, such as color, weight and shape [67, 134]. This allows the user to identify equal or similar elements across diverse views, find outliers and keep track of changes between groups, such as the addition or deletion of data elements [92]. enRoute [116], ConTour [117] and Pathfinder [115] present table-based approaches for pathway analysis coordinated with network visualizations, where path lists can be extracted to the table through brushing nodes or selected from the table to be highlighted in the network (Figure 2.8). Owing to the number of possible paths, pathway analysis through path lists is particularly reliant on queries. As such, these tools provide sorting methods that bring interesting pathways to the top and linear representations embedded into the cells for easier comparisons between attributes.

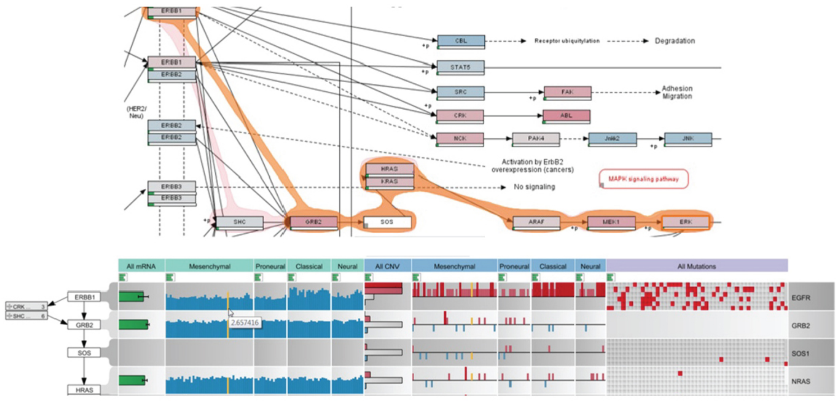


Figure 2.8: enRoute employs a pathway visualization through a network, using colored shapes around nodes and edges to show selected paths and a table that has multiple linear visualizations to represent attributes in each cell.

2.3.3 *Editing*

Data analysis algorithms and sorting methods, such as clustering and force-directed layouts, are essential in the organization and visualization of large datasets. However, these techniques often do not provide ideal results, and it may be necessary to introduce human input. As such, the user should be given the option to fine-tune the visualization, while also being provided with analytical tools that increase the transparency of the implemented methods to better understand the problem and more quickly solve it [70].

— *Setting Parameters*

Regarding tools that provide clustering approaches, these may allow the manual definition of clustering parameters [72, 140]. In particular, MLCut [157] uses sliders to identify and refine gene expression clusters (Figure 2.1), while Furby [145] lets the user to locally or globally adjust the threshold of the biclusters membership values to create well-defined clusters (Figure 2.9).

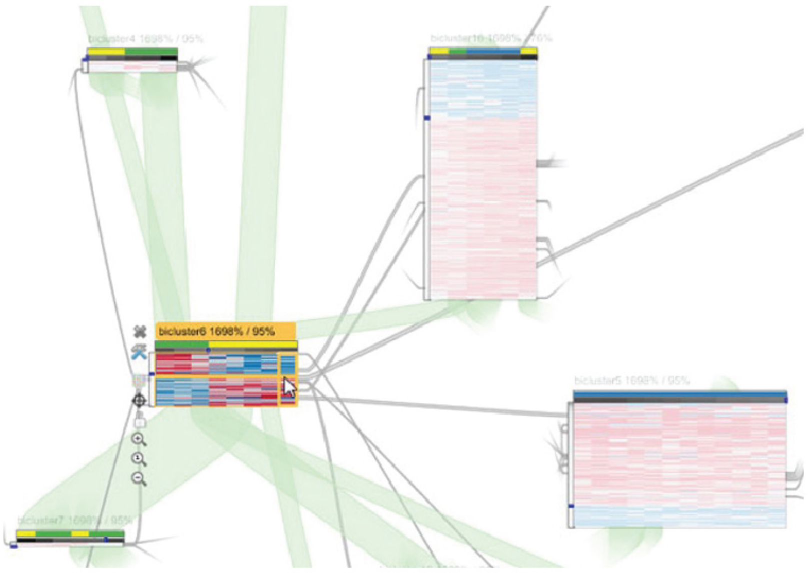


Figure 2.9: Furby provides adjustable biclustering of data, where fuzzy clusters can be converted into hard clusters, and visualization through multiple heatmaps displayed as a network, organized with a force-based layout. To reduce visual clutter, selecting a bicluster will fade out edges that are not connected to it.

— *Custom Visuals*

In regard to editing the visualization itself, some tools allow the selection of which variable is mapped to the color and size of nodes and edges [89, 134]. The visualization environment developed by Abello et al. [2] stands out by allowing the user to set multiple thresholds within an attribute and mapping each of these intervals to a color. A direct method for manipulating the position of elements is dragging, which has been used in networks to rearrange nodes [12, 84], and to move data across multiple views [8, 69]. Further control over biological networks can involve drawing new pathways and managing their visual properties [52, 87].

— *Action History*

The ability to undo and redo actions is helpful in allowing users to edit and explore options and parameters without fear of mistakes. Contour [117] provides a list of the users' previous actions, while PivotSlice [165] presents a user's history visually by saving a thumbnail of the visualization to a separate panel whenever the user performs an edit, which can be used to go back to any of the recorded points. This concept can be explored further by providing methods to record the actions performed in the current and previous sessions and share this history with other users to confirm and extend discoveries.

2.4 DISCUSSION

Visualization tools seek to provide users with the means to analyze diverse datasets to find meaningful relationships that could lead to advances in research. However, these relationships are often complex, and their discovery is hindered by the characteristics of these datasets, in particular their heterogeneity and volume, which are at the base of prominent challenges in data visualization.

2.4.1 *Managing Volume*

Modern visualization tools must contend with the representation of large volumes of data, which is not only computationally demanding, but can also obscure relationships in the data, particularly in network representations. For instance, when a user enlarges a section of a tree in TreeJuxtaposer [110], the remainder of the visualization is dynamically shrunk and kept on screen, but it is difficult to discern information because of the quantity and size of the shrunk elements (Figure 2.7). As such, overviews of the data should avoid overwhelming users with information and instead point them toward regions of interest, while facilitating comparisons. This is a problem of scalability, where the number of visual elements should be reduced while taking into consideration all available data, which may be achieved through statistical analysis and clustering.

Dunne and Shneiderman [39] and Maguire et al. [100] presented approaches to network reduction by representing common motifs through symbols and glyphs with graphical attributes that represent the motif's properties. However, while these methods only aggregate partitions of a larger network, Bach et al. [10] and Elzen et al. [43] proposed approaches that are able to translate states of the data into points on a 2D plane, which can portray behaviors such as outliers and cycles. In the surveyed tools, network aggregation was applied through hierarchy. For instance, VisANT [72] uses interactive aggregated nodes to open or close lower levels of the network. However, the lack of descriptive visual characteristics on the nodes hinders the user's navigation, unlike in AVOCADO [144], where nodes are labeled with symbols and quantities (Figure 2.10). Additionally, semantic approaches are under-explored, as they could be used in this context to dynamically aggregate elements not in focus and keep them for context when the user drills down on demand.

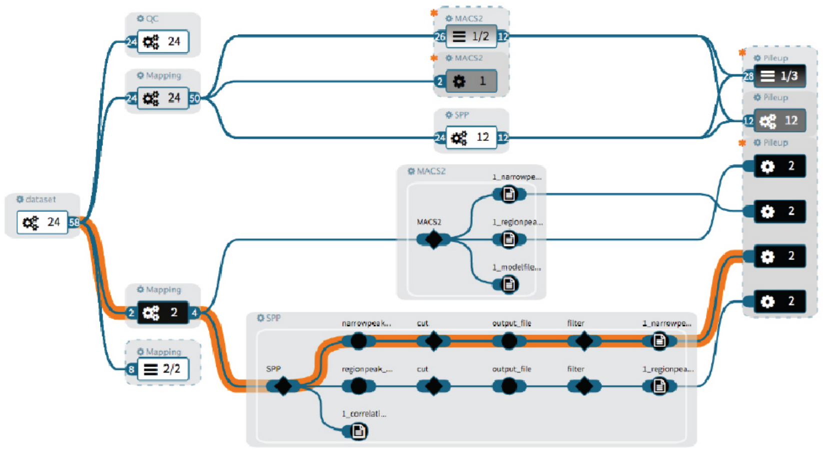


Figure 2.10: AVOCADO utilizes a provenance graph visualization that is aggregated based on the graph's topology, which can then be interactively expanded by the users as they drill-down. Aggregated nodes contain values and symbols that describe group.

2.4.2 *Managing Heterogeneity*

Many biological visualization tools provide a range of models and layouts to represent diverse data structures, including annotations from external databases, but the depiction of relationships between them is still an issue. The most common solution is CMV, as multiple models can organize and represent individual datasets, while coordinated interaction is used to highlight common data points. For instance, ConTour [117] uses coordinated tables, networks and compound visualizations in the analysis of pathways. Its table-based approach is capable of organizing diverse attributes through columns that can be nested, and by using various graphical representation forms within each cell (Figure 2.11). Alternatively, CMV can be used to navigate through different levels of a dataset. Mizbee [105] simultaneously displays four different interactive scales for comparing two sets of chromosomes (Figure 2.12).

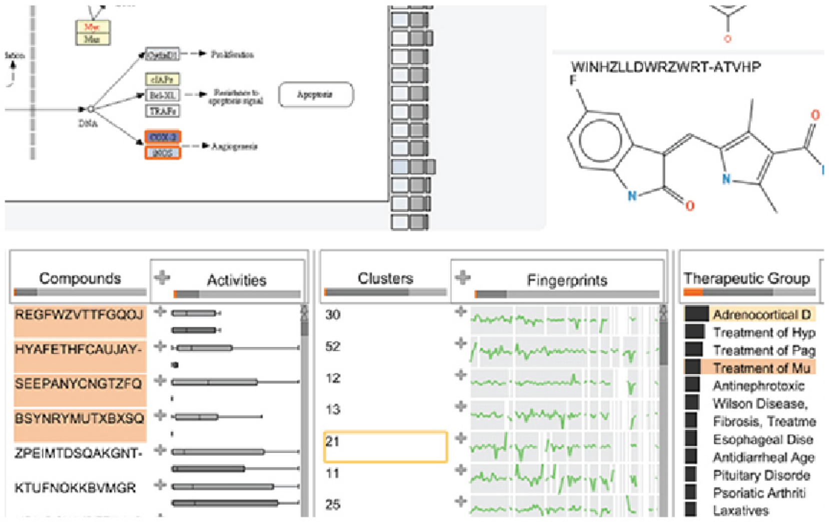


Figure 2.11: Contour is a visualization tool focused on data discovery that displays data through tables with nested columns and sorting methods that bring relevant information to the top, along with a network visualization for pathways and a view for compounds. Variables, such as sequences, can be clustered and then represented with a line chart that uses the median values of the group.

A prevalent challenge in the representation of heterogeneous datasets is the visual encoding of data, as different variables should be graphically consistent and easily identifiable. In particular, time-series gene expression has been a target of multiple visualization approaches that seek to represent relationships between genes that have similar expression patterns. While heatmaps have been a standard approach in representing time-series, line charts have risen in popularity in the biological visualization tools developed during the past decade. Not only are the values of expression profiles more easily interpreted through a line chart than a row of colors, but they can also be compared by overlaying multiple charts. Superimposition is useful for detecting trends, but it is also a significant source of uncertainty, as individual elements may be difficult to distinguish, so it should be used purposefully. For instance, Matse [27] uses superimposition to overview time-series profiles and allow the user to directly apply thresholds based on the resulting visualization, while MLCut [157] encodes superimposed profiles with color to distinguish between the clusters created by the user's choice of parameters (Figure 2.1). Alternatively, by calculating the average between time-series, the profile of a group can be represented with a single linear visualization. Cerebral [12] makes use of this to list clustered time-series profiles through small multiples, which can be selected to highlight each group of genes in a network model (Figure 2.2).

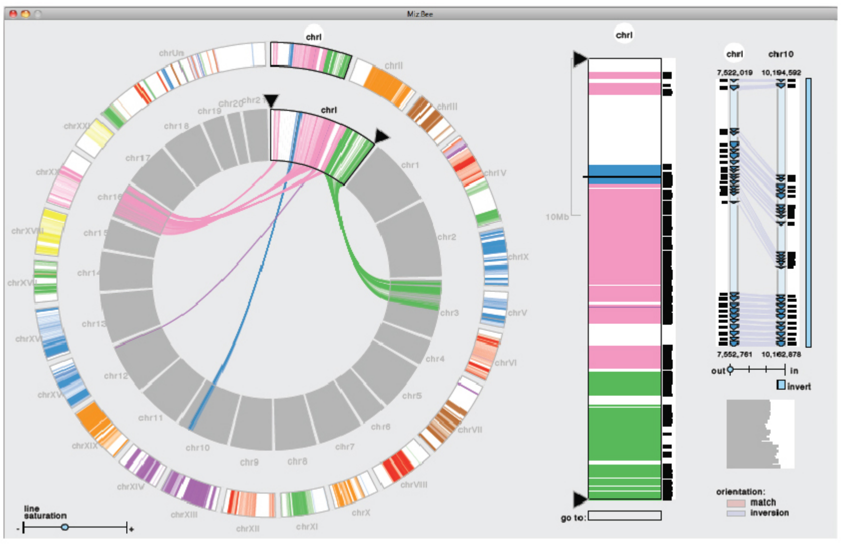


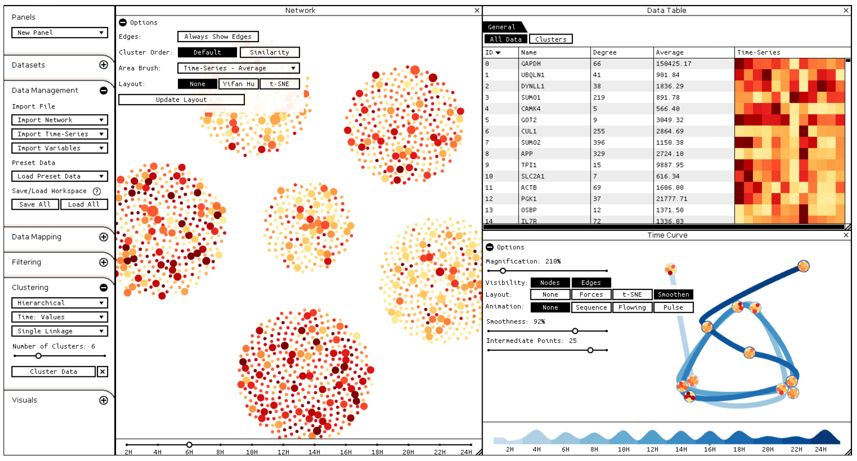
Figure 2.12: MizBee is a visualization tool for comparing chromosomes at multiple scales shown simultaneously through coordinated multiple views, which include a circular network with edge-bundling.

While the temporal attributes from gene expression can be clustered and compared through compact graphical representations, we must also work toward new visualization metaphors to analyze relationships between other attributes and datasets. In this regard, Pathline [103] and MulteeSum [104] relate a curvemap, a grid of time-series profiles that shows trends using superimposition, with both pathway and spatial data through interactive coordination. However, these tools highlight the need for encoded visualizations that directs users toward patterns that may be of interest, in particular between views (Figure 2.3, 2.4). As such, this warrants the exploration of new models that integrate multiple biological datasets, designed with the purpose of portraying their relationships. For instance, while VisBricks [95] uses heatmaps to represent multiple gene expression datasets, time-series can be clustered and represented through small multiform visualizations, where relationships are drawn between the clusters across datasets (Figure 2.5). These are integrated in a single visualization with a parallel coordinates layout, but this does present a limitation, as each dataset can only be compared against those on each side.

In summary, tools for visualizing multidimensional data are becoming more comprehensive and flexible but still present limitations in their visualization approaches. Ideally, the development of new visualization tools should focus on user-centered interaction and coordinated environments, as well as new visualization metaphors capable of showing patterns, key changes and outliers by enabling the comparison between large multivariate datasets. Data can be aggregated into simple graphical representations that provide an informative overview of the data, while interactivity provides users with the ability to navigate through different levels of detail by drilling down and access details on demand through brushing. Future developed visualization environments should automatically support users in their queries by predicting regions of interest and dynamically adapting the visualization to the type and amount of information on screen. While most of the surveyed tools still use static representations, force-based layouts can be used to react to dynamically changes in the environment and user inputs in real time using fluid transitions. At the same time, users should have control to make manual adjustments, both in customizing the visualization and fine-tuning parameters, as human input may help discover key relationships between elements and groups, which would not be easily discernible solely through data analysis.

3 OVERVIEW

CroP is a data visualization tool developed in Java using the Processing library [126], designed to represent and analyze multivariate data, in particular relational and temporal data. While it is able to process generic datasets, there is additional support for biological datasets, such as the integration of an external database for cross-referencing gene proprieties, allowing it to be used to explore and discover patterns across PPI networks and gene expression time-series.



Edges: Always Show Edges
Cluster Order: [✓ Default] Similarity
Area Brush: Time-Series - Average
Layout: [✓ None] Yifan Hu t-SNE

Hierarchical
Time: Values
Single Linkage
Number of Clusters: 6

ID	Name	Degree	Average
0	GAPDH	66	150425.17
1	UBQLN1	41	901.84
2	DYNLL1	38	1836.29
3	SUMO1	219	891.78
4	CAMK4	5	566.40
5	GOT2	9	3049.32
6	CUL1	255	2864.69
7	SUMO2	396	1150.38
8	APP	329	2724.10
9	TPI1	15	9887.95
10	SLC2A1	7	616.34
11	ACTB	69	1606.00
12	PGK1	37	21777.71
13	OSBP	12	1371.50
14	IL7R	72	1336.03

Magnification: 210%
Visibility: [✓ Nodes] [✓ Edges]
Layout: None Forces t-SNE [✓ Smoothen]
Animation: [✓ None] Sequence Flowing Pulse
Smoothness: 92%
Intermediate Points: 25

Figure 3.1: CroP's user interface with a loaded temporal dataset, showcasing the options sidebar (left) and three different panels in its workspace: a network panel (middle), a data table panel (top right) and a time curve panel (bottom right).

CroP uses a CMV layout, where interactions may be shared between views to facilitate the navigation between different levels of detail, as well as the analysis of multivariate data and discovery of meaningful patterns. Loaded data is represented through visualization models within flexible panels that can be arranged according to each user's objectives and queries. Data can also be sorted, clustered and filtered, which is reflected through its dynamic visualization models and flexible environment. Additionally, multiple datasets can also be loaded and visualized simultaneously, allowing users to compare them through multiple panels and a differences view.

In this section, we will provide an overview of CroP and its functionalities from a user perspective. We will present its capabilities for data analysis, the representation of multiple variables, the layouts provided by each type of panel, and its ability to handle multiple datasets.

3.1 USER INTERFACE

CroP's user interface is divided into an options sidebar and a workspace (Figure 3.1). In the options sidebar, users can import data files and manage datasets, while the workspace consists of a modular environment where panels containing the visualization models are set on a grid-based layout (Figure 3.2). The size of CroP's window can be adjusted be dragging the edges or maximizing it which will adjust the size of the panel grid and its contents accordingly.

The options within the sidebar are categorized into different groups and their visibility can be toggled based on the needs of the user (Figure 3.3). Circular icons with question marks are located next to options with complex functionalities in order to provide users with instructions or context regarding that function. This information is contained within a text prompt that appears when the icon is hovered with the mouse.

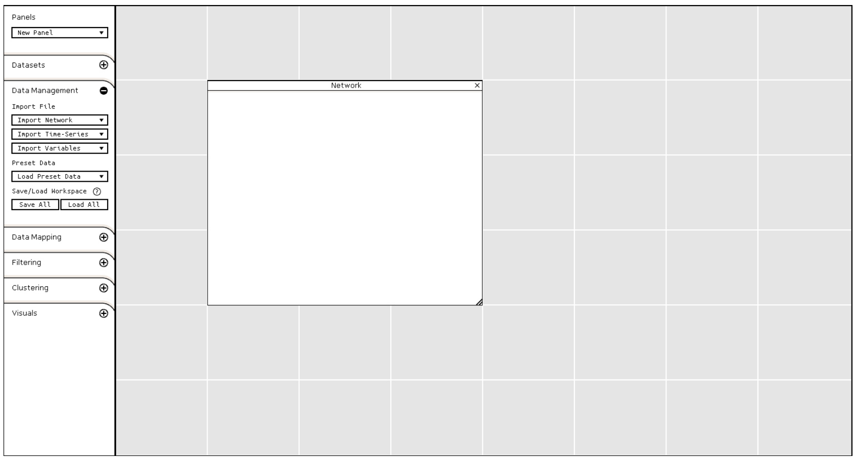


Figure 3.2: CroP's user interface when data has not been loaded and a single empty network panel in its workspace, showing the grid to which panels adjust their size and position automatically.

3.1.1 *Dataset Management*

Under "Data Management", users can import relational data, time-series and multivariate data, whose files can be formatted as either comma-separated values or tab-separated values: Relational data describes the edges of a network, containing all of the direct relationships between the existing data points; time-series data contains ordered lists of values, describing how a propriety of each point varies over time; multivariate data consists of a set of independent quantitative proprieties, which can be used define general attributes for each data point without a defined order.

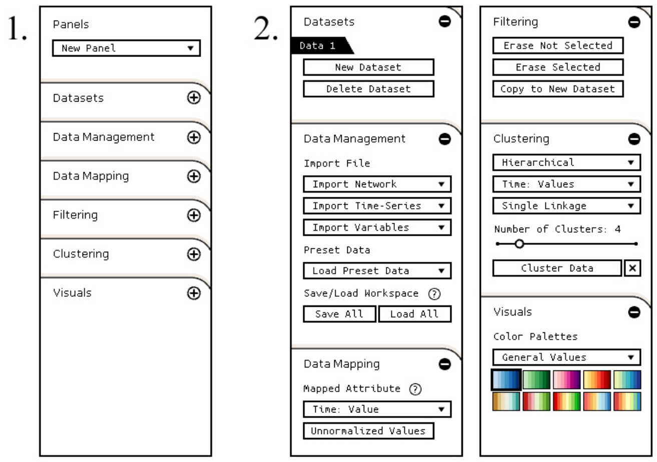
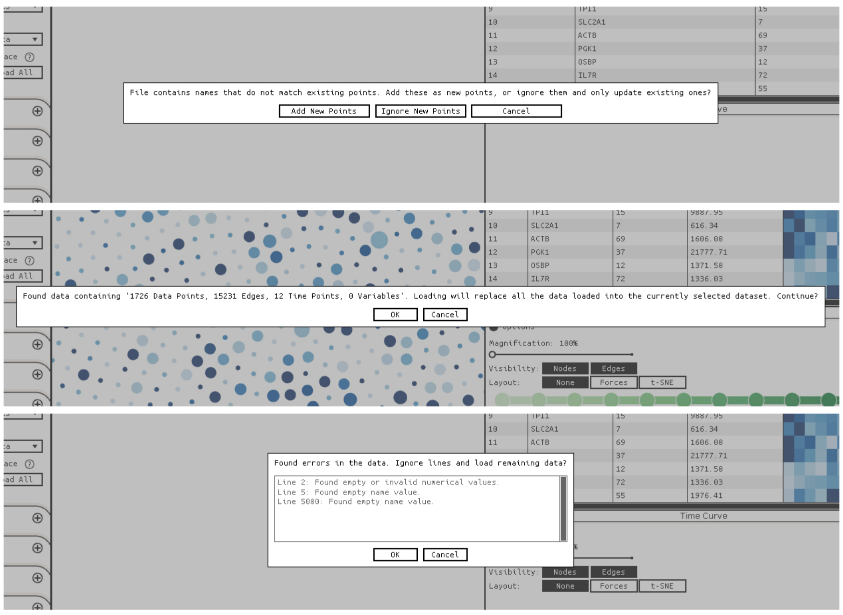


Figure 3.3: Options sidebar with all of its groups collapsed (1.) and opened (2.).

Loaded files will be parsed and the user will be alerted any detected errors, along with the line numbers in which they occurred so that these can be more easily located and corrected (Figure 3.4). When multiple files are loaded, alerts will either allow users to merge data points referenced with the same name on multiple files into single elements, or immediately filter existing data points that do not contain certain loaded variables. If any values in time-series and multivariate files are left blank, the application will interpret this as nodes being "inactive" at those time points or for those variables, and they will be represented in the visualization models accordingly.

In support of biological data analysis, the Gene Ontology (GO) databases were integrated into the application in order to provide additional information and the ability to compare biological elements. If any data points are loaded with names that correspond to the proteins in the database, CroP will associate them with the biological processes of their corresponding proteins within some of the visualization models. Additionally, there are integrated biological datasets can be selected and loaded from the "Preset Data" dropdown, which includes a human PPI network.

After a dataset has been loaded, data can be clustered into groups of nodes with similar proprieties using the options in the sidebar located under "Clustering". Users can select the type of clustering, the attribute being clustered, and the merging criteria. Data elements can be clustered by their position in the network panel, by temporal attributes, or by the values of variables. Data points and clusters selected in the visualization panels can be removed or copied using the options provided in the sidebar under "Filtering". The first two options will either remove the selected data points or remove all but those that are selected, while the last option will instead copy the data points into new dataset, without removing them from the current one. The simultaneous visualization of multiple datasets is further explained in the “Managing Multiple Datasets” section.



File contains names that do not match existing points. Add these as new points, or ignore them and only update existing ones?

Found data containing '1726 Data Points, 15231 Edges, 12 Time Points, 0 Variables'. Loading will replace all the data loaded into the currently selected dataset. Continue?

Found errors in the data. Ignore lines and load remaining data?

```
Line 2: Found empty or invalid numerical values.
Line 5: Found empty name value.
Line 5000: Found empty name value.
```

Figure 3.4: Examples of alerts when merging data (top), replacing a dataset (middle), and finding errors in loaded data (bottom).

At any point, users can select the “Save All“ option to generate a file containing the current state of CroP, which includes the entire dataset as parsed by the application, clustering, panel positions, settings and parameters. This file can be loaded at any point through the “Load All“ option in order to restore the workspace to its previously saved state with minimal loading times, as it bypasses the need to recalculate layouts or clustering.

3.1.2 *Data Mapping*

Data mapping refers to how colors are mapped to numerical values loaded from time-series or multivariate data files. Mapping uses normalized values by default, but they can be unnormalized within the previous options. Additionally, while values can be mapped between the colors of any of the available palettes, we’ll be using the "RdYlGn" palette as a reference in this section, which maps values from red to yellow and then to green, as it better distinguishes between extreme and middle values. Value mapping simply maps colors between the minimum and maximum values of each time-series or variable, from red to green respectively (Figure 3.5). However, if the values are unnormalized, then the colors will be mapped between the minimum and maximum values across the entire dataset.

As time-series data describes a list of ordered values, this allows for color to also be mapped to how the values change over time, using either its variation or tendency. We define variation as the difference between the current value and that of the previous time point, as to represent the variation of values over time. Color is then mapped to the intensity of the variation and whether it is negative or positive, approaching either red and green respectively (Figure 3.6). When values didn't change significantly from the previous time point, points will approach with middle tones, in this case yellow.

Tendency consists of a simple approach that does not take into account values, only how the data shifts between the previous and next time points. Each of the color palette's extremes corresponds to a shift in variation, where green represents a peak of values and red represents a valley, while middle colors represent other behaviors, such as light green representing increasing values, orange representing decreasing values, and yellow representing values that didn't change (Figure 3.7). Tendency mapping is aimed at the analysis of datasets where shifts between positive and negative variation mark significant moments in the data, such as gene expression time-series where peaks of expression represent when proteins have become over-expressed. The type of data mapping can be switched in the options sidebar, within "Data Management" and under "Data Mapping".

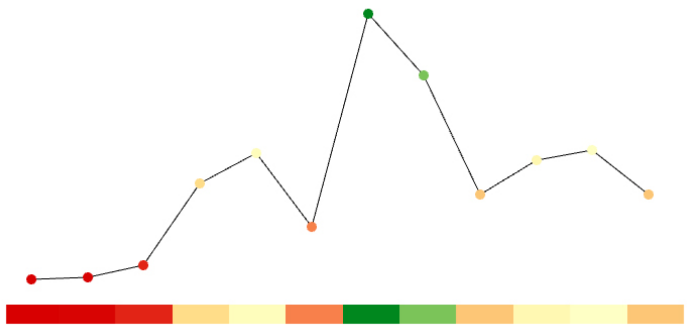


Figure 3.5: Example of value mapping.

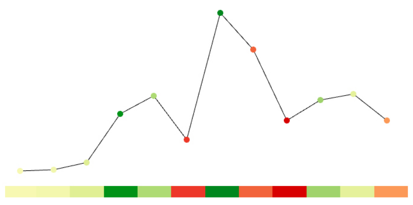


Figure 3.6: Example of variation mapping.

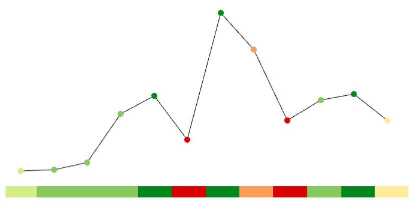


Figure 3.7: Example of tendency mapping.

3.1.3 *Color Palettes*

In order to consistently represent the same types of values across different visualization models, we established a set of color palettes from which users are able to choose from in the options sidebar. Different palettes may be chosen to distinguish between and map general numerical values, temporal progression, and differences between multiple datasets. As default, general values are mapped from light yellow to dark orange, representing minimum and maximum values respectively, while temporal progression is mapped from light blue to dark blue, representing time from the first time point to the last. However, these color palettes can be changed under the "Visuals" tab in the sidebar, allowing users to switch between ten different palettes to represent either type of values.

These palettes were referenced from ColorBrewer [62], a work in which such color schemes have been shown to distinctively represent different ranges of values while also being appropriate for users with any of the common types of colorblindness. However, as these palettes were originally defined for map visualization, they were subjected to minor modifications that allowed them to be used more effectively within the context of our program. In particular, very light colors were either removed or altered to make nodes without outlines more visible against a white background.

The variety in palettes chosen for CroP were in response to a potential need to discern between various ranges of values across different types of datasets (Figure 3.8). The first five color palettes are sequential, representing low values with light tones and high values with dark tones: "Blues" and "Greens" vary between brightness within the similar hues, while "RdPu", "YlOrRd" and "YlGnBl" present variation in both their brightness and hue. While sequential palettes will generally emphasize higher values, those with higher variation in hue (in particular "YlGnBl") allow users to differentiate between values more effectively. The remaining five color palettes are divergent, representing either two very distinct colors at each of their extremes with light colors in the middle, as is the case for "BrBG", "PiYG", "RdYlGn" and "RdYlBl", or representing a larger gamut of colors, as does "Spectral". In general, divergent palettes represent low values with hot colors, such as red, and high values with cold colors, like green, highlighting values on either extreme while also distinctly representing different ranges of values.

Furthermore, as CroP is able to receive and process *null* values as to explore patterns of inactivity in datasets, we established a color for each palette that differentiates inactive nodes from others. While each sequential palette was assigned a bright color that was able to contrast against any other color mapped between its values, divergent palettes simply represent inactive nodes with black, as to contrast with their generally large variation in hue.

3.2 VISUALIZATION PANELS

The visualization panels are windows dedicated to analyzing different types of data, containing multiple visualization models and layouts that represent loaded datasets. While CroP starts up with a predefined set-up, new panels

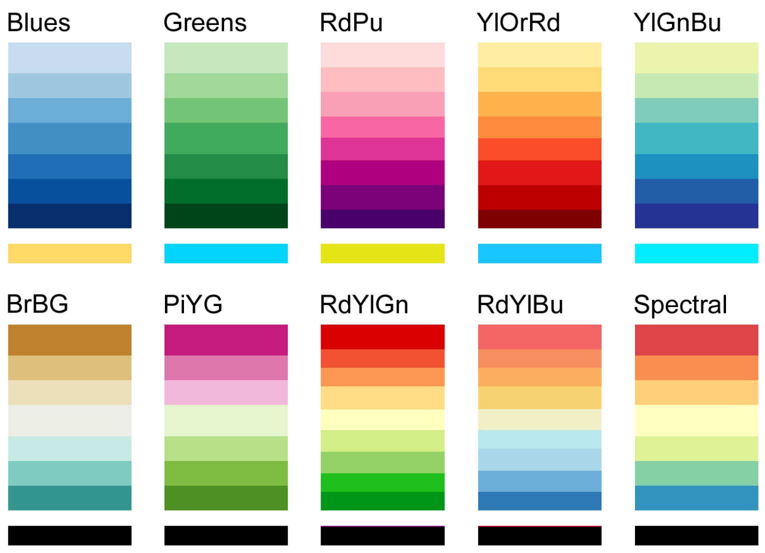


Figure 3.8: List of the color palettes available in CroP. The final color in each palette is used to represent inactivity.

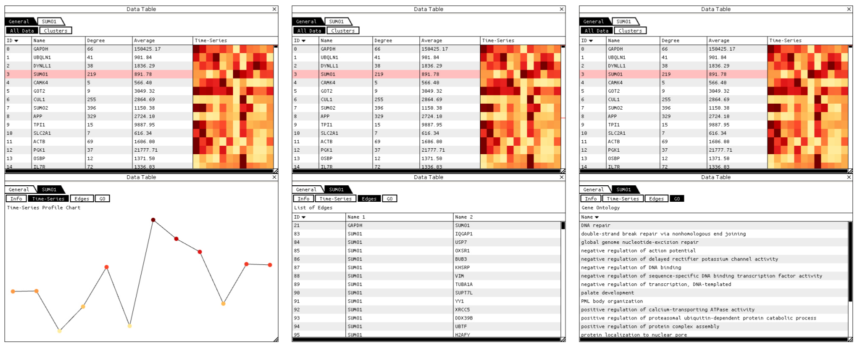
can be added to the workspace from "New Panel" dropdown in the options sidebar. Every panel contains some consistent elements that allow users to organize them: the top bar can be dragged to move the panel, and the corner of the panel can be dragged to resize it. Additionally, the bar contains a button to close the panel and a dropdown that selects the dataset being visualized (when there exist multiple datasets).

When a panel is moved or resized, its corners will always snap to the closet point on the grid of the workspace. This grid layout ensures that the organization of the workspace is maintained, as panels can be sorted and adjusted to make use of the available space. This, for instance, allows users to easily place panels next to each other and resize them to consistent sizes when comparing between multiple similar visualizations. Overlapping panels are handled automatically, where the overlapped panels are resized or moved to accommodate the new changes.

There are four types of visualization panels: Data Tables, Networks, Time Curves, and Multivariate Views. In this section, we will primarily utilize the "YlOrRd" color palette for values and the "Blues" color palette for time.

3.2.1 *Data Table*

The data table panel shows every data point at its lowest level, listing them in a sortable table where rows can be selected to access the proprieties of each individual element (Figure 3.9) or the aggregated proprieties of clusters (Figure 3.10). Different options will be available depending on the type of data loaded and options selected: The "General" button will always contain a list of the dataset's data points; clustering the data adds the "Clusters" button to the top of the panel, which opens a table of aggregated time-series and multivariate data for each of the created groups, where columns display the means of each group's values; loading multivariate data will add a "Variables" button, which provides access to a table of all the existing variables and their values across the dataset.

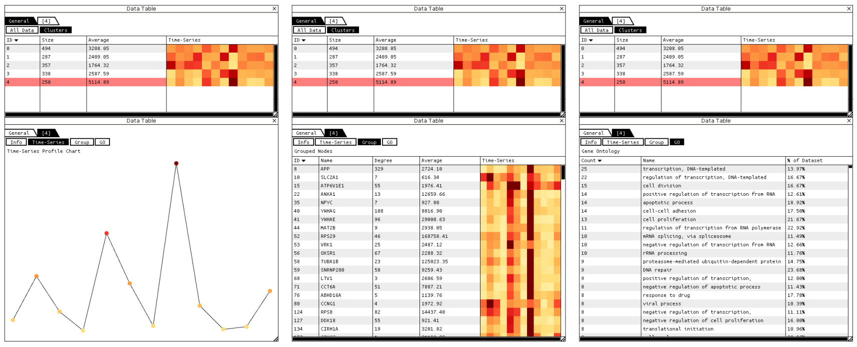


ID	Name	Degree	Average	Time-Series
0	GAPDH	66	150425.17	
1	UBQLN1	41	901.84	
2	DYNLL1	38	1836.29	
3	SUMO1	219	891.78	
4	CAMK4	5	566.40	
5	GOT2	9	3049.32	
6	CUL1	255	2864.69	
7	SUMO2	396	1150.38	
8	APP	329	2724.10	
9	TPI1	15	9887.95	
10	SLC2A1	7	616.34	
11	ACTB	69	1606.00	
12	PGK1	37	21777.71	
13	OSBP	12	1371.50	
14	IL7R	72	1336.03	

ID	Name 1	Name 2
21	GAPDH	SUMO1
83	SUMO1	IQGAP1
84	SUMO1	USP7
85	SUMO1	OXSR1
86	SUMO1	BUB3
87	SUMO1	KHSRP
88	SUMO1	VIM
89	SUMO1	TUBA1A
90	SUMO1	SUPT7L
91	SUMO1	YY1
92	SUMO1	XRCC5
93	SUMO1	DDX39B
94	SUMO1	UBTF
95	SUMO1	H2AFY

Name
DNA repair
double-strand break repair via nonhomologous end joining
global genome nucleotide-excision repair
negative regulation of action potential
negative regulation of delayed rectifier potassium channel activity
negative regulation of DNA binding
negative regulation of sequence-specific DNA binding transcription factor activity
negative regulation of transcription, DNA-templated
palate development
PML body organization
positive regulation of calcium-transporting ATPase activity
positive regulation of proteasomal ubiquitin-dependent protein catabolic process
positive regulation of protein complex assembly
protein localization to nuclear pore

Figure 3.9: Sets of table panels showing an element being selected (top) and various information about that element (bottom): a temporal profile (left), a list of its edges (middle), and a list of its Gene Ontology proprieties (right).



ID	Size	Average	Time-Series
0	494	3208.05	
1	287	2409.05	
2	357	1764.32	
3	338	2587.59	
4	250	5114.89	

ID	Name	Degree	Average	Time-Series
8	APP	329	2724.10	
10	SLC2A1	7	616.34	
15	ATP6V1E1	55	1976.41	
22	ANXA1	13	12659.66	
35	NFYC	7	927.80	
40	YWHAG	108	8016.90	
41	YWHAE	96	29000.63	
44	MAT2B	9	2938.05	
52	RPS29	46	168758.41	
53	VRK1	25	2487.12	
56	OXSR1	67	2288.32	
58	TUBA1B	23	125023.35	
59	SNRNP200	58	9259.43	
68	LTV1	3	2606.59	
71	CCT6A	51	7887.21	
76	ABHD16A	5	1139.76	
80	CCNG1	4	1972.92	
124	RPS8	82	14437.40	
127	DDX18	55	921.41	
134	CIRH1A	19	3201.82	

Count	Name	% of Dataset
25	transcription, DNA-templated	13.97%
22	regulation of transcription, DNA-templated	16.67%
15	cell division	16.67%
14	positive regulation of transcription from RNA	12.61%
14	apoptotic process	18.92%
14	cell-cell adhesion	17.50%
13	cell proliferation	21.67%
11	regulation of transcription from RNA polymerase	22.92%
10	mRNA splicing, via spliceosome	11.49%
10	negative regulation of transcription from RNA	12.66%
10	rRNA processing	11.76%
9	proteasome-mediated ubiquitin-dependent protein	14.75%
9	DNA repair	23.68%
9	positive regulation of transcription,	12.00%
8	negative regulation of apoptotic process	11.43%
8	response to drug	17.78%
8	viral process	10.39%
8	negative regulation of transcription,	11.11%
8	negative regulation of cell proliferation	16.00%
8	translational initiation	10.96%

Figure 3.10: Sets of table panels showing a list of clusters with one being selected (top) and various information about that cluster (bottom): an aggregated temporal profile (left), a list of its elements (middle), and a list of the Gene Ontology proprieties within the cluster (right).

— *Row Selection*

Tables can be scrolled by using the mouse wheel or by using the sliders on its side, while data points can be selected or deselected by clicking on their respective rows. Selections are coordinated across visualization models, highlighting these data points in all network and table visualizations of the same dataset. The positions of selected elements are also marked in the scroll bar with colored bars, allowing users to easily discover their positions on the table.

For each selected data point, a new tab on the top of the table is created. These contain the existing proprieties of the selected data point, including a line chart of its temporal profile, a bar chart depicting the values of its

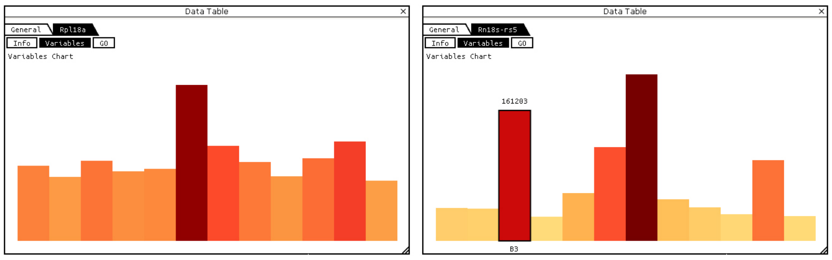


Figure 3.11: Two examples of multivariate data profiles, which utilize bar charts.

variables (Figure 3.11), a list of its edges between other nodes, and a table of corresponding GO terms (if the loaded data matches that of the integrated biological database). If the number of tabs exceeds the width of the panel, tabs can be dragged horizontally to scroll through them. Additionally, selecting a cluster will also create a tab with detailed proprieties of that group, including aggregated profiles of its data, a list of its nodes and a table of aggregated GO terms.

Holding the "CTRL" key will allow the selection of multiple rows simultaneously, each creating an individual tab for the selected element. Alternatively, to select large sections of data points, holding the "SHIFT" key will toggle start/stop points for row selection, where selecting two separate rows in this manner will automatically select every row between them. Due to the high number of rows that could selected simultaneously, this type of selection does not create individual tabs for each row. Instead, a single tab called "Highlighted" is created which contains the aggregate data on every data point selected. Furthermore, to distinguish this type of selection, tabs selected with the "SHIFT" key are colored differently.

— *Column Order*

Columns are kept updated with the current data and loading new attributes will add any new proprieties as columns. Rows can be sorted by any of these attributes by clicking on the respective column's title cell, where clicking it a second time will invert that order. If time-series or multivariate data has been loaded, columns will be added with a sequence of colored squares that match the current value mapping of the data points, as described previously in the "Basic Representation" section. These columns can only be sorted after data has been clustered as it orders nodes by the similarity calculated by the algorithm.

3.2.2 *Network Panel*

The network panel's main visualization model is a dynamic node-link graph that represents data points and their relationships in two-dimensional space.

The position of nodes can be sorted to reflect the relationships between their attributes through multiple layouts, which allows the network panel to be used even when visualizing data that does not possess relational attributes.

— *Basic Representation*

The network visualization utilizes conventional circles and lines to respectively represent nodes and edges, with visualization settings that can be switched in the options menu on the top left of the panel. The visualization can be panned by clicking and dragging the mouse, or zoomed in/out on the current mouse position using the mouse wheel.

If either time-series data or multivariate data have been loaded, nodes will be colored and sized according to the type of color mapping selected (as described in the previous "Data Mapping" section). This will also create a slider at the bottom of the panel that will either display a timeline of the time-series data, or a list of all the variables. These visual proprieties are mapped to the selected time point and variable on the slider, creating an animation when the slider is dragged as each node smoothly transitions between colors and size.

In regards to edge representation, a common issue with the visualization of complex networks as graphs is that drawing hundreds of thousands of edges will often result in a visual "hairball" that provides practically no perceptible information, while also potentially obscuring others visual elements. In our approach, edges have their transparency mapped to amount of edges being drawn, meaning that they fade out as their number increases until they are no longer drawn. However, transparency mapping can be removed by selecting the "Always Show Edges" button in the options menu, making it so that every edge will be drawn regardless of their amount (Figure 3.12).

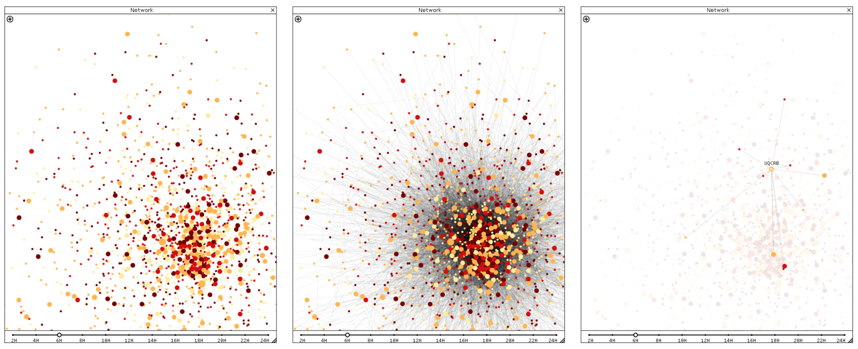
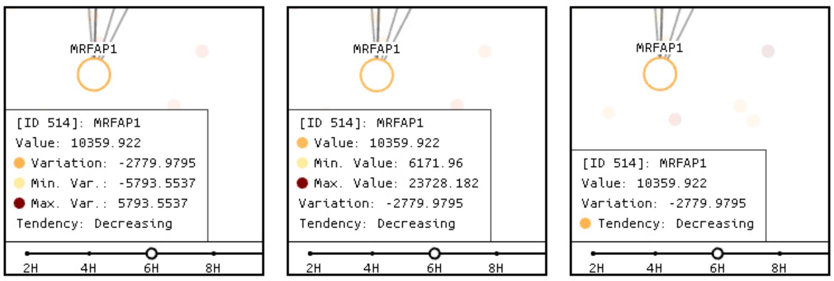


Figure 3.12: In high amounts, edges are hidden by default (left), but can be shown by selecting the “Always Show Edges” button (middle). Selecting a node will only highlight that node’s edges (right).

— *Node Selection*

Nodes can be hovered and selected with the mouse. This will highlight them and draw their names, while also highlighting every node related to those selected including the edges between them. However, these edges will also be subjected to transparency mapping, which will be based only on the amount of edges between the currently selected nodes. Clicking on the panel outside of a node will reset all selections. Moreover, any selections will also be coordinated with other network and data table panels that are showing the same dataset.

Hovering a node with temporal values will also open a small information section on the bottom-left of the panel that displays the values associated with its respective data point, including its current, minimum and maximum values, as well as variation and tendency, and alongside circles that indicate the colors corresponding to these values (Figure 3.13).



```
[ID 514]: MRFAP1
Value: 10359.922
● Variation: -2779.9795
● Min. Var.: -5793.5537
● Max. Var.: 5793.5537
Tendency: Decreasing
```

```
[ID 514]: MRFAP1
● Value: 10359.922
● Min. Value: 6171.96
● Max. Value: 23728.182
Variation: -2779.9795
Tendency: Decreasing
```

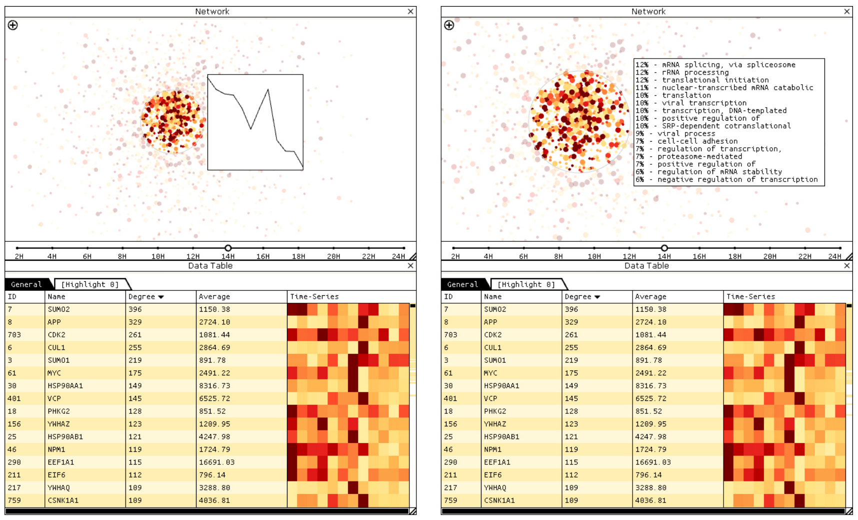
```
[ID 514]: MRFAP1
Value: 10359.922
Variation: -2779.9795
● Tendency: Decreasing
```

Figure 3.13: Hovering a node will display information on its current value in relation to the current value mapping and selected time point or variable. The image displays temporal information shown for the same node when colors are mapped by value (left), variation (middle) and tendency (right).

— *Mouse Lens*

If time-series or multivariate data have been loaded, or if the names of biological nodes correspond with those of the GO database, then an additional analysis tool will be available: the mouse lens. Right-clicking anywhere on the network panel will create a circle around the mouse which will act as a lens, following the mouse and selecting every node inside of it. The aggregated data of every node that is selected in this manner will be displayed in a small visualization next to the lens (Figure 3.14).

For time-series and multivariate data, the visualization will be of a line chart that depicts an average of all the values for every node selected. For biological nodes, the lens will show the percentage of each GO propriety that exists within the selected group. If there exist multiple types of data that can be represented within the lens, the preferred type of data can be chosen in the options menu. The size of the lens can be increased or decreased with



ID	Name	Degree	Average	Time-Series
7	SUMO2	396	1150.38	
8	APP	329	2724.10	
703	CDK2	261	1081.44	
6	CUL1	255	2864.69	
3	SUMO1	219	891.78	
61	MYC	175	2491.22	
30	HSP90AA1	149	8316.73	
401	VCP	145	6525.72	
18	PHKG2	128	851.52	
156	YWHAZ	123	1209.95	
25	HSP90AB1	121	4247.98	
46	NPM1	119	1724.79	
290	EEF1A1	115	16691.03	
211	EIF6	112	796.14	
217	YWHAQ	109	3288.80	
759	CSNK1A1	109	4036.81	

```
12% - mRNA splicing, via spliceosome
12% - rRNA processing
12% - translational initiation
11% - nuclear-transcribed mRNA catabolic
10% - translation
10% - viral transcription
10% - transcription, DNA-templated
10% - positive regulation of
10% - SRP-dependent cotranslational
9% - viral process
7% - cell-cell adhesion
7% - regulation of transcription,
7% - proteasome-mediated
7% - positive regulation of
6% - regulation of mRNA stability
6% - negative regulation of transcription
```

ID	Name	Degree	Average	Time-Series
7	SUMO2	396	1150.38	
8	APP	329	2724.10	
703	CDK2	261	1081.44	
6	CUL1	255	2864.69	
3	SUMO1	219	891.78	
61	MYC	175	2491.22	
30	HSP90AA1	149	8316.73	
401	VCP	145	6525.72	
18	PHKG2	128	851.52	
156	YWHAZ	123	1209.95	
25	HSP90AB1	121	4247.98	
46	NPM1	119	1724.79	
290	EEF1A1	115	16691.03	
211	EIF6	112	796.14	
217	YWHAQ	109	3288.80	
759	CSNK1A1	109	4036.81	

Figure 3.14: Brushing a network with the mouse lens (top) to view the average temporal profile (left) or a count of Gene Ontology proprieties of the selected nodes. Selected nodes are also highlighted in the data table panel (bottom).

the mouse wheel, and the lens can be turned off by right-clicking a second time. By holding the "CTRL" key, multiple nodes can be selected, either by clicking them, or by brushing them with the lens.

— *Layouts*

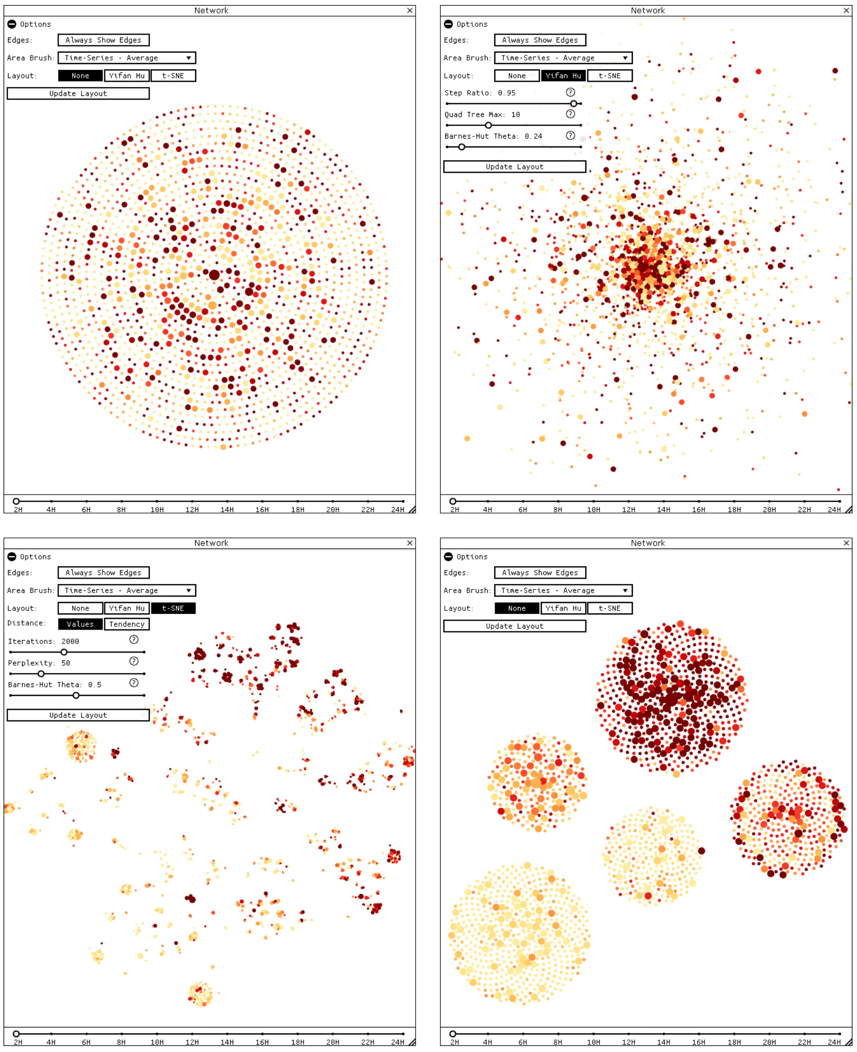
Nodes are initially displayed in a sunflower spiral layout, which utilizes the order loaded from the original file. The options menu provides two additional layouts for the network that can sort nodes based their attributes: the Yifan Hu layout and the t-SNE layout (Figure 3.15). These layouts are described in detail in the "Framework" chapter, under "Visualization Models & Layouts".

The Yifan Hu layout sorts nodes based on their edges, positioning them so that related nodes will be closer to each other. Several parameters of the layout are mapped to sliders in the options menu which can be switched to balance the calculation time of the layout against its accuracy in relatively positioning nodes.

The t-SNE layout will only be available if time-series data or multivariate data is available, as it positions nodes based on the similarity between their values instead of their edges, similarly to clustering but only through position mapping. Like in the previous layout, several parameters of the t-SNE layout can be adjusted through sliders in the options menu where user can choose whether to prioritize computational time over accuracy, as well as whether the layout should focus on local structure or preserve the global structure.

— *Node Clusters*

Clustering the data will apply a force-directed layout over the nodes, grouping them into the selected number of clusters. Each cluster is contained within its own circular area that can be hovered and selected like a node by placing the mouse on its border. Nodes are sorted within each cluster in a sunflower spiral layout where their order is determined by the clustering algorithm, meaning that neighboring nodes may also be more similar to each other (Figure 3.15). Additionally, the relative position of the clusters reflects the relationships between their nodes, as clusters with nodes that have edges between them will be placed closer to each other.



Edges: Always Show Edges
Area Brush: Time-Series - Average
Layout: [✓ None] Yifan Hu t-SNE

Edges: Always Show Edges
Area Brush: Time-Series - Average
Layout: None [✓ Yifan Hu] t-SNE
Step Ratio: 0.95
Quad Tree Max: 10
Barnes-Hut Theta: 0.24

Edges: Always Show Edges
Area Brush: Time-Series - Average
Layout: None Yifan Hu [✓ t-SNE]
Distance: [✓ Values] Tendency
Iterations: 2000
Perplexity: 50
Barnes-Hut Theta: 0.5

Edges: Always Show Edges
Area Brush: Time-Series - Average
Layout: [✓ None] Yifan Hu t-SNE

Figure 3.15: Network panels with different layouts: Sunflower (top-left), Yifan Hu (top-right), t-SNE (bottom-left), and force-directed clusters (bottom-right).

3.2.3 *Time Curve*

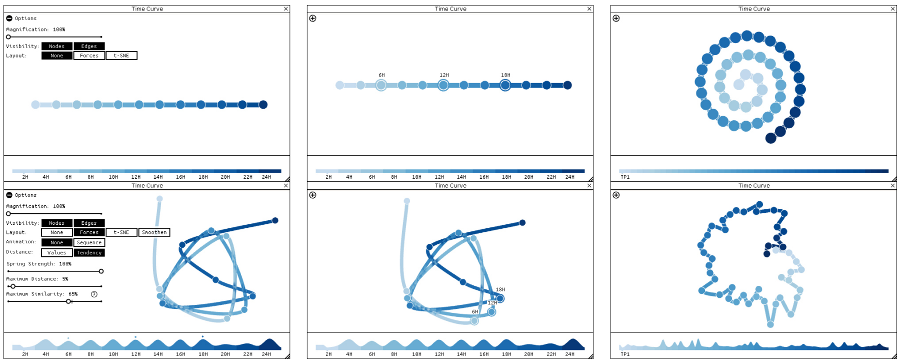
The time curve panel focuses primarily on the representation and analysis of time-series data through a time curve model. This model consists of a timeline visualization that is bent so that its time points are placed relatively to each other according to their similarity. As such, time points that are closer to each other represent moments at which the same data points have similar values which can portray the general behaviors of a dataset over time, such as significant changes in values, regressions and cyclical shifts.

— *Navigation*

Regarding navigation, hovering a node will show the name of its time point and clicking will select it, changing the current time step depicted in the network panels visualizing the same dataset. The mouse can also be dragged to pan the visualization, while the mouse wheel is used to zoom in/out of the current mouse position.

The bottom of the panel contains a timeline slider where all the time points are displayed in sequence. Dragging the slider will highlight every time node in the layout in sequence while hiding every edge except those between the time points that the slider crossed. This results in an animated transition that follows the sequence of time points that the user brushed.

The top left of the panel contains a list of options that offer control over visual elements and the layout of the time nodes. Under "Visibility", nodes or edges can be hidden in order to highlight one or the other. The "Animation" options available will depend on the layout chosen but they create different animated flows between time points. The "Layouts" option will change the position of time points, and are discussed in the following two sections.



Options
Magnification: 100%
Visibility: Nodes Edges
Layout: None Forces t-SNE

Options
Magnification: 100%
Visibility: Nodes Edges
Layout: None Forces t-SNE Smoothen
Animation: None Sequence
Distance: Values Tendency
Spring Strength: 100%
Maximum Distance: 5%
Maximum Similarity: 65%

Figure 3.16: Time curve panels displaying initial timeline layouts (top) and corresponding time curves (bottom). The first timeline is sequential due to a low amount of time points (left), and when transformed into a timeline, time points representing similar states are pulled together (middle). When the number of time nodes would exceed the available space for a timeline, they are placed in a spiral layout (right).

— *Bending Time*

Each time point is initially converted into a node and displayed sequentially as a timeline, either in a horizontal line or as a spiral, with the latter being used when the length of the former surpasses the amount of available space due to a large quantity of time points, as this would have resulted in nodes overlapping (Figure 3.16). Through the options menu, the timeline visualization can then be distorted using one of two layouts: Forces or t-SNE. These layouts are described more extensively in the "Framework" chapter, under "Visualization Models & Layouts".

The force-directed layout utilizes attraction and repulsion forces to pull similar nodes closer while repelling those that are different based on adjustable parameters, which dynamically update the layout. These parameters consist of multiple sliders located in the options menu, under the chosen layout, and they control the strength of the forces, the size of the layout, and the maximum similarity that is mapped to the distance between nodes. In other words, the maximum similarity slider defines the maximum percentage of similarity between two time points that will be mapped to their minimum distance from each other. For instance, at a maximum similarity of 55%, time points placed together will represent states where the dataset is at least 55% as similar, as illustrated in Figure 3.17.

The t-SNE layout also positions nodes based on the chosen distance metric. However, unlike the previous layout, it is not dynamical, being calculated only when the "Update" button is pressed. Additionally, it is also not deterministic, meaning that each recalculation of the layout will yield different results. The quality of these results can be influenced by the parameters, which offer a balance between quality and processing time.

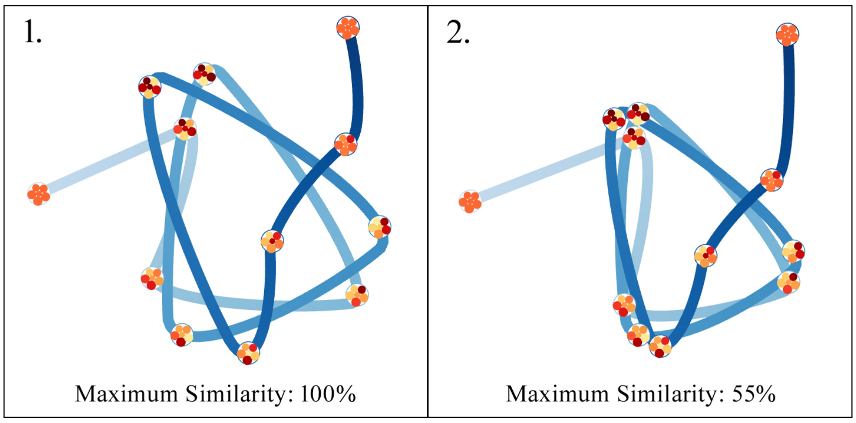


Figure 3.17: Time curve visualizations of the HIV-1 virus gene expression time-series dataset (7589 data points) showing how maximum similarity is mapped to the minimum distance between time points, reflecting their percentage of similarity.

— *Edge Smoothing*

While the time curve layout is able to position time points relatively to their similarity, visual complexity increases with the number of time points, making edge representation particularly important. The Processing library provides several options to manage the proprieties of the edges, including the creation of curved edges across multiple points, but these methods offer limited control over visual attributes. To resolve this, we implemented Time Paths, a layout that creates segmented edges with adjustable proprieties that allows us to smoothen time curves by controlling their trajectory and curvature.

The Time Paths layout must be applied over an existing time curve, as it redraws the visualization to create smoother edges with gradual transitions between time points. As such, the option for this layout becomes available in the options menu after either of the previous layouts have been applied to the visualization. These parameters control the level of detail of a time curve, either smoothing it to remove minor visual perturbations caused by small variations in the data, creating visualizations that only portray the main overall behaviors, or emphasizing the variations by distorting the timeline further, allowing users to create a different types of visualizations.

Furthermore, the increased control over edge proprieties also allowed for the addition of two animations that convey the flow and direction of the time-series more easily: a pulse created from increasing and decreasing the weight of each segment in sequence, and arrow particles that move inside the time curve. These animations react to the intensity of the variations, highlighting stronger shifts in the data through either size or speed.

— *Supporting Timeline Graph*

As the force-directed layout bends the timeline, the temporal order of the time points becomes harder to perceive. While interaction or animation can help users more easily follow the timeline and better perceive its temporal order, these methods may not be ideal in quickly identifying significant moments when dealing with large amounts of time points. In order to help better understand the sequence of events in a time series regardless of the visual complexity of the resulting time curve, a graph was added to the bottom slider which portrays the shifts in the data over time, as shown under each time curve in Figure 3.16.

The waves represent the distance between sequential time points, where a large wave represents a time point that is located at a significant distance from its predecessor. As distance is mapped to similarity, large waves portray moments when significant changes in the data occurred. On the other hand, flat sections without waves represent low changes in the data, which can be interpreted as periods of stability. As such, while the graph cannot convey similarity between non-sequential time points, it is capable of highlighting both moments and periods with significant behaviors in the time-series, allowing users to more easily identify and explore these time points in the other visualizations.

— *Cluster Glyphs*

To help better understand how data changes over time, we created glyphs that represent the dataset at each time point, providing users with a way to discern between different states of the data without having to solely rely on other views. As representing every data point through a simple glyph is unfeasible for significantly large datasets, we instead represent the groups of similar points created by clustering the dataset. As such, applying clustering will convert the time nodes in the time curve into glyphs, which can take the form of either a miniature network or a pie chart (Figure 3.18).

The miniature network glyph is a simplified representation of the visualization in the network panel, converting every cluster into a circle whose size and position is mapped to that of the cluster and colored based on the average values of the cluster at that time point. However, this representation may present a loss of readability at smaller sizes, particularly when there is a large quantity of clusters. As such, when the glyphs are drawn below a size threshold, they are represented with a pie chart, whose visual elements make use of the entire available area.

Each slice of the pie chart glyph represents a cluster, where the width of its arc represents the number of nodes in the cluster and the color corresponds to the average properties of its data points. The pie chart slices are sorted relatively to the positions of the clusters on the network visualization, allowing users to more easily match each slice to its corresponding cluster. The width and order of the slices is also consistent across every glyph to facilitate their comparison.

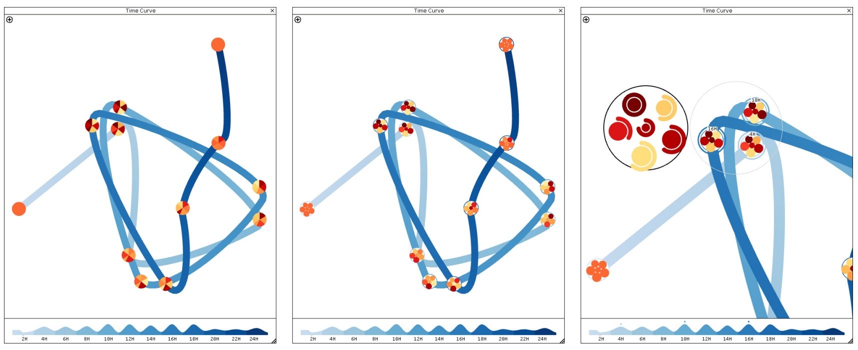


Figure 3.18: Time curve with pie chart glyphs (left) and miniature network glyphs (middle). The data lens is used to select three time points (right), creating an aggregated visualization of their average values that depicts the similarity between them.

— *Similarity Lens*

To further explore the similarities between time points or multivariate data utilizing these glyphs, we utilize the mouse lens previously introduced in the network panel. Right-clicking over a time curve panel or a multivariate view panel will create a circular area around the mouse, allowing users to brush over multiple nodes simultaneously.

This will create a larger glyph of the miniature network next to the lens that represents an aggregate of all the selected glyphs, where the color of each circle now represents the average values of each cluster between the highlighted time points (Figure 3.18). More importantly, arcs are also drawn around every circle, indicating the percentage of similar behaviors exhibited by data points within that cluster at the selected time points. In other words, if around half the nodes within a cluster are exhibiting the same behaviors at the time points hovered by the mouse lens, then the arc around that cluster's respective circle would be drawn as a semi-circle.

The network panel will also adapt its visualization to the nodes selected through the mouse lens, mapping the transparency and saturation of each node in the network to their similarity between the selected time points. As such, this will highlight data points with consistent values, allowing users to more easily identify those responsible for certain temporal patterns. Furthermore, the previously described arcs drawn around the circles in the miniature network glyphs are also drawn around the clusters in the network to represent the percentage of similarity within each cluster.

3.2.4 *Multivariate View*

The multivariate view panel is used for the discovery and analysis of patterns between quantitative variables that do not necessarily have a set order. Each variable is displayed is displayed as a point in two-dimensional space, and their position can be manipulated through the existing layouts in order to reflect their similarity. The layouts in this panel are used to explore the relationships between variables similarly to how the time curve panel explores the relationships between time points. However, as each variable is expected to be independent, the employed methods only focus on correlations between their values.

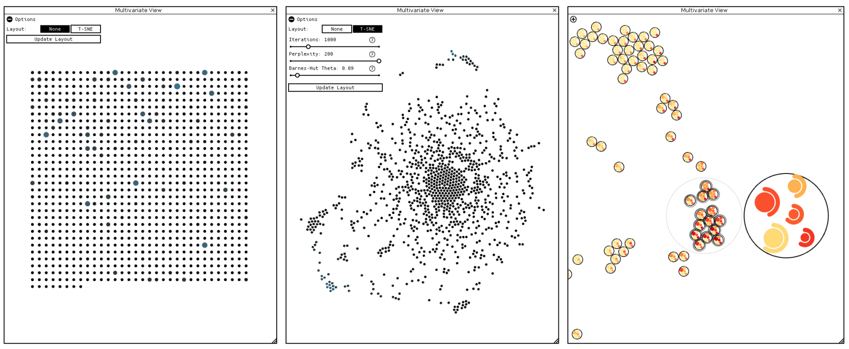
— *General Navigation*

The navigation of the multivariate view panel is very similar to that of the other panels with node layouts. Dragging the mouse will pan the visualization, using the mouse wheel will zoom in/out of the current mouse position and clicking nodes will select their respective variable. This is coordinated with other multivariate views, including network panels, where every network node is mapped to the color and size of the selected variable's value. Likewise, selecting a data point in a network panel or data table will map the proprieties of multivariate nodes to that data point's values.

— *Variable Analysis*

In order to represent variables as being independent while visually distinguishing the multivariate view panel from others, nodes are initially displayed in a grid layout (Figure 3.19). The top left of the panel contains an options menu where this layout can be changed. Through the t-SNE layout, nodes are positioned based on their values, being placed together when they have similar value patterns across different data points. As in other panels, there are multiple sliders that influence the accuracy of the layout and its processing time, allowing these to be adjusted to different types of datasets. The t-SNE parameters are described in the "Framework" chapter, under "Visualization Models & Layouts".

In addition to changing the layout, variable analysis is also aided by the cluster glyphs and mouse lens, whose implementation and functionalities are the same as those described in the previous time curve panel section, except applied to variable values rather than temporal. Likewise, right-clicking will create a mouse lens, allowing users to brush multiple variable nodes to visualize the miniature network glyph representing their aggregate proprieties and similarities (Figure 3.19). These similarities are also mapped to the transparency of nodes in the network panel, highlighting data points and clusters that contain a high percentage of variables with similar values between them. As such, while the analyzed variable may be independent, this panel seeks to highlight any potential patterns that may exist between them.



Iterations: 1000
Perplexity: 200
Barnes-Hut Theta: 0.09

Figure 3.19: Visualization of multivariate data in a grid layout (left) and sorted by the t-SNE layout (middle). Network glyphs and the mouse lens are also available in this visualization panel when the data is clustered (right).

3.3 MANAGING MULTIPLE DATASETS

While multiple files can be loaded and merged to combine different sets of data points and attributes, CroP also allows datasets to be stored, accessed and managed individually. Multiple panels can then be used to visualize different datasets simultaneously and compare them side-by-side, or instead utilize difference views to compare between the variables of datasets with the same data points.

3.3.1 *Dataset Tabs*

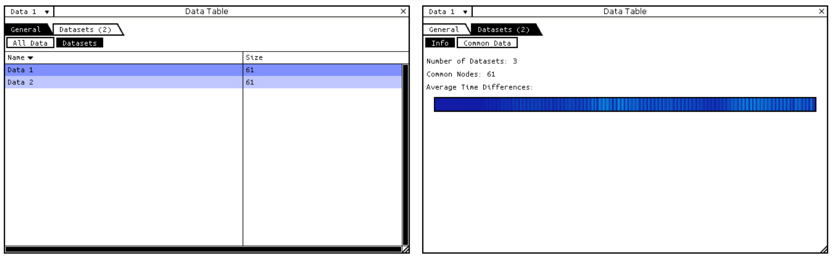
Individual datasets are stored within tabs, accessed and managed under the "Datasets" section in the options sidebar. Any number of dataset tabs can be created and if these exceed the size of the sidebar, they can be scrolled through by clicking and dragging the mouse through them. Data from loaded files will be stored in the dataset whose tab is currently selected, merging with any previously existing data within. Furthermore, options selected in the sidebar will also only affect the currently selected dataset, such as filtering and clustering.

A selected portion of data from a loaded dataset can also be transferred onto a new tab, which can be used to visualize and analyze a section of the dataset considered significant without affecting previous work. This is achieved by selecting the relevant data points and then using the "Copy to New Dataset" button under "Filtering". This will create a new dataset tab with a copy of the selected data.

3.3.2 *Differences View*

As a visualization panel can only represent one dataset at a time, a dropdown on the top left of each panel selects which dataset is currently being visualized, which is added whenever multiple dataset tabs have been created. Additionally, a "Datasets" button will be added to data table panels under the "General" tab, which opens a table containing a list of every dataset loaded (Figure 3.20). Selecting two or more datasets will create a new tab with the name "Datasets".

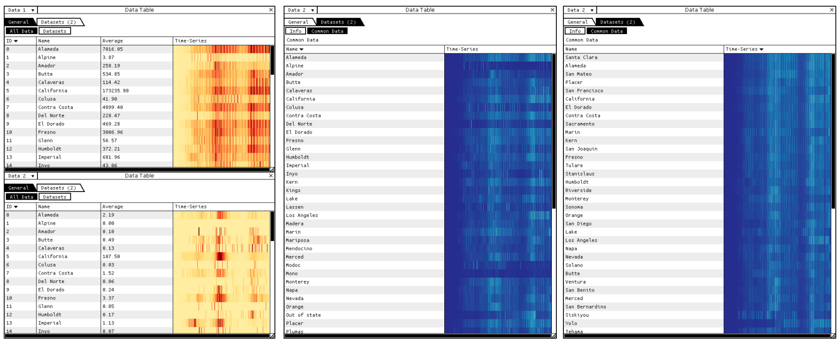
This tab contains two sub-tabs, where the first depicts information about the selected datasets and a color matrix visualization that depicts the average of all the similarities and differences across every common point. The second sub-tab contains a list of every common data point shared between them, where a column depicts a color matrix visualization of the time-series or multivariate data from each data point that represents the difference between the values across all selected datasets (Figure 3.21). The color differ-



Name	Size
Data 1	61
Data 2	61

Number of Datasets: 3
Common Nodes: 61
Average Time Differences:

Figure 3.20: Datasets are selected in the "Datasets" table (left), and their average differences can be seen in the tab that was created (right).



ID	Name	Average	Time-Series
0	Alameda	7016.05	
1	Alpine	3.87	
2	Amador	258.19	
3	Butte	534.85	
4	Calaveras	114.42	
5	California	173235.80	
6	Colusa	41.90	
7	Contra Costa	4099.40	
8	Del Norte	228.47	
9	El Dorado	469.28	
10	Fresno	3006.96	
11	Glenn	56.57	
12	Humboldt	372.21	
13	Imperial	681.96	
14	Inyo	43.06	

ID	Name	Average	Time-Series
0	Alameda	2.19	
1	Alpine	0.00	
2	Amador	0.10	
3	Butte	0.49	
4	Calaveras	0.13	
5	California	107.50	
6	Colusa	0.03	
7	Contra Costa	1.52	
8	Del Norte	0.06	
9	El Dorado	0.24	
10	Fresno	3.37	
11	Glenn	0.05	
12	Humboldt	0.17	
13	Imperial	1.13	
14	Inyo	0.07	

Common Data

Name	Time-Series
Alameda	
Alpine	
Amador	
Butte	
Calaveras	
California	
Colusa	
Contra Costa	
Del Norte	
El Dorado	
Fresno	
Glenn	
Humboldt	
Imperial	
Inyo	
Kern	
Kings	
Lake	
Lassen	
Los Angeles	
Madera	
Marin	
Mariposa	
Mendocino	
Merced	
Modoc	
Mono	
Monterey	
Napa	
Nevada	
Orange	
Out of state	
Placer	
Plumas	

Common Data

Name	Time-Series
Santa Clara	
Alameda	
San Mateo	
Placer	
San Francisco	
California	
El Dorado	
Contra Costa	
Sacramento	
Marin	
Kern	
San Joaquin	
Fresno	
Tulare	
Stanislaus	
Humboldt	
Riverside	
Monterey	
Sonoma	
Orange	
San Diego	
Lake	
Los Angeles	
Napa	
Nevada	
Solano	
Butte	
Ventura	
San Benito	
Merced	
San Bernardino	
Siskiyou	
Yolo	
Tehama	

Figure 3.21: The original data of the two selected datasets is represented in data tables on the left, while their average differences are depicted in the table in the middle. This table can be sorted by its differences (right).

ence is represented by a separate palette that can be changed in the options sidebar. To differentiate between values, we depict similarity by default with the YlGrBu palette, which emphasizes extreme and middle values, where dark blue represents high similarity and light yellow represents high differences. Selecting this column will order all the data points by the sum of their differences, allowing users to either sort by those that are the most similar or the most different.

In network panels, the color of every node will also be mapped to their differences across every dataset, just as in the data table. However, while the data table's dataset tab does not show data points that aren't common between all datasets, the network will represent uncommon nodes that belong to the currently selected dataset with transparency.

3.4 DISCUSSION

In this chapter, we presented an overview of CroP, describing its user interface, visualization models and data analysis functions, contained within a CMV framework. We propose CroP as a visualization tool for multivariate datasets, in particular relational data and time-series, that is able to identify patterns, in addition to providing the tools to identify their sources and impact.

In summary, CroP is comprised of an options sidebar and a workspace for visualization panels, which are automatically organized within a grid layout to help users maintain consistent panel sizes and avoid overlapping. The options available in the sidebar can be used to import and manage datasets, while also providing general functionalities for their analysis, such as data mapping, clustering and filtering. The sidebar also allows users to create visualization panels in the workspace, which will not only represent their respective types of data from loaded datasets, but also provide tools for their specific analysis:

- That *data table panel* employs tables and linear visualizations to simply list every current data point and their current attributes. While data points can be sorted by their attributes to quickly identify and select those with extreme values, the data table panel excels as a support visualization, being coordinated with other panels to showcase the proprieties of individual points or groups that have been brushed.

- The *network panel* is able represent relational data through a dynamic node-link diagram, employing a layout that positions nodes to reflect their direct relationships. Moreover, the available layouts can also sort nodes based on temporal attributes and other quantitative proprieties, allowing them to represent similarity between data points that do not have edges.

- The *time curve panel* represents time-series data through a timeline visualization that bends to reflect the similarity between the state of the dataset at each time point, creating visualizations that portray different types behaviors through abstraction. While abstracting a visualization will often increase its complexity, the panel provides tools to help explore and understand the resulting artefact, such as a supporting timeline graph and the time paths layout, which can smoothen the visualization to reduce visual noise.

- The *multivariate view panel*, similarly to the time curve panel, also positions nodes based on their values, reflecting the similarities between quantitative variables that do not have a set order.

In what regards general analysis tools, groups of data points that were created through clustering will be sorted in the network panel into circular groups, whose relative positions reflect the direct relationships between their nodes. These clusters are then used to create glyphs in the time curve and multivariate view panels, providing a simple representation of the whole dataset at each time step and for each variable, respectively. The mouse lens can then be used on these glyphs to highlight which clusters are behaving similarly or differently, providing users with the means to identify the data points that are responsible for the observed behaviors.

Finally, we also described CroP's ability to receive and store multiple datasets, providing the means to compare them by either juxtaposing their individual visualizations through different panels, or representing their direct similarities through a differences view in the data table panel.

4 ANALYSIS WORKFLOW

Through its CMV framework, CroP's functionalities can be used in conjunction in order to different types of relationships, such as analyzing potential patterns between multiple proprieties across large sets of data points, in particular relational, temporal and certain biological proprieties. As described previously, some of these functionalities are also common between visualization models, although their purpose and outputs may vary according to the type of data. In this section, we will describe how the implemented models and functions can be used cooperatively to explore different datasets and discover meaningful patterns.

4.1 RELATIONAL DATA

Datasets that describe direct relationships between data points can be represented in the network panel. While this panel is also capable of handling sets of points that do not have edges, the network model will visually represent these relationships and provide a layout to analyze their distribution.

4.1.1 *Analysis by Position*

The Yifan Hu layout [71] sorts nodes according to their edges, positioning them closer to related nodes and taking into consideration their degree (the amount of relationships they have in total). In smaller networks this will prevent edge-crossings and make the relationships between nodes more perceptible, while large networks will be sorted to highlight their overall structure, bringing nodes with high degrees closer and dispersing those with few relationships. While the latter may help identify the nodes that are central to complex networks with high amounts of edges, the layout alone may not be enough to resolve the visual complexity that is characteristic of such networks.

To manage some of this visual complexity, the transparency of edges is mapped to amount of edges being drawn, where edges fade out as their number increases until they stop being drawn. While edges between specific nodes may still be viewed either in the data table or when those nodes are highlighted, this approach to edge representation emphasizes the analysis of relationships between elements through their relative position. In this respect, nodes can clustered based on the positions determined by the previous layout, which categorizes closely related nodes into defined groups, further sorting the network visually. Moreover, clusters are positioned relative to each other based on the edges between their nodes, meaning that clusters with related nodes will be pulled closer. The general proprieties of these groups can be analyzed and compared through the data table, which includes temporal attributes, biological proprieties and other variables, allowing users to identify potential correlations with the relationships between those nodes.

4.1.2 *Attribute Correlations*

Grouped nodes positioned by layouts can also be analyzed through the mouse lens. The lens acts as an brush that can be used to select portions of the network, creating a dynamic visualization that shows the aggregated proprieties of the brushed nodes. This will also highlights nodes in the data table, marking the location of their respective rows along the vertical scroll bar. These colored markers allow users to quickly identify the selected nodes on the data table, and can also be used to correlate between the brushed area and the table's order. For instance, if the table is ordered by degree in descending order, a concentration of colored marks along the top of the scroll bar would immediately indicate that the brushed nodes have a high amount of edges (which can be seen in Figure 3.14, showing the selection of a cluster created by the Yifan Hu layout), and the same logic can be applied to any other ordered attribute. In either case, both clustering and area brushing facilitate partitioning nodes according to their direct relationships, allowing users to either focus on a specific subset or filter them using the available tools. Additionally, it is possible to characterize the created clusters based on their most prevalent biological characteristics by analyzing the distribution of GO annotations in the data table panel. This allows for the analysis of the relationships between specific gene expression profiles and other biological proprieties.

4.2 VALUE PROFILES

CroP is able to receive two types of data files with numerical values: time-series files that contain sequential values describing a propriety changing over time, and multivariate data files that contain lists of values for independent attributes.

4.2.1 *Value Normalization*

A key difference between how these datasets are processed is value normalization. Time-series values are normalized between the minimum and maximum temporal values of each data point, so that the variations over time can be compared relatively to other data points even if their values are widely different. However, in the case of multivariate data files, the values are normalized for each variable and not across each data points as these proprieties are expected to be independent and it is more likely for values to present significant differences between variables in the same data point than between data points for the same variable. However, if users intend to analyze the raw values, this process can be reverted by selecting the "Unnormalized Values" switch in the options sidebar. The depiction of each type of these values is exemplified in Figure 4.1.

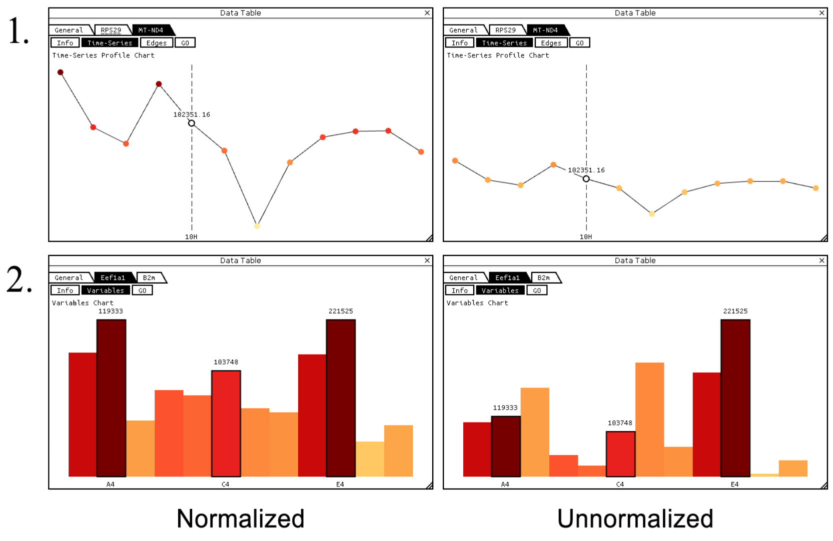


Figure 4.1: Data table profile visualizations of time-series (1.) and multivariate (2.) data, normalized and unnormalized: Normalized time-series are mapped between their data point's maximum and minimum values, while normalized variables are mapped between the extremes of that variable; unnormalized are mapped in relation to the whole dataset.

4.2.2 *Value Representation*

While CroP processes these data files different, the representation and analysis of their values is preformed through shared methods. Processed values are primarily represented through color, being mapped to the chosen palette between the calculated minimum and maximum values. These colors are congruent throughout the different visualization panels, allowing users to more easily identify correlations not just between data points but also across their attributes. Additionally, the color palette can be changed to define either a smaller or wider range of colors that reveal either more general or more precise similarities between data points, respectively. For instance, the spectral palette utilizes a wide range of colors that allow small differences to become more apparent, but this may also visually overwhelming the viewer. On the other hand, sequential palettes can clearly depict the difference between high and low values with a small amount of colors, but middle colors may be hard to distinguish.

The network panel is also capable of revealing relationships between points based on their value profiles, even when edge data does not exist. This is achieved through the t-SNE layout, which positions nodes relatively to each other based on the similarity between either their time-series profile or loaded set of variables. Combined with color representation, it is possible to identify distinct sets of grouped nodes that share specific proprieties, as well as scattered nodes with little correlation between each other. Furthermore, the size of nodes in the network is mapped using unnormalized values, allowing users to also distinguish nodes in relation to the whole dataset. The mouse lens allows for grouped nodes to be easily selected and their common proprieties quickly visualized, either through visualization created by the lens, or through the data table's tabs. Additionally, scattered nodes that were not selected by the lens can be filtered out through the sidebar's options, allowing users to focus their analysis on nodes with stronger correlations.

4.2.3 *Clustering by Value*

Data can also be divided by its values into groups of points with similar profiles through clustering. CroP provides a wide range of clustering options, each with a set of parameters with varying complexity that can be adjusted by the user to refine the similitude within the resulting clusters. For instance, DBSCAN and OPTICS feature sensitive parameters that calculate cohesive clusters and filter outliers, while k-means clustering only requires the user to define the number of clusters. On the other hand, hierarchical clustering only needs to be calculated once before allowing the user to dynamically switch between any number of clusters, although it has a higher computational cost. Depending on the data and the type of relationships being analyzed, users are able to experiment with the various options and save the results onto a file through the options in the sidebar, allowing them to quickly reload the created groups without the clustering execution time. In addition to value clustering, position clustering can also be applied to the network layout calculated by the t-SNE, assisting users in the creation of groups without the need to brush nodes manually.

When clustered, nodes in the network will be grouped into circular areas representing each of the calculated clusters. In order to organize nodes within each cluster and prevent overlapping, these are sorted with a sunflower spiral layout which is ordered by the chosen clustering algorithm, meaning that neighboring nodes may also be more similar. In the network, the common proprieties of these groups of nodes are more discernable, facilitating the analysis of discovered patterns. For instance, when scrolling through the timeline of time-series data, nodes within the same clusters should present increases and decreases of values at the same points in time, which is visually represented through changes in size and color. Individual clusters can be focused by clicking on their borders and analyzing their aggregated attributes on the data table panel. Multiple data tables can be juxtaposed to compare between multiple individual or aggregated profile charts, as well as to quickly navigate through attribute information, as panels focused on attribute tabs will update dynamically with the user's selections. Moreover, the similarity order determined by the cluster algorithm can applied to the data table, sorting its rows by the clustered attribute when the top of its respective column is clicked.

4.3 PATTERN ANALYSIS

Time-series and multivariate data can also be visualized through their respective panels: the time curve panel and the multivariate view panel. While the time curve panel features methods that are exclusive to the analysis of values changing over time, there are visual elements, layouts and functionalities that are common to both panels. Similarly to the network panel, time points and variables are represented as nodes in two-dimensional space, where they can be sorted through the t-SNE layout to visually represent relationships between their proprieties in relation to the whole dataset. For instance, if two variable nodes are placed together we may surmise that a significant portion of points in the dataset present similar values between these two variables. However, relative position alone cannot convey neither the values nor the data points that are common between these variables.

4.3.1 *Aggregating Values*

Brushing nodes on either the time curve panel or the multivariate view panel will update the network panel with the corresponding values, but this is not an efficient method for comparing the values between these proprieties. As such, CroP can represent each node as a glyph, a simplified graphical representations of the dataset at each time point or variable which can be visualized and compared without having to rely on additional views.

However, as scalability must be considered in a simple representation of a large amount of data, glyphs only portray clusters and not individual data points.

By representing the groups of data points calculated through clustering, we can reduce the number of elements that will be represented in the glyph, while ensuring that each one represents a concrete group of points possessing similar proprieties. Furthermore, the nodes filtered as noise by the clustering algorithms DBSCAN and OPTICS will not be visible in the glyph, allowing users to focus on nodes with more significant relationships. This gives users some control over what each glyph represents, including their visual complexity, due to being able to choose the number of clusters in certain algorithms.

The time curve panel and the multivariate view panel share the same type of glyphs: a miniature network and a pie chart. Since the former is directly based on the network visualization, users can more easily identify and compare clusters between panels. However, as this glyph consists of small elements surrounded by white space it may lose readability as its size is decreased, which can be the case for large quantities of nodes that require the user to zoom out significantly. In such cases, the glyph is switched with a pie chart, as it provides a very simple representation for multiple elements with different sizes and values without requiring white space, which increases its visibility.

4.3.2 *Glyph Similarity*

As glyphs provide a snapshot of the dataset for each time point and variable, their general differences and similarities become more apparent when positioned by the available layouts. These patterns can then be further explored through the mouse lens, which creates a larger version of the network glyph next to the lens for any node that is inside of it. When multiple nodes are brushed, this network will act as a differences view, drawing an arc around each of the network glyph's clusters.

The arc length around each cluster directly represents the similarity of the clusters between the selected nodes, meaning that similar clusters will be represented with more complete arcs and appear more prominent than those with low similarity. Furthermore, the color of each node in the network will represent the average value of the respective cluster for all the selected time points or variables, meaning that a high similarity percentage will indicate that value as the most predominant.

4.3.3 *Network Coordination*

While glyphs and the data lens provide the means to identify patterns of values between attributes without relying on additional panels, these functionalities are also coordinated with the network panel to help dig-down into the discovered patterns. When nodes in the time curve and multivariate view panels are selected through the mouse lens, the arcs drawn around the clusters on the lens will also be drawn around their respective clusters on the network panel. Moreover, the transparency and saturation of nodes in the network will be mapped to their similarity between the selected nodes. As such, the more consistent each data point's values are across the selected time points or variables, the less transparent their corresponding network nodes will be, highlighting them over data points with inconsistent values.

To help better identify individual nodes with high similarity in each group, the cluster can be ordered by the similarity of each node, resulting in the most similar nodes being ordered from the center to the outside (Figure 4.2). This allows users to more easily select the most similar nodes by brushing the center of the cluster with the data lens and isolate them if needed. However, this requires the order of each cluster to be recalculated every time that the user changes the nodes brushed by the mouse lens, which may be visually overwhelming if the user is actively using the lens, as every affected node must be re-positioned. Due to this, the user can switch between ordering clusters by similarity, or to maintain their original order in the network panel's options menu.

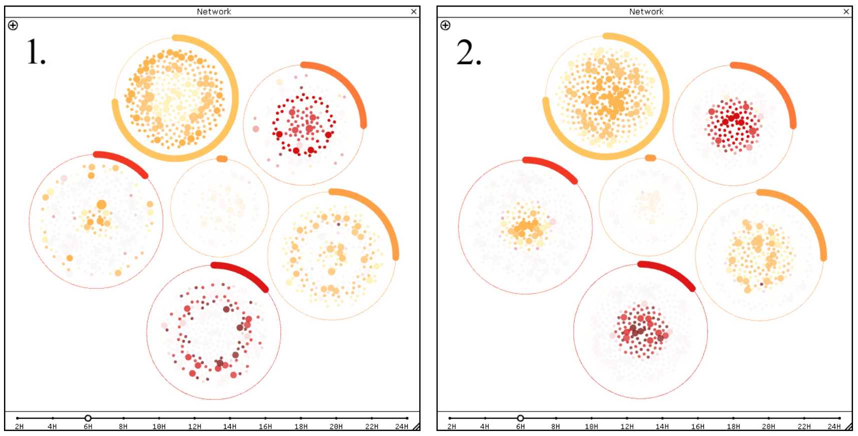

Network
1.
2H 4H 6H 8H 10H 12H 14H 16H 18H 20H 22H 24H
Network
2.
2H 4H 6H 8H 10H 12H 14H 16H 18H 20H 22H 24H

Figure 4.2: Clustered networks where the clusters are ordered by their default order (1.) and ordered by similarity (2.), where the nodes with the highest similarity between the current selection are closer to the center.

4.3.4 *Exploring Time-Series*

In what concerns the analysis of temporal data, it is important to consider how values change over time as these behaviors may be significant to the dataset and help the discover meaningful patterns that can lead to new insights or even help predict future events. When representing time-series data, color can be mapped either to its values, to its variation over time, or to tendencies (as described in the Data Mapping section of the Overview chapter). This allows visualizations to portray different types of temporal behaviors, such as periods during which values are increasing or decreasing with various intensities, as well as the moments when these tendencies shift, known as peaks and valleys. Such events are particularly significant in specific datasets, such as gene expression time-series data where peaks mark the moments when proteins are over-expressed. In regard to the analysis of such behaviors, data points can also be clustered by the variation and tendency in time-series.

Temporal behaviors can also be analyzed through the time curve, a layout that bends the timeline by positioning time points relatively to their similarity in both values and variation, similarly to previous layouts. As the resulting time curve maintains the connections between sequential time points, it is able to represent how the values change over time. For instance, large distances between two sequential nodes indicates a notable shift in values, while clusters of sequential nodes portray periods of stagnation, when the data did not change significantly. Moreover, it can also represent more complex behaviors such as regressions and cycles, portrayed by the curve travelling between two or more distinct sets of non-sequential nodes. In addition to the t-SNE layout, the time curve can also be created through a force-directed layout. Compared to t-SNE, the force-directed layout is capable of more quickly discovering positions that best reflect the relative similarity between time points, but may present issues with sorting large quantities of nodes, in which cases the t-SNE layout would be more effective.

The complexity of the resulting time curve visualization will depend on the characteristics of the data, as the layout is prone to visual noise problems similar to those in network visualizations. To help create more comprehensible visualizations, users may also apply Time Paths, a force-directed layout that smoothens time curves based on user-defined parameters, creating visualizations that can either be more or less sensitive to variations in the data. While increasing the level of smoothing also increases data abstraction, this can be used to reduce visual clutter and also promote the representation of predominant behaviors. Time Paths also allows for additional control over the visual proprieties of edges, creating smoother transitions between time-points which can be used to more clearly represent the flow of time. Through these, users may also employ additional animated transitions that better convey the direction and intensity of changes in the data. Furthermore, in parallel to any of the available layouts, the timeline graph portrays the variation between each time point, allowing users to more quickly identify moments of significant changes in the data and periods of stability, regardless of the complexity of the time curve visualization.

As noted before, relative positions can only represent the general behaviors of data points between themselves. As such, two time points placed close to each other may represent that the majority of the data points are repeating their behaviors at these two moments, but this not necessarily mean that there is only one predominant behavior. To further explore such behaviors, the mouse lens and network coordination can help highlight the types and distribution of values throughout the data at each group of time points. This is showcased in Figure 4.3 which shows a dataset that appears to alternate between two states of values, evidenced by the two distinct groups of time points. When each of these two groups are selected by the mouse lens in a. & b., it highlights the clusters that have the highest percentages of consistent behaviors throughout these time points. In both instances, it is possible to observe that the data points alternating between the two states are mostly contained within the two left-most clusters, making up between one-third and two-thirds of each cluster, as indicated by the arc surrounding each cluster.

a.

b.

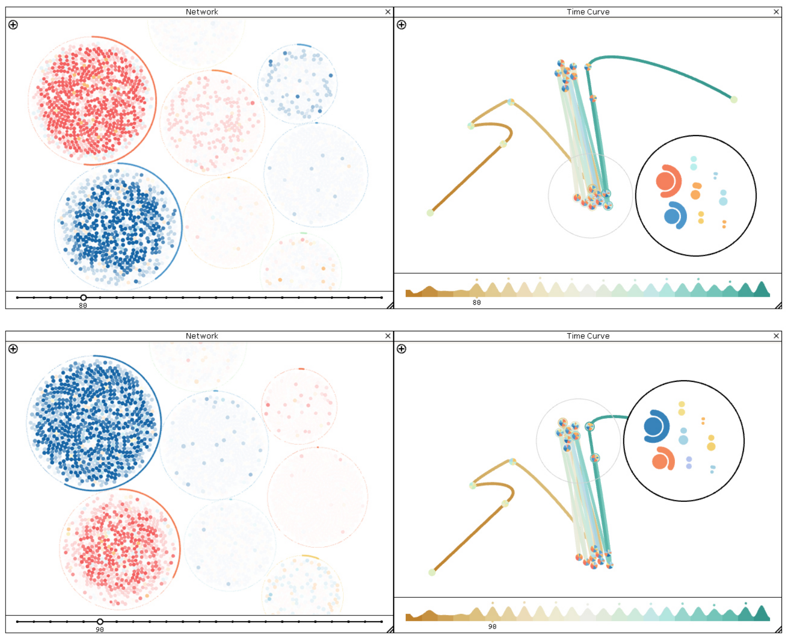


Figure 4.3: Network (left) and time curve (right) visualizations of the Saccharomyces cerevisiae dataset, where the data alternates between two states throughout most of its time-series (a. & b.). The time points corresponding to these two states are selected by the mouse lens, highlighting the network's data points that present similar behaviors in each state.

4.4 DISCUSSION

In this chapter, we described how CroP's visualization models, layouts and functionalities that were introduced in the previous chapter can be used cooperatively between multiple panels to explore different types of datasets, discover patterns and analyze their characteristics, sources and impact.

In what pertains to the analysis of relational data, the network panel serves as the main visualization model, managing edge representation by mapping their volume to transparency in order to prevent visual noise, while focusing on representing relationships through position. In this respect, the Yifan Hu layout can position nodes to prevent edge crossings while highlighting those that are central to the overall structure of the network. Nodes that have been grouped by layouts or clustering can then be brushed with the mouse lens to visualize common proprieties, an analysis that is supported by the highlighted elements and linear visualizations in the data table panels, identifying correlations between their direct relationships and their other attributes, including biological proprieties. Moreover, brushed groups of data points can be quickly filtered out or isolated.

Regarding the visualization of multivariate data, time-series data is mainly represented through the time curve panel and individual quantitative variables are portrayed in the multivariate view panel. Users are provided with options on how values are mapped, and may also choose a color palette that best suits the range of values being explored. In general, the initial analysis of value profiles can be achieved with similar tools to those described previously: the t-SNE layout in the network panel can position nodes according to the similarity of their attributes, while the layouts of the time curve and multivariate view panels can position the attributes themselves to reveal potential relationships between them, with the analysis of either being supported by the data table panel.

Clustering also provides several options to partition the dataset into defined groups of data points with similar attributes, providing insight into the diversity of existing value profiles. Moreover, the overall proprieties of each cluster will be used to create glyphs in the time curve and multivariate view panels, providing a simple representation of the dataset at each time step or for every variable. While this serves to more easily compare between the similarity of theses states through juxtaposition when applying one of the aforementioned layouts, it may also be used in pattern analysis. Brushing over glyph nodes with the mouse lens will create a side visualization that increases the visibility of the glyph, while multiple brushed nodes will be combined to show their aggregated proprieties and the percentage of similarity between the nodes in each cluster. Additionally, while the data lens portrays the similarity between clusters, the network panel will map each node to their individual similarity, allowing users to identify and select specific nodes in that panel, which is further supported by the ability to sort clusters by the current similarity of each node. As such, the lens can be used to brush groups of time points or variables and identify the primary type of value profiles that they have in common, thus revealing significant points of data at the source of behaviors being analyzed.

Due to the ordered nature of time-series, such data can also be analyzed by how it varies over time, which CroP explores visually and functionally. As described previously, the time curve is capable of portraying behaviors by showing the flow between time points as their position is changed to reflect their similarity. This flow is based off not just values but also variation, separating increasing values from those decreasing, representing cycles through loops, stagnation through clusters and strong variations through wide arcs. The analysis of these artefacts is further supported by a timeline graph, animations and Time Paths, a layout which graphically improves the initial time curve and provides options to either smoothen the visualization in favor of representing general behaviors, or highlight minor variations through exaggeration. In addition to variation, the concept of tendency was also applied to analyze time-series by its peak values and positive or negative trends, as to explore patterns in datasets where shifts in variation represent significant events. Such attributes can be mapped to color, allowing them to be analyzed through the aforementioned tools, with support from the network and data table panels.

While it may be unfeasible to devise a set of solutions for every kind of problem, it is through the combination of the presented visualisation models and functionalities that CroP is able to provide a wide range options to tackle the representation and exploration of different types of datasets, while facilitating the analysis of specific types of biological datasets.

5 FRAMEWORK

The development of a multiple coordinated views framework for CroP involved the implementation of diverse methods and visualization models that could function cooperatively in the exploration and analysis of varied datasets. Through interaction, it was possible to create an environment that could be adapted to different types of problems, while also provide various functionalities to be used in the search of the most apt solutions. However, as the number of options increases so does the tool's complexity, not only in its development but also for its users. As such, it is necessary to ensure that both the developed visual and interactive elements can be easily understood and utilized.

In this chapter, we will review CroP's interface, visualization models, layouts, and various algorithms in further detail, with particular regards to user interaction. In this sense, we will discuss the implementation and refinement of methods in consideration of the principles of fluid interaction [42], which seek to anticipate common usability issues that interrupt the user experience, such as uncertainty and lack of feedback.

5.1 USER INTERFACE

In what concerns the user interface, it is necessary to consider the basic structure and organization of all of CroP's functionalities in regards to the user experience and across different contexts. As described previously, the main interface consists of a grid layout where visualization panels can be placed and organized, next to sidebar of options involving different aspects of dataset management.

The grid layout promotes a flexible environment, allowing for multiple panels to be used simultaneously with different configurations. While its number of rows and columns is immutable, the grid and its panels will adapt to any changes to size of CroP's window. Moreover, the various options in the sidebar are sorted into different collapsible sections, which allows users to focus on the methods relevant to their specific problem, while also facilitating the addition of future functions. Any changes made to the current workspace, including the placement of panels and parameters in the sidebar, can be can be saved or restored at any point.

In this section we will review these methods further, focusing on the primary interface elements and internal processes that are consistent throughout the analysis of diverse datasets, such as dataset management, visualization panel arrangement and visual settings.

5.1.1 *Data Processing*

CroP supports user-provided dataset files containing network data, time-series data or multivariate data, formatted as either comma-separated values or tab-separated values. Network data files consist of a list of pairs of names, from which data points and edges will be extracted. Time-series data files contain ordered list of values that correspond to existing or new data points, as well as names that can be associated to each time point, such as a date or hour. Multivariate data files consist of lists of numerical values corresponding to independent variables, so the structure of the file is identical to the time-series data file, except each column corresponds to a different variable.

When loaded, datasets are subjected to multiple validation processes which start by checking if the file contains the minimum number of expected values, and then each line is checked for missing values or invalid number formats. If the file does not contain enough values or if every line has been found to contain errors, then the user will be notified and the file will not be loaded. However, if only certain lines are found to contain errors, then the corresponding line numbers and error types will be logged and displayed to the user, allowing them to choose to exclude those lines and load the remainder of the dataset.

After a dataset has been validated, CroP checks for data that has already been loaded, in which case the new dataset must either replace or merge with the existing data. For instance, if network data has been loaded previously, upon loading a time-series or multivariate dataset CroP will verify if the existing data point names match with those in the new file. If they do, then data can be merged so that the new data points will contain both network and time-series or multivariate data. However, if there are data points that do not match, the user will be given the option to immediately filter these points out of the dataset and keep only those that match both datasets. Furthermore, datasets of the same type can also be merged, such as loading two network datasets to create a single, larger network, where matching data points will inherit edges from both datasets. The same would occur with multivariate data, as data points can inherit the variables from multiple files. However, in the case of two temporal datasets containing time-series with different lengths, data cannot be merged and the user will be prompted that the new dataset will completely replace the other.

5.1.2 *Integrated Biological Database*

In addition to an integrated protein-protein network dataset, CroP also integrates GO databases [26] which allow the tool to identify and annotate matching biological elements with additional data. The GO project provides a set of structured, controlled vocabularies for community use in annotating genes, gene products and sequences [26]. It is a collaboration between databases to create shared vocabularies that facilitate uniform queries across all of them. These databases allow the application to identify, classify and sort protein datasets by utilizing their respective GO terms. These classifications can then be accessed by the user in the network and data table panels, where they used to characterize compare between groups of biological elements.

The integration process consisted of building the network that maps the GO terms (which act as identifiers) to their respective biological processes. This is a complex directed network with an inexplicit hierarchy, as there is a root node from which it is possible to define different levels. Each protein is associated with several GO terms, ranging from a few to hundreds, and some of these terms may also have multiple designations. Part of the classification process consisted of grouping GO terms that referred to the same biological processes and creating the relationships between the protein data and the GO databases. Furthermore, while CroP currently only supports the GO databases, its framework has been prepared to integrate additional databases that have the same purpose of classifying data points.

5.1.3 *Mapping Color Palettes*

Each color palette is defined in the program by an array that contains each of its colors in sequence, which are then used to map values to their respective colors, which are defined in the HSV (Hue, Saturation, Value) format. Any value can then be mapped into a color by using these arrays, including the color transitions between values when scrolling through the network panel's timeline, which allows us to guarantee that these transitions are smooth and only use colors from the chosen palette. Furthermore, the length of these arrays is variable, meaning that the range of colors available to any palette can be increased with the addition of more intermediate colors, or instead simplified by removing them. As such, this also means that the existing palettes can be easily altered and new additions only require the establishment of a new color array.

However, the color used in the representation of inactive nodes must also be manually chosen for each palette. As stated previously, such nodes should be represented with a color that contrasts with the currently chosen palette, meaning that the hue value on these contrasting colors may be very distinct from the remaining. Due to this, we cannot create a smooth transition by simply mapping each of the HSV values between them as this would result in a turbulent transition that cycles through every hue between the two colors. To avoid this, we defined a method that transitions between such colors: first we map the initial saturation value to zero, at which point we switch the initial hue value with the target hue value, before mapping the saturation from zero to its target value. In other words, the color transition goes from its original color to grey and then to its target color. While this may not be ideal for every situation, this can be easily applied to any two distinct colors, while also avoiding transitions through colors that would clash with the current palette.

5.1.4 *Panel Management*

Each visualization model is contained within a panel that has a variable size and position. Through these, the user can place, interact with, and compare between various visualization models simultaneously, including multiples of the same model representing different datasets. A panel is characterized by a surrounding border, ridges on its bottom-right, and a top bar with a title and buttons. Dragging the ridges with the left mouse button resizes the panel, while dragging the top bar moves the panel. When changes are made to the panel's size and position, the panel is adjusted to a defined grid, which can be seen in the background when it is not covered. This means that each corner of the panel will snap to the closet point on the grid.

The grid layout keeps the work environment organized, as the user can place panels next to each other and easily resize them in relation to other panels and the available space. To further ease panel management, several functions were developed to prevent overlapping and automatically re-position or resize panels based on the user's actions. For example, if the user places a panel over another, the panel that was overlapped will either be resized or moved to a new location, depending on the area of overlapping panels and the amount of available space in the work environment. The management of overlapping panels is described in Figure 5.1. If multiple datasets exist, the dropdown located on the left of the top bar of every panel can be used to change the dataset that is being represented.

5.2 VISUALIZATION MODELS & LAYOUTS

CroP integrates a variety of visualization models and layouts with the objective of allowing users to visualize, explore and analyze different types of data. For instance, the data table panel provides simple visualizations of datasets at a low level, while integrating methods focused on sorting and

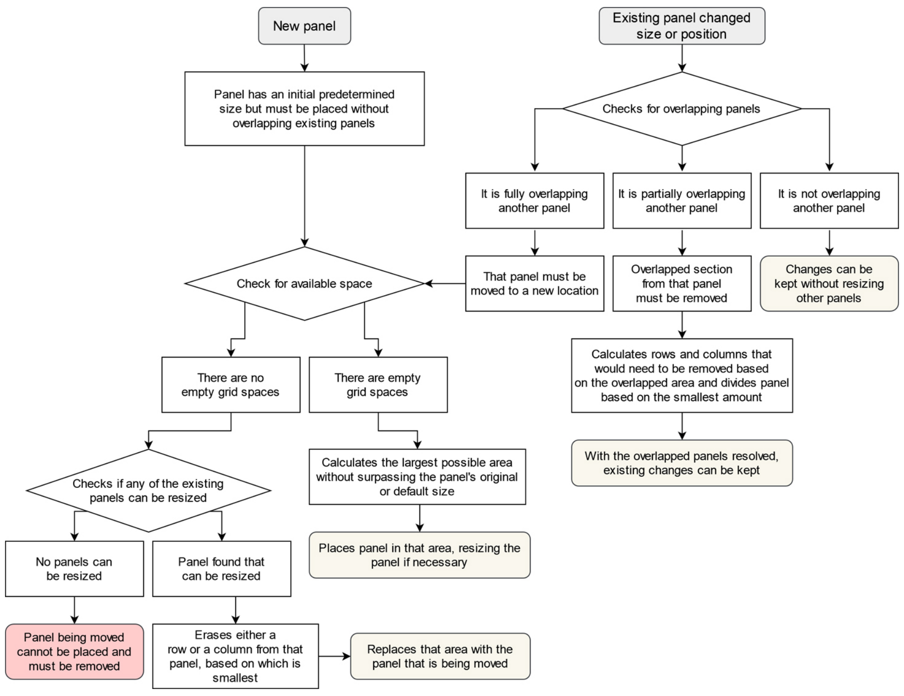

New panel
Panel has an initial predetermined size but must be placed without overlapping existing panels
Check for available space
There are no empty grid spaces
There are empty grid spaces
Checks if any of the existing panels can be resized
No panels can be resized
Panel found that can be resized
Panel being moved cannot be placed and must be removed
Erases either a row or a column from that panel, based on which is smallest
Replaces that area with the panel that is being moved
Calculates the largest possible area without surpassing the panel's original or default size
Places panel in that area, resizing the panel if necessary
Existing panel changed size or position
Checks for overlapping panels
It is fully overlapping another panel
It is partially overlapping another panel
It is not overlapping another panel
That panel must be moved to a new location
Overlapped section from that panel must be removed
Changes can be kept without resizing other panels
Calculates rows and columns that would need to be removed based on the overlapped area and divides panel based on the smallest amount
With the overlapped panels resolved, existing changes can be kept

Figure 5.1: Flowchart describing the management of overlapping panels.

brushing data that may be relevant to each dataset. On the other hand, the network and time curve panels make use of different types of layouts to position data relatively to their relationships, providing functions to highlight otherwise hidden patterns. In this section, we will describe the implementation of these models and the range of parameters of their layouts, as well as discuss the addition of functionalities necessary to facilitate exploration, discovery and analysis of diverse datasets.

5.2.1 *Data Table*

The data table panel may be considered the least complex visualization panel as it describes data at its lowest levels through lists and linear visualization models. However, various precautions had to be taken to ensure that data can be easily explored and analyzed by users through interaction and while coordinated with other panels. For instance, only data belonging to currently visible rows is loaded and updated, ensuring that performance is not affected even when the dataset contains over a hundred thousand data points. In consideration of large datasets, discovering and accessing specific points on the table was made easier by marking selected or highlighted data points on the scroll bar, which can be focused on instantly by clicking on that mark.

In what regards brushing, the actions associated with selecting rows were based on common actions performed on files in current operating systems, making such interactions more intuitive: holding the CTRL key will allow users to select multiple individual rows, while holding the SHIFT can be used to select every row between the current and previously selected row. The latter will group every selected data point into its own group rather than creating individual tabs for each one, providing users with aggregated information on all the brushed rows. These actions are also coordinated with other panels which creates tabs for selected data points and updates currently displayed graphs: if the data table is displaying the time-series chart of a selected point, then changing the selection to another data point will update that chart respectively, rather than closing the tab because the previous point was deselected.

Finally, as data table panels must be able to be resized, table columns were also made to be flexible and adapt proportionally to the size of the panel. However, each column was also given a minimum width based on their contents, to ensure a minimum level of readability in any situation. If the total width of all the columns surpasses the width of the data table panel, then vertical scrolling can be used to navigate between the columns.

5.2.2 *Initial Layouts*

When a visualization with nodes is initialized, these nodes must be attributed starting positions. In the case of networks, nodes could be initially positioned based on their relationships or values, but calculating such a layout would require parameters that best fit the type of dataset in addition to significant execution times for large quantities of nodes. A common method for initializing nodes with a very fast execution time is to simply randomize their positions, scattering them throughout the available space. However, while the initial position of nodes may not be particularly important, they can still be organized with a simple structure that does not need to take any of their attributes into consideration. In this regard, we chose to initialize networks using a Fibonacci sequence layout, also known as a sunflower spiral, which sequentially positions nodes around a center location, creating a pattern that resembles that of sunflower seeds. This is a space-filling layout that assures that nodes are structured, centered, without overlaps, and without the implicit sequential order of normal spiral layouts.

In the case of timelines, time points have an intrinsic sequential order that must be considered in their representation. As such, time nodes in the time curve panel are initialized with equal spacing in a straight horizontal line, ordered by time. However, large amounts of time points would result in either excessively long lines or short lines with overlapping time points. To prevent this, whenever the number of time points would require a timeline that surpasses the length of the window, the layout is changed into a standard spiral. Unlike the previous sunflower spiral, this layout displays the time nodes with their temporal order clearly represented, from the middle outwards, allowing more points to be displayed simultaneously without overlapping.

Finally, the multivariate view panel needs to display nodes that represent numerical attributes which may or may not have an implicit order, depending on the dataset's contents and organization. In this panel, nodes are initialized using a square grid layout, positioned in sequential rows with lengths that are based off the total number of nodes. Similarly to the spiral layouts, this is space-filling and keeps nodes organized for datasets in which their order is pertinent to their analysis. Furthermore, this layout is distinctive, which helps the multivariate view panel differentiate itself from the others.

5.2.3 *Force-Directed Layout*

While the position of each node can be calculated by different layouts to reflect that node's attributes and relationships, the visualization may need to react and adapt to options chosen by the user without having to recalculate a new layout. As such, we chose to implement force-directed layout which utilizes attraction and repulsion forces between nodes to dynamically adapt the visualization to any changes. While repulsion forces generally serve to separate two nodes by a set distance, attraction forces can be used to both bring nodes together and pull them towards a set position. We can use these forces to promote self-organizing behaviors in nodes, making them push others away to prevent overlapping, as well as direct nodes individually to their new positions when the layout changes, all through fluid movements that convey the changes that are happening more clearly to the users.

— *Overlap Prevention*

Regardless of how the position of nodes is originally mapped, overlapping may still occur through user interaction. For instance, the radius of nodes in the network panel is mapped to their current temporal values, meaning that their size shifts as users switch between time points, which could result in nodes expanding and overlapping their neighbors. By applying a constant repulsion force to every node based on their current size, we can dynamically prevent overlapping in any node visualization regardless of the changes made to the layout.

However, calculating the distance between every node at every frame can be significantly computationally intensive, particularly when dealing with thousands nodes. In order to ease this process, a grid is built based on the number of nodes in the visualization and it is kept updated as to the positions of every node in its squares. Through this, each node only needs to check their distance to nodes located in neighbour squares. As updating the grid at every frame is significantly less computationally intensive than calculating distances, this process has allowed CroP to handle overlap prevention in datasets containing several thousand nodes with minimal slowdowns. While the repulsion grid does present limitations on particularly large networks (>20.000 nodes), the importance of overlap prevention is diminished in such large datasets where identifying individual nodes within a network is less important than discerning different groups, their common proprieties and potential patterns between them. In such cases, we opted to halt repulsion forces in favor of performance, considering that low-level exploration can still be conducted through selection, filtering and the data table panels.

— *Network Clusters*

When a dataset is clustered, the force-directed layout plays a significant role in the creation of clusters within the network panel. First, circular cluster areas are created based on the groups of data points determined by the clustering algorithm: their size is proportional to the amount of points in their respective group, and their initial position is centered on the average network position of every node in their group. Using the average position assures that the cluster will be created near the majority of its nodes, which is particularly important when clustering the data by position. After the circular areas are created, the force-directed layout is applied on them as if they were nodes, utilizing repulsion forces to ensure that the areas do not overlap and maintain at least a set distance from each other. Additionally, if there exists relational data then attraction forces will be applied between the cluster areas based on the edges that exist across nodes in different clusters, pulling clusters together which have higher numbers of nodes related between them. The final positions of the cluster areas are determined through several hundred iteration of these forces, which are performed immediately after the clustering algorithm is applied and not frame-by-frame.

In previous iterations of CroP, once the cluster areas have been created and positioned, we would use attraction forces to pull nodes towards the center of their respective clusters, allowing them to populate the cluster area and self-organize through repulsion forces. However, this method resulted in nodes being messily pulled into each area and requiring a significant amount of time to spread evenly throughout it, particularly for large clusters. To avoid this, we pre-determine the position of each node within their cluster through the Fibonacci sequence layout (or sunflower spiral, as described in the Initial Layouts section). By utilizing attraction forces to pull each node towards their individual positions, we can ensure that they are immediately evenly distributed throughout each cluster, which circumvents the need for nodes to self-organize for an extended period, but also creates clusters with clean layouts that reflect the general node order calculated by the clustering algorithms.

As such, whenever clustering is applied every node will travel to their pre-determined positions in their respective clusters. This presents users with the opportunity to witness how data points are distributed between clusters, especially when experimenting with different types of clustering algorithms. For instance, changing the number of clusters in hierarchical clustering will result in clusters getting dynamically split or merged, which is visually represented as nodes break apart for their clusters or rejoin existing ones. In the case of fine-tuning parameters in some of the algorithms, it is possible to identify even minor changes, as individual nodes travel between clusters.

— *Time Curve Forces*

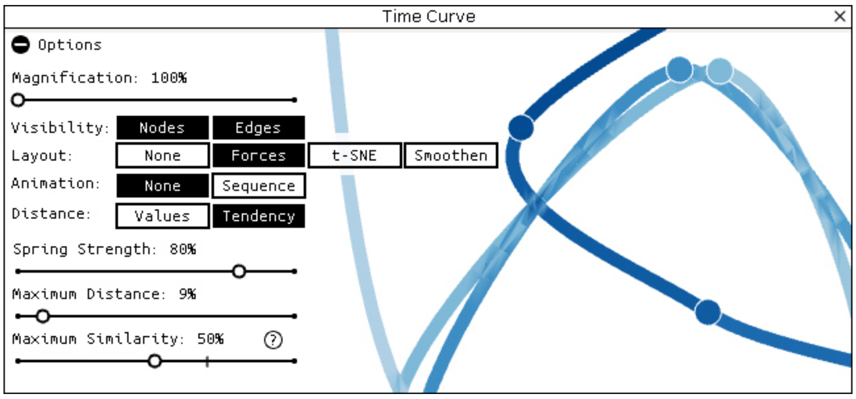
A time curve visualization is a result of the initial timeline being distorted by a different layout that re-positions the time nodes based on the similarities between their proprieties. This is achieved through the force-directed layout, where springs are placed between every node which maintain the distances between them proportional to their similarities. The formula used to calculate the force applied to each spring is based on Hooke's law [131] and each spring's ideal stretching length is determined using a similarity matrix. The similarity between any two time points t_i and t_j is determined through the function $f(i,j)$, described below, where N is the number of nodes in a dataset, and P is a node containing the time series t. It is performed by calculating both the difference of the values of the time points $v(i,j)$ and the difference of their variation $b(i,j)$, averaged between every point in the dataset. By using both of these operations in calculating similarity, time points are positioned based on their current values and in relation to whether these values are increasing or decreasing similarly over time.

$$v(i,j) = \sum_{P=0}^{N} |P_{t_i} - P_{t_j}|$$

$$b(i,j) = \sum_{P=0}^{N} |(P_{t_i} - P_{t_{i-1}}) - (P_{t_j} - P_{t_{j-1}})|$$

$$f(i,j) = \begin{cases} (i,j), & \text{if i=0} \\ \frac{(i,j)+b(i,j)}{2}, & \text{else} \end{cases}$$

Values are averaged across every node in the dataset when calculating similarity, meaning that the distance between time points will reflect the percentage of nodes that are behaving similarly between them. As such, closer time points reflect higher amounts of nodes manifesting similar behaviors at those moments in time. However, if only a smaller percentage of the dataset is manifesting a certain behavior pattern, time points could be scattered without any perceivable correlation. For such cases, we added the maximum similarity slider that controls how similarity is mapped to the distance between time points. For instance, if time points are very close to each other when the slider is set to 55% maximum similarity, then at least 55% of all of the data in the dataset should be behaving similarly at those time points (as illustrated previously in Figure 3.17). By manipulating this threshold, it is possible to identify patterns occurring between smaller percentages of large datasets. Additionally, to more quickly identify such behaviors in any dataset, the highest value of maximum similarity between any time point in the current dataset is marked on the slider (Figure 5.2).


Time Curve

Options

Magnification: 100%

Visibility: [✓ Nodes] [✓ Edges]

Layout: None [✓ Forces] t-SNE Smoothen

Animation: [✓ None] Sequence

Distance: Values [✓ Tendency]

Spring Strength: 80%

Maximum Distance: 9%

Maximum Similarity: 50%

Figure 5.2: Options menu for the Forces layout in the time curve panel. While the Maximum Similarity slider is set to 50%, a mark at 70% indicates that there is no pair of time points whose similarity is superior to that value.

In short, the time curve's force-directed layout is defined by three parameters that can be controlled through sliders in the options menu:

- Spring Strength – Controls the speed at which nodes move into positions that reflect their similarity, but high values may also cause instability.

- Maximum Distance – Determines the maximum distance between the most dissimilar nodes; increasing this value will expand the size of the visualization.

- Maximum Similarity – Maps the percentage of the dataset that must be similar between two time points in order for them to be at maximum proximity.

5.2.4 *Yifan Hu Layout*

The Yifan Hu layout was integrated into CroP to provide the means to spatially organize nodes based on their edges when exploring relational data, meaning that this layout is exclusive to the network panel. It re-positions nodes over continuous iterations in order to find placements that adequately reflect their relationships, visually sorting networks to highlight nodes that are central to the data while also reducing visual noise by preventing edge-crossings [71]. The layout was implemented using the Gephi library [13], and it was chosen based on its balance of graph quality and performance speed on large graphs when compared to other algorithms [73].

While there are various parameters that control the output of the Yifan Hu layout, we opted to simplify the number options available to the users, allowing them to manipulate the layout enough to generate a variety of networks that represent relationships differently while not overwhelming them with choices. The three chosen parameters can be changed in the network panel's options menu through various sliders, providing a way to balance the calculation time of the layout against its accuracy in depicting relationships through position:

- Step Ratio – The ratio used to update the step size across each iteration of the calculation of the layout, in which higher ratios indicate a faster convergence but lower accuracy.
- Quad Tree Maximum – The maximum value to be used in the quad tree representation, where greater values increase accuracy at the cost of execution time.
- Barnes Hut Theta – Theta of the Barnes Hut optimization, where lower values increase accuracy at the cost of execution time.

5.2.5 *t-SNE Layout*

The t-SNE (t-Distributed Stochastic Neighbor Embedding) technique is a non-linear dimensionality reduction algorithm used in the visualization of high-dimensional datasets, attributing a position on a two-dimensional map for every data point based on their proprieties and the implicit structure of the dataset. In the context of CroP, t-SNE is used in both the network and time curve panels to sort nodes based on either their time-series or other multivariate values, depending on the type of dataset. For instance, when handling time-series data, a network layout calculated using t-SNE will spatially organize nodes based the similarity of their temporal profiles (for each pair of nodes it compares the values of each time point), while a time curve layout will position time nodes based on the similarity of every data point at each temporal step (for each pair of time nodes it compares the value of the respective time points for every data point).

While t-SNE does not sort data points into specific groups, it is capable of creating visual clusters of points by placing those that are most similar closer to each other. However, defined clusters can then be created by clustering the data by position, diving groups of nodes into clusters based on their spatial density. The definition of the original visual groups is dependant on the parameters of the t-SNE algorithm, which can be adjusted through sliders in the options of the layout:

- Iterations – Number of iterations performed, where higher values increase accuracy at the cost of execution time.
- Perplexity – Approximate guess on the number of neighbors for each point, where higher values help preserve global structure, but may obscure local structure and increase execution time.
- Barnes Hut Theta – Lower values increase accuracy at the cost of execution time.

5.2.6 *Time Paths*

Time Paths is a layout that smoothens an existing time curve visualization, redrawing it through a brush controlled by parameter-based attraction forces, creating segments that allow us to create better transitions between colors, opacity and line weight. The brush consists of a moving point which is first placed at the initial time node on the original time curve and it is then pulled towards the following time node using a spring, calculated using Hooke's law [131] and a fixed attraction strength. The brush's route is mapped by intermediate points that are left behind as it moves, and after placing a set number of points it is pulled towards the next time node. However, the new spring does not immediately replace the previous one, as we apply momentum: a percentage value that defines how quickly the attraction force from the previous time node is converted into the attraction force to the next node.

Once the brush reaches the final time node, every edge of the new curve is defined by the sets of intermediate points that were left in its path. This provides increased control over the visual representation throughout each edge as we can define gradual transitions of visual proprieties between any time node, as well as animate visual elements along the drawn trajectory. Through time paths, we added two animations that convey the flow of time: a pulse created from increasing and decreasing the weight of each segment in sequence and arrow particles that move across the time curve. Both of these animations convey the intensity of variation between sequential time points, where the size of pulses and speed of arrows both increase proportionally to the difference of similarity between two time points.

We defined two variables that can be controlled through sliders which update the layout dynamically:

- Intermediate Points – Defines the number of points that make up the edges drawn between time points, controlling the visual definition of each curve; extreme values will cause distortions.
- Smoothness – Controls the speed of forces converging between time points, previously described as momentum, where decreasing it will create sharp turns between points, and increasing it will create wider loops.

The calculation of a Time Path only needs to be performed once for each set of parameters, as all of the intermediate points are saved along with their properties. During this, overlapping intermediate points are removed, cleaning the visualization and improving drawing speed. Furthermore, it should be addressed that time paths naturally distort the position of the time nodes from the original time curve, in which their position best reflected their similarity. To diminish this distortion, after the time path has been calculated, we move time nodes along the new path to a point that is closest to their original position on the time curve.

5.2.7 *Timeline Graph*

The graph on the timeline slider of the time curve panel represents how data shifts over time, allowing users to identify different behaviors, such as moments with significant shifts in values and periods of stagnation where variation is low. This is achieved by mapping the height of each bar in the graph to the distance between that time point and the previous one, meaning that the size of the bar represents the intensity of the changes in the data sequentially throughout time. As such, periods of stagnation can easily be identified by portions of the graph with very low bars. However, if several intense shifts happened sequentially then these would be depicted by multiple concurrent large bars, which could incorrectly be perceived by the viewer as a sequence of time points with similar high values (Figure 5.3.a). While these bars do depict large values of variation in this case, we want to convey that not only are these time points dissimilar, but also that large shifts in the data occurred at those moments. For this reason, we depict the bars as spikes, matching the intensity of the shifts in values to their sharpness.

To achieve the spike shape, the height of the point between two sequential time points is calculated based on their height. The formula to calculate the middle point h located between time points $t1$ and $t2$ (exemplified in Figure 5.3.b) is described below where T is an array of the average values of every time point, $max(T)$ and $min(T)$ are the maximum and minimum values in T, respectively, and $avg(t1, t2)$ is the average value of $t1$ and $t2$. This formula describes how h is calculated by mapping $avg(t1, t2)$ from the set $[min(T), max(T)]$ into the set $[avg(t1, t2), min(T)]$, meaning that the height of both time points increases (which would indicate two sequential large shifts of data) the middle point's height decreases.

$$h = (1 - \frac{avg(t1, t2) - min(T)}{max(T) - min(T)}) \times (avg(t1, t2) - min(T)) + min(t) \quad (5.1)$$

Additionally, we smoothen the spike shapes to make their points wider and easier to visualize in compact visualizations, while also creating a graph that better conveys the flow of time. Through the Processing library, this could be achieved by turning the graph into a shape formed by vertex curves, smoothing each of the graph's edges. However, the shape would have to be a solid color, meaning it would not be possible to maintain the temporal color gradient throughout the graph. To resolve this, we instead created a white shape formed by vertex curves (created by inverting the spikes calculated previously) and applied it on top of the bar graph, using it as a mask that creates the intended smoothing effect (Figure 5.3.c,d).

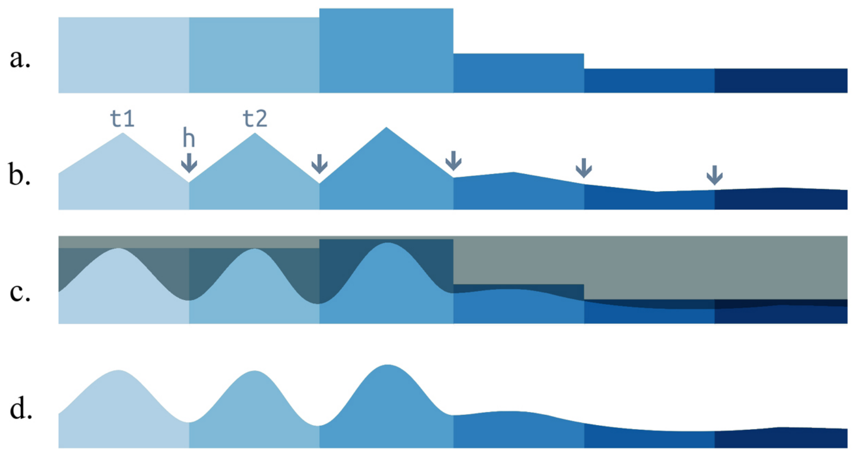


Figure 5.3: Illustration of a segment from timeline graph that represents high variation for the first three time steps and then a period of minimal changes; First it is drawn as bar chart (a), then the middle points are calculated to highlight high variation (b); These points are used to create a shape that masks the initial bar chart (c), smoothing the spike shapes (d).

While the spike graph is capable of representing hundreds of time points without compromising its visual representation, as the number of time points increases, the size of each bar decreases. To improve the scalability of the graph, we established a minimum pixel width for each bar so that the spikes will not be drawn whenever the amount of time points or the graph width would result in the spikes being too small to be perceived properly. While the previously described visual effect of the spike representations is lost in such situations, by drawing only bars we can better highlight significant moments in what would be a complex graph, while also reducing the necessary processing power for its representation, as drawing spikes is more computationally complex.

5.2.8 *Mouse Lens & Glyphs*

The mouse lens is a circular brush for selecting large areas of nodes and analyzing their combined proprieties through small data visualizations. While this lens can also be used to quickly view information from individual nodes, such as line graph of their temporal profile or a table of GO proprieties, it is best used to visualize the predominant proprieties of groups of nodes that have been positioned by the various layouts that seek to identify and reveal hidden patterns in the data. As to help prevent the small data visualization from being drawn outside of the panel, if the mouse lens is currently on the left half of the panel then its respective visualization will be drawn to the right of the lens, or vice-versa.

In the time curve panel, the distribution of values throughout the data points at each time step is represented through glyphs, provided that the dataset has been clustered. This glyph portrays the average values of each cluster at each time point, allowing users to visualize the states of the dataset over time without needing use selections or other views, utilizing either a miniature network or a pie chart. The miniature network glyph utilizes the force-directed layout to position its cluster nodes so it works even without clusters having been originally positioned in the network panel, allowing the time curve panel to be used independently if necessary. Furthermore, this also allows the glyph to be created without clusters that have been classified as "noise" by the DBSCAN or OPTICS clustering algorithms, allowing the remaining clusters to be more visible and easier to analyze. Each cluster node also has a minimum size so that they remain visible even when they contain very few points relatively to the other clusters. However, the miniature network becomes more difficult to be read when the glyph is too zoomed out, mainly due to the white space between cluster nodes, in which case the pie chart glyphs are used. The ordered and space-filling slices allow for glyphs to be more visible and easy to compare at smaller sizes.

The Time Curve's lens will create a larger version of the glyph so it can be more easily visualized, while also highlighting the similarities and differences between several time points, allowing users to identify and analyze the key elements in an observable temporal pattern. When multiple time nodes are brushed over, the large glyph will display a miniature network using the average color of the respective cluster nodes across all the selected time nodes. Moreover, each cluster node will now be drawn with a surrounding arc whose diameter is mapped to their similarity, a percentage that is calculated using the sum of differences of each data point between all the selected time points. As such, if every data point in a cluster was behaving the exact same way across three brushed time points, then that cluster would be drawn with an arc that fully surrounds its node. Likewise, if only half the data points were exhibiting similar behaviors, the cluster node would be surrounded by a semicircle arc.

To ensure that all the cluster nodes fit within the glyph we calculate the circular area that they would need to occupy by comparing the two most distant points between every pair of cluster nodes: the sum of their radius and the distance between their centers. The largest measurement obtained is the diameter necessary for a circle to contain all the cluster nodes, which is used to calculate the difference with the size of the glyph in order to discover the ratio necessary to resize all the clusters and their vectors to fit within the glyph. Moreover, we also organize the clusters inside each glyph by moving the average centroid of each one to the center of the glyph, to further prevent them from being drawn outside the glyph's area.

5.3 CLUSTERING ALGORITHMS

Clustering is the unsupervised classification of patterns into groups, known as clusters, consisting of sets of points that present more similarities to other points within the same cluster than to those in others. However, the ideal number of clusters depends on the attributes used to determine the similarity between each node and on the problem being solved. Furthermore, different clustering algorithms require different parameters, approach classification differently and have different processing speeds, meaning that there isn't a single clustering algorithm that could be considered ideal for every problem [83].

Clustering may be an important step in the discovery of new knowledge through the identification of meaningful data patterns. In order to provide a range of distinct approaches for discovering patterns in different types of datasets, five clustering algorithms have been integrated into CroP: Hierarchical, K-means, Bisecting K-means, DBSCAN and OPTICS. These clustering options can be accessed in the main sidebar, along with sliders that can be used to change their respective parameters. Data points may be clustered based on any of the attributes available, which depend on the data that has been loaded. If a dataset is represented in a network panel, points can be clustered spatially, while if time-series data has been loaded, data points can be clustered by their values, variation, or tendency. Additionally, if the time-series contains null values, we consider these as moments or periods of "inactivity" and the dataset can be respectively clustered in order to potentially discover temporal patterns between points missing values.

5.3.1 *Hierarchical Clustering*

Hierarchical agglomerative clustering algorithms define every data point as its own cluster and then applying a bottom-up strategy that successively groups the closest clusters until only a single cluster remains, which creates a hierarchical tree that represents the nested grouping of patterns and similarity levels at which groupings change [74]. While this algorithm may be more computationally intensive than others, it is also more versatile, as the resulting hierarchical tree can be used to sort the dataset between any number of clusters without any additional calculations.

The implemented algorithm is the generic algorithm based on Michael Anderberg's approach [5] described by Daniel Müllner [109]. Before the algorithm is initiated, every node is compared to each other based on the chosen attribute in order to build a similarity matrix. From this point onwards, every data point is treated as a cluster object and any two objects can be merged in order to create a new cluster that contains the points from both. The algorithm then iteratively locates the lowest value in the matrix which corresponds to the currently two closest clusters, combining these into a single cluster.

In each iteration, the distance matrix is updated with the distance between the new cluster and every remaining cluster, calculated through the Lance-Williams formula [90]. This is a flexible formula that contains variable parameters that define which distance metric will be used. The most appropriate metric may depend on the data and the user's intentions as it affects how the nodes are distributed across the clusters, determining their shape, size and how discernible they are from each other. Through the Lance-Williams formula, we implemented the seven most common distance metrics, known as: single-linkage, complete-linkage, average-linkage, weighted-average linkage, centroid-linkage, median-linkage and Ward's linkage [46]. Additionally, in order to speed up the searches in each iteration, the algorithm also maintains a list of the candidates for nearest neighbors of the clusters.

Throughout the algorithm's run, we record how clusters are merged as an hierarchical tree, which at the end will allow clusters to be combined and split in the order that was previously calculated, although there is an upper limit to the number of clusters which was established in order to conserve memory and prevent the dilution of patterns. As stated previously, this hierarchical tree allows users to switch between any number of clusters without running the algorithm additional times, dynamically updating all the visualization models accordingly. This allows users to quickly adjust the number of clusters to better suit the dataset being visualized, creating either large quantities of small clusters with potentially very similar points, or lower amounts of clusters that are easier to sort and compare but also contain more diluted patterns.

5.3.2 *K-Means & Bisecting K-Means*

K-means is considered one of the simplest unsupervised machine learning algorithms, identifying a user-defined number of clusters by creating the same number of random centroids and iteratively identifying the closest data points which are used to adjust their positions [98]. Once the position of the centroids stabilizes, the closest data points to each one are assigned to a cluster. One of the main drawbacks of this algorithm is that the number of clusters must be pre-assigned, which is cannot naturally be determined in most problems. However, its execution time is very fast, especially when compared to hierarchical clustering, making them suitable for discovering patterns even in time-series datasets [3]. Furthermore, while the algorithm measures the Euclidean distance between centroids and data points as its default, other metrics are also available, namely the Correlation, Cosine and Manhattan distance metrics.

In addition to k-means, CroP also integrates the Bisecting K-means algorithm, a modification of the original that begins by considering the entire dataset as one cluster and then iteratively performs bisections, splitting one of the current clusters into two through k-means until the desired number of clusters is obtained [143]. As such, this algorithm also requires the number of clusters to be pre-defined, but, unlike k-means, it is able to recognize clusters of any shape and size. Furthermore, each bisection step can be iterated multiple times while keeping the best discovered result. Increasing the number of iterations will naturally result in longer processing times, but the quality and consistency of the results are also superior when compared to its alternative. While K-means may return different results with each execution due to the stochastic nature of the initialization of its centroids, running bisecting k-means with a high number of iterations can yield similarly composed clusters between executions, as it seeks ideal placements.

Both of these algorithms were implemented into CroP through the SPMF open-source data mining library [48].

5.3.3 *DBSCAN & OPTICS*

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a clustering algorithm that finds clusters of daypoints based on the density, grouping points that have many neighbors while marking points from areas with low-density as noise [45]. As such, clusters are determined by the neighbourhoods of each point, which are defined by two parameters: the radius of the neighbourhood of a point (epsilon), and the minimum number of points that need to exist within that neighbourhood in order to consider that point as part of a cluster. Points that lack enough neighbors are considered noise, which CroP handles by placing them into their own cluster. This cluster is classified with the name "Noise" and represented with a grey background in the network panel, as to distinguish it from other clusters. This cluster is also excluded from the time curve’s glyphs, allowing the remaining clusters to be more visible and easier to analyze, especially when there is a large quantity of points classified as noise.

On one hand, DBSCAN requires more knowledge of its parameters and more exploration in order to find well-defined clusters when compared to the other integrated algorithms, which otherwise could result in the classification of hidden patterns as noise. On the other hand, being able to filter out points that do not adequately fit existing patterns allows for the creation of clusters with increased similarity between its points, which can be aided by CroP's ability to filter groups of points from the dataset in order to focus on those with higher relevance to the problem. Furthermore, DBSCAN is also proficient at discovering clusters of various shapes.

CroP also integrates OPTICS (Ordering Points to Identify the Clustering Structure), a density-based algorithm that follows the principles of DBSCAN but does not explicitly cluster data, instead producing an augmented ordering of the database that represents its clustering structure based on its density, from which clusters can then be extracted [9]. Similarly to DBSCAN, OPTICS utilizes two parameters to define the radius of the neighbourhood of each point (epsilon) and the minimum points required within that neighbourhood to consider it as a core point. However, unlike DBSCAN, sorts data points based on their neighbors, creating an hierarchical order for each cluster. Furthermore, after the clusters are formed, an additional distance that represents their density is taken into consideration (epsilon prime), which is used to further filter noise. While its additional parameter makes it more complex than DBSCAN, it surpasses it by being able to detect clusters of varying density, making it better suited for larger datasets.

Both of these algorithms were implemented into CroP through the SPMF open-source data mining library [48].

5.4 DISCUSSION

In this chapter, we reviewed CroP's user interface, visualization models and data analysis methods, describing the technical aspects of their implementation, choice of parameters, and the considerations that were taken in their addition, such as error prevention measures and methods to improve performance.

Regarding the user interface, interactive aspects were designed with consideration to the principles of fluid interaction, as to aim towards a smooth experience for the average user that is uninterrupted neither uncertainty in their actions or a lack of feedback from the tool itself. While it may be unfeasible to prevent all human error, imported data is parsed and any detected errors are listed and presented to the user in alert messages along with the respective line numbers in the original file. These alerts also serve to provide users with options when handling multiple imported files, allowing them to merge or filter them based on matching or missing attributes. Moreover, data is automatically matched with integrated databases to append additional proprieties, namely the GO database. The same principles are also applied to panel management, as to facilitate the addition, removal, and organization of the workspace. With particular regard to the latter, CroP handles panel arrangements dynamically, resizing and re-positioning overlapped accordingly, taking into consideration existing free space and other panels.

The choice of visualization models and their respective parameters also aim to provide users with additional options on visualizing and analyzing different types of datasets. While the data table represents the lowest levels of data through lists and linear visualizations, it is able to consistently support other panels through coordination. By loading only necessary rows, these lists are able to contain large datasets with minimal processing cost, providing tools to order and brush between large sections, creating aggregate data visualizations.

The remaining visualization panels rely primarily on node networks sorted by layouts that portray the relationships between data points. While the initial positions are determined by simple space-filling layouts that aim to display all nodes without overlaps, users are then provided with other layout options specific to each panel with adjustable parameters. For instance, the Yifan Hu and t-SNE layouts use parameters that balance the quality of their results and the processing time required for their calculation, allowing users to choose between quality and speed. While the Yifan Hu layout is specific to sorting networks based on edge data, the t-SNE layout can be used to both sort data points based on their attributes (network panel) or sort attributes based on the whole dataset (time curve and multivariate view panels). Alternatively, there is also a dynamic force-directed layout that is used to bend timelines through adjustable parameters in the time curve panel. In comparison to the t-SNE layout, the force-directed layout generally has more difficulties in dealing with larger datasets, as the convergence of large quantities of nodes may be very slow. However, it is comparatively more effective on smaller datasets, displaying consistent results with more accuracy. The same forces are also used to create clusters and position nodes within them, placing nodes in a sunflower layout where they can be ordered by their attributes.

In addition to the visualization models, clustering plays an important role in the analysis of both simple and complex datasets. With consideration to the analysis of a wide range of datasets, CroP provides multiple clustering algorithms with different inputs, outputs and processing speeds. Hierarchical clustering may result in long processing speeds but allows the user to select between any number of clusters after being calculated only once. K-means has a fast execution time, but requires the user to specify the number of desired clusters beforehand. These proprieties are shared by the bisecting k-means algorithm, which is also able to recognize clusters of any shape and size, iterating over each bisection step multiple times to improve its results. DBSCAN and OPTICS are also effective at discovering clusters of different shapes and sizes, but require very specific search parameters, as points that do not meet the established requirements are sectioned off as noise. In particular, OPTICS orders points hierarchically and utilizes an additional parameter to further filter noise.

As such, hierarchical clustering may be best suited for smaller datasets, allowing users to quickly explore the diversity of existing value profiles based on how clusters form and divide using just default parameters. Meanwhile, large datasets may be more quickly clustered through k-means and even further refined through bisecting k-means if the user is willing to search for an optimal number of clusters. On the other hand, DBSCAN and OPTICS are able to quickly determine very cohesive clusters in large and complex datasets with the correct parameters, but these are also much more sensitive than those of other algorithms and require more effort to approach desirable results.

Through this, we show how choice of visualization models and data analysis methods contributed towards the flexibility of the tool, taking advantage of its multiple coordinated views framework to provide the user with an environment that they can adapt to a variety of problems, while improving usability through fluid interaction, which includes dynamic methods and visual feedback.

6 EXPERIMENTATION

In this section, we present the visualizations created by our models and discuss their performance in representing behaviors over time in diverse datasets, in addition to the role they play in the discovery of significant data points or temporal events. The first experiments are produced from simple datasets comprised of single time-series that can be easily matched with their results, and then from biological datasets that contain thousands of nodes with individual time-series, in addition to a multivariate dataset. Throughout these experiments, we employ the developed methods to explore each dataset, identify patterns, and analyze their composition, sources and impact. Regarding representation, we will primarily utilize the "YlOrRd" color palette for values and the "Blues" color palette for time, unless specified otherwise.

6.1 INITIAL TESTS

While the representation of a single time-series may not offer much insight that is not already observable in a linear representation of the data, the following datasets allowed us to test the representation abilities of our layouts, as any of the general data behaviors that could be represented can be easily compared with a line chart of the time-series.

6.1.1 *Sine Waves*

Firstly, in order to demonstrate our implementation of the time curve layout, we conducted tests using a sine wave dataset with 500 time points, which depicts a simple consistent behavior: a cycle, where values increase and decrease repeatedly between a minimum and a maximum. Additionally, we altered the dataset to depict other predictable behaviors. The resulting visualizations are shown in Figure 6.1. The visualization of the basic sine wave dataset resulted in an oval consisting of overlapping loops, whose shape represents the shifts in variation over time, while the number of loops matches the number of cycles in the dataset (Figure 6.1.a). Regarding the transformations applied to the dataset, altering the minimum and maximum values affected the shape's height, increasing or decreasing the oval to match the amplitude of each cycle (Figure 6.1.b). The following dataset was changed to present an overall consistent increase in its values, which resulted in a matching gradual position shift for the ovals representing each cycle (Figure 6.1.c). Finally, increasing frequency resulted in stronger value variations. These were represented with increases to the width of the cycle's loops, making the larger variations also more visually noticeable (Figure 6.1.d). In these initial experiments, our implementation of the time curves layout was able to position time points relatively to each other based on their values and variations, creating visualizations that convey the overall behaviors of each time-series.

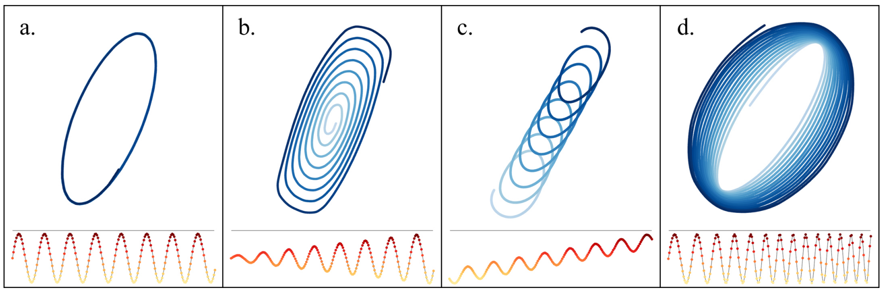


Figure 6.1: Time curve visualization of a sine wave dataset (a), followed by time curves of the same dataset with different gradual transformations: an increase of its amplitude (b), an overall increase of values (c), and an increase of its frequency (d). A linear representation of each dataset is displayed below their respective visualizations.

6.1.2 *Individual Time-Series*

To further test CroP's visualization models, we utilized time-series gathered from real events which describe various behaviors and trends. These datasets were chosen to test our implementation of the Time Curve's model, while also demonstrate how the Time Paths layout can utilize different parameters to smoothen curves, reduce visual clutter and highlight general trends.

The first dataset is "Wolfer's Sunspot Numbers", a yearly measurement of sunspots, small dark areas caused by concentrations of the magnetic field flux on the sun's surface, from 1770 to 1869 [23]. This time-series contains 100 time points and presents cycles with similar minimums but varying peak values, resulting from periodic value increases of different intensities. The second dataset depicts "Monthly Milk Production", measuring pounds per cow from January 1962 to December 1975 [107]. The dataset contains 168 time points and is characterized by a yearly production cycle with minor jumps in variation and a consistent increasing trend until the final five years, where the production increase stabilizes.

In both cases, the Time Curve depicted each cycle with circular patterns whose sizes and positions matched the respective variations and trends observable in the dataset, which is consistent with previous results (Figure 6.1).

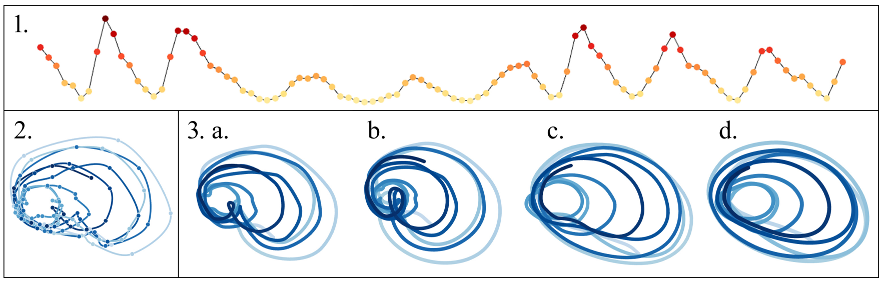


Figure 6.2: Visualizations of the "Wolfer's Sunspot Numbers" time-series, depicted as a line chart (1.), a Time Curve (2.), and then transformed through Time Paths with different parameters (3.) where the level of smoothing is increased (a to d).

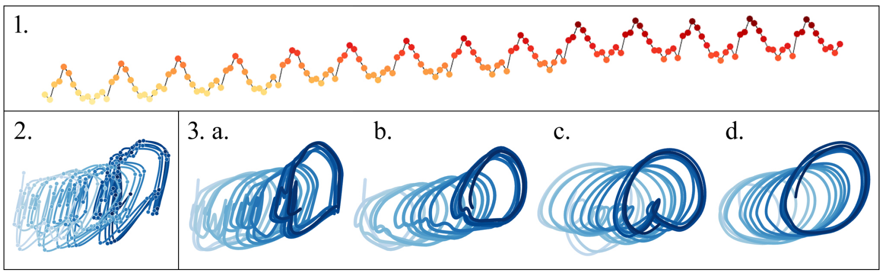


Figure 6.3: Visualizations of the "Monthly Milk Production" time-series, depicted as a line chart (1.), a Time Curve (2.), and then transformed through Time Paths with different parameters (3.) where the level of smoothing is increased (a to d).

For instance, the "Monthly Milk Production" dataset is represented with consistently shifted loops, up until the final cycles, when they begin to overlap as production no longer increases yearly. The Time Path layout was then applied with four different sets of parameters, increasing the level of smoothing with each one. The results are depicted in Figures 6.2 & 6.3.

The first of each set of results (3.a) represents the default parameters, which generally smoothed the initial time curve while remaining sensitive to smaller variations in the values. We can note a small loop in the center that highlights a slight increase in values within a decreasing trend. In the second result set (3.b), the number of intermediate points is reduced while momentum is increased, which reduces the level of details on the visualization. This results in the previous loops being reduced into small perturbations that still indicate moments with significant data shifts, but less prominently. The third set (3.c) was a result of a higher increase in momentum, removing almost all visual clutter in favor of representing the overall cycles. In the last parameter set (3.d), momentum was increased again to create exaggerated depictions of the previous cycles, particularly large data variations such as the two largest increases in the sunspots dataset (Figure 6.2.3.d).

In general, the time curves model was generally able to represent how time-series vary over time, while time paths helped reduce visual noise and highlight the main tendencies of the time-series. Additionally, while these cycles and variations in the data may be observed through linear representations of these time-series, time path visualizations offer various alternative representations of otherwise simple datasets.

6.1.3 *Autumn Weather*

In addition to previous experiments, we also visualized datasets with containing simultaneous time-series. Due to their low-dimensionality, the created visualization should not reveal significant insights that are not already observable in the linear time-series graphs, but this allows us to easily relate the results to original datasets. The first of these datasets describes the changes in temperature, humidity and pressure across two days (between the 4th and 6th of November in 2013). As these were measured every 4 minutes, the dataset contains 722 time points.

Figure 6.4 depicts each time-series through graphs from the data table panel (a., b. & c.), as well as a timeline visualization (d.) marking shifts in the data. In the time curve visualization (1.), we highlighted five time points (T1 through T5) which stand out in the time-series graphs as moments of significant changes, in addition to being notable in the time curve visualization as points of transition between groups of time points. For instance, T1 marks the highest value of humidity in the dataset, preceded by a large increase of temperature and decrease in pressure. T2 then occurs after both a steep increase and decrease of humidity, followed by a small spike in temperature.

After T3, the time curve appears to close a loop, as all three sets of variables return to very similar values as those at the start of each time-series.

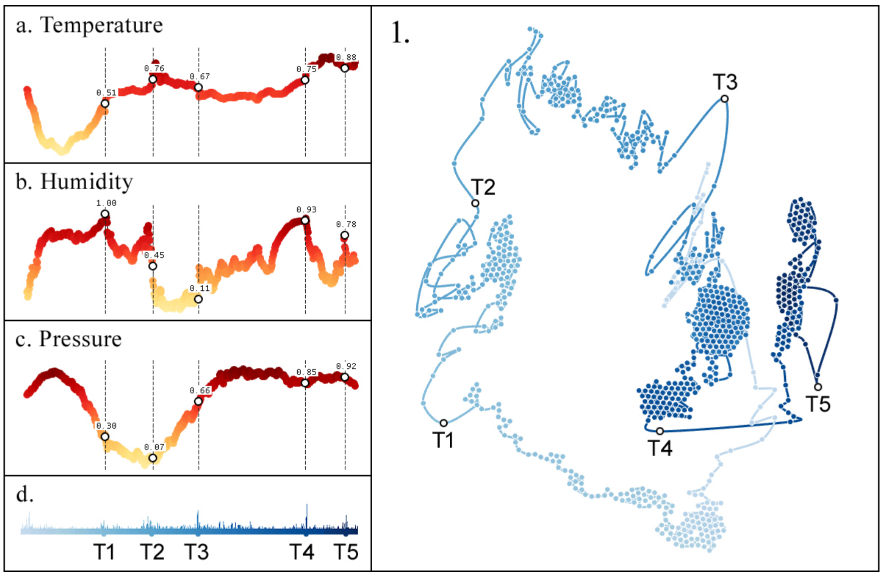


Figure 6.4: Line graphs depicting the time-series for temperature (a.), humidity (b.) and pressure (c.), lined with the corresponding timeline graph (d.) and the resulting time curve visualization (1.) for this dataset.

Following this, there is a large cluster of time points between T3 and T4, marking a period of stagnation. This can be explained by the low variation in values across both temperature and pressure, while humidity shows small but repetitive increases and decreases until there is a slow and continuous increase in values. One it reaches T4, humidity drops quickly alongside a overall increase in temperature, causing the time curve to shift. The data seems to continue to stabilize, with the exception of a notable peak in humidity, marked by T5. In general, we were able to observe that the time curve visualization reflects the behaviors of all three variables, while also highlighting some significant moments.

6.1.4 *U.S. Economy*

Another dataset that allowed us to explore visualizations created from multiple simultaneous time-series is a United States economic time-series that lists personal consumption expenditures (in billions of dollars), the personal savings rate and the number of unemployed (in thousands) from July of 1967 to April of 2015. When observing each of these time-series through the data table panels, we can note that expenditures have had a consistent increasing trend, personal savings had an overall decreasing trend with some outliers, and unemployment numbers are characterized by slow changes with various peaks.

Figure 6.5 shows these time-series along with the resulting time curve visualization, highlighting six time points that signal moments of significant changes in the data (T1 through T6). Some of these appear to be a result

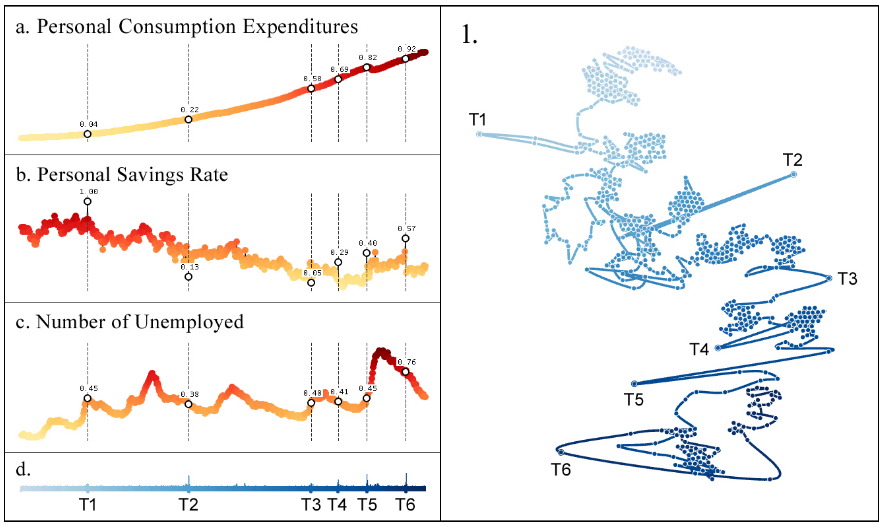


Figure 6.5: Time-series visualizations of personal consumption expenditures (a.), personal savings rate (b.) and number of unemployed (c.) in the U.S. between 1967 and 2015, along with their timeline (d.) and time curve (1.) visualizations. Time points marking significant changes in the data are highlighted (T1 through T5).

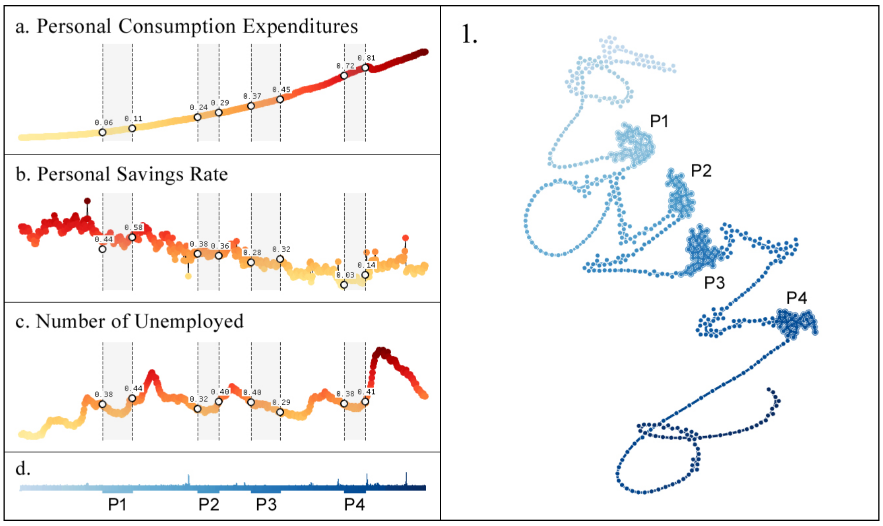


Figure 6.6: Time-series visualizations U.S. economic time-series (a., b. & c.), and respective timeline (d.) and time curve (1.) visualizations. Time curve is smoothed by the time paths layout, and four clusters of data points are highlighted, indicating periods of low variation in the data.

from outliers in personal savings, although some can be observed to match events on other time-series. For instance, T1 marks the highest point in the rate of personal savings as well as one of the initial peaks of unemployment numbers. More notably, T5 marks the moment where consumption expenditures broke its consistent raising trend, followed by a significant increase of unemployment. In the time curve, this period of time is represented with wider distances between time points, indicating stronger shifts in values and setting it apart from the rest of the visualization. This period matches the deep recession of 2007 and 2008, caused by the collapse of the housing bubble.

Additionally, the time curve was also smoothed by time paths in order to decrease some of its visual noise in favor of portraying the general behaviors observed across its time-series, as shown in Figure 6.6. The resulting visualization presents an overall direction which reflects the main tendencies observed in expenditures and savings, with several loops and clusters that match the peaks of unemployment number and their stabilization, respectively. In this regard, four clusters of time points have been highlighted, which match periods of low changes both in unemployment numbers and rate of savings.

6.2 BIOLOGICAL DATASETS

Biological datasets can be generally characterized as complex, containing large quantities of data points with multiple variables. Unlike in the previous datasets, tendencies and significant moments are not easily identified without data analysis methods. Proteins that exhibit similar behaviors may contribute to the condition being tested, such as infections or cancer. Analyzing these behaviors is necessary to build knowledge on the basic molecular mechanisms in cells, used in the development and testing of new treatments. Throughout these experiments, we will analyze various datasets through the visualizations created by CroP in order to discover unique behaviours, while utilizing the available tools to dig-down and identify the data points at the source of these patterns.

6.2.1 *HIV-1 Virus*

The first gene expression time-series dataset that we visualized shows human proteins reacting to the HIV-1 infection. This dataset was obtained from Mohammadi et al. [108], which measured gene expression every 2 hours for 24 hours after transfection with HIV-1 in Sup-T1 cell line. Expression was profiled using SAGE-Seq and normalization was done using DESeq [6]. The network dataset is comprised of a human PPI network with 7589 proteins, after being filtered by CroP to exclude proteins that do not contain time-series data. The dataset was initially clustered by tendency into 6 groups through bisecting k-means, and its time curve visualization revealed cyclical patterns consisting of the same groups of proteins behaving similarly at non-sequential points in time (Figure 6.7). Each of these clusters consists of proteins that at first glance are not related, but in fact may be considered as co-expressed. We are able to discern at least two cycles through three distinct groups of non-sequential time points, where at least half of the dataset presented very similar behaviors (as maximum similarity has been set at 60%).

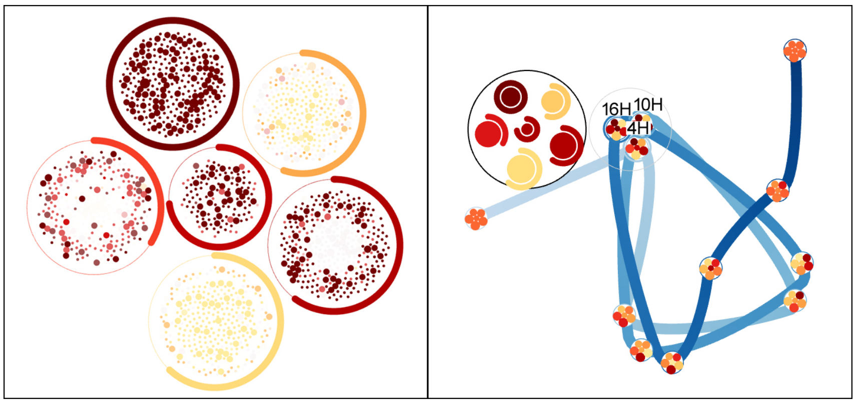


Figure 6.7: Network (left) and time curve (right) representations of the HIV-1 virus gene expression time-series dataset. Data has been clustered into 6 groups through the bisecting k-means clustering algorithm, and the 4H, 10H and 16H time points are selected with the mouse lens.

When analyzing this dataset using the mouse lens on the time curve, it was possible to quickly identify the groups of nodes that were presenting similar behaviors at the same points in time. For instance, in Figure 6.7 we show the lens being used on the group of time points representing 4, 10 and 16 hours, allowing us to see that, while there is a significant percentage of nodes in each cluster presenting similar behaviors across the three time points, one cluster stands out with full similarity. We can also interpret from the dark red color of this cluster that its proteins present peaks of expression

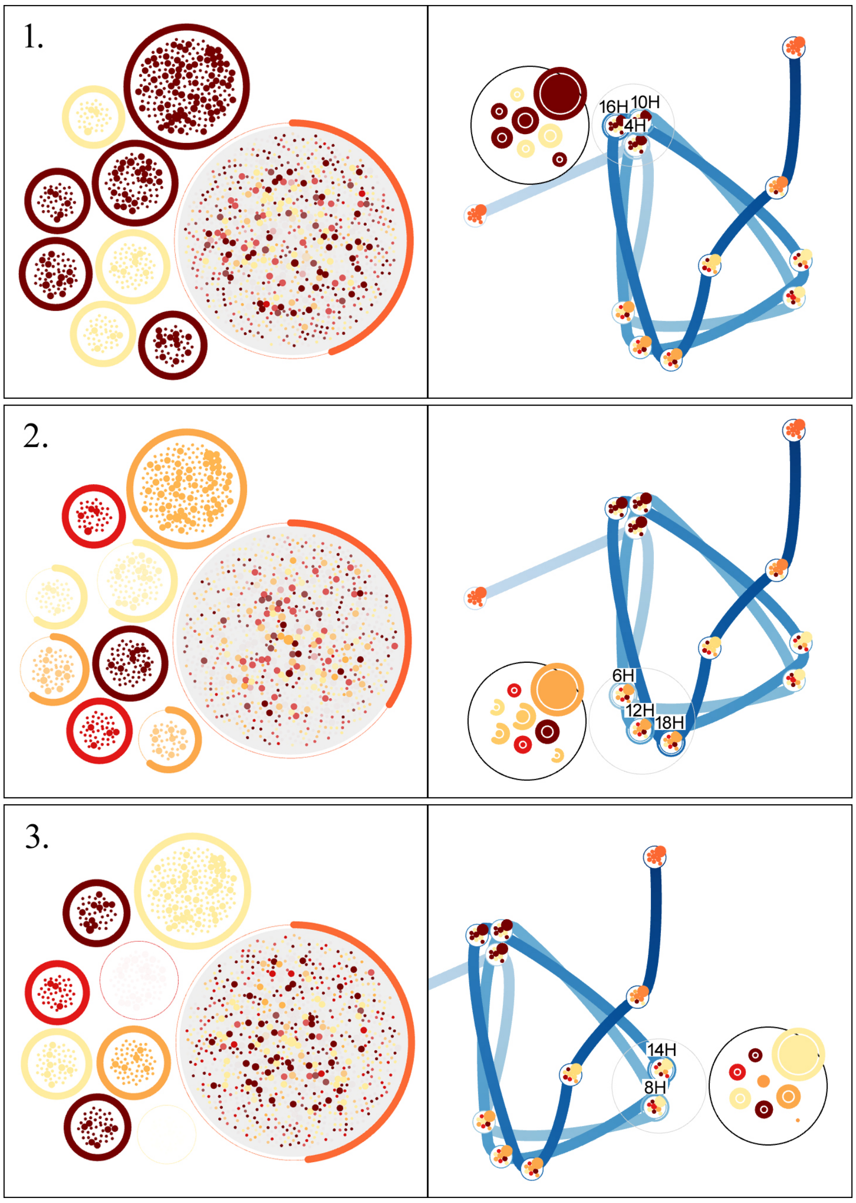


Figure 6.8: Visualizations of the HIV-1 virus gene expression time-series dataset clustered using the DBSCAN algorithm, as shown in the network panel (left). The cluster with a grey background represents genes classified as noise and it is not represented in the time curve's glyphs (right). Each visualization reflects the behaviors across each group of time points selected with the mouse lens (1, 2 & 3).

at these points in time, while most others appear to show either increasing tendencies or valleys of values.

To further analyze these behaviors, we clustered the dataset using DBSCAN, which resulted in about half the proteins getting filtered as noise, but creating clusters with more consistent behaviors (Figure 6.8). When comparing the previous group of time points with the mouse lens again, we can clearly identify that the resulting clusters have full similarity across all time points, where 5 are exhibiting peaks of expression and the remaining show valleys of values (Figure 6.8.1). The other two groups of time points were also examined using this method, revealing many of the same clusters to also present consistent behaviors, although more varied between them (Figure 6.8.2,3). In particular, we can discern 2 clusters with minimum similarity between the 8 and 14 hour time points (Figure 6.8.3). Through such exploration, these types of clusters can be identified, selected and either isolated to be studied further or filtered out of the dataset.

6.2.2 *Malaria Virus*

Following the previous dataset, we analyzed a time-series of the gene expression for the intraerythrocytic developmental cycle of Plasmodium Falciparum, the agent responsible for human malaria. This dataset contains 5080 genes, whose expression values were measured with an hour interval over a 48-hour period. The time curve visualization of this dataset shows that the dataset has a general continuous behavior throughout, with each time node sticking close to the one that follows without loops or overlapping, as shown in Figure 6.9. The time curve visualization was created with near 90% maximum similarity, meaning that genes present very similar behaviors overall. This matches the description of this dataset by an existing study [21] which refers to the behavior of the genes as a cascade of continuous expression that lacks sharp transitions. Moreover, near the end, the dataset appears to return to a state similar to where it began, characteristic of a cycle. By using the lens, we can compare two time points at the beginning and end of the timeline (TP1 and TP48) to observe that most of the dataset presents very similar values and variations at those instances (Figure 6.9.a).

However, while expression values do not appear to shift drastically, we can observe that the data does not present consistent variations. In the supporting timeline graph, we can easily identify both moments of stable variation and some larger shifts in the data. We can use the lens to analyze some of these larger shifts, by identifying the groups of nodes with that present differences between time points rather than similarities. For instance, when comparing TP13 and TP14 (Figure 6.9.b), we can quickly identify that the two clusters of nodes with higher values (dark colors) were responsible for that spike in the time curve, as they have less consistent values between the two selected time points. When comparing their glyph colors, we can also observe that this was the result of a relatively significant increase in values across both nodes.

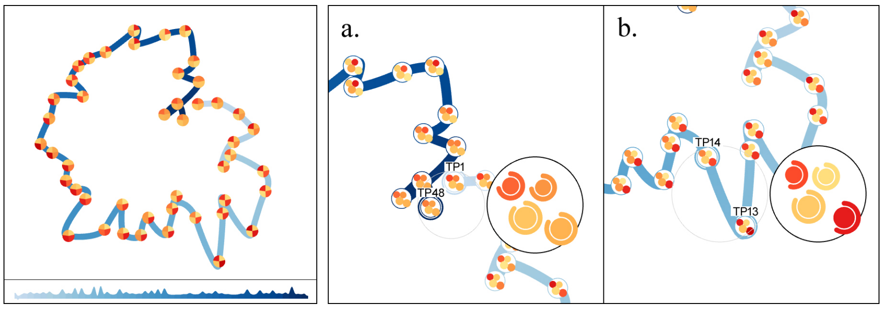


Figure 6.9: Time curve visualization of the Plasmodium Falciparum dataset with supporting timeline graph below it (left), along with close-ups of an analysis of two sets of time points (right): TP1 and TP48, which show overall similarities (a.), and TP13 and TP14, which present two clusters with a higher degree of differences (b.).

6.2.3 *Yeast Cells*

The next dataset to be visualized contained transcription profiles across a yeast cell cycle, where alpha factor synchronized cells were followed across two cell cycles [121]. 4381 cells were sampled every 5 minutes throughout 2 hours, resulting in 25 samples per cell. The dataset was clustered using the DBSCAN algorithm, which discovered 5 clusters of cells with unique expression patterns, as depicted in Figure 6.10. To better differentiate between extreme and middle values, we have chosen the "RdYlGn" color palette. The time curve visualization presents a loop in its middle, which may be related to the second cell cycle. Despite maximum similarity being set at 80%, the

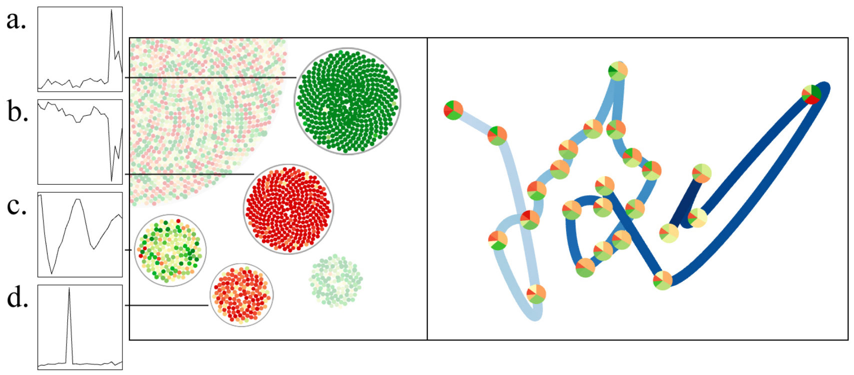


Figure 6.10: Network visualization of the yeast cell cycle dataset clustered by the DBSCAN algorithm (left) with line charts of the average temporal profiles of four clusters (a. through d.), and a time curve visualization (right).

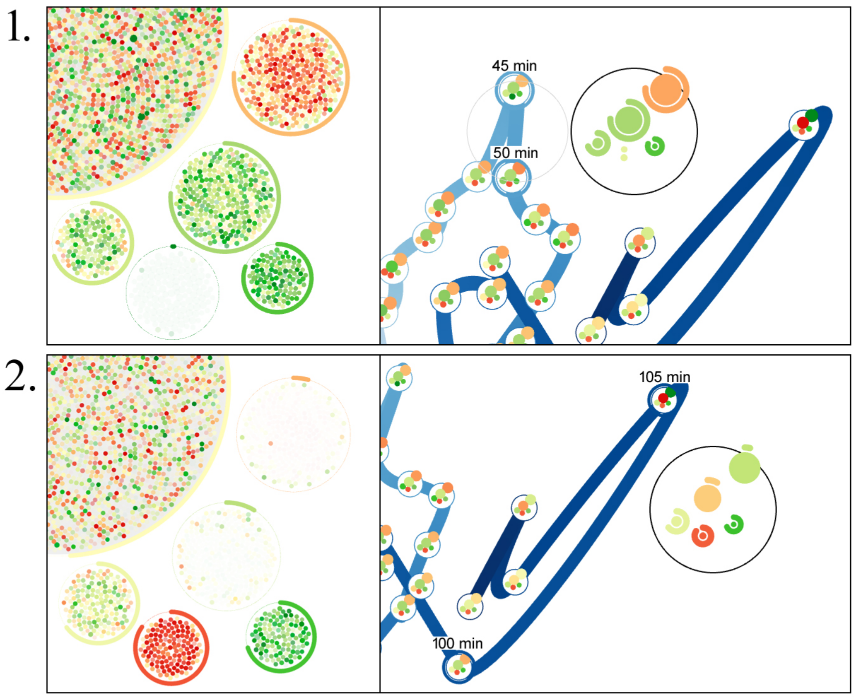


Figure 6.11: Network (left) and time curve (right) visualizations of the yeast cell cycle dataset. The mouse lens is used to select the time points at 45 and 50 minutes (1.), resulting in the bottom-most cluster presenting almost no similarity between these times. The lens is also used to select the 100 and 105 minutes, resulting in the two top-most clusters also presenting very little similarity between those sequential times, indicating a large shift.

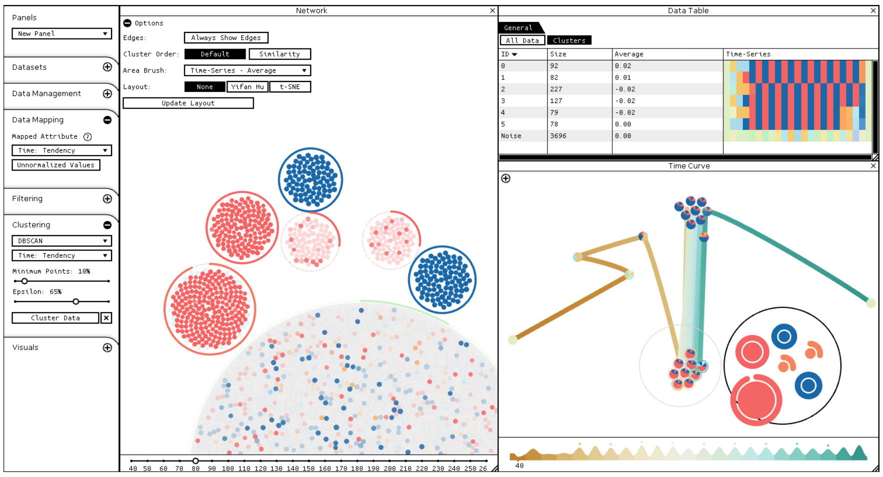
cycles appear to be significantly different from each other, which has been observed in a previous study of this dataset [121].

During the first cycle, we can observe a spike in the time curve at 45 minutes, which breaks away from the expected tendency (Figure 6.11.1). Comparing this time point with the following one reveals one cluster with very high dissimilarity, highlighting it as the cause for this spike. Moreover, examining this cluster shows that it presents very little activity throughout the dataset until that point in time, during which the values of all its nodes increase sharply (Figure 6.10.d). We can also observe that the end of the second cycle (at 105 minutes) is marked by a very significant shift in values (Figure 6.11.2), characterized mainly by significant shifts in values in the cells contained in two clusters. We can identify these as the two top-most clusters, where one presents a large increase of values (Figure 6.10.a) and another presents a large decrease (Figure 6.10.b), immediately followed by the same expression values shifting inversely.

6.2.4 *Saccharomyces Cerevisiae*

In addition to the previous dataset, we visualized gene expression data measured in Saccharomyces cerevisiae cell cultures, another species of yeast. These cells have been synchronized at different points of the cell cycle through a temperature-sensitive mutation (CDC15) which arrests cells late in mitosis. The dataset contains 4816 cells with expression values measured every 5 minutes for 2 hours.

The dataset was first clustered into 7 groups using the hierarchical clustering algorithm, while the time curve visualization was created by positioning time points by tendency, using a maximum similarity of 50% (Figure 4.3). To better differentiate between extreme values we have chosen the "RdYlBu" color palette, while time is mapped across the "BrBG" palette. The resulting time curve shows that most of dataset is initially comprised values increasing or decreasing with little correlation, changing between unique states during the first four time points. However, this followed by a behavior that is repeated throughout the remaining time points: the dataset alternates between two states where two clusters alternate oppositely between peaks and valleys of expression. Towards the end of this consistent behavior, it is possible to discern one time point that is located relatively farther from the top group. This may indicate the occurrence of an event that resulted in a break of the cycle.



Mapped Attribute: Time: Tendency
Unnormalized Values
Clustering: DBSCAN
Time: Tendency
Minimum Points: 10%
Epsilon: 65%
Cluster Data

Edges: Always Show Edges
Cluster Order: Default | Similarity
Area Brush: Time-Series - Average
Layout: None | Yifan Hu | t-SNE

ID	Size	Average	Time-Series
0	92	0.02	
1	82	0.01	
2	227	-0.02	
3	127	-0.02	
4	79	-0.02	
5	78	0.00	
Noise	3696	0.00	

Figure 6.12: Screenshot of CroP visualizing the Saccharomyces cerevisiae dataset. The clusters created by the DBSCAN algorithm are represented in the network panel and listed in the data table panel. The time curve panel shows the data alternating between two states, one being selected with the mouse lens to highlight consistent peaks and valleys of values across several clusters.

Additionally, we clustered the dataset using the DBSCAN algorithm. While a significant portion of the dataset was classified as noise, likely due to a high amount of variation in temporal patterns, the algorithm was capable of grouping the primary genes responsible for the previously discussed behaviors. This can be observed in Figure 6.12, where using the mouse lens to select the bottom group of nine time nodes shows that 4 out of the 6 clusters consist almost entirely of nodes with the same behaviors. Furthermore, the inconsistencies within the remaining two clusters appear to be caused by genes that stop behaving consistently towards the end of the timeline, which was also noted previously.

6.2.5 *Lung Cancer*

To demonstrate CroP's ability to process larger datasets, we represented a dataset that explored the human gene expression responses to glucocorticoids [101]. This dataset contains 119208 cells of the human lung adenocarcinoma exposed the synthetic glucocorticoid dexamethasone, and describes their changes in gene expression every 2 hours for 6 time points. The time curve revealed a simple circular pattern with significant similarities between the data at the 3 hour and 9 hour time points (Figure 6.13.1). To better compare the dataset between these time points, we clustered the data using the OPTICS clustering algorithm (Figure 6.13.2), as it was one of the fastest available algorithms for a dataset with these characteristics. Due to the size of the dataset, the amount of distinct temporal patterns resulted in the creation of a high number of clusters. However, from these we are able to discern some particularly large clusters, one characterized by having minimal values throughout the dataset, while the remainder showed significantly high values at the 3 hour and 9 hour time points.

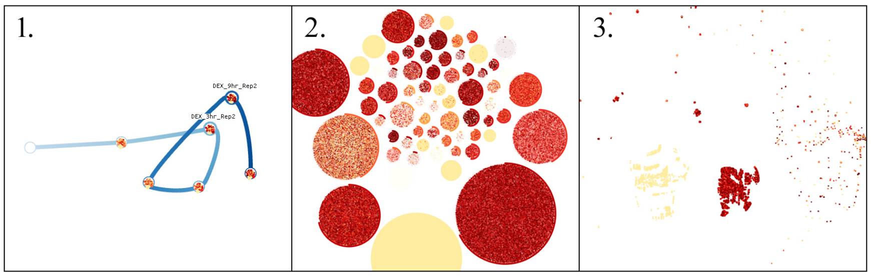


Figure 6.13: Lung cancer dataset represented through a time curve visualization (1.), a network clustered by the OPTICS algorithm (2.), and a network sorted using t-SNE (3.). While OPTICS discovered a large diversity of temporal patterns, the t-SNE layout divided most cells into two groups.

To further analyze these potential patterns, the data was spatially sorted using the t-SNE layout (Figure 6.13.3). Although the layout does not create cluster objects, it was capable of effectively sorting the points into visually distinct groups: two large groups in the center, surrounded by some small groups and a "cloud" of scattered points on the right. It should be noted that node repulsion is disabled for datasets beyond a certain size in order to improve performance, which results in condensed groups of nodes. In comparison to the clusters created with OPTICS, the t-SNE layout more clearly divided the data points that contain consistently low values into the left group and those with significantly high values into the right group, while also separating data points with temporal patterns that do not fit any particular group. We were able to conclude that the large group of data points with high values was primarily responsible for the pattern observed in time curve: a large increase that led to peaks of values at the 3 hour and 9 hour time points, followed by a slow decrease. While there is a limited amount of time points, this may describe a potential cyclical pattern of expression that would continue happening across this group of points.

6.2.6 *Seminavis Robusta*

Another large dataset that we analyzed through CroP was a gene expression time-series dataset of the Seminavis robusta, a benthic species found in biofilms along shallow coastal regions. The dataset is part of a study of the effects of associated bacterial spent medium on the gene expression and metabolic processes of the Seminavis robusta organism, which impacts its sexual reproduction. The dataset contains 25557 genes and describes the changes in expression values over 18 time points.

To initially sort the most predominant temporal patterns, the dataset was clustered using k-means into 5 clusters. These are depicted in Figure 6.14 (labeled C1 through C5), and each presents a unique profile with different overall tendencies and peaks of values. The time curve visualization created from this dataset is depicted in Figure 6.15, where we highlight specific

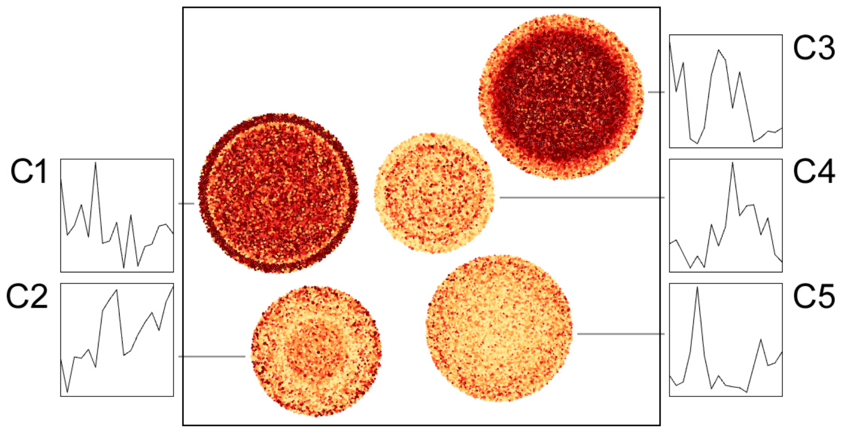


Figure 6.14: Network visualization of the Seminavis robusta dataset, clustered into 5 groups through the bisecting k-means algorithm. It is represented alongside time-series graphs of the average values for each cluster (C1 to C5), obtained through the network's mouse lens.

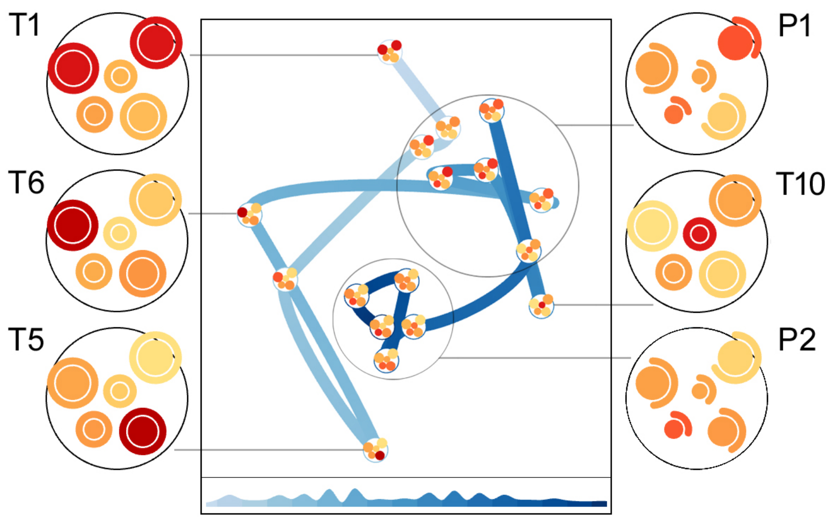


Figure 6.15: Time curve visualization of the Seminavis robusta dataset, whose data was clustered into 5 groups. It is represented alongside glyphs representing four moments (T1, T4, T5 & T10) and two groups of time points (P1 & P2), obtained through the time curve's mouse lens.

moments (T1, T4, T5 and T10) and two groups of time points (P1 and P2) which help understand how expression values behave. The first major event occurs at T4, which appears to be caused by an significant drop in expression values in the genes in C3 and a fast increase in expression in the genes in C5, which peaks at T5 and then drops again, resulting in another notable shift. The dataset appears to return to a state similar to that of the start represented by the group P1. In this group, we take note of the time point T10, whose larger shift in position appears to be caused by the genes in C4, which present a significant peak of values. This is followed by a period of relatively smaller shifts in gene expression, represented as the group P2.

Although these groups of temporal patterns could be further refined through other clustering options, they are able to help portray how the dataset generally varies over time, particularly when analyzing the resulting time curve visualization. From this overview of the behavior of the gene expression values, it was already possible to identify significant moments and narrow down the responsible genes.

6.2.7 *Coronavirus Disease*

In addition to the previous temporal datasets, we also visualized a multivariate dataset detailing the effects of the COVID-19 pandemic on the population of the state of California in the United States of America. This data was obtained from the California Health and Human Services Open Data Portal [63] and describes the number of tests, cases and deaths across every county in California between February of 2020 and January of 2022. In order to better compare data between counties, we divided the total number of tests, cases and deaths by the population as to obtain these values per capita.

We can observe the unnormalized distribution of these values across two counties in Figure 6.16, which shows a comparison between the general proportion of tests, positive tests, cases and deaths per capita. However, due to the vast differences in values between these variables, it may be cumbersome to identify the relative significance of each variable across the whole dataset.

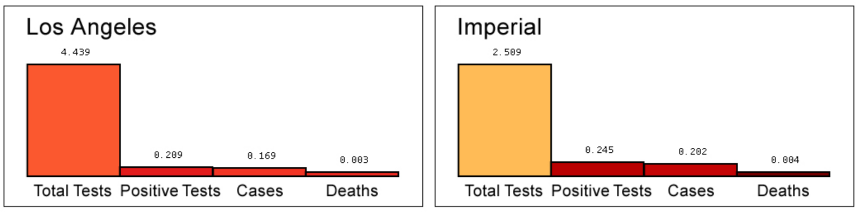


Figure 6.16: Bar charts depicting the unnormalized values of total tests, positive tests, cases and deaths per capita registered during the COVID-19 pandemic in the counties of Los Angeles and Imperial.

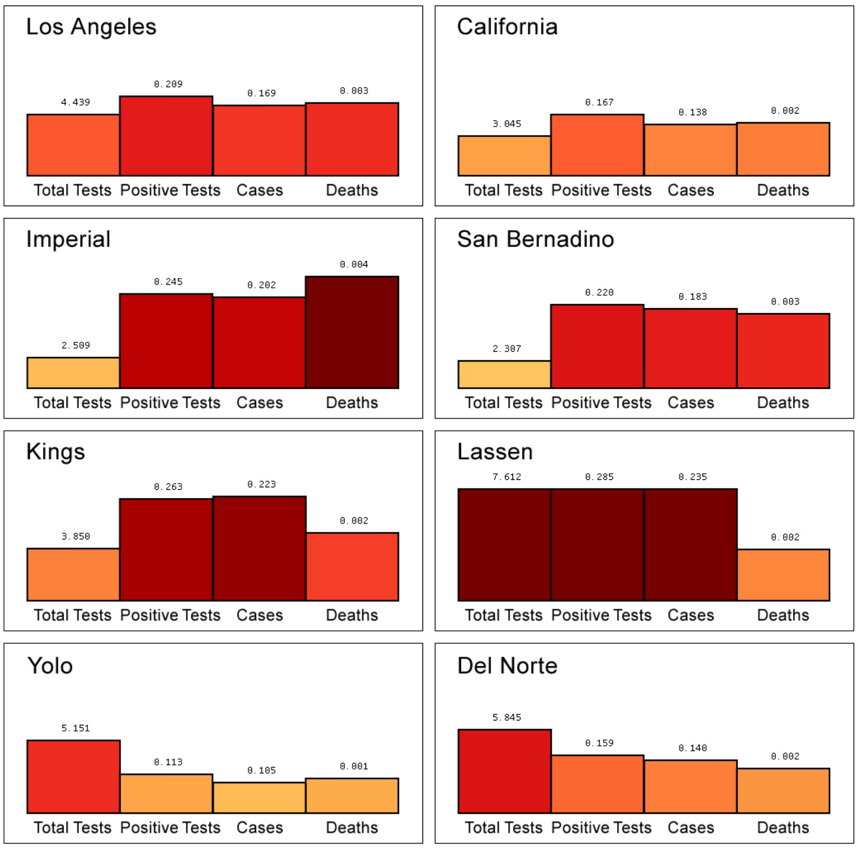


Figure 6.17: Bar charts depicting the normalized values of total tests, positive tests, cases and deaths per capita registered during the COVID-19 pandemic for several counties in the state of California.

In this regard, we can utilize normalized variables to represent the intensity of each value in relation to every other county, as shown in Figure 6.17. For instance, the bars for total tests, positive tests and cases in Lassen are completely filled, indicating that this county had the highest number of these per capita in comparison to all other counties.

Similarly as with the previous datasets, we clustered the data into a small number of groups in order to quickly identify any patterns in the distribution of values (Figure 6.18). The four clusters, obtained through bisecting k-means, presented distinct profiles, where middle cluster appears to contain all the counties with the highest values per capita across the dataset. Additionally, each variable is depicted as a node in the multivariate view, which

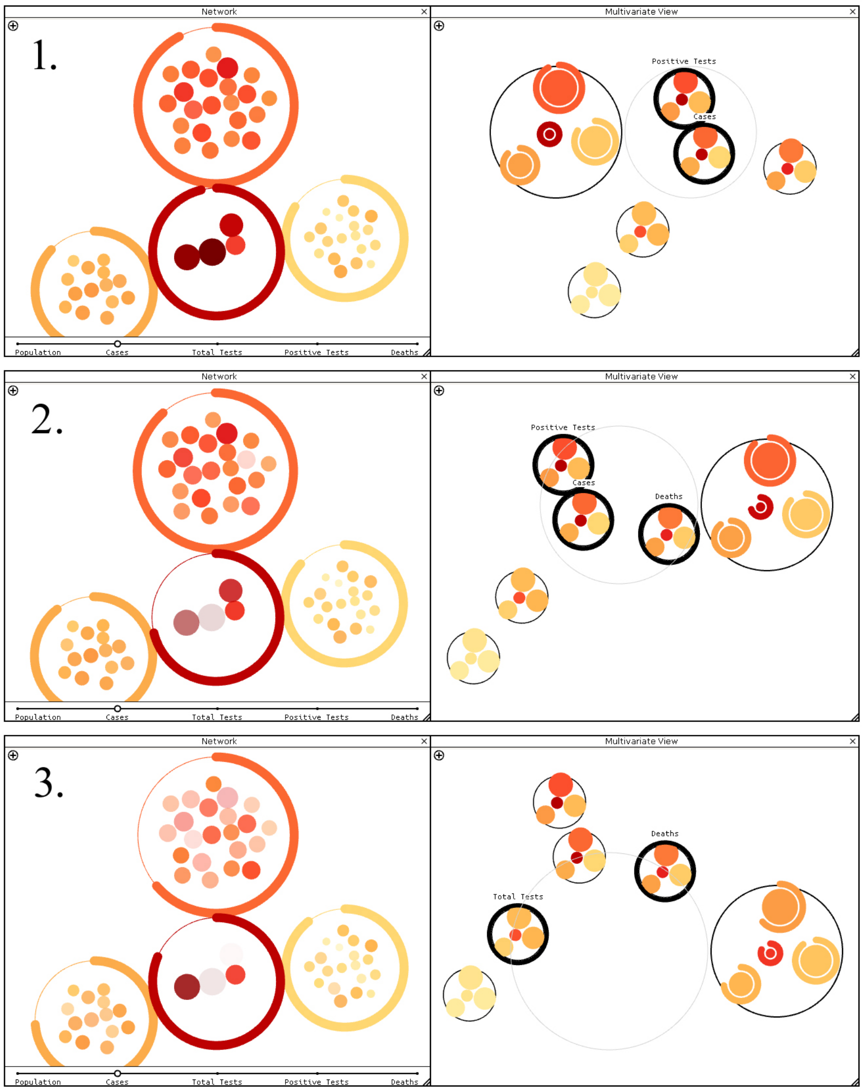


Figure 6.18: Data visualizations of the COVID-19 pandemic dataset, where each county is represented as a node in the network (left) that has been clustered by its variables, which are represented as nodes in the multivariate view (right). Using the mouse lens, multiple variables are compared across all counties: the number of positive tests and cases (1.); positive tests, cases and deaths (2.); total tests and deaths (3.).

have been positioned by the t-SNE layout. The two variables with the most similar distribution of values are "Positive Tests" and "Cases" (Figure 6.18.1), which is expected as a positive test would be an indication of the COVID-19 infection, unless it was a false positive. There is also a correlation between "Cases" and "Deaths" (Figure 6.18.2), although it is more inconsistent, possibly due to the variation in factors related to the ability of each county to handle the pandemic.

Finally, the largest difference in distribution appears to be between "Total Tests" and "Deaths". The glyphs in the multivariate view show that the number of total tests per capital is relatively lower than their other variables, which the exception of the counties in the right-most cluster, where this trend appears to be inverted. In order to better understand this, we can look at the graphs in Figure 6.17 where we see that "Total Tests" and "Death" are more consistently inversely proportional across the selected counties. It is possible that such a correlation could be attributed to prevention measures, as a higher number of tests per capita would lead to infections being detected and treated earlier, potentially lowering death rates (and vice versa). However, such conclusions would have take into consideration additional factors throughout the counties such as hospital availability, number of people with health insurance and other preconditions that may have contributed to these values.

6.3 DISCUSSION

Throughout the preformed experiments, we were able to visualize and explore various types of datasets, starting with low-dimensional datasets used in testing basic representation features, up to high-dimensional datasets which exhibited diverse behaviors across thousands of data points.

Due to the inherent complexity in creating comprehensible abstract visualizations, the initial experiments focused on the time curve visualization as its ability to represent various behaviors results from the distortion of a timeline. Through simple datasets, we were able to more easily compare the created time curve visualizations with the original time-series, allowing us to match its visuals with any observed behaviors, such as periods of stagnation and moments with intense shifts of values, as well as regressions and cycles. By following these experiments with the visualization of datasets containing multiple time-series, it was possible to identify that the representation of similar behaviors was consistent with previous tests. The resulting visualizations were shown as combinations of the behaviors of all the time-series in the dataset, with the most common behaviors being represented more prominently. Furthermore, the time curves also highlight significant events, such as moments where major shifts in values occurred.

In relation to our research questions, we may consider that these experiments have already shown that the generated time curve visualizations are able to promote the discovery of meaningful relationships and patterns. Moreover, in seeking to answer how visual abstractions can be used to manage visual complexity, we developed the time paths layout to reduce visual noise in favor of representing prominent behaviors, but this comes at the cost of data fidelity. This was reflected in the preformed experiments, where manipulating the layout's parameters showed that increasing the level of smoothing highlighted overall tendencies by removing smaller variations, in addition to some outliers that could have marked significant events in these datasets. Additionally, high parameters resulted in exaggerated deformations of simple behaviors, reinforcing the need for balance between accuracy and abstraction when seeking to achieve readability. However, such exaggerations could be considered for artistic representations of datasets.

In further regard to managing visual complexity, we can also consider how time curves are able to represent the overall behaviors across thousands of data points through a single visualization. While this may allow users to identify moments and periods marking significant events, it is not possible to determine the data points that are responsible for such behaviors through the time curve model alone. It was in analyzing such behaviors that we sought to answer two other research questions: how to comprehensibly represent multivariate datasets and how CMVs can facilitate their exploration.

Through CroP's interactive multiple views framework, we are able to use animated visual elements while giving users varied navigation options such as panning and zooming and the ability to change the parameters of diverse layouts with the visualization adapting dynamically. However, in seeking to create comprehensible visualizations, we had to consider how these representations of the data would be visualized in a static environment, such as when exported as images. Functionalities like selecting consistent palettes across diverse views, controlling the size of elements, glyphs with different levels of detail, directional elements, parameter-based smoothing, juxtaposed views and coordinated brushing were implemented for the creation of visualizations that support readability while minimizing loss of information.

More importantly, many of the same functionalities are used in support of the exploration of these visualizations to facilitate their analysis and discovery of patterns of information, in particular the juxtaposed views and coordinated brushing. As shown throughout the performed experiments, the different types of clustering allowed for varying degrees of precision in the creation of groups of data points containing similar patterns. Utilizing the simpler types of clustering (such as hierarchical and k-means, which require few parameters) was enough to reveal the diversity in patterns across multiple datasets and provide a better understanding of the patterns revealed by the time curve visualizations. Alternatively, other algorithms (such as DBCANS and OPTICS) were able to define more uniform groups while isolating independent patterns as noise.

The composition of these groups can be clearly viewed in the data table and network panels, while the time curve and multivariate view panels uses them in glyphs to represent the state of the dataset at different instances so they can be compared. The mouse lens then allows for further exploration into the patterns revealed by the layouts of these panels, facilitating the identification of groups of nodes that are responsible for unique behaviors and relationships. For instance, while a general cyclical tendency was identified in the HIV-1 dataset it was only through the time lens that we identified the nodes that followed this behavior, despite the existence of multiple groups with different temporal profiles that followed the same cyclical pattern. Similar analysis was performed for other behaviors, such as the large shifts of values and regressions identified in the Malaria Virus and Yeast Cells datasets, whose responsible cells were highlighted by the differences represented in the data lens.

While the discussion on comprehensible visualization should also be validated through user tests, through the performed experiments it is possible to conclude that the combination of the functionalities implemented into CroP facilitated the exploration of varied datasets, while highlighting patterns and providing the tools to analyze them. The identification of significant variables and groups of data points can further help understand complex datasets and extract new knowledge that may help solve problems.

7 VALIDATION

The development of CroP can be characterized as iterative and incremental as it resulted from multiple periods of testing and validation. In this chapter, we will describe the user tests performed throughout the development of CroP to evaluate its usability and the effectiveness of the implemented visualization models. Throughout the development of this visualization tool, we performed model and interface tests with people of different fields of study at three separate instances.

Due to the complexity of the time curves model, some preliminary tests were performed early in the development to determine the viability and effectiveness of the model in the context of analyzing temporal patterns and discovering significant moments. Once all the planned visualization models and functionalities had been implemented into CroP, we were able to produce a prototype to be used for interface tests. These were initially performed with a reduced number of participants, after which both the tests and the prototype were refined for a more comprehensive round of tests. In addition to testing the tool itself, we also simultaneously asked users to evaluate the visualization models. We will describe each test and the conditions in which their were performed, as well as discuss the results and feedback that was obtained.

7.1 PRELIMINARY MODEL SURVEY

Throughout the experiments preformed using our implementation of the time curve model we were able to observe the creation of visualizations featuring characteristics that described the datasets that were being portrayed, such as various types of cycles and variation, periods of stagnation and moments of significant value increases. However, as the time curve depicts these temporal behaviors through an abstraction of a timeline, one predominant concern is the comprehensibility of the model by different types of users, in particular those minimal knowledge of data visualizations. To this end, we performed an early survey that would test how users from different fields would fare in decoding time curve visualizations and identify patterns and significant moments.

— *Survey Questions*

Each participant was given a form that consisted of various multiple-choice questions whose objectives were twofold: to determine subjects' ability or difficulties in perceiving and interpreting the information encoded by each visualization, and surveying participants' preferred model for data representation in regard to functionality and aesthetics. As such, the survey featured multiple tasks where participants would need to solve problems using only static visualizations created by our implementation of the time curves model, in addition to questions that allowed them to give a rating to visualizations created with different parameters in regard to their legibility and aesthetics.

While animated and interactive components may aid users in interpreting the visualization, particularly in identifying edge direction and specific time points, we chose to exclude them from this survey in order to focus on determining the ability of the base visual elements to encode information. In this respect, all the visualizations use consistent color encoding for time progression, where from black represents the initial time point and orange represents the final.

To start, participants were only provided with context over the nature of the data visualization. Given the learning curve that is inherent to the interpretation of data abstractions, we intended the visualization model to be gradually explained throughout the survey. As such, the survey featured increasingly more complex visualizations across each set of questions, where we noted their initial impressions, abilities and difficulties in perceiving the visualizations at each stage. The survey was divided into seven sets of questions (S1 through S7):

S1: Matching a visualization with a behavior. The first question in the survey presented the visualization shown in Figure 7.1 without showing the line chart which describes the original dataset. Participants were then asked to identify the main behavior they think is being represented out of a list: whether it shows a continuous increase in values, erratic values, stagnated values or cyclical values. After their answer, the line chart describing the original dataset was shown and they were told that the stable cycles generate overlapping ovals shapes. On one hand this was meant to verify if viewers would associate the gener-

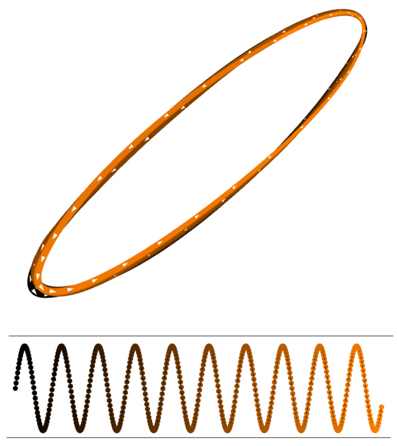


Figure 7.1: Time curve of a sine wave for S1 of the initial model tests, where users had to identify the general behavior being depicted.

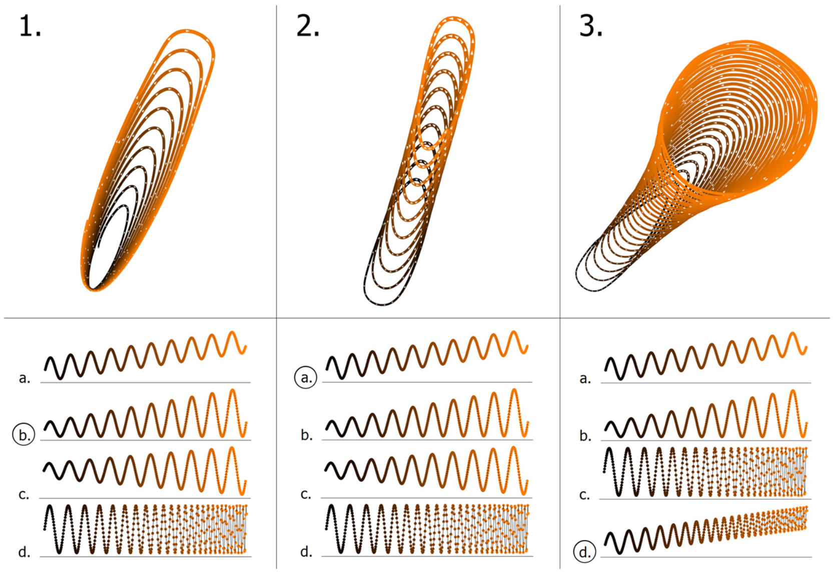


Figure 7.2: Set of three time curves for S2 of the initial model tests, where users had to pick between four line charts to match each time curve with its corresponding dataset. The correct answer is circled.

ated circular shapes to cyclical behaviors, while on the other present them with the most basic input and output for the model.

S2: Matching a visualization with multiple behaviors. This section consisted of a set of three questions where participants were tasked with matching increasingly complex visualizations drawn using our model with line charts representing their original datasets. Participants were presented with the visualizations shown in Figure 7.2 and asked to choose from the four line charts that showed cycles with different emerging behaviors: variations in value, amplitude and frequency. Through this, we intended to identify whether viewers could associate the variations in the circles' displacement, shape and size to their correspondent behaviors.

S3: Analyzing real data –- Monthly Milk Production. This section tasked participants with deciding the veracity of statements regarding the behaviors exhibited by the represented cycles represented in the "Yearly Milk Production" data visualization, shown in the left side of Figure 7.3. The goal was to determine if participants could analyze time curves created from noisy time-series and correctly discern the following behaviors: an overall increasing trend, consistent cycle duration, the absence of a significant decrease in production before the final cycles, and a stabilization in production during the final cycles.

S4: Classifying visualizations –- Monthly Milk Production. In this set, participants were presented with six visualizations of the previous dataset created by the model, each with an increasing level of smoothness (Figure 7.3), where the initial and final visualizations showed the results of parameters being pushed to either extreme, while those in the middle showed the re-

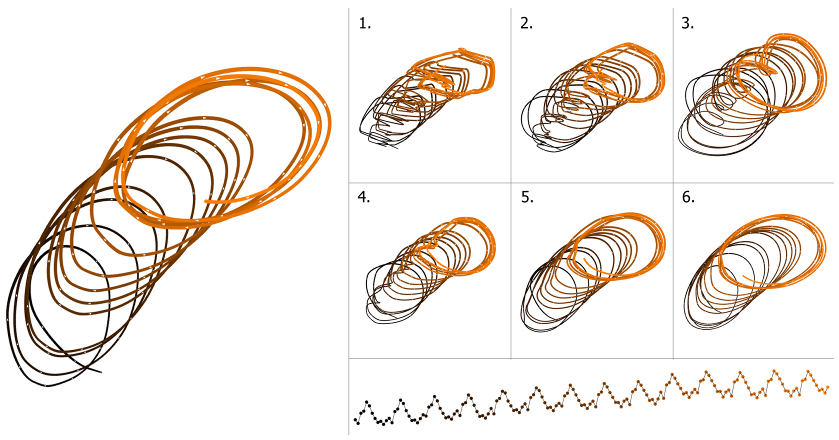


Figure 7.3: Time curve of the "Monthly Milk Production" dataset used for the questions in S3 of the initial model tests (left), and six variations of it utilizing different smoothing parameters for S4. The line chart that describes this dataset is represented at the bottom.

sults from near default parameters. Participants were asked to choose the visualization that they thought most accurately represented the data, the one they thought was the most visually appealing, and finally the one that they would overall choose to represent the dataset. These questions served to primarily to understand the preferences of participants and the role of smooth continuous lines in the representation of the datasets.

S5: Analyzing real data — Wolfer's Sunspot Numbers. Here participants were presented the "Wolfer's Sunspot Numbers" data visualization, shown on the left of Figure 7.4, where fluctuations do not have a consistent trend. Considering the visualization's characteristics, participants were asked to indicate at which points in the timeline they could identify two specific events: the largest value increase and an outlier value peak. These tasks were meant to test viewer's abilities to understand the time progression in a visualization where values do not have a consistent trend and, more importantly, identify an unusual visual element as an outlier, meaning the small loop that occurs during the final cycle in the visualization.

S6: Classifying visualizations –– Wolfer's Sunspot Numbers. This set of questions mirrors S4, where participants were once again shown six visualizations generated by the model for the same dataset with increasing levels of smoothness (Figure 7.4) and ask to choose the most informative, the most visually appealing and their overall preference.

S7: General feedback. Finally, based on the visualizations that were presented throughout the survey, participants were asked to grade the model's utility (its ability to describe information or highlight the behaviors in the data) and the resulting visualization's aesthetics (whether they were considered visually appealing in comparison to traditional visualization approaches), from 1 (Very Poor) to 5 (Excellent). Participants were also encouraged to provide additional feedback at this point.

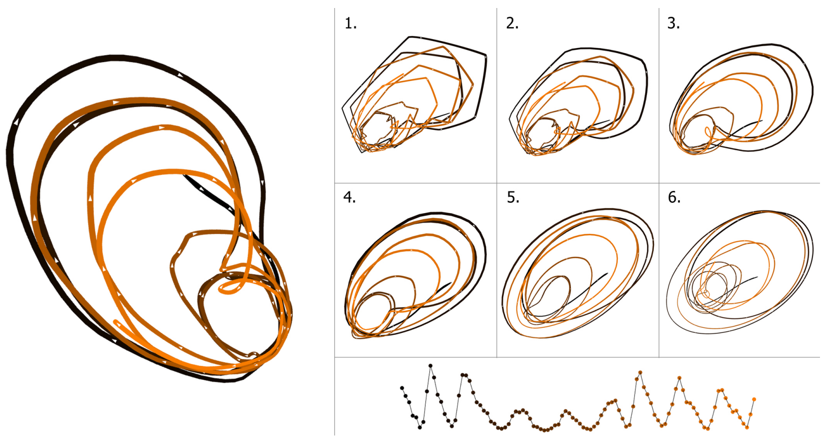


Figure 7.4: Time curve of the "Wolfer's Sunspot Numbers" dataset used for the questions in S5 of the initial model tests (left), and six variations of it utilizing different smoothing parameters for S6. The line chart that describes this dataset is represented at the bottom.

— *Results Analysis*

The study was conducted in person with university students from various fields of study: out of the 25 students surveyed, 8 had an Information Visualization background, 4 had a Computational Creativity background, 6 had a Computer Science background, and 7 had a Biomedical Science background. Of those without a visualization background, 4 had a high level of expertise in the visualization field, 8 had a medium level, and 5 had a low level of knowledge of data visualizations. Our discussion on the results is structured based on evaluating the performance of our implementation of the time curve's model. As such, focuses on the model's ability to represent general behaviors, significant moments and events, as well as analyzing the effects of smoothing a time curve visualization in regards to how it is perceived.

In the first two sets of questions, we intended to test whether viewers could identify general temporal behaviors from simple time curves. We considered the sine wave dataset to be our baseline, as it describes a simple, continuous cycle without noise, with its corresponding time curve (Figure 7.1) being an oval created from multiple loops that represent the cycles in the dataset. Participants were shown this time curve in S1 and, out of the four behavior options provided, all 25 participants associated the circular shape to a cyclical behavior.

Continuing with the identification of overall behaviors, S2 presented three variations of the sine wave time curve and were asked to choose the one out of four line charts that they thought best corresponded to the data represented. In S2.1, participants were shown the visualization that depicts an overall value increase (Figure 7.2.1) and 48% of participants chose the correct line chart. While participants showed some confusion while trying to understand this first layout, the corresponding dataset was still the one most chosen. When shown the visualization with constant minimum values and increasing maximums in S2.2 (Figure 7.2.2), 84% of participants matched it with the correct chart. This showed a significant increase in correct answers which, based on participant feedback, was a result of a increased understanding of the model's mapping of variables. However, in S2.3, where participants were asked shown a visualization that presented an increase of values and frequency (Figure 7.2.3), only 64% of them were able to do so correctly. Relatively to the others, participants generally took longest or had the most difficulties in identifying this chart, which was relayed as an inability to match the width of the time curve's arc to stronger variations in values, instead associating it to increases in values.

In S3 and S5, we asked participants multiple choice questions regarding two different datasets that depicted various types of behaviors. S3 featured the "Monthly Milk Production" dataset (Figure 7.3), which is characterized by multiple cycles with an overall increasing trend in production over the years until it stabilizes. Over 90% the participants were capable of correctly identifying the increasing trend, the moment of significant increase, and the period of stagnation through the visualization. However, 42% of participants incorrectly perceived the cycles has not having the same length in duration, which may have been a result from the time curve's parameters that resulted in cycles being drawn with different widths.

S5 featured the "Wolfer's Sunspot Numbers" dataset (Figure 7.4), which is characterized by values increasing and decreasing with an inconsistent frequency, where minimums are consistent but maximum values vary for each peak. When asked to identify when the first significant increase in values occurred, 96% of the participants were able to correctly identify that it occurred at the start of the dataset. When asked to identify a sudden shift in variation that occurred in the data, only 56% of the participants were able to identify this outlier as the small loop that was represented at the end of the time curve. Despite this, three other participants did identify the loop as an inconsistency in the time-series, but also deemed the other fluctuations to be of equal importance, and thus could not pinpoint the specific event. However, this did show that most viewers were able to potentially identify both trends and significant events.

Although the S3 and S5 utilized time paths to smoothen the original time curves and emphasize general behaviors, the ideal parameters may vary between each dataset or even depending on the context of the visualization's application. To better understand viewer's preferences, in S4 and S6 we presented participants with six visualizations with increasing levels of smoothing from the "Monthly Milk Production" dataset and the "Wolfer's Sunspot Numbers" dataset, respectively. Out of these, participants were asked which one they considered to be the most informative, the most aesthetically appealing, and their overall choice. From the results, we can discern some general trends: in both S4 and S6, most participants chose the same visualizations as being both the most informative and their overall choice, with 6 (44% & 46%) and 4 (28% & 24%) being the most popular choices in S4, and 3 (68% & 62%) and 4 (22% & 30%) in S6. Based on these results and feedback, we can conclude that these choices were based on readability. However, regarding aesthetical preference, participants seems to prefer time curves with round shapes, as the most popular choices in S4 were 6 (52%) and 3 (32%), and the most chosen in S6 were 6 (32%), 5 (26%) and 4 (26%). In general, we can infer that participants preferred minimal variation details with rounder curves, as long as the visualization was still able convey the overall behaviors of the original dataset. This is supported by the unpopularity of visualizations 1 and 2 all around, which in both datasets depict angular changes between time points, while visualizations comprised primarily of loops were chosen as the most aesthetically pleasing.

Finally, in S7 we asked participants for feedback regarding our model's performance based on the visualizations presented and their experience. Nine participants commented on the existence of a learning curve, which was considered as steep by those without a visualization background. However, the average score given to the model's ability to represent behaviors was of 4.08 out of 5, with participants justifying their scores by stating that the model can be useful for data analysis after the learning curve is surpassed. Additionally, participants commented on having been able to identify behaviors in the Time Path visualizations that were not apparent in the line graphs, and mentioned the potential of saving space by using the model to represent large datasets. When reviewing their difficulties in solving the previous tasks, participants mentioned that it was difficult to identify the temporal position of each time node. They also commented on the limitations of the model, such as overlapping edges potentially obscuring information and a lack of visual highlighting of significant time points. In regard to the aesthetics of the presented Time Path visualizations, participants gave an average score of 3.88 out of 5, commenting that they were generally more visually appealing than traditional visualization models, but that their application would be context-sensitive. Some participants considered that there was a lack of exploration of the visual elements, such as line width and color, in particular if the visualizations were to be used within an artistic or computational creativity context.

7.2 INTERFACE & MODEL TESTING

As the visualization models and functionalities within CroP were refined, interface tests had to be performed in order to determine its accessibility to our target audience, as well as to detect any usability flaws that should be fixed. These consisted of a series of short tasks that simulates an average user's process of navigating the tool, loading data, exploring it and then utilizing the provided data analysis functions to potentially identify patterns. These user tasks had to be created while keeping the complexity of the application and some of its models in mind, with particular regard to the level of experience of the target audience in handling visualization tools, as well as the available amount of time that could be allotted to each test.

7.2.1 *Initial Tests*

The initial interface tests were conceived with the purpose of detecting overall usability problems and to better understand how users with minimal knowledge of data visualization would perform in solving their assigned tasks. Due to the limited availability of participants that would fit within our expected target audience, the tests were performed with a group comprised of 9 college students from a Biochemistry degree.

User tests were performed individually and in person, each beginning with a short demonstration of CroP using small, randomly-generated network and time-series datasets, followed by a brief explanation of the dataset that they would be interacting with. For these tests, we chose datasets from our previous experiments: a human PPI network paired with the gene expression time-series dataset depicting the HIV-1 infection [108]. We chose these datasets due to the large number of proteins but small number of time points, giving users a large dataset to explore with a low temporal complexity. The time curve resulting from this dataset is also relatively simple, as it portrays two clear loops, allowing analyze how users decode this pattern, with minimal training.

After users answered profile questions regarding their field and level of expertise, they were allowed to interact with a prototype of CroP and asked to perform six tasks (T1 through T6), followed by two questions regarding the generated time curve model (Q1 and Q2), and ending with a set of five general feedback questions (F1 through F5).

— *Interface Tasks*

As each session was conducted in person, users could ask for help whenever they got lost in their navigation or didn't understand a task or how to complete it. However, such difficulties were noted down and the solution to a task was never given until the user had given up, in which case this was considered as a failure to complete the task. This test was comprised of six interface tasks and the time to complete each task was noted down and the sessions were recorded.

To begin, users were asked to navigate the tool and first import a network dataset file, then a time-series dataset file, containing the human PPI network and gene expression time series of the HIV-1 infection respectively.

Additionally, they were asked to filter out data that was not present in both datasets when prompted.

T1: Import the network dataset file.

T2: Import the time-series dataset file and keep only points that match both datasets.

After the previous tasks loaded the data into the default network, data table and time curve panels, users were asked to find and select a protein with a particular characteristic. The objective was to lead them towards the data table panel, as it is the simplest visualization model of those available, requiring the user to only sort it by the value attribute and selecting the top row. T4 then required users to note the tab that was created when selecting the previous protein so that they could access its proprieties and view its temporal profile. Task completion times are represented in Figure 7.5.

T3: Select the protein with the highest average value.

T4: Open the temporal profile of that protein in the Data Table.

Here, users were tasked with clustering the data into various groups and discovering which group contained proteins similar to the previously selected one. It required both the navigation of the tool's functionalities, and the users to understand the relation between a cluster and the proteins belonging to it.

T5: Use clustering to find and select a cluster of nodes that have the same temporal pattern as the protein selected in the previous task.

Finally, users were asked to freely use the time curve panel so that we could determine whether they could apply a layout and interpret the resulting time curve visualization as a temporal pattern. However, this task is further complimented by the next set of questions.

T6: Use the Time Curve panel search for a potential pattern of behavior.

— *Model Questions*

Upon applying the layout in the time curve panel on the final task, the users were asked if they were able to discern a pattern in the created time curve visualization. Due to the level of abstraction of the time curve visualization, we provided options with the most common temporal patterns portrayed in these visualizations, as well as options for the users that were not able to detect any patterns and those that inferred a pattern that was not listed.

Q1: In the Time Curve visualization, what predominant behavior pattern of expression can you discern?

a) Stagnant / Continuous behavior

b) Erratic / Unpredictable behavior

c) Cyclical / Repeating behavior

d) Cannot discern any pattern

e) Other

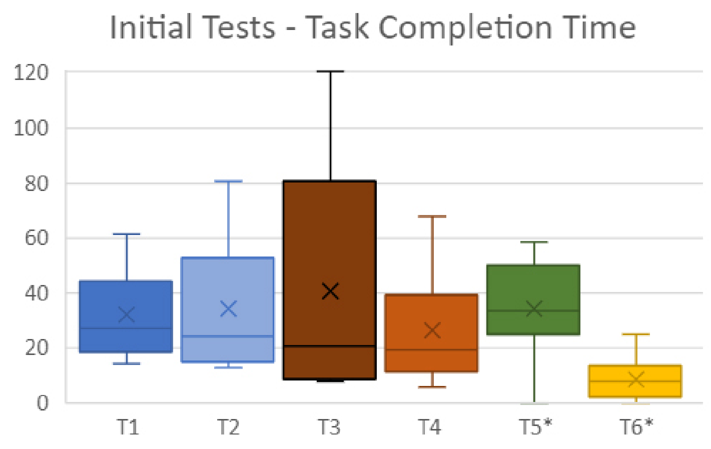


Figure 7.5: Box plot of the time taken by all participants to complete each of the interface tasks of the initial tests. T5* and T6* only present partial task completion as to be compared with similar tasks of the extended tests, where T5* shows the time taken to cluster the dataset and T6* the time taken to apply the forces layout on the time curve. Tasks that are similar with those of the extended tests, in Figures 7.7 & 7.8, have matching colors.

To further understand the level of confidence in the user's previous answer, we asked them how clearly they were able to perceive the previous pattern, if any, on a scale from 1 (Confusing) to 7 (Clear).

Q2: If you discerned a pattern, how clearly do you think it was depicted?

— *General Feedback*

At the end of the test, users were asked to select how much they agreed with the following affirmations from a 1 (Strongly Disagree) to 5 (Strongly Agree) scale:

F1: "I thought the tool was easy to use."

F2: "I thought the tool was needlessly complex."

F3: "I think I'd need more time to learn how to use this tool."

F4: "I could see myself using this tool, now or in the future."

To finalize the test, users were provided with the following open ended question where they could provide a longer answer regarding any of the difficulties encountered:

F5 - Did you feel any difficulties in navigating the tool, or in understanding any of the data representations?

— *Results Analysis*

As these tests featured a small amount of users, of which all had very low experience with visualization tools, our primary objectives in these tests were to discern the more prominent problems with the interface so that they may be corrected before initiating a new round of tests. As such, this analysis will focus primarily on the main difficulties encountered by the participants, but these results will also be taken into consideration in the analysis of the extended versions of the tests that were performed afterwards.

At this stage of testing, the prototype utilized a single dropdown for loading datasets from which users could choose between all the supported types of data. During T1 and T2, 3 of the 9 participants selected the wrong type of loading in one of the two initial tasks. While this could be attributed to a natural learning curve derived from using a new tool, this was taken into consideration when updating the user interface. The participants, however, did not have significant difficulties utilizing the data table panel in T3 and T4, with the only notable problem being two participants in identifying the panel as the means to discover the intended proteins.

While T5 and T6 allowed users more freedom to explore the visualization panels and arrive at their own conclusions, they also proved to be too vague and therefore confusing, especially for users with minimal experience. For instance, while all participants were able to use the menus to apply clustering, mostly with little difficulties, the lack of understanding of clustering resulted in many of them expressing confusion at identifying which cluster contained proteins with similar profiles to the one selected. However, of those that were able to perform the task, some were able to take advantage of the hierarchical clustering by increasing the number of clusters in order to create smaller groups with increasingly similar profiles, and thus discover which proteins were the closest to the one selected. Moreover, despite the limited interactions with the time curve model in T6, it was possible to detect usability problems when using the timeline slider to pinpoint specific time points. We identified this issue as deriving from a lack of visual feedback, as the hovered nodes in the timeline slider were not highlighted in the time curve.

Regarding the model questions, 7 out of the 9 participants were able to identify the pattern of the time curve as representing multiple cycles, then rated the clarity of the representation with an average score of 5.6 out of 7. The remaining two users did not think the visualization represented neither continuous nor cyclical behaviors.

For the final feedback, users were asked to disagree or agree with statements using scores of 1 to 5, respectively. Distribution of these scores is represented in Figure 7.6, and the average scores were as follows: participants generally agreed that CroP was accessible with 3.9, generally disagreed that the tool was complex with 1.6, generally agreed that they required more time to use CroP properly with 3.9, and finally generally agreed that they could see themselves using this tool in the future with 4.1.

7.2.2 *Extended Tests*

Following the feedback obtained from the previous tests, we refined both the interface and the tests themselves, extending them in order to include additional interface tasks and further model testing. These tests were conducted with 26 college students, where 16 were from the field of computational biology and 11 from information science, with varying degrees of data visualization knowledge and experience. Tests were once again performed individually, either in person or through a conference call, and personal assistance was provided in the case of significant difficulties, particularly when a task could not be completed. Sessions were recorded, and the time taken to perform each task was noted, in addition to any issues and necessary assistance.

Each user was initially introduced to CroP through a video that presented an overview of its functionalities, with minimal details, as well as a description of the data and how it is represented. Some precautions were taken in regard to the feedback received from the previous tests, such as the inclusion of a short explanation on clustering. Users were then asked several profile questions about their field of study, knowledge of data visualization and previous experience with visualization tools before performing the interface test. This test consisted of 16 tasks divided into 5 categories (T1 through T5), followed by a set of five feedback questions (F1 through F5), and ending with 10 questions regarding the visualization models, which are divided into 3 categories (Q1 through Q3).

— *Interface Tasks*

For these tests we once again used the gene expression time-series dataset of HIV-1 infection, not only so that these tests could be potentially compared to those done previously, but also due to the characteristics of this dataset continuing to be favorable for user testing (being complex but with a simple temporal dimension, relatively to other similar datasets). Task completion

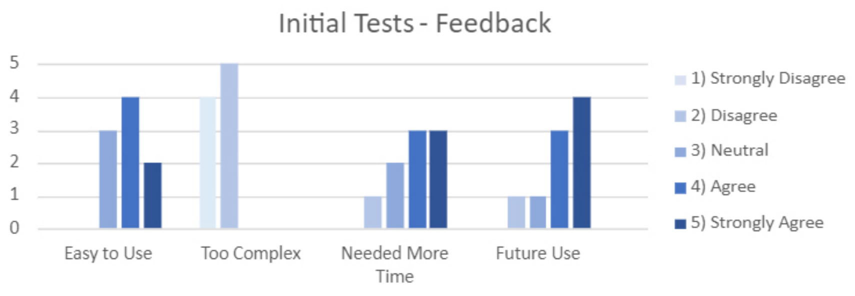


Figure 7.6: Score total given by participants for each of the affirmations in the feedback section of the initial tests, from F1 to F4.

time is represented for those with a biological background in Figure 7.7, and those without in Figure 7.8.

Similarly to the previous tests, users began by loading the two datasets into the tool. However, upon the data being loaded into the visualization panels, users were encouraged to navigate the tool freely for a minute as to reduce the potential fear of interacting with a new system.

T1 —- Initialization

T1.1: Import the network dataset file.

T1.2: Import the time-series dataset file and ignore proteins that don't exist in the previously loaded file.

Users were then directed to the data table panel as it is, once again, the most potentially familiar environment even to those with minimal experience with data visualization tools. Through these tasks we intended to test how users search, sort, and select one or groups of points (T2.1, T2.3 & T2.5), whether users are able to notice the tabs created for each selected row and recognize them as ways to dive down (T2.2), and if they were able to filter unwanted points from the dataset (T2.4). Additionally, through T2.6 we checked how users deselect previous selections, as this should be a simple and intuitive action.

T2 —- Data Table

T2.1: In the Data Table panel, find and select the protein with the highest "degree" value.

T2.2: In the properties of that protein, find the highest expression value of its time-series.

T2.3: Select the 3 proteins with the highest "degree" values.

T2.4: Filter the proteins that you selected to remove them from the dataset.

T2.5: Select every protein from ID 0 until ID 16.

T2.6: Reset your current selections.

In the next set of tasks, we tested how a new user would perform a simple data analysis task in order to identify a particular subset of the data. While T3.1 directly asks users to apply a specific type of clustering, T3.2 simply provides an objective, requiring users to navigate the tool and first discover how to change the current time point and then to decode the visualization to determine the average expression values represented by each cluster. Following this, T3.3 tested users' intuition in resetting selections.

T3 — Clustering

T3.1: Cluster the data by tendency and create 5 groups.

T3.2: Identify and select the cluster(s) that contain(s) a group of proteins with the lowest expression values at 16 hours.

T3.3: Reset current selections.

As we first needed to observe how tendency changed over time, users were then asked to change the "Data Mapping" option before focusing on the time curve panel. Given the initial complexity of the time curves model for new users, we encouraged participants explore the time curve panel during this time, including changing the color scheme or the sliders that control the parameters of the layout. T4.3 and T4.4 then tested users ability to recognize the position of the time points as describing similarity between them, and whether they were able to apply this knowledge to solve the task. Additionally, T4.4 could also be solved by looking at the timeline graph, which would highlight the time points where significant value shifts had occurred.

T4 — Time Curve

T4.1: Change "Data Mapping" so that colors are mapped to values by tendency.

T4.2: On the "Time Curve" panel, apply the "Forces" layout.

T4.3: Identify if there are other temporal points at which the network has a similar tendency shifts to that at 10 hours.

T4.4: Change the distance option to "Values" and identify at which time point(s) occurred the largest variations in expression values.

To finalize the interface portion of the test, users were asked to interact with the panels themselves. As CroP allows panels to be moved and resized within a fixed grid, this task was meant to gauge the difficulties of managing the workspace, while also detecting potential issues with CroP's automatic panel adjustments, such as overlap detection and resolution.

T5 — Panels

T5.1: Create a new "Data Table" panel and organize the workspace so that both data table panels are placed next to each other, with the same size.

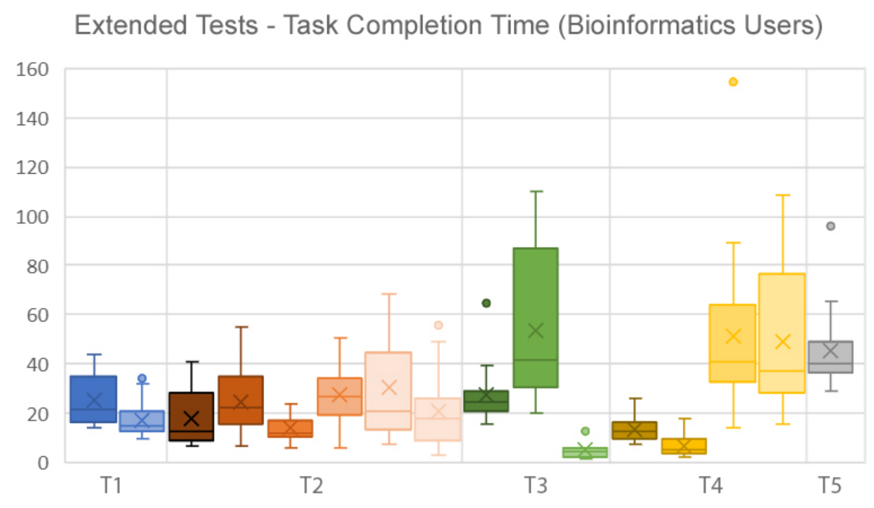


Figure 7.7: Box plot of the time taken by participants with a biological background to complete each of the interface tasks of the initial tests. Tasks that are similar with those of the initial tests, in Figures 7.5, have matching colors.

— *General Feedback*

Following the interface tasks, we gathered additional feedback by asking users to select how much they agreed with the following affirmations from a 1 (Strongly Disagree) to 5 (Strongly Agree) scale:

F1: "I thought the tool was easy to use."

F2: "I thought the representations of the data were easy to interpret."

F3: "I think I'd need more time to learn how to use this tool."

F4: "If I had to analyze network data or temporal data, I could see myself using this tool."

The distribution of answers is depicted in Figure 7.9. Participants were also provided with an open-ended question where they could write down any difficulties they felt when using the tool:

F5 - Did you feel any difficulties in navigating the tool, or in understanding any of the data representations?

— *Visualization Model Questions*

Following our previous study on the efficacy of the time curves model, participants were presented with several questions regarding CroP's time curve visualizations and cluster glyphs as to obtain feedback on these models.

To ensure that the participants had the same level of knowledge regardless of their performance within the interface tests, this section was introduced with a summary of the type of data being handled and how it is represented within CroP, including how it is clustered and an explanation of the time curve layout with an example. Before moving to the questions, participants

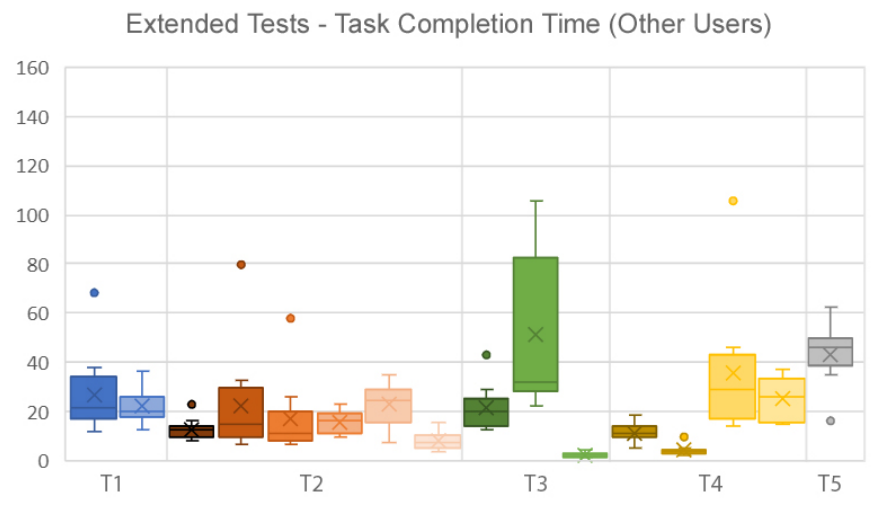


Figure 7.8: Box plot of the time taken by participants without a biological background to complete each of the interface tasks of the initial tests. Tasks that are similar with those of the initial tests, in Figures 7.5, have matching colors.

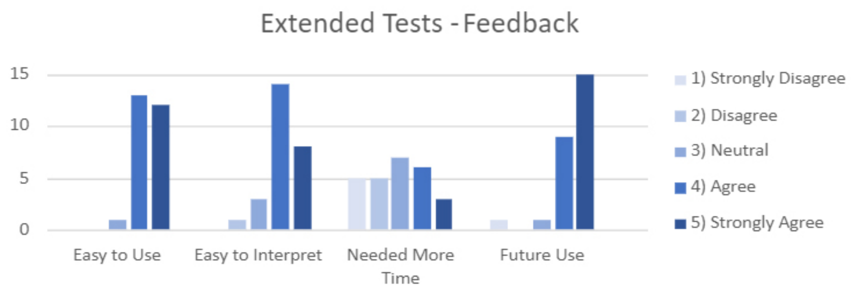


Figure 7.9: Score total given by participants for each of the affirmations in the feedback section of the extended tests, from F1 to F4.

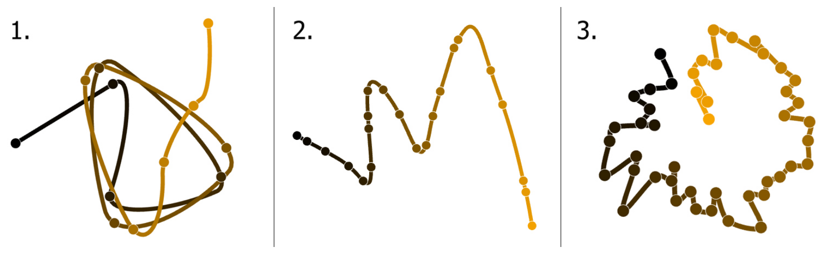


Figure 7.10: Three time curves shown to the participants of the third round of user testing, where they were asked to choose the behaviors that were represented in each visualization.

were first asked if they felt any difficulties in understanding the previous explanation from a scale from 1 ("No difficulties") to 5 ("Very difficult to understand"), and, if so, to note what they considered to be complex.

The first set of questions presented three time curves (Figure 7.10) and, in parallel to the previous tests, multiple options that contained statements that could describe the behaviors depicted in each of the visualizations. The same statements were presented for each of the time curves, from which the participants could choose those that they thought were fitting:

M1 — Time Curves

M1.1 - In regard to the order of the time points and their relationships, select the statement(s) that you think best describe(s), in general, the variation of values over time:

a) I can identify moments with strong variation between time points

b) I can identify moments with low variation between time points

c) There exists one or multiple cycles in the data (repetition between states)

d) I cannot identify patterns of variation

e) Other

The first time curve showed the cycle of the gene expression time-series dataset of the HIV-1 infection, the second time curve represented a single time-series without cycles but with inconsistent variations, and the third represented the malaria virus cycle. The distribution of statements chosen by the users for each time curve is depicted in Figure 7.11.

The second set of questions pertained to the time curve glyphs, which represented the state of the network at each point in time. As our objective is to abstract a large quantity of data into a simple comprehensible glyph, we presented users with three different glyphs representing the same type of data, first in a large size so it could be properly viewed (Figure 7.12), and then in a smaller size that better reflected how the glyph would be viewed in the tool itself (Figure 7.13). The first glyph can be described as a miniature representation of a clustered network, reducing its elements with minimal abstraction: every cluster is represented as a circle with its position and color matching the original cluster, and its color mapped to the average values of its group. The second glyph abstracts the clustered network into a circular graph by converting each cluster into a slice, where its color represents the average values, its size represents the size of the cluster, and its relative position reflects the position the cluster in the network. The third glyph, the bar chart, is similar to the former, where each bar represents a cluster in color and size, while its order reflects the horizontal position of clusters on the network.

When presented with each of the large glyphs, participants were provided an explanation of what each visual variable represented, and then they were shown statements regarding that glyph's comprehensibility. For the small glyphs, the statements reflected their legibility and how well they could be compared to each other. For each statement, participants could choose whether they agreed or disagreed using a scale from 1 (Strongly Disagree) to 5 (Strongly Agree), and the totals for the scores given are displayed in Figure 7.14.

M2 — Cluster Glyphs

M2.1 - "By solely looking at the glyph I can understand the state of the network at that point in time (how values are distributed)."

M2.2 - "I think I would need more time to comprehend this type of glyphs."

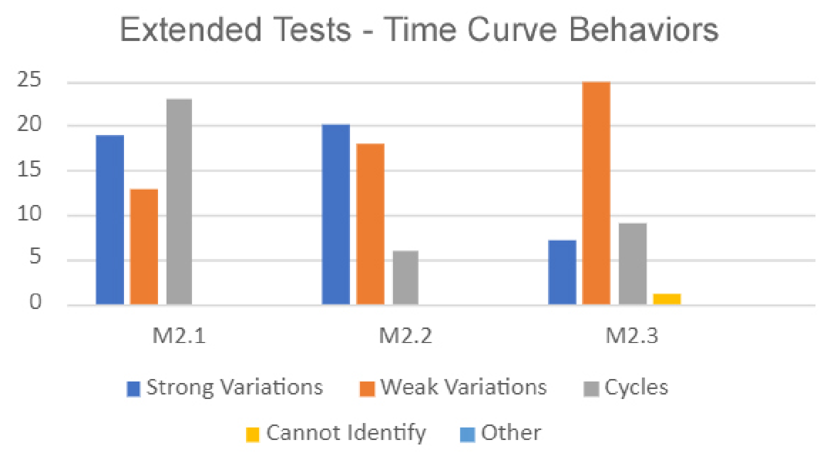


Figure 7.11: Total behaviors perceived by all participants for each of the time curves (M2.1, M2.2 & M2.3) in the model questions of the extended tests.

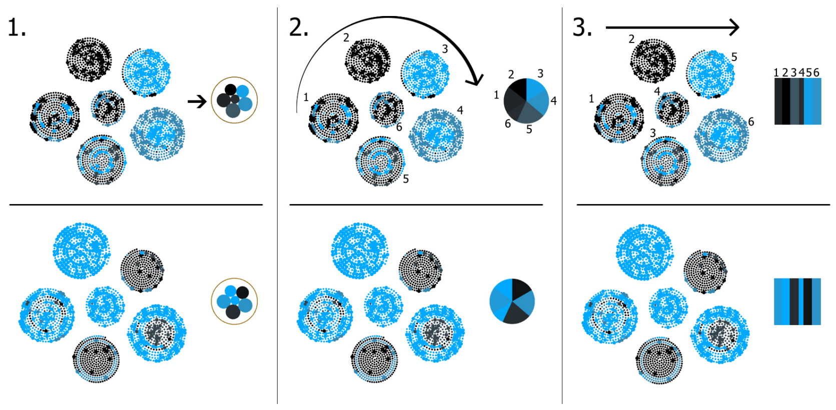


Figure 7.12: Three glyph representations of a network (1: miniature, 2: circular, 3: bars) at two different time points (top and bottom) used in the model tests performed during the third round of user testing.

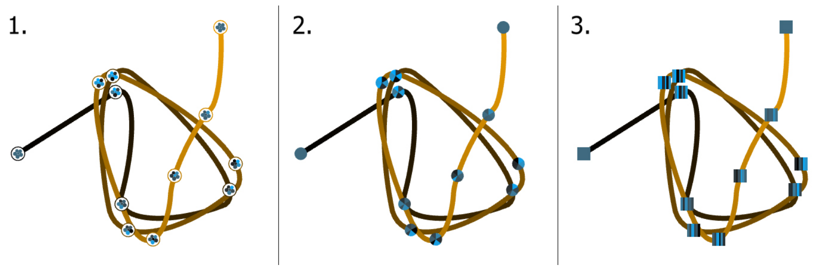


Figure 7.13: Example of an application of the three types of glyphs (1: miniature, 2: circular, 3: bars) on a time curve, used in the model tests performed during the third round of user testing.

M2.3 - "I think this glyph is legible in a small size."

M2.4 - "I am able to easily distinguish similarities and differences between time points."

The test is then finalized with another open-ended question concerning the participant's preferences:

M3 - "Do you have any comments regarding the visualizations show, or personal preferences regarding the glyphs in each size?"

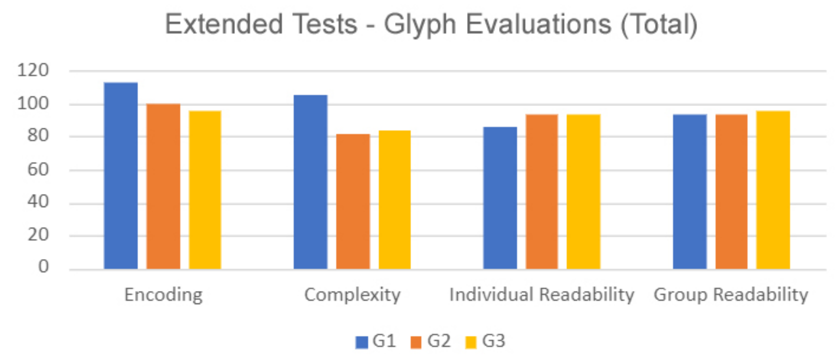


Figure 7.14: Sum of score values given by all participants for each of the attributes (from M2.1 to M2.4) of each of the three glyphs (G1, G2 & G3).

— *Interface Results Analysis*

In analyzing the results, we must take into account the background of the participants and their level of experience with visualization tools. Among those who participated, we can compare between the performances between those with a visualization background and those with a computational biology background in order to understand how different types of users are able to interact with and interpret our visualization models.

Similarly to the previous tests, the first set of tasks required users to simply load the network (T1.1) and time-series (T1.2) datasets. However, unlike in the previous tests, there were no users that selected the wrong type of loading in neither T1.1 or T1.2, which may have resulted from separating the types of loading into different dropdowns. To some extent, we can also compare the average time that it took users to load these datasets (Figure 7.5) with the previous tests (Figure 7.7) as the tasks are very similar: in the previous tests, without considering failed tasks and outliers, users took an average of 29 seconds to load network data and 25 seconds to load time-series data; in the current tests, users with a background in computational biology took an average of 25 seconds to load network data and 17 seconds to load time-series data.

After loading data, participants were directed to the data table panel to resolve the next set of tasks, starting with two tasks that were once again similar to those performed in the previous tests. Regardless of experience with visualization tools, participants did not show appear to have significant problems with T2.1. However, 7 participants did have difficulties in identifying the tabs of selected nodes on the data table as a source of information for T2.2, likely as it is a system that is not common to interactive tables.

T2.3 and T2.5 required users to utilize keyboard buttons to have multiple rows selected simultaneously in the table. The former only required three rows to be selected, which could be achieved by simply holding "CTRL" while clicking each one. Despite some difficulties from two users, all others were eventually able to perform these selections. In T2.5, users were prompted to select 17 subsequent nodes, which directed them to use "SHIFT" to select every row between a first and last selection, rather than selecting each individual row while holding "CTRL". However, many users had difficulties in completing this task, with four users not being able to complete it without help. This revealed several usability flaws that could be attributed to two factors: the keyboard shortcuts were a simplified version of the file selection in the Windows operating system and were missing some actions that users were used to, and, by being based off this operating system, we were not taking into account users from other operating systems. To this end, we resolved the main issues that we identified, such as the "SHIFT" button no longer needing to be held down between two selections, and mapping keyboard shortcuts according to the target operating system for each version of CroP.

In T2.4, participants were asked to erase the nodes they had selected by filtering them out, requiring them to move out of the data table and use the options sidebar. While all except one participant were able to discover the function and finish the task, it is of note that users without a visualization background took an average of 80% longer to complete this task than the remaining participants. T2.6 only required users to deselect their previous selections, which should be achievable with a click outside of the panel's area. Once again, participants without a visualization background had more difficulties in finding where to click, but this allowed us to identify more areas that users would feel intuitive to clicked to undo selections. For instance, 4 participants clicked the "ESC" key at this point resulting in the application shutting down due to the default shortcuts of programs made in Processing. This was then resolved by rebinding the key.

In T3, the participants were asked to cluster the data (T3.1) and discover a groups of nodes particular temporal pattern (T3.2). In T3.1, none of the participants presented any issues in clustering the data with the requested parameters. However, T3.2 can be considered one of the most complex tasks in the test as it required the participants to change the current time point to 16H, identify the clusters with the lowest expression, and select both. This originated the largest range of values in task completion time, where 8 participants were able be to complete it in less than 30 seconds, while 5 others required more than 90 seconds. Despite this, only 4 participants required any kind of assistance, and only 1 of those was not able to complete the task due to not being able to change the current time point. In T3.3, similarly to T2.6, users were asked cancel the selection of the previous clusters. This time, all users simply clicked outside of the clusters to deselect them, as intended, with the exception of one participant which instead tried to use the data table panel.

T4 directed users to perform tasks in the Time Curve panel, but required them to first change the type of data mapping to "tendency" (T4.1), but no participants showed any significant difficulties in locating the Data Mapping option and changing it. Participants were also able to easily apply the forces layout in the time curve panel (T4.2), but interpreting the resulting visualization in T4.3 proved to be more complicated. In total, 5 participants were unable to identify the relationships between time points in the resulting time curve (4 of a computational biology background). The same difficulties could be observed when analyzing the values time curve in T4.4, where 3 users were not able to detect any pattern that would indicate when the largest changes in values occurred. However, despite these difficulties, most participants were able to complete both of the previous tasks. Of those who succeeded, the average completion time for both tasks for those with a visualization background was 53 seconds, and 86 seconds for those with a computational biology background.

Finally, T5 asked participants to create a new panel and position it next to another, allowing us to understand how users would interact with the windows to organize the environment within its grid. In general, participants did not show any significant difficulties in managing the panels, with the only observed issues being the location of the option to create new panels and the initial learning curve of manipulating the panels, although both of these issues were either minor or rare.

Overall, participants with a bioinformatics background took an average of 24% longer to resolve tasks (Figure 7.7) than those with a visualization background (Figure 7.8), when excluding outliers. However, performing tests with this diverse group of users helped us detect and correct prominent usability problems, while also considering new options. For instance, when interacting with a minimized section in the options sidebar, many of the users first tried to click the title of a section to open it before using the plus button. As this would allow users to more easily access or hide sections of the interface due to having a wider clickable area, we changed titles to also toggle sections open.

Regarding the general feedback section that followed the interface tasks, where participants rated statements about their experiences from 1 (Strongly Disagree) to 5 (Strongly Agree), the responses were overall positive and similar across those from different backgrounds, as seen in Figure 7.9. In general, participants thought the tool was easy to use with an average score of 4.4, while the ease of interpreting the data representations got an average score of 3.8. These difficulties in interpreting some of the visualization models was also notable when participants were asked if they needed more time to learn how to use the tool, which got an average of 3. Based on these scores and obtained feedback, CroP was considered to be generally intuitive and easy to pick up, but they also reflected the inherent learning curve to utilize all of its functions, particularly the time curve panel. Despite this, participants showed a positive interest in the tool in regards to potentially using it in the future, giving that statement an average score of 4.4.

— *Model Results Analysis*

In the model tests, participants were first asked to pick which behaviors they could interpret from three different time curves without knowledge of the original dataset (Figure 7.10), and the distribution of these choices is represented in Figure 7.11. As in the previous test, the first time curve is that of the HIV-1 Virus dataset, which depicts multiple cycles with strong variations of values between time points. Over 88% of the participants were able to associate the circular pattern to a cyclical tendency, and about 73% interpreted these variations as being strong. However, 50% also discerned small variations between time points, but this may have been due to the close proximity of non-sequential time points, according to feedback.

The second time curve depicted a dataset with a constant tendency but with fluctuating intensities that resulted in both moments of high and low variation. About 23% of the participants interpreted these shifts in variation as a cycle, likely due to the time curve going up and down over time, although this does not actually translate to the data regressing to a previous state. Additionally, only 53% of the participants discerned the existence of both moments of strong and low variations of values. The third time curve represents the Malaria Virus dataset, characterized by a single overall cycle with mostly constant low shifts in variation, and a few stronger shifts. While 96% of participants were able to identify the small shifts in variation, only about 35% interpreted the overall shape of the time curve as representing a cycle.

Following this, we surveyed participants on different types of glyphs as to either validate our glyphs or guide the creation of a new one. Overall, the reception to the presented glyphs was mixed as the average scores attributed to each statement range between 3 and 4, with the only exception being M2.1.1 which received an average score of 4.4, as shown in Figure 7.14. While this shows that the majority of participants agreed that the miniature network is the most intuitive glyph of those presented, the remaining scores did not show a clear preference, regardless of whether the glyph was presented in a small or large size. We can, however, note that the range of scores was higher for the evaluations of the bar graph glyph in comparison to the circular graph glyph. Additionally, at the end of the survey about

54% of the participants expressed a preference for the miniature network as a large glyph, followed by 23% preferring the circular graph glyph, while the remaining either preferred the bar chart or had no preference. Regarding the glyphs in smaller size, the support for all three glyphs was once again balanced, although the bar chart was overall the least favorite among the participants that provided feedback.

Despite the mixed feedback, we were able to take the following conclusions: while the miniature network was the easiest to comprehend for participants, it appeared to be more difficult to comprehend at a smaller size unless the graphical elements were enhanced; regarding the other glyphs, the bar chart glyph was considered to be easier to follow and compare due to its order of elements, while other participants preferred the circular chart glyph due to its simplicity and circular shape that matches the original nodes, unlike the former.

7.3 DISCUSSION

While in the experimentation chapter we were able to test the ability of CroP as a tool for representing and analyzing different types of data, it is only through user validation that we can evaluate its usability and the comprehensibility of the visualization it creates. It is in this respect that the surveys and tests that were performed focused particularly on the time curve model, being the most uncommon visualization model in CroP as it represents the relationships between time points through the distortion of a timeline.

Our conclusions regarding the interpretation of the time curves visualizations were generally consistent throughout all the tests, where we must acknowledge the existence of a learning curve that is inherent to a model that creates abstracted data visualizations. For instance, in the preliminary survey, nearly half the participants misinterpreted the first time curve (Figure 7.2.1) despite having correctly identified the cyclical behavior previously. However, over 80% of participants were able to identify the behavior that followed it (Figure 7.2.2), and throughout the rest survey most participants correctly answered all of the questions through the use of time curves, with half of these problems being answered correctly by over 90% of the participants.

In the final model tests, some participants also misinterpreted weak variations in the first time curve (Figure 7.10.1) and the existence of cycles in the second time curve (Figure 7.10.2), which may have been a result of trying to compare these visualizations to common linear graphs. Additionally, only a quarter of the participants identified a cycle in the third time curve (Figure 7.10.3), although a contributing factor may have been that the circle was not closed. However, despite these difficulties, even participants with little experience with data visualization were able to discern prominent behaviors, significant events and trends. As these visualizations were created from datasets containing thousands of data points, these results show that it is possible for CroP to create visualizations that comprehensibly represent complex datasets and promote the discovery of meaningful patterns.

In what regards to testing CroP's usability, the interface tests contributed towards answering some our research questions, namely how CMVs facilitate the exploration of multivariate datasets and whether visualization and data analysis approaches promote the discovery of meaningful relationships and patterns. As the initial interface tests were performed by a small group of participants with low experience with visualization tools and data analysis, there existed exceptional difficulties with concepts such as clustering. However, despite their inexperience, most users were able to navigate the tool and perform the indicated tasks, including loading data, brushing nodes, applying filters and analyzing data from their visual elements.

These tasks were adjusted and expanded in the extended tests, further testing the ability of users to use CroP to analyze data, now with participants from various fields of study. While those with low experience with visualization tools had on average the longest task completion times, all of them were eventually able to solve all of the data analysis tasks involving the data panel and only 4 out of the 26 participants presented any significant difficulties with the tasks involving network clustering and the time curves. This showed how different types of users were able to utilize the available tools to identify elements or groups with specific proprieties, as well as analyze of one dataset across multiple visualization panels to discover different types of relationships.

Lastly, in what pertains to our final research question, we can overview the results of the validation tests in relation to the considerations needed to be taken to accommodate users with varying levels of experience into CroP. Many of the preemptive measures that were taken with regard to usability were based on the fluid interaction principles, with particular regard to error prevention: failing to load a dataset will return an appropriate error message and list of the lines containing errors when appropriate; buttons and sliders are clearly labeled, and those that describe uncommon features are accompanied a "?" icon that contains a relevant description; actions performed on either the visualization models or the user interface give immediate visual feedback. However, it was through validation with a wide variety of individuals that we were able to obtain new insight into the development of CroP and resolve issues that were not initially anticipated. In this respect, we can review several meaningful usability problems that were identified and resolved:

- In the initial interface tests, CroP utilized a single dropdown for loading datasets from which users could choose between all the supported types of data. After observing multiple occurrences of users choosing the wrong type of loading, the available options were categorized into different dropdowns, one for each type of data. This revision was performed before the extended interface tests, during which the average time in loading time-series datasets was reduced and participants did not present significant issues in resolving the task.

- When navigating the sidebar, users attempted to open and close the collapsible sections by clicking their title instead of the intended button. This was not reported as an issue, as users simply utilized the button afterwards, but we were able to recognize that having a wider area for executing this action would facilitate navigation. In this regard, the titles were updated with button proprieties as to open and close their respective sections.

- Some users pressed the "ESC" key with the intent of cancelling node selections, resulting in CroP shutting down as that is the default action for programs developed in Processing. While the key was originally overlooked, its action was changed to reset any current selections.

- Multiple interactions with sliders and visualization models utilize dragging, which was initially considered as a different action from clicking in order to prevent the unintentional activation of elements dragged over by the mouse. However, this led to an issue that was only identified by user testing: users unaccustomed to high mouse sensitivity would consistently drag the mouse while clicking buttons, cancelling the action. This was resolved by selecting dragged elements as long as the mouse did not leave the area of that element.

- Hovering time points in the time slider originally did not highlight their respective nodes in the time curve as to not interfere with time point selections, but users were observed having difficulties when utilizing the slider to pinpoint specific time points. To resolve these issues, visual highlights were added to nodes hovered using the timeline without removing current selections, either when scrolling or when clicking on it.

- Originally, values were mapped using a single color palette, from black to bright blue, representing the lowest values to the highest respectively. Its objective was to highlight high values with bright colors and represent low values with the darker, less saturated tones. However, this originated two issues: middling values were difficult to distinguish and compare and some users reported interpreting black as representing the highest values, due to its contrast. Despite these issues only being reported by a very low percentage of users, it ultimately lead to the modification of CroP's simplest palettes and the addition of several new options.

- Another similar issue was brought up by a user, commenting on the lack of information regarding what the colors represent. While the numerical values can be visualized in the data table panel, such context is missing from the network panel. This resulted in the addition of a text box in the bottom-left corner of the network panel containing textual and numerical information on any hovered node or cluster, including their current values and respective colors, based on the selected value mapping.

In conclusion, working with a wide range of participants from different fields of study served to quickly identify both minor and general usability issues with the implemented functionalities, resulting in continuous improvements throughout the development of the tool. Moreover, even users lacking experience in data visualization were able to navigate the tool and solve tasks, while also generally understand the time curve visualizations and interpret its patterns, despite the clear existence of a learning curve. While some participants expressed the need for additional time to get accustomed to the tool, overall feedback was consistently positive in regard to CroP's usability and the comprehensibility of its models.

8 CONCLUSION

The continuous advancements in the field of Information Visualization, along with those in HCI and Data Analysis, resulted in the development of new visualization tools that can process, represent and analyze large and complex datasets. As we stated in the research hypothesis, interactive visualization can be a powerful tool for data analysis as it provides the means to represent high-dimensional datasets comprehensibly. This creates an environment that helps explore such datasets and gives users the tools needed to discover new meaningful information and extract new knowledge.

Throughout this thesis, we addressed the research questions that we proposed which entailed various related and relevant subjects: the representation of complex multivariate data, the discovery of patterns in high-dimensional data, the management of visual complexity through visual abstractions, the discovery of new insights through multiple coordinated views, and the user experience with a data visualization tool. The subjects of these questions were present throughout this document and served to describe both the challenges and motivation behind the development of CroP, a visualization tool with a CMV framework, integrated with dynamic visualization models and diverse novel visual and interactive approaches. This development focused on the representation and analysis of temporal and relational data, in particular those from biological fields of study as they are often characterized as complex, due to their volume and high-dimensionality.

In the *State of the Art* chapter, we discussed the management of heterogeneous data through diverse visualization approaches for relational and multivariate datasets, including reviewing layouts that employ simultaneous views and principles for encoding and perception to improve readability and data fidelity. Furthermore, in regards to the expected challenges in representing complex datasets, we also discussed the management of visual complexity both through data analysis methods, such as dimensionality reduction and clustering, and through data aggregation and visual abstractions. In what concerns the development of interactive visualizations, we reviewed methods that involve the navigation through multiple levels of detail, querying data, and brushing elements, of different proprieties and also coordinated between different views. Regarding usability, we also surveyed interface options for custom parameters, providing users with additional control over visualizations, as well as error prevention measures.

The developed visualization tool, CroP, was initially presented in the *Overview* chapter, where we reviewed all of its main functionalities within its CMV framework. The user interface is comprised of an options sidebar and a modular environment where multiple visualization panels can be placed and resized within a grid, building a workspace that best suits the dataset being analyzed. These panels can be used to visualize relational, temporal and multivariate datasets at different levels of detail, providing users with several types of layouts and tools to sort data points and variables in order to discover patterns of relationships.

In what concerns the visualization of time-series in particular, we presented our implementation of the time curves layout and demonstrated its ability to represent different types of behaviors in time-series datasets. We complemented this model with Time Paths, a parameter-based layout that dynamically transforms time curve visualizations to represent temporal behaviors with varying levels of sensitivity to shifts in the data. By increasing the level of smoothing, the layout can reduce visual clutter while also promoting the representation of predominant behaviors. Additionally, we can more easily control the visual proprieties of edges, which allows for smoother transitions between time points that more clearly represent the flow of time, including the creation of animated edges. The tool also features supporting visualization elements aimed at facilitating the identification of specific moments and behaviors in the timeline, particularly when dealing with complex time curve visualizations. Specifically, we implemented glyphs that represent the dataset at each stage, as well as a lens-based area brush that can be used to search across groups of nodes and highlight those with similar proprieties.

The *Workflow Analysis* chapter followed this introduction to CroP's functionalities with a description of how they can be used cooperatively to analyze different types of datasets. For relational data, we presented how the network panel's layouts can position data points based on their relationships, while the data table can be used in parallel to dig-down, and identify additional correlations between related nodes and their attributes. While temporal and non-temporal variables can be analyzed through the layouts in the time curve and multivariate view panels, respectively, we also showed how patterns could further explored and studied through the mouse lens and the network panel. Additionally, while interaction and animation could also be used to discover and better understand these patterns of behavior, the timeline graph also supports the exploration of complex time curve visualizations, while also providing a simple visualization of the general shifts of data in static environments, such as screenshots.

CroP's functionalities and visualization models were demonstrated in the *Experimentation* chapter, where we represented and analyzed datasets with various levels of complexity. As discussed, the network and data table panels were able to present the datasets with different levels of detail, while providing the means to create groups of data points with similar proprieties and explore the composition of these groups. In this respect, the implemented layouts and clustering algorithms helped in understanding the structure of each dataset, highlighting the diversity of patterns of variables while isolating potential noise. Moreover, the models developed for the time curve panel were capable of representing different types of behaviors over time, across both single time-series and large datasets, highlighting periods of stagnation and cycles, as well as events that mark significant shifts of values. Using the glyphs, mouse lens and timeline graphs, we were able to dig-down into these patterns and identify nature of their respective behaviors and the nodes at their origin.

While previous chapters presented CroP's inputs and outputs primarily from a user perspective, in the *Framework* chapter we reviewed the development of the tool and the implementation of its functionalities, such as error prevention measures, visual enhancements and optimization. This includes descriptions of the validation of processed datasets, of panel management in dynamically adapting the workspace to any changes, and of every layout that was implemented, including the considerations regarding the chosen visual variables. Moreover, we reviewed the advantages and disadvantages of all the integrated clustering algorithms, as well as the motivation for their inclusion.

Finally, the *Validation* chapter describes the surveys and usability tests that were conducted to evaluate the tool, its functionalities and visualization models, performed by participants from various fields of study with different levels of experience with data visualization. The model tests focused primarily on the time curves, testing their comprehensibility in representing various temporal behaviors. While these confirmed that the time curve model possesses a significant learning curve, even users with low knowledge of visualization were able to interpret patterns, both on visualizations created from simple and complex datasets, as well as use them to solve tasks.

The usability tests required participants to interact with the tool, navigate visualizations of a dataset and perform several tasks of varying complexity. As expected, participants with less experience with visualization tools presented longer times in completing tasks, particularly with those involving the identification of patterns in clusters and the time curve visualization. However, despite their inexperience, most participants were able to complete most tasks without issues, showing the ability to navigate the tool's multiple views, identify data points and behaviors, as well as managing the workspace. In general, the tests showed that a majority of users were able to use CroP, regardless of their background and in spite of the amount of time spent with the tool, with feedback being overall positive in regards to the tool's usability and models. Moreover, feedback was employed in the tool's iterative development and we were able to solve most of the detected issues, including that of visualization tests served which was used in the development of glyphs, visual feedback and addition of color palettes.

Regarding future work, the CMV framework facilitates the implementation of new visualization models and functionalities that target specific problems, while allowing for the addition of coordinated actions with existing models when applicable. As such, CroP can more easily be developed to tackle more domain-specific problems, particularly within the field of Bioinformatics. For instance, the development of additional relational visualization models and edge-based functionalities would contribute towards the representation, comparison and analysis of biological pathways. Moreover, the integration of additional databases, similar to the GO, could help provide additional context and proprieties to various biological elements, which could then be analyzed through CroP's existing tools.

In what concerns the general visualization of data, new layouts can also be integrated to explore datasets in new ways, such as a 3D plots to view networks through different angles, or additional methods to improve the representation of the relationships between data points, which could involve the implementation of novel machine learning methods for multidimensional scaling. Such developments could be employed to help surpass the limitations of the current models, such as managing the complexity caused by edge overlap in particularly complex time curves.

In closing, we would highlight the dissemination of the contributions produced throughout this thesis which resulted in publications across international, peer-reviewed conferences and journals within the fields of both information science and bioinformatics:

- *Conference Articles*
  - António Cruz, Joel P. Arrais, and Penousal Machado. Interactive Network Visualization of Gene Expression Time-Series Data. In 22nd International Conference Information Visualisation (IV), pp. 574-580. IEEE, 2018. [28]
  - António Cruz, Joel P. Arrais, and Penousal Machado. Exploring Time-Series Through Force-Directed Timelines. In 24th International Conference Information Visualisation (IV), pp. 328-335. IEEE, 2020. [30]
- *Journal Articles*
  - António Cruz, Joel P. Arrais, and Penousal Machado. Interactive and coordinated visualization approaches for biological data analysis. Briefings in Bioinformatics, v. 20, no. 4, pp. 1513–1523, 2019. [29]
  - António Cruz, Penousal Machado, and Joel P. Arrais. CroP — Coordinated Panel visualization for biological networks analysis. Bioinformatics, v. 36, no. 4, pp. 1298-1299, 2020. [31]

– António Cruz, Joel P. Arrais, and Penousal Machado. Force-Directed Timelines: Visualizing & Exploring Temporal Patterns. Big Data Research, v. 27, pp. 100291, 2022. [33]
– António Cruz, Joel P. Arrais, and Penousal Machado. Multivariate Data Exploration Through Coordinated Views. IEEE Access, 2022. [**cruz2022access**]

As we continue to develop CroP and make it more accessible, we also aim creation of a web-based version of the tool. While browser tools may present more limitations in their ability to process and display data, such a platform allows for easier access to up-to-date databases. Fundamentally, we will aim to continue exploring the representation of complex multivariate datasets with comprehensible visualizations, a wide-range of analysis functionalities and intuitive navigation.

BIBLIOGRAPHY

[1] John Aach and George M Church. "Aligning gene expression time series with time warping algorithms." In: *Bioinformatics* 17.6 (2001), pp. 495–508.

[2] James Abello, Steffen Hadlak, Heidrun Schumann, and Hans-Jorg Schulz. "A Modular Degree-of-Interest Specification for the Visual Analysis of Large Dynamic Networks." In: *IEEE Transactions on Visualization and Computer Graphics* 20.3 (Mar. 2014), pp. 337–350. ISSN: 1077-2626. DOI: 10.1109/TVCG.2013.109. URL: http://dx.doi.org/10.1109/TVCG.2013.109.

[3] Saeed Aghabozorgi, Ali Seyed Shirkhorshidi, and Teh Ying Wah. "Time-series clustering–a decade review." In: *Information Systems* 53 (2015), pp. 16–38.

[4] István Albert and Réka Albert. "Conserved Network Motifs Allow Protein–protein Interaction Prediction." In: *Bioinformatics* 20.18 (Dec. 2004), pp. 3346–3352. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/bth402. URL: http://dx.doi.org/10.1093/bioinformatics/bth402.

[5] Michael R Anderberg. *Cluster analysis for applications. Monographs and textbooks on probability and mathematical statistics*. 1973.

[6] Simon Anders and Wolfgang Huber. "Differential expression of RNA-Seq data at the gene level–the DESeq package." In: *Heidelberg, Germany: European Molecular Biology Laboratory (EMBL)* 10 (2012), f1000–research.

[7] Gennady Andrienko and Natalia Andrienko. "Coordinated multiple views: a critical view." In: *Coordinated and Multiple Views in Exploratory Visualization, 2007. CMV'07. Fifth International Conference on*. IEEE. 2007, pp. 72–74. DOI: 10.1109/CMV.2007.4.

[8] Paolo Angelelli, Steffen Oeltze, Judit Haász, Cagatay Turkay, Erlend Hodneland, Arvid Lundervold, Astri J Lundervold, Bernhard Preim, and Helwig Hauser. "Interactive visual analysis of heterogeneous cohort-study data." In: *IEEE computer graphics and applications* 34.5 (2014), pp. 70–82. DOI: 10.1109/MCG.2014.40.

[9] Mihael Ankerst, Markus M Breunig, Hans-Peter Kriegel, and Jörg Sander. "OPTICS: Ordering points to identify the clustering structure." In: *ACM Sigmod record* 28.2 (1999), pp. 49–60.

[10] Benjamin Bach, Conglei Shi, Nicolas Heulot, Tara Madhyastha, Tom Grabowski, and Pierre Dragicevic. "Time curves: Folding time to visualize patterns of temporal evolution in data." In: *IEEE transactions on visualization and computer graphics* 22.1 (2016), pp. 559–568. DOI: 10.1109/TVCG.2015.2467851.

[11] Michael Baitaluk, Mayya Sedova, Animesh Ray, and Amarnath Gupta. "BiologicalNetworks: visualization and analysis tool for systems biology." In: *Nucleic acids research* 34.suppl_2 (2006), W466–W471.

[12] Aaron Barsky, Tamara Munzner, Jennifer Gardy, and Robert Kincaid. "Cerebral: Visualizing Multiple Experimental Conditions on a Graph with Biological Context." In: *IEEE Transactions on Visualization and Computer Graphics* 14.6 (Nov. 2008), pp. 1253–1260. ISSN: 1077-2626. DOI: 10.1109/TVCG.2008.117. URL: http://dx.doi.org/10.1109/TVCG.2008.117.

[13] Mathieu Bastian, Sebastien Heymann, and Mathieu Jacomy. "Gephi: an open source software for exploring and manipulating networks." In: *Proceedings of the international AAAI conference on web and social media*. Vol. 3. 2009, pp. 361–362.

[14] Fabian Beck, Michael Burch, Stephan Diehl, and Daniel Weiskopf. "A Taxonomy and Survey of Dynamic Graph Visualization." In: *Computer Graphics Forum* 36.1 (2017), pp. 133–159. ISSN: 1467-8659. DOI: 10.1111/cgf.12791. URL: http://dx.doi.org/10.1111/cgf.12791.

[15] Ben Bederson and Jon Meyer. "Implementing a zooming User Interface: experience building Pad++." In: *Software: Practice and Experience* 28.10 (1998), pp. 1101–1135.

[16] Michael Behrisch, James Davey, Fabian Fischer, Olivier Thonnard, Tobias Schreck, Daniel Keim, and Jörn Kohlhammer. "Visual Analysis of Sets of Heterogeneous Matrices Using Projection-Based Distance Functions and Semantic Zoom." In: *Computer Graphics Forum* 33.3 (2014), pp. 411–420. ISSN: 1467-8659. DOI: 10.1111/cgf.12397. URL: http://dx.doi.org/10.1111/cgf.12397.

[17] Amir Ben-Dor and Zohar Yakhini. "Clustering Gene Expression Patterns." In: *Proceedings of the Third Annual International Conference on Computational Molecular Biology*. RECOMB '99. Lyon, France: ACM, 1999, pp. 33–42. ISBN: 1-58113-069-4. DOI: 10.1145/299432.299448. URL: http://doi.acm.org/10.1145/299432.299448.

[18] Jacques Bertin. *Semiology of Graphics: Diagrams, Networks, Maps*. ISBN: 978-1-589-48261-6. Redlands: Esri Press, 2010. ISBN: 978-1-589-48261-6.

[19] Krithika Bhuvaneshwar, Anas Belouali, Varun Singh, Robert M Johnson, Lei Song, Adil Alaoui, Michael A Harris, Robert Clarke, Louis M Weiner, Yuriy Gusev, et al. "G-DOC Plus–an integrative bioinformatics platform for precision medicine." In: *BMC bioinformatics* 17.1 (2016), p. 193. DOI: 10.1186/s12859-016-1010-0.

[20] Luc Bijnens, P Lewi, H Goehlmann, Geert Molenberghs, and Luc Wouters. "Spectral map analysis-a method to analyze gene expression data." In: *2003 Proceedings of the American Statistical Association, Biopharmaceutical Section [CDROM], Alexandria, VA: American Statistical Association* (2004), pp. 553–559.

[21] Zbynek Bozdech, Manuel Llinás, Brian Lee Pulliam, Edith D Wong, Jingchun Zhu, and Joseph L DeRisi. "The transcriptome of the intraerythrocytic developmental cycle of Plasmodium falciparum." In: *PLoS Biol* 1.1 (2003), e5. DOI: doi:10.1371/journal.pbio.0000005.

[22] Matthew Brehmer, Bongshin Lee, Benjamin Bach, Nathalie Henry Riche, and Tamara Munzner. "Timelines revisited: A design space and considerations for expressive storytelling." In: *IEEE transactions on visualization and computer graphics* 23.9 (2017), pp. 2151–2164. DOI: 10.1109/TVCG.2016.2614803.

[23] Peter J Brockwell and Richard A Davis. *Time Series: Theory and Methods*. Berlin, Heidelberg: Springer-Verlag, 1986. ISBN: 0-387-96406-1.

[24] Hong Chen. "Compound Brushing Explained." In: *Information Visualization* 3.2 (2004), pp. 96–108. DOI: 10.1057/palgrave.ivs.9500068. eprint: https://doi.org/10.1057/palgrave.ivs.9500068. URL: https://doi.org/10.1057/palgrave.ivs.9500068.

[25] Herman Chernoff. "The Use of Faces to Represent Points in k-Dimensional Space Graphically." In: *Journal of the American Statistical Association* 68.342 (1973), pp. 361–368. DOI: 10.1080/01621459.1973.10482434. eprint: http://www.tandfonline.com/doi/pdf/10.1080/01621459.1973.10482434. URL: http://www.tandfonline.com/doi/abs/10.1080/01621459.1973.10482434.

[26] Gene Ontology Consortium et al. "The gene ontology project in 2008." In: *Nucleic acids research* 36.suppl 1 (2008), pp. D440–D444.

[27] Paul Craig, Alan Cannon, Robert Kukla, and Jessie Kennedy. "MaTSE: The microarray time-series explorer." In: *Biological Data Visualization (BioVis), 2012 IEEE Symposium on*. IEEE. 2012, pp. 41–48. DOI: 10.1109/BioVis.2012.6378591.

[28] António Cruz, Joel P Arrais, and Penousal Machado. "Interactive Network Visualization of Gene Expression Time-Series Data." In: *2018 22nd International Conference Information Visualisation (IV)*. IEEE. 2018, pp. 574–580.

[29] António Cruz, Joel P Arrais, and Penousal Machado. "Interactive and coordinated visualization approaches for biological data analysis." In: *Briefings in bioinformatics* 20.4 (2019), pp. 1513–1523.

[30] António Cruz, Joel P Arrais, and Penousal Machado. "Exploring Time-Series Through Force-Directed Timelines." In: *2020 24th International Conference Information Visualisation (IV)*. IEEE. 2020, pp. 328–335.

[31] Antònio Cruz, Penousal Machado, and Joel P Arrais. "CroP - Coordinated Panel visualization for biological networks analysis." In: *Bioinformatics* (2019). btz688. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btz688. URL: https://doi.org/10.1093/bioinformatics/btz688.

[32] António Cruz, Penousal Machado, Filipe Assunção, and António Leitão. "ELICIT: Evolutionary Computation Visualization." In: *Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation*. GECCO Companion '15. Madrid, Spain: ACM, 2015, pp. 949–956. ISBN: 978-1-4503-3488-4. DOI: 10.1145/2739482.2768443. URL: http://doi.acm.org/10.1145/2739482.2768443.

[33] António Cruz, Joel P. Arrais, and Penousal Machado. "Force-Directed Timelines: Visualizing & Exploring Temporal Patterns." In: *Big Data Research* 27 (2022), p. 100291. ISSN: 2214-5796. DOI: https://doi.org/10.1016/j.bdr.2021.100291. URL: https://www.sciencedirect.com/science/article/pii/S2214579621001088.

[34] Ross E Curtis, Amos Yuen, Le Song, Anuj Goyal, and Eric P Xing. "TVNViewer: an interactive visualization tool for exploring networks that change over time or space." In: *Bioinformatics* 27.13 (2011), pp. 18–80–1881. DOI: 10.1093/bioinformatics/btr273.

[35] Edgar D Coelho, Joel P Arrais, and Jose Luis Oliveira. "From protein-protein interactions to rational drug design: are computational methods up to the challenge?" In: *Current topics in medicinal chemistry* 13.5 (2013), pp. 602–618.

[36] Ben Dalziel, Hui Yang, Rahul Singh, Matthew Gormley, and Susan Fisher. "XMAS: An Experiential Approach for Visualization, Analysis, and Exploration of Time Series Microarray Data." In: *BIRD*. Springer. 2008, pp. 16–31. DOI: 10.1007/978-3-540-70600-7_2.

[37] Aritra Dasgupta, Min Chen, and Robert Kosara. "Conceptualizing visual uncertainty in parallel coordinates." In: *Computer Graphics Forum*. Vol. 31. Wiley Online Library. 2012, pp. 1015–1024.

[38] Hao Ding, Chao Wang, Kun Huang, and Raghu Machiraju. "iGPSe: A visual analytic system for integrative genomic based cancer patient stratification." In: *BMC bioinformatics* 15.1 (2014), p. 203. ISSN: 1471-2105. DOI: 10.1186/1471-2105-15-203. URL: https://doi.org/10.1186/1471-2105-15-203.

[39] Cody Dunne and Ben Shneiderman. "Motif Simplification: Improving Network Visualization Readability with Fan, Connector, and Clique Glyphs." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '13. Paris, France: ACM, 2013, pp. 3247–3256. ISBN: 978-1-4503-1899-0. DOI: 10.1145/2470654.2466444. URL: http://doi.acm.org/10.1145/2470654.2466444.

[40] Michael B. Eisen, Paul T. Spellman, Patrick O. Brown, and David Botstein. "Cluster analysis and display of genome-wide expression patterns." In: *Proceedings of the National Academy of Sciences* 95.25 (1998), pp. 14863–14868. eprint: http://www.pnas.org/content/95/25/14863.full.pdf. URL: http://www.pnas.org/content/95/25/14863.abstract.

[41] Niklas Elmqvist and Jean-Daniel Fekete. "Hierarchical aggregation for information visualization: Overview, techniques, and design guidelines." In: *IEEE Transactions on Visualization and Computer Graphics* 16.3 (2010), pp. 439–454. DOI: 10.1109/TVCG.2009.84.

[42] Niklas Elmqvist, Andrew Vande Moere, Hans-Christian Jetter, Daniel Cernea, Harald Reiterer, and TJ Jankun-Kelly. "Fluid interaction for information visualization." In: *Information Visualization* 10.4 (2011), pp. 327 –340. DOI: 10.1177/1473871611413180. URL: https://doi.org/10.1177/1473871611413180.

[43] Stef van den Elzen, Danny Holten, Jorik Blaas, and Jarke J van Wijk. "Reducing snapshots to points: A visual analytics approach to dynamic network exploration." In: *IEEE transactions on visualization and computer graphics* 22.1 (2016), pp. 1–10. DOI: 10.1109/TVCG.2015.2468078.

[44] Jason Ernst and Ziv Bar-Joseph. "STEM: a tool for the analysis of short time series gene expression data." In: *BMC bioinformatics* 7.1 (2006), p. 191. ISSN: 1471-2105. DOI: 10.1186/1471-2105-7-191. URL: https://doi.org/10.1186/1471-2105-7-191.

[45] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. "A density-based algorithm for discovering clusters in large spatial databases with noise." In: *kdd*. Vol. 96. 1996, pp. 226–231.

[46] Brian Everitt. *Cluster analysis*. Londres: Heinemenn Educational Books, 1977. ISBN: 0-435-82297-7.

[47] A. Fahad, N. Alshatri, Z. Tari, A. Alamri, I. Khalil, A. Y. Zomaya, S. Foufou, and A. Bouras. "A Survey of Clustering Algorithms for Big Data: Taxonomy and Empirical Analysis." In: *IEEE Transactions on Emerging Topics in Computing* 2.3 (Sept. 2014), pp. 267–279. ISSN: 2168-6750. DOI: 10.1109/TETC.2014.2330519.

[48] Philippe Fournier-Viger, Jerry Chun-Wei Lin, Antonio Gomariz, Ted Gueniche, Azadeh Soltani, Zhihong Deng, and Hoang Thanh Lam. "The SPMF open-source data mining library version 2." In: *Joint European conference on machine learning and knowledge discovery in databases*. Springer. 2016, pp. 36–40.

[49] Michael Friendly. "A brief history of data visualization." In: *Handbook of data visualization*. Springer, 2008, pp. 15–56.

[50] Thomas M. J. Fruchterman and Edward M. Reingold. "Graph drawing by force-directed placement." In: *Software: Practice and Experience* 21.11 (1991), pp. 1129–1164. ISSN: 1097-024X. DOI: 10.1002/spe.4380211102. URL: http://dx.doi.org/10.1002/spe.4380211102.

[51] Ben Fry. *Visualizing data: Exploring and explaining data with the processing environment*. " O'Reilly Media, Inc.", 2008.

[52] Akira Funahashi, Yukiko Matsuoka, Akiya Jouraku, Mineo Morohashi, Norihiro Kikuchi, and Hiroaki Kitano. "CellDesigner 3.5: a versatile modeling tool for biochemical networks." In: *Proceedings of the IEEE* 96.8 (2008), pp. 1254–1265. DOI: 10.1109/JPROC.2008.925458.

[53] George W Furnas. "Generalized fisheye views." In: *Acm Sigchi Bulletin* 17.4 (1986), pp. 16–23.

[54] Andreas Gerasch, Daniel Faber, Jan Küntzer, Peter Niermann, Oliver Kohlbacher, Hans-Peter Lenhof, and Michael Kaufmann. "BiNA: A Visual Analytics Tool for Biological Network Data." In: *PLOS ONE* 9.2 (Feb. 2014), pp. 1–11. DOI: 10.1371/journal.pone.0087397. URL: https://doi.org/10.1371/journal.pone.0087397.

[55] Michael Gleicher, Danielle Albers, Rick Walker, Ilir Jusufi, Charles D. Hansen, and Jonathan C. Roberts. "Visual Comparison for Information Visualization." In: *Information Visualization* 10.4 (Oct. 2011), pp. 289–309. ISSN: 1473-8716. DOI: 10.1177/1473871611416549. URL: http://dx.doi.org/10.1177/1473871611416549.

[56] Joana P Gonçalves, Sara C Madeira, and Arlindo L Oliveira. "BiGGEsTS: integrated environment for biclustering analysis of time series gene expression data." In: *BMC research notes* 2.1 (2009), p. 124. ISSN: 1756-0500. DOI: 10.1186/1756-0500-2-124. URL: https://doi.org/10.1186/1756-0500-2-124.

[57] Lisa Graham. "Gestalt theory in interactive media design." In: *Journal of Humanities & Social Sciences* 2.1 (2008).

[58] Anna C. Greene, Kristine A. Giffin, Casey S. Greene, and Jason H. Moore. "Adapting bioinformatics curricula for big data." In: *Briefings in Bioinformatics* 17.1 (2016), pp. 43–50. DOI: 10.1093/bib/bbv018. eprint: /oup/backfile/content_public/journal/bib/17/1/10.1093_bib_bbv018/2/bbv018.pdf. URL: +http://dx.doi.org/10.1093/bib/bbv018.

[59] Isabelle Guyon and André Elisseeff. "An Introduction to Variable and Feature Selection." In: *J. Mach. Learn. Res.* 3 (Mar. 2003), pp. 1157–1182. ISSN: 1532-4435. URL: http://dl.acm.org/citation.cfm?id=944919.944968.

[60] John Gómez et al. "BioJS: an open source JavaScript framework for biological data visualization." In: *Bioinformatics* 29.8 (2013), pp. 1103–1104. DOI: 10.1093/bioinformatics/btt100. URL: +http://dx.doi.org/10.1093/bioinformatics/btt100.

[61] Steffen Hadlak, Heidrun Schumann, and Hans-Jörg Schulz. "A survey of multi-faceted graph visualization." In: *Eurographics Conference on Visualization (EuroVis). The Eurographics Association*. 2015, pp. 1–20. DOI: 10.2312/eurovisstar.20151109.

[62] Mark Harrower and Cynthia A. Brewer. "ColorBrewer.org: An Online Tool for Selecting Colour Schemes for Maps." In: *The Cartographic Journal* 40.1 (June 2003), pp. 27–37. ISSN: 0008-7041. DOI: 10.1179/000870403235002042.

[63] California Health and Human Services Open Data Portal. *Statewide COVID-19 Cases Deaths Tests*. Accessed: 2022-03-01. 2022. URL: https://data.chhs.ca.gov/dataset/covid-19-time-series-metrics-by-county-and-state/resource/046cdd2b-31e5-4d34-9ed3-b48cdbc4be7a.

[64] Jeffrey Heer, Michael Bostock, and Vadim Ogievetsky. "A Tour Through the Visualization Zoo." In: *Queue* 8.5 (May 2010), 20:20–20:30. ISSN: 1542-7730. DOI: 10.1145/1794514.1805128. URL: http://doi.acm.org/10.1145/1794514.1805128.

[65] Julian Heinrich, Corinna Vehlow, Florian Battke, Günter Jäger, Daniel Weiskopf, and Kay Nieselt. "iHAT: interactive hierarchical aggregation table for genetic association data." In: *BMC bioinformatics* 13.8 (2012), S2. ISSN: 1471-2105. DOI: 10.1186/1471-2105-13-S8-S2. URL: https://doi.org/10.1186/1471-2105-13-S8-S2.

[66] Ivan Herman, Guy Melançon, and M Scott Marshall. "Graph visualization and navigation in information visualization: A survey." In: *IEEE Transactions on visualization and computer graphics* 6.1 (2000), pp. 24–43. DOI: 10.1109/2945.841119.

[67] Matthew A Hibbs, Nathaniel C Dirksen, Kai Li, and Olga G Troyanskaya. "Visualization methods for statistical analysis of microarray clusters." In: *BMC bioinformatics* 6.1 (2005), p. 115. ISSN: 1471-2105. DOI: 10.1186/1471-2105-6-115. URL: https://doi.org/10.1186/1471-2105-6-115.

[68] Yuen Ho, Albrecht Gruhler, Adrian Heilbut, Gary D Bader, Lynda Moore, Sally-Lin Adams, Anna Millar, Paul Taylor, Keiryn Bennett, Kelly Boutilier, et al. "Systematic identification of protein complexes in Saccharomyces cerevisiae by mass spectrometry." In: *Nature* 415.6868 (2002), pp. 180–183.

[69] Harry Hochheiser, Eric H Baehrecke, Stephen M Mount, and Ben Shneiderman. "Dynamic querying for pattern identification in microarray and genomic data." In: *Multimedia and Expo, 2003. ICME'03. Proceedings. 2003 International Conference on*. Vol. 3. IEEE. 2003, pp. III–453. DOI: 10.1109/ICME.2003.1221346.

[70] Andreas Holzinger, Markus Plass, Katharina Holzinger, Gloria Cerasela Crisan, Camelia-M Pintea, and Vasile Palade. "A glass-box interactive machine learning approach for solving NP-hard problems with the human-in-the-loop." In: *arXiv preprint arXiv:1708.01104* (2017).

[71] Yifan Hu. "Efficient, high-quality force-directed graph drawing." In: *Mathematica Journal* 10.1 (2005), pp. 37–71.

[72] Zhenjun Hu, Joe Mellor, Jie Wu, Takuji Yamada, Dustin Holloway, and Charles Delisi. "VisANT: Data-integrating visual framework for biological networks and modules." In: *Nucleic acids research* 33 (Aug. 2005), W352–7. DOI: 10.1093/nar/gki431.

[73] Mathieu Jacomy, Tommaso Venturini, Sebastien Heymann, and Mathieu Bastian. "ForceAtlas2, a continuous graph layout algorithm for handy network visualization designed for the Gephi software." In: *PloS one* 9.6 (2014), e98679.

[74] A. K. Jain, M. N. Murty, and P. J. Flynn. "Data Clustering: A Review." In: *ACM Comput. Surv.* 31.3 (Sept. 1999), pp. 264–323. ISSN: 0360-0300. DOI: 10.1145/331499.331504. URL: http://doi.acm.org/10.1145/331499.331504.

[75] Anil K. Jain, Robert P. W. Duin, and Jianchang Mao. "Statistical Pattern Recognition: A Review." In: *IEEE Trans. Pattern Anal. Mach. Intell.* 22.1 (Jan. 2000), pp. 4–37. ISSN: 0162-8828. DOI: 10.1109/34.824819. URL: http://dx.doi.org/10.1109/34.824819.

[76] Siddhartha Jain, Joel Arrais, Narasimhan J Venkatachari, Velpandi Ayyavoo, and Ziv Bar-Joseph. "Reconstructing the temporal progression of HIV-1 immune response pathways." In: *Bioinformatics* 32.12 (2016), pp. i253–i261.

[77] Daxin Jiang, Chun Tang, and Aidong Zhang. "Cluster Analysis for Gene Expression Data: A Survey." In: *IEEE Trans. on Knowl. and Data Eng.* 16.11 (Nov. 2004), pp. 1370–1386. ISSN: 1041-4347. DOI: 10.1109/TKDE.2004.68. URL: http://dx.doi.org/10.1109/TKDE.2004.68.

[78] Ian T Jolliffe. "Principal Component Analysis and Factor Analysis." In: *Principal component analysis*. New York, NY: Springer, 1986, pp. 115–128. ISBN: 978-1-4757-1904-8. DOI: 10.1007/978-1-4757-1904-8_7. URL: https://doi.org/10.1007/978-1-4757-1904-8_7.

[79] G. Joshi-Tope et al. "REACTOME: a knowledgebase of biological pathways." In: *Nucleic Acids Research* 33 (Jan. 2005), pp. 428–432. DOI: 10.1093/nar/gki072.

[80] Eser Kandogan and Ben Shneiderman. "Elastic Windows: evaluation of multi-window operations." In: *Proceedings of the ACM SIGCHI Conference on Human factors in computing systems*. 1997, pp. 250–257.

[81] Natalie Kerracher, Jessie Kennedy, and Kevin Chalmers. "The Design Space of Temporal Graph Visualisation." In: *EuroVis - Short Papers*. Ed. by N. Elmqvist, M. Hlawitschka, and J. Kennedy. The Eurographics Association, 2014. ISBN: 978-3-905674-69-9. DOI: 10.2312/eurovisshort.20141149.

[82] Andreas Kerren, Kostiantyn Kucher, Yuan-Fang Li, and Falk Schreiber. "MDS-based Visual Survey of Biological Data Visualization Techniques." In: *EuroVis 2017 - Posters*. Ed. by Anna Puig Puig and Tobias Isenberg. The Eurographics Association, 2017. ISBN: 978-3-03868-044-4. DOI: 10.2312/eurp.20171175.

[83] Jon Kleinberg. "An impossibility theorem for clustering." In: *Advances in neural information processing systems* (2003), pp. 463–470.

[84] Nobuaki Kono, Kazuharu Arakawa, Ryu Ogawa, Nobuhiro Kido, Kazuki Oshita, Keita Ikegami, Satoshi Tamaki, and Masaru Tomita. "Pathway Projector: Web-Based Zoomable Pathway Browser Using KEGG Atlas and Google Maps API." In: *PLOS ONE* 4.11 (Nov. 2009), pp. 1–10. DOI: 10.1371/journal.pone.0007710. URL: https://doi.org/10.1371/journal.pone.0007710.

[85] Martin Krzywinski, Inanc Birol, Steven JM Jones, and Marco A Marra. "Hive plots—rational approach to visualizing networks." In: *Briefings in Bioinformatics* 13.5 (2012), pp. 627–644. DOI: 10.1093/bib/bbr069. eprint: /oup/backfile/content_public/journal/bib/13/5/10.1093/bib/bbr069/2/bbr069.pdf. URL: +http://dx.doi.org/10.1093/bib/bbr069.

[86] Martin Krzywinski, Jacqueline Schein, Inanc Birol, Joseph Connors, Randy Gascoyne, Doug Horsman, Steven J Jones, and Marco A Marra. "Circos: an information aesthetic for comparative genomics." In: *Genome research* 19.9 (2009), pp. 1639–1645. DOI: 10.1101/gr.092759.

[87] Martina Kutmon, Martijn P. van Iersel, Anwesha Bohler, Thomas Kelder, Nuno Nunes, Alexander R. Pico, and Chris T. Evelo. "PathVisio 3: An Extendable Pathway Analysis Toolbox." In: *PLOS Computational Biology* 11.2 (Feb. 2015), pp. 1–13. DOI: 10.1371/journal.pcbi.1004085. URL: https://doi.org/10.1371/journal.pcbi.1004085.

[88] Martina Kutmon et al. "WikiPathways: capturing the full diversity of pathway knowledge." In: *Nucleic Acids Research* 44.D1 (2016), pp. D–488–D494. DOI: 10.1093/nar/gkv1024. URL: http://dx.doi.org/10.1093/nar/gkv1024.

[89] Jacob Köhler, Jan Baumbach, Jan Taubert, Michael Specht, Andre Skusa, Alexander Rüegg, Chris Rawlings, Paul Verrier, and Stephan Philippi. "Graph-based analysis and visualization of experimental results with ONDEX." In: *Bioinformatics* 22.11 (2006), pp. 1383–1390. DOI: 10.1093/bioinformatics/btl081. URL: +http://dx.doi.org/10.1093/bioinformatics/btl081.

[90] Godfrey N Lance and William Thomas Williams. "A general theory of classificatory sorting strategies: 1. Hierarchical systems." In: *The computer journal* 9.4 (1967), pp. 373–380.

[91] T. von Landesberger, A. Kuijper, T. Schreck, J. Kohlhammer, J.J. van Wijk, J.-D. Fekete, and D.W. Fellner. "Visual Analysis of Large Graphs: State-of-the-Art and Future Research Challenges." In: *Computer Graphics Forum* 30.6 (2011), pp. 1719–1749. ISSN: 1467-8659. DOI: 10.1111/j.1467-8659.2011.01898.x. URL: http://dx.doi.org/10.1111/j.1467-8659.2011.01898.x.

[92] M. Lawrence, Eun-Kyung Lee, D. Cook, H. Hofmann, and E. Wurtele. "exploRase: Exploratory Data Analysis of Systems Biology Data." In: *Coordinated and Multiple Views in Exploratory Visualization, 2006. Proceedings. International Conference on*. July 2006, pp. 14–20. DOI: 10.1109/CMV.2006.7.

[93] Cosmin Lazar, Jonatan Taminau, Stijn Meganck, David Steenhoff, Alain Coletta, Colin Molter, Virginie de Schaetzen, Robin Duque, Hugues Bersini, and Ann Nowe. "A Survey on Filter Techniques for Feature Selection in Gene Expression Microarray Analysis." In: *IEEE/ACM Trans. Comput. Biol. Bioinformatics* 9.4 (July 2012), pp. 1106–1119. ISSN: 1545-5963. DOI: 10.1109/TCBB.2012.33. URL: http://dx.doi.org/10.1109/TCBB.2012.33.

[94] A. Lex, M. Streit, H.-J. Schulz, C. Partl, D. Schmalstieg, P.J. Park, and N. Gehlenborg. "StratomeX: Visual Analysis of Large-Scale Heterogeneous Genomics Data for Cancer Subtype Characterization." In: *Computer Graphics Forum* 31.3pt3 (2012), pp. 1175–1184. ISSN: 1467-8659. DOI: 10.1111/j.1467-8659.2012.03110.x. URL: http://dx.doi.org/10.1111/j.1467-8659.2012.03110.x.

[95] Alexander Lex, Hans-Jorg Schulz, Marc Streit, Christian Partl, and Dieter Schmalstieg. "VisBricks: multiform visualization of large, inhomogeneous data." In: *IEEE Transactions on Visualization and Computer Graphics* 17.12 (2011), pp. 2291–2300. DOI: 10.1109/TVCG.2011.250.

[96] Xiaoqing Liu, Jun Wu, Feiyang Gu, Jie Wang, and Zengyou He. "Discriminative pattern mining and its applications in bioinformatics." In: *Briefings in bioinformatics* 16.5 (2014), pp. 884–900. DOI: 10.1093/bib/bbu042.

[97] Ben D. MacArthur, Alexander Lachmann, Ihor R. Lemischka, and Avi Ma'ayan. "GATE: software for the analysis and visualization of high-dimensional time series expression data." In: *Bioinformatics* 26.1 (2010), pp. 143–144. DOI: 10.1093/bioinformatics/btp628. eprint: /oup/backfile/content_public/journal/bioinformatics/26/1/10.1093_bioinformatics_btp628/2/btp628.pdf. URL: +http://dx.doi.org/10.1093/bioinformatics/btp628.

[98] James MacQueen et al. "Some methods for classification and analysis of multivariate observations." In: *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. Vol. 1. Oakland, CA, USA. 1967, pp. 281–297.

[99] Sara C. Madeira and Arlindo L. Oliveira. "Biclustering Algorithms for Biological Data Analysis: A Survey." In: *IEEE/ACM Trans. Comput. Biol. Bioinformatics* 1.1 (Jan. 2004), pp. 24–45. ISSN: 1545-5963. DOI: 10.1109/TCBB.2004.2. URL: http://dx.doi.org/10.1109/TCBB.2004.2.

[100] Eamonn Maguire, Philippe Rocca-Serra, Susanna-Assunta Sansone, Jim Davies, and Min Chen. "Visual compression of workflow visualizations with automated detection of macro motifs." In: *IEEE transactions on visualization and computer graphics* 19.12 (2013), pp. 2576–2585. DOI: 10.1109/TVCG.2013.225.

[101] Ian C. McDowell, Dinesh Manandhar, Christopher M. Vockley, Amy K. Schmid, Timothy E. Reddy, and Barbara E. Engelhardt. "Clustering gene expression time series data using an infinite Gaussian process mixture model." In: *PLOS Computational Biology* 14.1 (Jan. 2018), pp. 1–27. DOI: 10.1371/journal.pcbi.1005896. URL: https://doi.org/10.1371/journal.pcbi.1005896.

[102] Chen Meng, Oana A. Zeleznik, Gerhard G. Thallinger, Bernhard Kuster, Amin M. Gholami, and Aedín C. Culhane. "Dimension reduction techniques for the integrative analysis of multi-omics data." In: *Briefings in Bioinformatics* 17.4 (2016), pp. 628–641. DOI: 10.1093/bib/bbv108. URL: +http://dx.doi.org/10.1093/bib/bbv108.

[103] M. Meyer, B. Wong, M. Styczynski, T. Munzner, and H. Pfister. "Pathline: A Tool For Comparative Functional Genomics." In: *Computer Graphics Forum* 29.3 (2010), pp. 1043–1052. ISSN: 1467-8659. DOI: 10.1111/j.1467-8659.2009.01710.x. URL: http://dx.doi.org/10.1111/j.1467-8659.2009.01710.x.

[104] Miriah Meyer, Tamara Munzner, Angela DePace, and Hanspeter Pfister. "MulteeSum: a tool for comparative spatial and temporal gene expression data." In: *IEEE transactions on visualization and computer graphics* 16.6 (2010), pp. 908–917. DOI: 10.1109/TVCG.2010.137.

[105] Miriah Meyer, Tamara Munzner, and Hanspeter Pfister. "MizBee: a multiscale synteny browser." In: *IEEE transactions on visualization and computer graphics* 15.6 (2009), pp. 897–904. DOI: 10.1109/TVCG.2009.167.

[106] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. "Network Motifs: Simple Building Blocks of Complex Networks." In: *Science* 298.5594 (2002), pp. 824–827. ISSN: 0036-8075. DOI: 10.1126/science.298.5594.824. URL: http://science.sciencemag.org/content/298/5594/824.

[107] J Minichino. *Recurrent Neural Networks course project: time series prediction and text generation*. Accessed: 2019-12-02. 2017. URL: https://github.com/techfort/aind2-rnn.

[108] Pejman Mohammadi, Sébastien Desfarges, István Bartha, Beda Joos, Nadine Zangger, Miguel Muñoz, Huldrych F. Günthard, Niko Beerenwinkel, Amalio Telenti, and Angela Ciuffi. "24 Hours in the Life of HIV-1 in a T Cell Line." In: *PLOS Pathogens* 9.1 (Jan. 2013), pp. 1–11. DOI: 10.1371/journal.ppat.1003161. URL: https://doi.org/10.1371/journal.ppat.1003161.

[109] Daniel Müllner. "Modern hierarchical, agglomerative clustering algorithms." In: *arXiv preprint arXiv:1109.2378* (2011).

[110] Tamara Munzner, François Guimbretière, Serdar Tasiran, Li Zhang, and Yunhong Zhou. "TreeJuxtaposer: Scalable Tree Comparison Using Focus+Context with Guaranteed Visibility." In: *ACM Trans. Graph.* 22.3 (July 2003), pp. 453–462. ISSN: 0730-0301. DOI: 10.1145/882262.882291. URL: http://doi.acm.org/10.1145/882262.882291.

[111] C. Niederer, H. Stitz, R. Hourieh, F. Grassinger, W. Aigner, and M. Streit. "TACO: Visualizing Changes in Tables Over Time." In: *IEEE Transactions on Visualization and Computer Graphics* PP.99 (2017), pp. 1–1. ISSN: 1077-2626. DOI: 10.1109/TVCG.2017.2745298.

[112] Alexander Nikitin, Sergei Egorov, Nikolai Daraselia, and Ilya Mazo. "Pathway studio—the analysis and navigation of molecular networks." In: *Bioinformatics* 19.16 (2003), pp. 2155–2157.

[113] Steven Noel and Sushil Jajodia. "Managing Attack Graph Complexity Through Visual Hierarchical Aggregation." In: *Proceedings of the 2004 ACM Workshop on Visualization and Data Mining for Computer Security*. VizSEC/DMSEC '04. Washington DC, USA: ACM, 2004, pp. 109–118. ISBN: 1-58113-974-8. DOI: 10.1145/1029208.1029225. URL: http://doi.acm.org/10.1145/1029208.1029225.

[114] Sabrina Nusrat, Theresa A. Harbig, and Nils Gehlenborg. "Tasks, Techniques, and Tools for Genomic Data Visualization." In: *Comput. Graph. Forum*. 2019. DOI: https://doi.org/10.1111/cgf.13727.

[115] C. Partl, S. Gratzl, M. Streit, A. M. Wassermann, H. Pfister, D. Schmalstieg, and A. Lex. "Pathfinder: Visual Analysis of Paths in Graphs." In: *Computer Graphics Forum* 35.3 (2016), pp. 71–80. ISSN: 1467-8659. DOI: 10.1111/cgf.12883. URL: http://dx.doi.org/10.1111/cgf.12883.

[116] C. Partl, A. Lex, M. Streit, D. Kalkofen, K. Kashofer, and D. Schmalstieg. "enRoute: Dynamic path extraction from biological pathway maps for in-depth experimental data analysis." In: *2012 IEEE Symposium on Biological Data Visualization (BioVis)*. Oct. 2012, pp. 107–114. DOI: 10.1109/BioVis.2012.6378600.

[117] C. Partl, A. Lex, M. Streit, H. Strobelt, A. M. Wassermann, H. Pfister, and D. Schmalstieg. "ConTour: Data-Driven Exploration of Multi-Relational Datasets for Drug Discovery." In: *IEEE Transactions on Visualization and Computer Graphics* 20.12 (Dec. 2014), pp. 1883–1892. ISSN: 1077-2626. DOI: 10.1109/TVCG.2014.2346752.

[118] Georgios A Pavlopoulos, Dimitris Malliarakis, Nikolas Papanikolaou, Theodosis Theodosiou, Anton J Enright, and Ioannis Iliopoulos. "Visualizing genome and systems biology: technologies, tools, implementation techniques and trends, past, present and future." In: *GigaScience* 4.1 (2015), p. 38. DOI: 10.1186/s13742-015-0077-2.

[119] Georgios A Pavlopoulos, Theodoros G Soldatos, Adriano Barbosa-Silva, and Reinhard Schneider. "A reference guide for tree analysis and visualization." In: *BioData mining* 3.1 (2010), p. 1. ISSN: 1756-0381. DOI: 10.1186/1756-0381-3-1. URL: https://doi.org/10.1186/1756-0381-3-1.

[120] Ken Perlin and David Fox. "Pad: An Alternative Approach to the Computer Interface." In: *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '93. Anaheim, CA: ACM, 1993, pp. 57–64. ISBN: 0-89791-601-8. DOI: 10.1145/166117.166125. URL: http://doi.acm.org/10.1145/166117.166125.

[121] Tata Pramila, Wei Wu, Shawna Miles, William Stafford Noble, and Linda L Breeden. "The Forkhead transcription factor Hcm1 regulates chromosome segregation genes and fills the S-phase gap in the tran-

scriptional circuitry of the cell cycle." In: *Genes & Development* 20.16 (2006), pp. 2266–2278. DOI: doi:10.1101/gad.1450606.

[122] Adam Price, Adrian Caciula, Cheng Guo, Bohyun Lee, Juliet Morrison, Angela Rasmussen, W Ian Lipkin, and Komal Jain. "DEvis: an R package for aggregation and visualization of differential expression data." In: *BMC bioinformatics* 20.1 (2019), pp. 1–7.

[123] Teresa M. Przytycka, Mona Singh, and Donna K. Slonim. "Toward the dynamic interactome: it's about time." In: *Briefings in Bioinformatics* 11.1 (2010), pp. 15–29. DOI: 10.1093/bib/bbp057. eprint: /oup/backfile/content_public/journal/bib/11/1/10.1093/bib/bbp057/2/bbp057.pdf. URL: +http://dx.doi.org/10.1093/bib/bbp057.

[124] Qlucore. *Omics Explorer*. 2017. URL: http://www.qlucore.com.

[125] Sangeeta Rani and Geeta Sikka. "Recent techniques of clustering of time series data: a survey." In: *International Journal of Computer Applications* 52.15 (2012). DOI: 10.5120/8282-127.

[126] Casey Reas and Ben Fry. "Processing: programming for the media arts." In: *AI & SOCIETY* 20.4 (Sept. 2006), pp. 526–538. ISSN: 1435-5655. DOI: 10.1007/s00146-006-0050-9. URL: https://doi.org/10.1007/s00146-006-0050-9.

[127] Alberto Rezola, Jon Pey, Luis Tobalina, Ángel Rubio, John E. Beasley, and Francisco J. Planes. "Advances in network-based metabolic pathway analysis and gene expression data integration." In: *Briefings in Bioinformatics* 16.2 (2015), pp. 265–279. DOI: 10.1093/bib/bbu009. eprint: /oup/backfile/content_public/journal/bib/16/2/10.1093/bib/bbu009/2/bbu009.pdf. URL: +http://dx.doi.org/10.1093/bib/bbu009.

[128] J. C. Roberts. "State of the Art: Coordinated Multiple Views in Exploratory Visualization." In: *Fifth International Conference on Coordinated and Multiple Views in Exploratory Visualization (CMV 2007)*. 2007, pp. 61–71. DOI: 10.1109/CMV.2007.20.

[129] Hendrik Rohn, Astrid Junker, Anja Hartmann, Eva Grafahrend-Belau, Hendrik Treutler, Matthias Klapperstück, Tobias Czauderna, Christian Klukas, and Falk Schreiber. "VANTED v2: a framework for systems biology applications." In: *BMC systems biology* 6.1 (2012), p. 139. ISSN: 1752-0509. DOI: 10.1186/1752-0509-6-139. URL: https://doi.org/10.1186/1752-0509-6-139.

[130] Jean-François Rual, Kavitha Venkatesan, Tong Hao, Tomoko Hirozane-Kishikawa, Amélie Dricot, Ning Li, Gabriel F Berriz, Francis D Gibbons, Matija Dreze, Nono Ayivi-Guedehoussou, et al. "Towards a prote-ome-scale map of the human protein–protein interaction network." In: *Nature* 437.7062 (2005), pp. 1173–1178.

[131] J. Rychlewski. “On Hooke’s law.” In: *Journal of Applied Mathematics and Mechanics* 48.3 (1984), pp. 303 –314. ISSN: 0021-8928. DOI: `https://doi.org/10.1016/0021-8928(84)90137-0`. URL: `http://www.sciencedirect.com/science/article/pii/0021892884901370`.

[132] Alok J. Saldanha. “Java Treeview—extensible visualization of microarray data.” In: *Bioinformatics* 20.17 (2004), pp. 3246–3248. DOI: `10.1093/bioinformatics/bth349`. eprint: `/oup/backfile/content_public/journal/bioinformatics/20/17/10.1093/bioinformatics/bth349/2/bth349.pdf`. URL: `+http://dx.doi.org/10.1093/bioinformatics/bth349`.

[133] Nathan Salomonis, Jan Baumbach, Thomas Lengauer, Bruce Conklin, and Mario Albrecht. “AltAnalyze and DomainGraph: Analyzing and visualizing exon expression data.” In: *Nucleic acids research* 38 (July 2010), W755–62. DOI: `10.1093/nar/gkq405`.

[134] Rodrigo Santamaria, Roberto Therón, and Luis Quintales. “BicOverlapper 2.0: Visual Analysis for Gene Expression.” In: *Bioinformatics* (June 2014). DOI: `10.1093/bioinformatics/btu120`.

[135] Maximilian Scherr. “Multiple and coordinated views in information visualization.” In: *Trends in Information Visualization* 38 (2008).

[136] Hans-Jörg Schulz, Mathias John, Andrea Unger, and Heidrun Schumann. “Visual analysis of bipartite biological networks.” In: *Eurographics Workshop on Visual Computing for Biomedicine*. 2008. DOI: `10.2312/VCBM/VCBM08/135-142`.

[137] Maria Secrier, Georgios A Pavlopoulos, Jan Aerts, and Reinhard Schneider. “Arena3D: visualizing time-driven phenotypic differences in biological systems.” In: *BMC bioinformatics* 13.1 (2012), p. 45. ISSN: 1471-2105. DOI: `10.1186/1471-2105-13-45`. URL: `https://doi.org/10.1186/1471-2105-13-45`.

[138] Maria Secrier and Reinhard Schneider. “Visualizing time-related data in biology, a review.” In: *Briefings in bioinformatics* 15.5 (2013), pp. 771–782.

[139] Jinwook Seo and Ben Shneiderman. “A Rank-by-feature Framework for Interactive Exploration of Multidimensional Data.” In: *Information Visualization* 4.2 (July 2005), pp. 96–113. ISSN: 1473-8716. DOI: `10.1057/palgrave.ivs.9500091`. URL: `http://dx.doi.org/10.1057/palgrave.ivs.9500091`.

[140] Paul Shannon, Andrew Markiel, Owen Ozier, Nitin S Baliga, Jonathan T Wang, Daniel Ramage, Nada Amin, Benno Schwikowski, and Trey Ideker. “Cytoscape: a software environment for integrated models of biomolecular interaction networks.” In: *Genome research* 13.11 (2003), pp. 2498–2504. DOI: `10.1101/gr.1239303`.

[141] Zeqian Sheny and Kwan-Liu Maz. “Path Visualization for Adjacency Matrices.” In: *Proceedings of the 9th Joint Eurographics / IEEE VGTC Conference on Visualization*. EUROVIS’07. Norrk&#246;ping, Sweden: Eurographics Association, 2007, pp. 83–90. ISBN: 978-3-905673-45-6. DOI: 10.2312/VisSym/EuroVis07/083-090. URL: http://dx.doi.org/10.2312/VisSym/EuroVis07/083-090.

[142] Milton H Shimabukuro, Edilson F Flores, Maria Cristina F de Oliveira, and Haim Levkowitz. “Coordinated views to assist exploration of spatio-temporal data: A case study.” In: *Coordinated and Multiple Views in Exploratory Visualization, 2004. Proceedings. Second International Conference on*. IEEE. 2004, pp. 107–117. DOI: 10.1109/CMV.2004.1319531.

[143] M. Steinbach, G. Karypis, and V. Kumar. “A comparison of document clustering techniques.” In: *KDD Workshop on Text Mining*. 2000. URL: http://citeseer.ist.psu.edu/steinbach00comparison.html.

[144] H. Stitz, S. Luger, M. Streit, and N. Gehlenborg. “AVOCADO: Visualization of Workflow–Derived Data Provenance for Reproducible Biomedical Research.” In: *Computer Graphics Forum* 35.3 (2016), pp. 481–490. ISSN: 1467-8659. DOI: 10.1111/cgf.12924. URL: http://dx.doi.org/10.1111/cgf.12924.

[145] Marc Streit, Samuel Gratzl, Michael Gillhofer, Andreas Mayr, Andreas Mitterecker, and Sepp Hochreiter. “Furby: fuzzy force-directed bicluster visualization.” In: *BMC bioinformatics* 15.Suppl 6 (2014), S4. ISSN: 1471-2105. DOI: 10.1186/1471-2105-15-S6-S4. URL: https://doi.org/10.1186/1471-2105-15-S6-S4.

[146] A. Taylor, K. McLeod, C. Armit, R. Baldock, and A. Burger. “Visualization of gene expression information within the context of the mouse anatomy.” In: *ArXiv e-prints* (July 2014). arXiv: 1407.2117 [cs.HC].

[147] Athanasios Theocharidis, Stjin Van Dongen, Anton J Enright, and Tom C Freeman. “Network visualization and analysis of gene expression data using BioLayout Express3D.” In: *Nature protocols* 4.10 (2009), pp. 1535–1550. ISSN: 1754-2189. DOI: 10.1038/nprot.2009.177.

[148] Oliver Thimm, Oliver Bläsing, Yves Gibon, Axel Nagel, Svenja Meyer, Peter Krüger, Joachim Selbig, Lukas A. Müller, Seung Y. Rhee, and Mark Stitt. “mapman: a user-driven tool to display genomics data sets onto diagrams of metabolic pathways and other biological processes.” In: *The Plant Journal* 37.6 (2004), pp. 914–939. ISSN: 1365-313X. DOI: 10.1111/j.1365-313X.2004.02016.x. URL: http://dx.doi.org/10.1111/j.1365-313X.2004.02016.x.

[149] C. Tominski, S. Gladisch, U. Kister, R. Dachselt, and H. Schumann. “Interactive Lenses for Visualization: An Extended Survey.” In: *Computer Graphics Forum* 36.6 (2017), pp. 173–200. ISSN: 1467-8659. DOI: 10.1111/cgf.12871. URL: http://dx.doi.org/10.1111/cgf.12871.

[150] Christian Tominski, James Abello, Frank Van Ham, and Heidrun Schumann. "Fisheye tree views and lenses for graph visualization." In: *Information Visualization, 2006. IV 2006. Tenth International Conference on*. IEEE. 2006, pp. 17–24. DOI: 10.1109/IV.2006.54.

[151] Ngoc Tam L. Tran, Sominder Mohan, Zhuoqing Xu, and Chun-Hsi Huang. "Current innovations and future challenges of network motif detection." In: *Briefings in Bioinformatics* 16.3 (2015), pp. 497–525. DOI: 10.1093/bib/bbu021. eprint: /oup/backfile/content_public/journal/bib/16/3/10.1093/bib/bbu021/2/bbu021.pdf. URL: +http://dx.doi.org/10.1093/bib/bbu021.

[152] Edward R. Tufte. *The Visual Display of Quantitative Information*. USA: Graphics Press, 1986. ISBN: 096139210X.

[153] John W. Tukey. *Exploratory Data Analysis*. Addison-Wesley, 1977.

[154] Leif Väremo, Francesco Gatto, and Jens Nielsen. "Kiwi: a tool for integration and visualization of network topology and gene-set analysis." In: *BMC bioinformatics* 15.1 (2014), p. 408. ISSN: 1471-2105. DOI: 10.1186/s12859-014-0408-9. URL: https://doi.org/10.1186/s12859-014-0408-9.

[155] Corinna Vehlow, Fabian Beck, and Daniel Weiskopf. "The State of the Art in Visualizing Group Structures in Graphs." In: *Eurographics Conference on Visualization (EuroVis) - STARs*. Ed. by R. Borgo, F. Ganovelli, and I. Viola. The Eurographics Association, 2015. DOI: 10.2312/eurovisstar.20151110.

[156] Jarkko Venna and Samuel Kaski. "Comparison of Visualization Methods for an Atlas of Gene Expression Data Sets." In: *Information Visualization* 6.2 (2007), pp. 139–154. DOI: 10.1057/palgrave.ivs.9500153. eprint: https://doi.org/10.1057/palgrave.ivs.9500153. URL: https://doi.org/10.1057/palgrave.ivs.9500153.

[157] A. Vogogias, J. Kennedy, D. Archambault, V. A. Smith, and H. Currant. "MLCut: Exploring Multi-level Cuts in Dendrograms for Biological Data." In: *Proceedings of the Conferece on Computer Graphics & Visual Computing*. CGVC '16. Bournemouth, United Kingdom: Eurographics Association, 2016, pp. 1–8. ISBN: 978-3-03868-022-2. DOI: 10.2312/cgvc.20161288. URL: https://doi.org/10.2312/cgvc.20161288.

[158] Michael E Wall, Andreas Rechtsteiner, and Luis M Rocha. "Singular value decomposition and principal component analysis." In: *A practical approach to microarray data analysis*. Springer, 2003, pp. 91–109.

[159] Michelle Q. Wang Baldonado, Allison Woodruff, and Allan Kuchinsky. "Guidelines for Using Multiple Views in Information Visualization." In: *Proceedings of the Working Conference on Advanced Visual Interfaces*. AVI '00. Palermo, Italy: ACM, 2000, pp. 110–119. ISBN: 1-58113-

252-2. DOI: 10.1145/345513.345271. URL: http://doi.acm.org/10.1145/345513.345271.

[160] Chaoli Wang and Jun Tao. "Graphs in Scientific Visualization: A Survey." In: *Computer Graphics Forum* 36.1 (2017), pp. 263–287. ISSN: 1467-8659. DOI: 10.1111/cgf.12800. URL: http://dx.doi.org/10.1111/cgf.12800.

[161] Michel A Westenberg, Sacha AFT Hijum, Andrzej T Lulko, Oscar P Kuipers, and Jos BTM Roerdink. "Interactive visualization of gene regulatory networks with associated gene expression time series data." In: *Visualization in Medicine and Life Sciences* (2008), pp. 293–311. DOI: 10.1007/978-3-540-72630-2_17.

[162] Michael AE Wright and Jonathan C Roberts. "Click and brush: A novel way of finding correlations and relationships in visualizations." In: *TPCG*. 2005, pp. 179–186.

[163] Rui Xu and D. Wunsch II. "Survey of Clustering Algorithms." In: *Trans. Neur. Netw.* 16.3 (May 2005), pp. 645–678. ISSN: 1045-9227. DOI: 10.1109/TNN.2005.845141. URL: http://dx.doi.org/10.1109/TNN.2005.845141.

[164] Joseph P Zbilut, Nitza Thomasson, and Charles L Webber. "Recurrence quantification analysis as a tool for nonlinear exploration of nonstationary cardiac signals." In: *Medical engineering & physics* 24.1 (2002), pp. 53–60.

[165] J. Zhao, C. Collins, F. Chevalier, and R. Balakrishnan. "Interactive Exploration of Implicit and Explicit Relations in Faceted Datasets." In: *IEEE Transactions on Visualization and Computer Graphics* 19.12 (Dec. 2013), pp. 2080–2089. ISSN: 1077-2626. DOI: 10.1109/TVCG.2013.167.