DynTrix: A Hybrid Representation for Dynamic Graphs

B. Vago[1], D. Archambault[2] , A. Arleo[3,1]

[1]TU Vienna, Austria   [2]Newcastle University, United Kingdom
[3]Eindhoven University of Technology, The Netherlands

Abstract

Hybrid graph representations combine two or more network visualization techniques in a unique drawing, simultaneously leveraging their strong traits. Since their introduction in the early 2000s, hybrid representations have gained significant research interest, with the introduction of new techniques and comparative user studies. However, all this research has not considered dynamic graphs. In this paper, we investigate hybrid graph representations in a dynamic network context and present DynTrix. *Our system uses the NodeTrix representation as a basis, but the research extends this representation to the dynamic network domain.* DynTrix *supports automatic or manually created clusters/matrices across time. Drawing stability is implemented through aggregation and users can rearrange the nodes/matrix positions and pin them.* DynTrix *visualizes the temporal dynamics of the network through a combination of movement and element highlighting. We also introduce the concept of volatility, that allows the identification of actors in the network that are the most volatile. Matrices can be ordered such that stable cores gravitate towards the centre of the matrix. We integrate this technique in a visual analytics application for the exploration of offline dynamic networks and evaluate our system through case studies and qualitative expert interviews. Experts agree on the capabilities of the system, noting its potential for the analysis of dynamic networks through hybrid representations.*

CCS Concepts

• *Human-centered computing* → *Information visualization; Visual analytics; Graph drawings;*

1. Introduction

A *graph* (or *network*) is a data structure comprised of a set of *nodes* (or *vertices*) connected by *edges* that represent their relationships. In a *dynamic* graph, the temporal dynamics are represented as changes (additions/deletions) to the node and edge sets, posing a significant visualization challenge over the static problem. Typically (as in this paper), the time dimension is represented as a discrete time axis, meaning that the evolution of the network is encoded as a series of snapshots at equally spaced moments in time (also known as *timeslices*). The added challenge over a static scenario (where the graph is not subject to change) is to visualize both the topology of the graph *and* its temporal dynamics (i.e., changes over time). Depending on the context, the temporal behavior of some elements, e.g., appearing intermittently in different timeslices or persisting for long periods of time, might be the result of higher-level phenomena that could be uncovered using visualization.

Visualization of dynamic networks is a mature and thriving research field. Two recent and comprehensive surveys on the topic [BBDW17, FAM23] discuss experimenting with hybrid visualizations as a promising yet under-investigated research direction. Existing approaches (such as *NodeTrix* [HFM07] and *Chordlink* [ADM*22]) show interesting results: however, the design space of hybrid representations is still far from being explored to its full potential. The basic principle of a hybrid representation is combining one or more techniques in the same visualization, retaining the advantages of both while mitigating their respective shortcomings. One of the most known examples is *NodeTrix* [HFM07]: it combines the adjacency matrix representation for dense communities and a node-link representation to connect them to each other, improving support for path finding and overview tasks. This type of techniques gained significant research attention for their potential in showing complex structures in different types of networks, including social, biological, financial, and collaboration networks [HFM07, GDL*22]. Research on hybrid representations however exclusively focused on the static scenario, disregarding whether their advantages could be leveraged in a dynamic context as well – a research question we tackle in this paper.

Within this motivation, we present DynTrix: a hybrid visualization technique for dynamic networks that combines a node-link plus adjacency matrix graph representation to support the exploration of the graph structure and investigate the temporal dynamics of the network. We apply the hybrid representation philosophy to the dynamic scenario: we introduce a selection of visual metaphors for each individual graph representation to emphasize, suggest, and visualize the changes of the data over time, experimenting with a combination of movement, color, and highlighting. We also introduce a *volatility* metric for nodes to rank their tendency to appear and disappear over time, which we encode within the graph representation and

use to assess the presence of both stable *cores* within dense communities and stable nodes in sparser areas of the network. Moreover, we engineer and implement DynTrix as a fully functional visual analytics (VA) prototype. We discuss the efficacy of our system with two case studies on real data, and we evaluate DynTrix using the *ICE-T* heuristic evaluation method [WAM*19]. We then discuss the empirical evidence we collected about the user experience and the effectiveness of our design decisions to conclude the paper.

2. Related Work

Hybrid Visualizations often mix node-link diagrams with other representations in order to leverage the advantages of each approach. The probably most known approach in this context is *Node-Trix* [HFM07]. In this technique, designed for static graphs, strongly connected communities are represented as matrices. Matrices are then connected to each other with straight edges, as in the node-link representation. These allow for the user to immediately connect the communities, as node-link representation is notoriously more effective in path-tracing and overview-first tasks. The result of this approach is an effective combination of techniques that provides a high-level view of the graph but visualizes only static networks. Agrawal et al. [ATS16] extend NodeTrix to support multiplex networks within the context of the exploration of small world graphs.

ChordLink [ADM*22] combines chord and node link diagrams for the visualization of dynamic networks. In a way similar to Node-Trix, chord diagrams can be used to simplify dense parts of the visualization. In a chord diagram, nodes are arranged as circular arcs around the circumference of a circle, and its edges "cross" the middle of the circle. The size of the sectors and the width of the edge can be used to encode further variables. Didimo et al. [GDL*22] recently performed a user study where the performance of various hybrid graph representations, including Nodetrix and Chordlink, are evaluated on tasks concerning the analysis of clustered networks. The results suggest that while they do not accelerate the analysis process, hybrid representations tend to be more accurate, with Nodetrix being recommended when the analysis task entails identifying attributes of single nodes. Zhao et al. [ZMC05] combine a space-filling technique (i.e., treemaps [Shn92]) and a node-link diagram to support the visualization of hierarchies. Compound graph visualizations involving both matrices and node link diagrams have been explored for large graphs [RMF12]. All of these techniques augment node link diagrams with other visualizations in order to gain advantages from the various representations. However, none of this has been designed for dynamic graphs – a gap we intend to fill with our proposed DynTrix design.

Dynamic Graphs Visualization has received significant attention from the visualization community. For a full survey, we refer to Beck et al. [BBDW17].

Methods for visualizing dynamic graphs primarily consist of time-to-time (see, e.g., [BPF14, FKN*04, GBPD04, BW04, FWSL12]) and time-to-space (see, e.g., [SA06, BVB*11, LHS*15, LAN19, BtBC*21]) techniques, whether if time is represented by replicating multiple time snapshots of the graph arranged closely on the drawing area or by the use of animation. Several human-centered experiments have been conducted to compare and investigate the effectiveness of different methods (see, e.g., [FHQ11, OJK17, OJK19,

LAN21, FABM23]). In particular, it was investigated the role of the user "mental map" [Pur98, BB99, APP11, AP13b, AP16] in typical dynamic network exploration tasks. Experiments often showed a stable drawing did not provide any significant advantage (in terms of task performance or accuracy), but preserving the mental map did help in identifying specific paths and vertices in the graph [AP13a]. Some papers propose a hybrid timeline/animation approach. Hadlak et al. [HSS11] introduce a technique for large dynamic graphs by embedding smaller *in situ* visualizations within the context of a base visualization. This enables animations to be embedded within small multiples. *DiffAni* [RM13] proposes a combination of difference maps, animations, and timeline visualization to show the evolution of the network. However, these representations are hybrid representations of time visualization and not graph structure.

Other than visualizations, a number of algorithms has been introduced to produce node-link drawings of dynamic graphs with timeslices (e.g., [BM11, EHK*03]). Brandes et al. [BM11] compared different drawing strategies to layout dynamic graphs investigating the best tradeoff between global layout stability and local timeslice drawing quality. Three strategies were tested: aggregating all timeslices and creating a layout (maximum stability), anchoring the nodes' positions to the initial timeslice, and linking adjacent timeslices together. The latter, within the experiments' boundaries, was found to be the preferable solution and has been integrated in a social network analysis software [BW04]. Cvorsanin et al. [CCM17] introduce an incremental drawing method for the online drawing problem. The approach extends the popular *FM*^{3} [HJ05] algorithm. As the addition/removal of nodes and edges perturbates the forces equilibrium, local refinements to the drawing are made in the areas of the drawing with high energy, yielding a stable and readable layout. For completeness, we also mention that research has been made also on the drawing of dynamic graphs without timeslices (see, e.g., [SAK18, AMA22]).

While extending the functionalities of traditional static graph drawing to the dynamic scenario, these techniques still suffer from the shortcomings of the node-link representation, most notably the tendency to clutter, which heavily impacts readability with dense communities. We believe that with empirical evidence showing the efficacy of hybrid representations in a dynamic scenario, the purpose of this paper, we would foster further research to apply these techniques in this context as well.

3. Design Considerations

3.1. Definitions

A *Graph*, or *Network*, $G = (V,E)$ is a data structure comprised of a set of *nodes*, or *vertices*, V and a set E of *edges*, which are pair of nodes that represent the relationships and connections between them. In this paper, edges are considered undirected. A *dynamic* graph $\Gamma(V,E) = (G_1, ..., G_n)$ consists of a set of subgraphs where each $G_i = (V_i \subseteq V, E_i \subseteq E)$ represents the state of the graph at the time *i*. Each G_i is a *timeslice* of Γ. All timeslices are known beforehand (offline dynamic graph drawing). An *adjacency matrix* represents a graph as a square matrix, with its rows/columns being the graph nodes and a non-zero value in each cell corresponding to a pair of nodes where an edge is present. As we consider undirected graphs,

adjacency matrices are symmetrical. A matrix *ordering* [BBHR*16] is the arrangement of the nodes on the rows/columns. Row and column ordering typically matches (and is the case of this paper), but independent ordering has been experimented as well [LRT21]. Finally, we define a *clustering* of a graph as a classification of its vertices into a non-overlapping collection of groups, called *clusters*. In this paper, such classification can be either given as an input or automatically computed.

3.2. Design Goals, Tasks, Intended Users

NodeTrix was designed to make social networks, i.e., networks with several dense subgraphs, more readable and, therefore, more accessible [HFM07]. In DynTrix, we aim to obtain the same result with large, dense, dynamic graphs with the same basic principles. Moving to a dynamic scenario poses its own set of challenges, and we designed our system not only to make complex dynamic graphs readable but also to be a powerful tool for analyzing their temporal evolution, taking advantage of the hybrid network representation. While our system shares its basic principles with NodeTrix, we wanted to give DynTrix its own distinct identity, with our design goals being as follows: (i) replicate the main NodeTrix functionalities and extend them to give users more interaction possibilities, especially considering the different scenario (dynamic vs. static); (ii) introduce a combination for visual metaphors specifically designed for time exploration within this hybrid graph representation; (iii) focus on a scalable web-based implementation to tackle large graphs in terms of node/edge count while remaining interactive.

Within these goals, DynTrix design tasks are as follows. We map our proposed tasks to the taxonomy by Kerracher et al. [KKCG15] (indicated in brackets at the end of each task) to ground and corroborate our task analysis and foster future comparisons and experiments with DynTrix.

T1. *Explore* the current timeslice topology, including identifying and exploring its clusters, creating matrices, and tweaking the layout (*Direct lookup/behavior characterization: Q1, Q2* – identify graph object's attributes and patterns at a single time point).

T2. *Navigate* through time and anticipate the changes in the vertex and edge sets coming in following timeslices (*Direct Comparison: Q3, Q4* – compare patterns of connectivity over time).

T3. *Identify* topological structures and temporal features within and between communities (*Relation seeking: Q3, Q4* – find temporal trends in connectivity).

We tackle **T1** by leveraging the node-link plus matrix graph representation, including automated clustering, user-modifiable layout, and specific interactions for matrix creation. We introduce movement to encode the modifications (**T2**) in the node set; changes in the edges are highlighted by changes in their thickness. Users can decide whether to focus on new (incoming) elements from the previously visited timeslice or the ones that will not be present in the following one. The introduction and visual representation of *volatility* is meant to support **T3** as a metric to inspect, assess, and anticipate changes within and between clusters and individual nodes. We refer to Section 4 for details about DynTrix design.

Concerning our intended users, we designed DynTrix with net-

work visualization experts in mind, familiar with the representation methods we use in our system.

4. DynTrix

In this Section, we describe the design details of DynTrix, including information about its implementation. We base our design decisions on the goals and tasks discussed in Section 3. A full view of the system, with all of its main components and different views, is shown in Figure 1.

4.1. Design Challenges

As we briefly mentioned in the introduction, the representation of the time dimension of a network presents additional challenges compared to the visualization of static graphs. When dealing with hybrid representations, the designer faces the additional problem of identifying temporal representations that not only work best with each of the hybrid components but also yield a legible, organic, non-redundant final visualization when combined. In this case, the main risk is to overload some of the visual channels, leading to potentially confusing and misleading design decisions. Each combination of network structural and time representation performs differently on typical dynamic network exploration tasks [FABM23] – having more structural representations in the same visualization potentially requires investigating multiple concurrent representations of time. Hybrid graph representations are as effective as the synergy between their individual components. Representing the time dimension should exploit such interplay, but it makes exploring the design space more difficult.

Another challenge lies in scalability, including designing interactions that would support the user in making sense of the clusters present in large networks, while maintaining interactivity with thousands of nodes and edges displayed.

4.2. A Dynamic Hybrid Representation

Volatility. We introduce the metric of *volatility* to rank nodes. We define it as the total sum of a node *entry* and *leave* events across all the available timeslices. We define an *enter* event when a node does not exist in G_i and it does in G_{i+1} and vice versa for *leave* events. Rather than "persistence", that is the number of timeslices where a node remains visible, we are more interested in this aspect of the network temporal behavior. This is beneficial when dealing with clusters that possess one or more rather "stable" cores that interact with many fleeting "acquaintances". In co-citation networks, for example, researchers are unlikely to publish with the same co-authors over the years – causing node neighborhoods to change drastically from one timeslice to the other (see, e.g., Figure 4).

Representing Topology. DynTrix uses a hybrid node-link and adjacency matrix representation as basic visualization for the graph topology. In node-link, we represent nodes as circles and edges as straight lines. Node labels are placed in the vicinity of each node. Circle radius is inversely proportional to node volatility (i.e., more stable nodes appear larger and vice versa), while its color encodes the cluster. In adjacency matrices, the diagonal cells are colored according to the corresponding node cluster, while the remaining
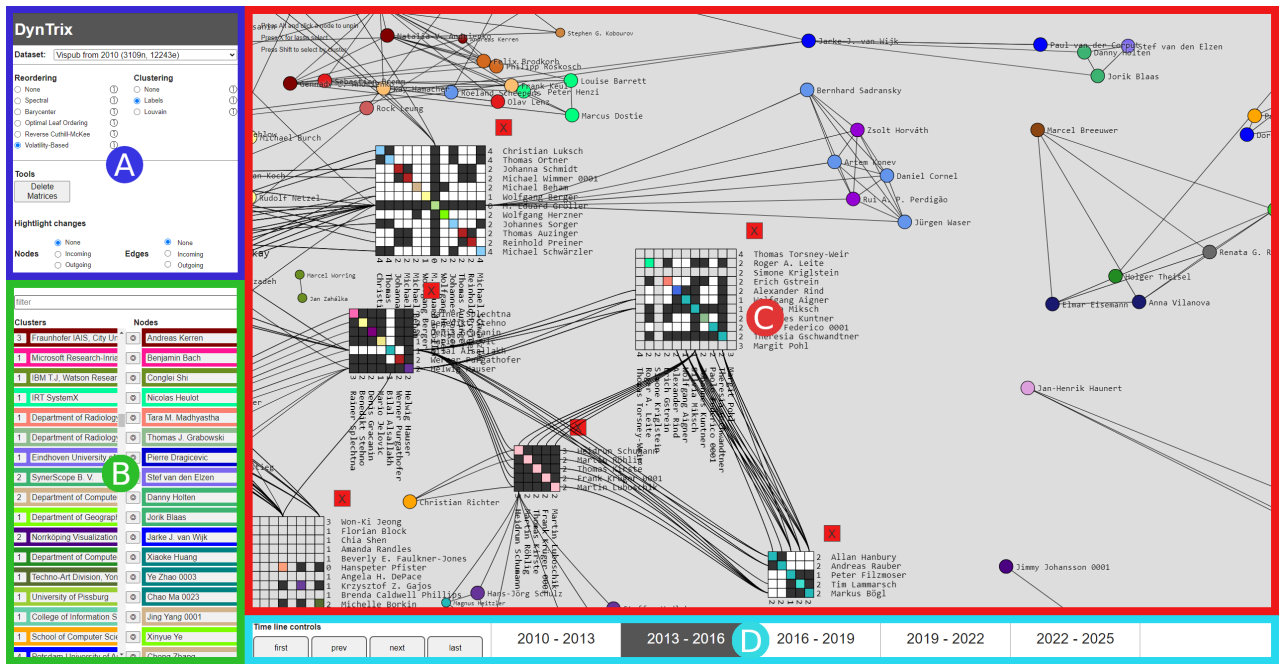
Figure 1: `DynTrix` *interface with its views: (A) sidebar, (B) cluster view, (C) main view, and (D) timeline. VisPub dataset (see Section 5) is shown. It is possible to see: the matrices "pushing" away the nodes, thus avoiding overlaps; the clusters encoding through color; the volatility encoding in the matrices, highlighting the stable elements; and the representation of empty nodes in matrices, as it is possible to see in the center and lower left side of the main view (C).*

(a) *Volatility Reordering*   **(b)** *Optimal Leaf Ordering*
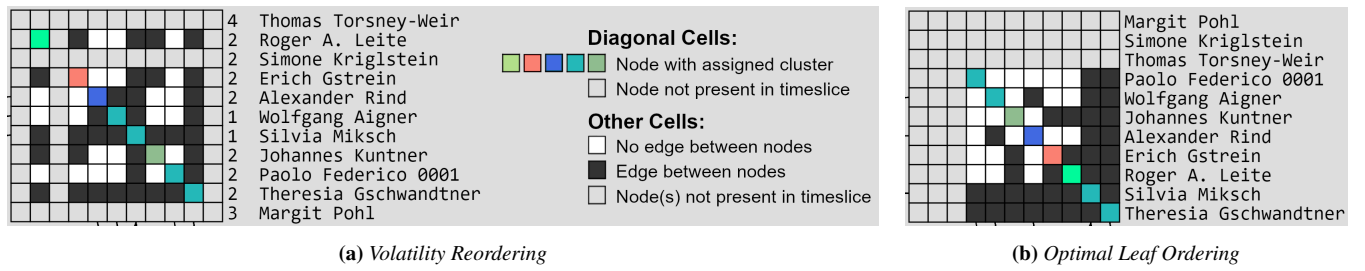
Figure 2: *Example of volatility ordering (a) compared to Optimal Leaf Ordering (b) on one of the matrices in Figure 1 (the closest to center). The numbers close to the names in the left matrix represent the computed volatility values. In (a), we also include a legend illustrating the matrix cells' color coding. While the community separation is visible on both sides, and the leaf ordering highlights the cluster connections, the volatility highlights actors most likely to remain in subsequent timeslices, suggesting the matrix most stable components and their connections.*

non-zero cells in the matrix are colored black if an edge exists between them and white otherwise. Node labels are displayed next to their corresponding rows and columns.

In adjacency matrices, row/column ordering can make a huge difference in highlighting hidden structures and patterns in the data [BBHR*16]. For this reason, we include several reordering algorithms (extracted from *Reorder.js* [Fek15]) to make sure that users can deeply explore the inner structure of strongly connected communities. Each reordering method shown in the interface is accompanied by a short explanation of its predicted performance and expected results. A change of ordering affects all existing and future matrices. Since we deal with dynamic networks that can change over time, the produced patterns can get broken up and shuffled by

the leaving and re-entering of nodes. For this reason, we include a volatility based reordering method designed to identify stable sub-communities and predict the changes in the matrix composition (see, e.g., Figures 1, 2, and 4). When this method is used, the volatility score is shown before each node label.

Representing and Navigating Time. The main novelty of `Dyn-Trix` over existing hybrid graph representations is the depiction of the time dimension. Timeslices are navigated by directly switching to a specific one or by using the interface *timeline* buttons (see Figure 1-D). One of the main challenges when visualizing dynamic graphs is making sure that the users can easily identify the changes between timeslices, that is when the user changes from G_t to any timeslice G_i, $i \neq t$. To avoid confusion, we allow the user to decide

whether to highlight in- or out-going changes for nodes and edges independently (see Figure 1-A). We refer to incoming changes as nodes and edges that were not present in G_t but are now in G_i. We refer to outgoing changes as elements present in G_i but not in G_{i+1}. While it is possible to highlight incoming changes when switching to any timeslice (also not adjacent), it is not possible to do the same with outgoing ones, as it would require knowing where the user intends to go next beforehand. Therefore, we decided to highlight which nodes and edges leave the graph in the succeeding timeslice as we believe it would be the most frequent use case.

Regardless of the type of change, we represent them as follows. Changing nodes start to "vibrate": if they are included in a matrix, their label bounces to attract attention. This motion-based approach, inspired by the work of Ware et al. [WB04, WB05], is particularly effective in this context as newly added nodes immediately "pop out" from the picture. As all nodes move in the same way, they are intuitively grouped by the user as "new" in an effect of "kinematic integration" [Mic17]. To make them stand out, changing edges depicted as lines are highlighted by increasing their thickness, while in matrices present a different hue of grey.

Mental Map Preservation. In dynamic graph visualization, making sure that the user does not get disoriented plays a major role in the design. In the following, we outline how we tackled this aspect in several elements of `DynTrix`.

The node-link layout plays a major role as it is the base visualization of all the elements of the graph, both for matrices and individual nodes. We implement the node-link layout as an aggregation approach [BM11], that is, aggregating all G_i together to form one single supergraph $\bar{G}$ which is laid out using our force-directed layout. This provides a stable drawing, supporting the user mental map – at the expense of the quality of each local layout. We attempt to mitigate this issue as follows. When loading a timeslice G_i, each node is placed at their coordinates in $\bar{G}$. Nodes and edges $\bar{v}, \bar{e} \notin G_i$ are marked as "invisible" and excluded from the drawing functions. However, they are kept in the layout, and the other nodes are still affected by their presence – though we significantly reduce the strengths of their linking and charging forces. The layout algorithm is re-run at every timeslice change and continuously updated as the user interacts with the graph. With this technique, we do not "lock" the nodes in place (as we would do in a typical aggregation approach) but rather leave them some freedom to move – potentially improving the local layout at the expense of increased computational effort.

When a user creates a matrix, some of its nodes might disappear when navigating to another timeslice. We decided not to splice any rows and columns from existing matrices when a vertex disappears. We believe that when a user creates a matrix, it represents a group of vertices of importance to the current analysis task and thus would not be in the user's intention to have it modified or disbanded from one timeslice to the other. Non-present nodes, however, are easily recognizable as there is no color in the respective diagonal cell (see Figure 1). If a node reappears, its corresponding diagonal will be colored again. Generally speaking, the composition of user-generated matrices never changes unless they are deleted.

Layout Algorithm. We compute the layout for the network on the plane using *d3* [BOH11] implementation of a force-directed layout. The graph's elements are modeled as objects that interact with each other in a simulated physical system (see, e.g., [FR91, Hu05, GHGH09]). Other than attractive and repulsive forces exerted by the edges and the individual nodes' charge respectively (as in any force-directed approach), we add a *center force* that gently draws all nodes towards the center of the canvas and a *collision force* that simulates the physical boundaries of each node in an effort to minimize overlap, especially for adjacency matrices. Adjacency matrices in the graph are simulated as invisible nodes inside the force layout, with their on-screen and charge appropriately changed according to the number of nodes within it to avoid overlaps.

Clustering. The generation of matrices is simplified by clusters specified in the data. If clusters are available, `DynTrix` applies them to the visualization and the group name is shown in the cluster view (see Figure 1). Since most datasets do not have pre-determined group assignments, we include an automated clustering algorithm to facilitate their identification. We chose the *Louvain algorithm* [BGLL08]. It is a methodology based on a modularity optimization heuristic, yielding good results with a convenient and efficient implementation. Community detection is run every time a timeslice changes and only on the portion of the graph currently visible. To emulate a dynamic community detection (as Louvain algorithm is designed for static graphs), when changing timeslice we pass the previous clustering to the algorithm as a parameter, which is used for initialization. Clustering information is presented to the user using color. Colors are selected from a cyclic categorical scale of 12 colorblind-friendly colors [HB03]. These are extended to 80 with an iterative algorithm that produces visually distinct colors by maximizing the perceptive distance from one another [mok]. Finally, we had the colors reviewed and adjusted by a red-green colorblind team member.

4.3. Implementation and Scalability

`DynTrix` is built in *Javascript*. The following core libraries are used: *d3.js* [BOH11] for the force-directed layout algorithm, *Reorder.js* [Fek15] for the matrix reordering, and *jLouvain* [jlo] provides the Louvain clustering. D3 is most often used with SVG as an output, as it represents each drawn object as a logical entity that can easily be interacted with. However, since we need to draw matrices – and matrix cells are individual objects – we deal with a quadratic rising number of objects to draw. Thus, we decided to use HTML5 canvas as a purely pixel-based output. This increases the amount of objects we can draw simultaneously by a factor of 100 compared to SVG. The largest graphs we could smoothly interact with on mid-range consumer hardware contain about 6k nodes and 23k edges (see supplemental video). As a drawback, since canvas offers no logical representation of its content, it is purely meant as an output device without native support for any interactions. Thus, we had to work around that by implementing our own library of interactions, based on assigning each element a unique color and polling the hovered pixel color when moving the mouse, which required significant additional implementation effort.

4.4. Interface and Interactions

Interface layout is presented in Figure 1, and we describe in this section each individual view.

Sidebar. In the sidebar, we place the controls to manipulate the visualization (see Figure 1-A). The top half of this bar is populated with controls for selecting datasets, matrix reordering, and graph clustering. We include a button to delete all matrices and a radio selection to highlight in- and out-going nodes or edges.

Cluster View. This view begins with a text box to search nodes by label. Below, the view is separated into two columns: on the left, there is the list of clusters, and on the right, there is the list of vertices – both lists relate to the currently visualized timeslice. Each entry in both lists is color-coded according to the cluster color (see Figure 1-B). Each vertex in the list has a button that, when pressed, centers the graph visualization on its coordinates. Linked highlighting and selection ensures that if a node in the graph is hovered over or clicked on, the corresponding node entry in the list is highlighted accordingly and vice versa. The same applies to clusters.

Main View. The main view supports zooming and panning via mouse input, as well as basic layout manipulation by dragging of nodes and matrices (see Figure 1-C). Once moved, a node/matrix will remain in place across all timeslices unless it is "freed" by the user, acting as reference points for the analysis. These nodes don't move but still exert their repulsive forces to other nodes. Nodes can be selected individually, per-cluster, or via lasso selection by dragging the mouse cursor over an area. If at least one node is selected, two buttons appear on the top side of the screen: a green button to convert the current selection of nodes into one adjacency matrix and a red button to dismiss the selection. Edges incident to selected nodes are highlighted in red (see Section 3). In an adjacency matrix, nodes can be selected by clicking on their corresponding row/column. A matrix can be removed (and its visualization "converted" to node-link) by clicking on the red "X" button close to it. A matrix remains on screen (even if empty) until it is removed by the user.

Timeline. A scrollable list of labeled timeslices at the bottom of the screen makes it possible to set the current instant out of the provided time sequence. A set of control buttons allows to rapidly navigate back and forth between adjacent timeslices (see Figure 1-D).

5. Evaluation

We evaluate `DynTrix` by discussing two case studies, to show different potential workflows when using the system on different datasets, and then by conducting a qualitative evaluation using the ICE-T protocol [WAM*19]. First, we discuss the datasets included in the system. To foster reproducibility, the source code of the system is available on GitHub [dyn], plus sample data and useful links.

- **Dialogs.** The graph represents the interactions between the characters of the "Pride and Prejudice" novel by Jane Austen in order [GWMG16]. Nodes are characters labeled by their names, and edges are inserted if the characters share a dialogue. It has 118 nodes, 501 edges, and 60 timeslices (1 per chapter).
- **InfoVis.** This dataset represents the co-authorships in the IEEE InfoVis conference from 2010 to 2019 [inf]. Each node is an author, and if two publish together, an edge exists between them. It has 708 nodes and 2,194 edges, with 3 timeslices. Each one of them groups three years, and we did that to provide some stability

to the graph, as every year the neighborhoods tend to change almost completely.
- **VisPub.** This dataset contains a co-authorship network of IEEE VIS conference, and, differently from InfoVIS, it spans all the different tracks of the conference (VAST, SciVIS, etc.) and goes from 1990 to 2022. Moreover, it presents, as metadata, the affiliation of the individual authors (which also changes over time). We simplified the dataset by using only InfoVis and VAST and limiting the time span from 2005 to 2025. This resulted in a graph with 4,181 nodes and 15,823 edges distributed over 5 timeslices (one every 5 years). For less performing systems, we also prepared a reduced version of the dataset spanning 2010-2025, with 3,109 nodes and 12,243 edges.

5.1. Case Studies

In the following, we describe two case studies where we explore real data using `DynTrix`, also showing how it achieves its design tasks.

Dialogs. In this first case study, we explore the beginning of Jane Austen's novel. Figure 3 shows two chapters from the novel. In Chapter 1, some of the main characters of the book are presented, that are the Bennet family (parents and 5 sisters) and Mr. Bingley. We cluster using the Louvain algorithm and use it as a basis to create matrices. We group together the 5 sisters (upper matrix in Figure 3) and the elder Bennets (lower matrix). We leave Mr. Bingley and other minor characters as separate nodes. Mr. Bingley, one of the novel protagonists, is introduced to the Bennets in this chapter and shares dialogues with elements from both of the previous clusters.

In Chapter 3, Mr. Darcy, one of the main characters of the novel, is introduced to the reader at the Meyrton Ball. Coming from Chapter 1, the visualization displays the incoming nodes by vibration, highlighting and indicating the other party guests. Mr. Bingley interacts with the majority of guests, remaining a very central node. It is possible to see that clustering changed slightly compared to the first chapter as the colours of the cells in the matrix changed. Elizabeth moved to the Bennets' cluster (dark blue), and Jane to the Lucas family (green). In contrast to Mr. Bingley, Mr. Darcy acts "*aloof*", interacting with fewer people during the ball. However, he interacts with both Jane and Elizabeth and declines a dance with the latter with the famous quote "*She is tolerable; but not handsome enough to tempt me.*" [Aus93]. Overall, in this example we have seen how `DynTrix` enables **T1** (timeslice exploration) and **T2** (time navigation).

InfoVis. In this second example, we examine the InfoVis co-authorship network. This version of the dataset has 1 timeslice per year. As authors tend to change their co-authors frequently, node neighborhoods tend to change drastically from one timeslice to the other. The majority of the node clusters that form are cliques, suggesting that traditional matrix ordering might provide limited insights in terms of cluster evolution. Therefore, we will now show two examples of how our volatility metric can support in finding the "stable cores" of some of these communities and, in general, how our system achieves **T3** (temporal structures investigation).

We focus on two communities in the time period 2010-2012: in
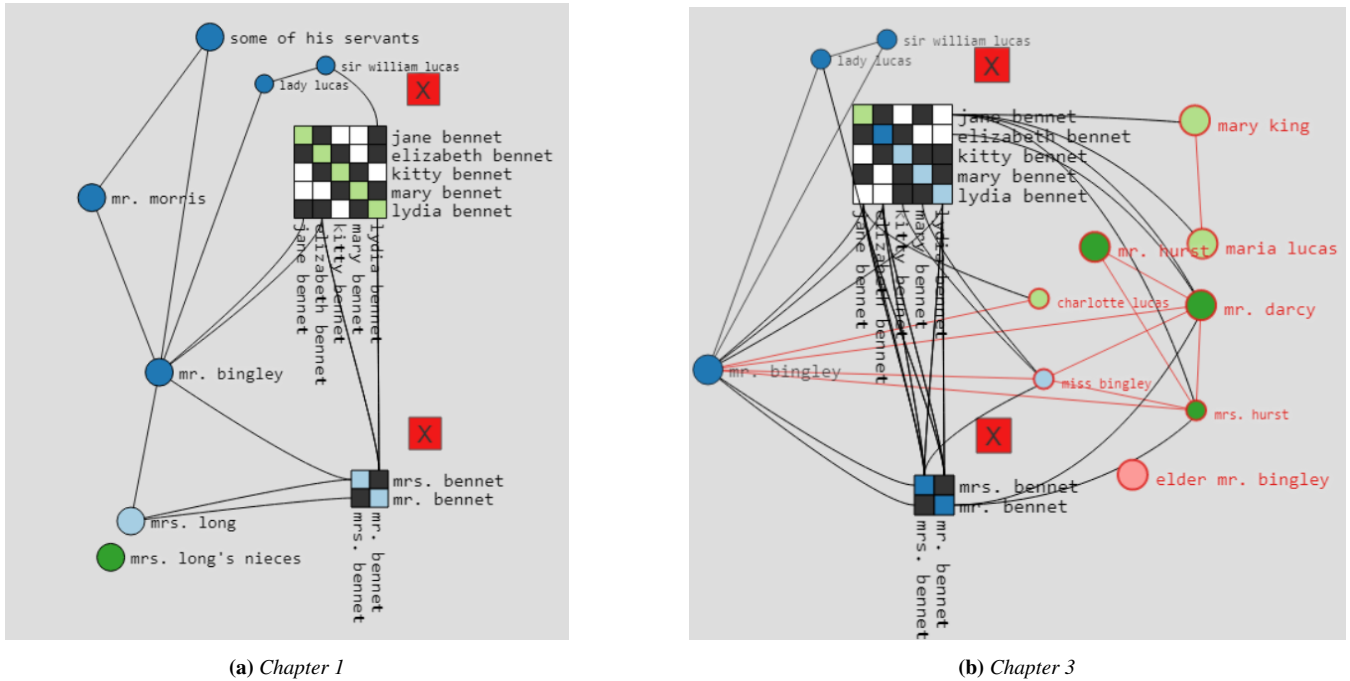
(a) *Chapter 1*

(b) *Chapter 3*

Figure 3: *Chapters from Pride & Prejudice as in Case Study 1.* **Left:** *Chapter 1 matrices including the 5 Bennet sisters (upper matrix) and their parents (lower). A stable drawing is preserved as we transition to later chapters.* **Right:** *Chapter 3 when Mr Darcy appears at the Meyrton ball. New nodes, highlighted by movement in the system, are shown here with a red outline as well as the edges incident to them. It is possible to see that matrices are not modified by DynTrix once created, but clustering can change from one chapter to the other, as shown in the matrix grouping the Bennet sisters.*

the first (*SG* in the following), we target how it evolved from its creation and forwards; in the other (*FC* in the following), we investigate its origin by seeing how different smaller clusters in previous years merge together over time (see Figure 4). *SG* is a cluster grouped in a matrix in 2010, including Stasko and Gotz. *FC* is created in 2012 and includes Fekete and Carpendale. First, we can see how the two clusters looked like in 2010. If Optimal Leaf Ordering is used, sub-communities can be easily identified (see supplemental video). By switching to volatility, temporal information regarding the nodes is included, as stable authors are moved toward the center of the matrices. In 2011, it is possible to see that only the internal core of *SG* is still visible, while *FC* is growing, showing the stable sub-communities. Finally, in 2012 *FC* is fully formed, and in *SG*, it is possible to see the cluster changes of its remaining stable members.

5.2. ICE-T Evaluation

We decided to conduct a qualitative study, following the protocol described in the paper by Wall et al. [WAM*19], to further investigate the efficacy of our system in a realistic dynamic network exploration scenario. Specifically, this protocol aims to evaluate the "value" of the visualization. John Stasko in 2014 defined the value of visualization [Sta14] as its ability to go beyond answering questions about the data, moving beyond typical tasks used in usability studies. Wall et al. developed a methodology to estimate and quantify the potential value of visualization, which goes through evaluating 4 of its components (i.e., important capabilities [Sta14]): "**I**nsight",

"**C**onfidence", "**E**ssence", "**T**ime" (hence the abbreviation ICE-T methodology – we refer to the original paper for their definition). Each component is divided into a set of guidelines that encapsulate its basic concepts, and the guidelines are finally divided into a set of 21 heuristics, formulated as actionable insights that are presented to the study participants for them to evaluate on a Likert scale from 1 (completely disagree) to 7 (completely agree). We chose this type of evaluation for different reasons. First, we thought about making a comparative user study as in the work by Didimo et al. [GDL*22], however we did not find suitable candidates to compare DynTrix with, as both NodeTrix and Chordlink (see Section 2) focus on static graphs. Therefore, the comparison would not include the temporal aspects of the visualization, While having users solving tasks about the data would have shed light on the readability and usability of the visualization, we believed that it would have been more interesting investigating the "holistic" value of DynTrix design, its ability to provide a "*true understanding of the data*" [WAM*19], and whether is capable of generating insights and knowledge beyond specific pre-determined tasks.

In the following, we describe the evaluation protocol and the quantitative results, with a discussion about the qualitative aspects of our evaluation in the Discussion section (see Section. 6). The study material (consent form, questionnaire, and result sheet) can be found as supplemental material.

Protocol. The study is structured as a set of expert interviews, where each participant is interviewed separately. As the majority of our

(a) *2010*

(b) *2011*

(c) *2012*
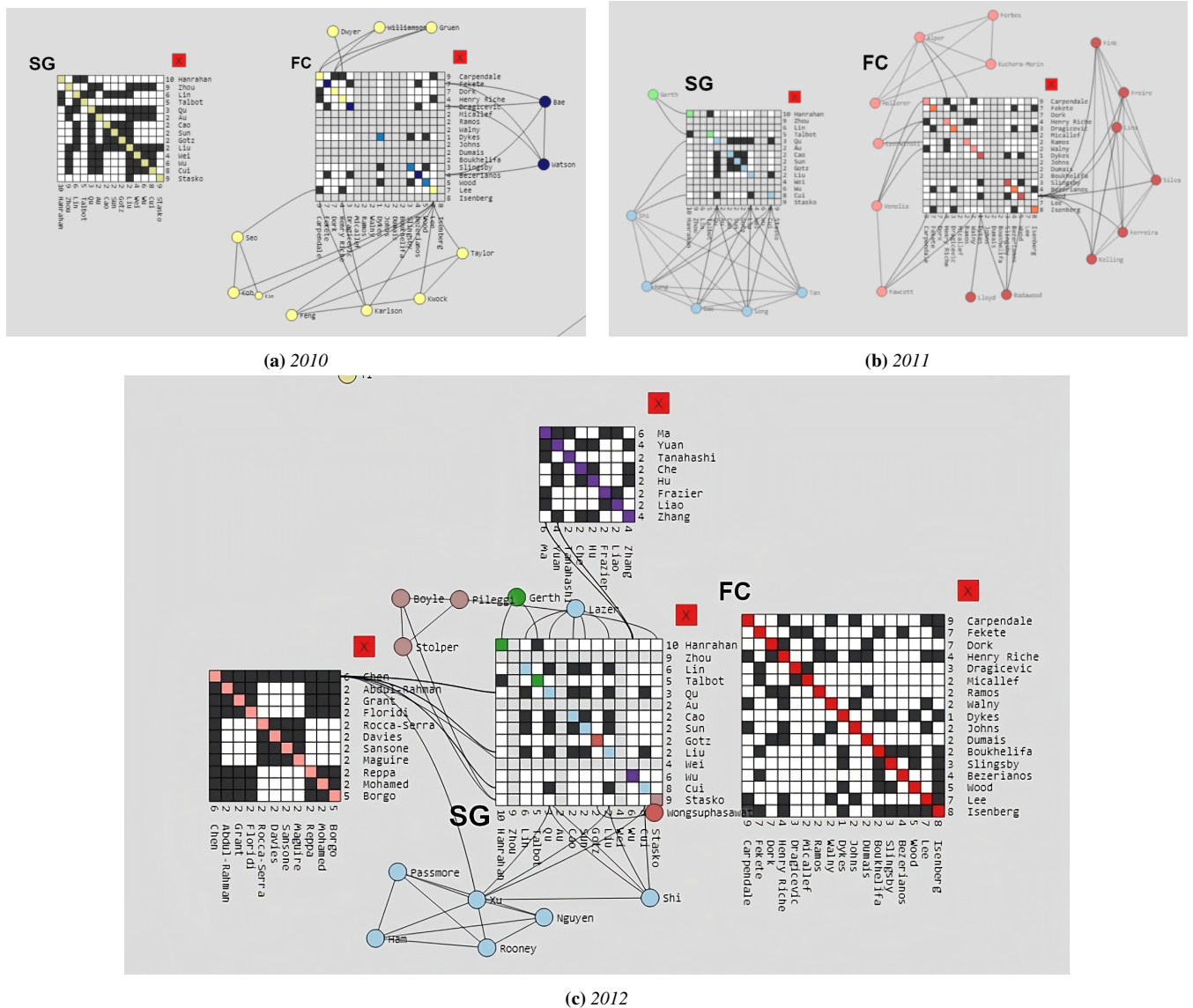
Figure 4: *Figures from case study 2: SG and FC matrices in 2010 (**a**), 2011 (**b**), and 2012 (**c**). All matrices are ordered by volatility. Color differences encode cluster changes. Volatility matrix ordering is used throughout. In (c), further matrices are included for context.*

participants could not be present physically, we opted for Zoom calls for everyone. Each individual interview was planned to last around 60-70', including a short onboarding (10'), free interaction with the system (5-10'), task solving (30'), and feedback session (10'). At the end of the interview, the participant is handed a copy of the ICE-T questionnaire. They can keep using and testing `DynTrix` on their own after the interview – participants typically sent us the compiled questionnaire a few days after. Consent forms were signed before the beginning of each interview.

Participants shared their screen with us so that we could record their interactions with `DynTrix`, as well as their voice as in a think-aloud protocol. We had no control over the experiment environment (as we let the participants use their own devices), but the ICE-

T protocol did not request it anyway. The recorded videos were then watched again to extrapolate particular behaviors, differentiate between the approaches of each participant, and extract quotes. Concerning the questionnaire results, we opted for the basic strategy described in [WAM*19]: the score for the top-level components is a simple average of the scores of the middle-level guidelines, which are a simple average of the scores obtained by the low-level heuristics. According to the protocol, an average score of at least 5 on a component means that it is a strength of the visualization, with higher scores being "better".

Tasks. The ICE-T protocol does not include any tasks, as its purpose is not to evaluate accuracy or performance. However, to bootstrap the analysis process on each of the available datasets, we devised

a set of simple tasks that would let the participants make use of all the available system features. In many cases, this fed the curiosity of the participants, who then continued using the system to pursue answers to their own questions about the data after the end of the interview. The tasks are as follows, grouped by dataset.

- **Dialogs.** (i) *Please identify who you believe are the protagonists of the novel*; (ii) *What can you tell about the evolution of relationships between the families in the novel?*
- **InfoVis.** (i) *Can you identify the changes in the ego-network of an author you recognize in this dataset?*; (ii) *Please do the same with an author you are not familiar with.*
- **VisPub.** (i) *Select a cluster from the network (manually or automatically) and inspect its composition. Could you identify the most stable and volatile elements?*; (ii) *Please select two or more institutions. What can you say about their relationships over time?*

The Infovis and VisPub represent prime examples of datasets that could be explored using `DynTrix`: they are dense graphs with changing communities over time. Moreover, they were chosen as all of our participants would be members of the visualization community. They would be able to navigate and understand the data with little onboarding, identifying (un-)expected behaviors and relationships (necessary to evaluate the heuristics about Confidence). Dialogs was selected because it was a smaller graph (and therefore would have been great as an "entry" challenge) while presenting natural communities and a long sequence of timeslices. However, we were unsure whether the participants were familiar with the novel within the data, hence the formulation of the task. As a final remark, while we always performed Dialogs and VisPub tasks, in 2 interviews we skipped InfoVis to remain within the 60' of the interview, as users preferred exploring the other datasets.

Participants. The ICE-T protocol requires a minimum of 5 experts [WAM*19]. We recruited 6, who were directly asked from us if they were interested in participating in the study. None of the experts was involved in the design process nor had any experience (or knowledge) of the system before the experiment. A consent form compliant with the European General Data Protection Regulation (GDPR) was signed beforehand, and no compensation was given. Out of the 6 participants, 4 are professors of graph drawing/network visualization, and 2 are post-doctoral researchers with papers published on the same topic. All were between expert and knowledgeable concerning hybrid graph representations, thus reflecting the profile of our intended users (see Section 3).

Quantitative Results. To consider our evaluation successful, we are aiming for an average score of 5 and above across all components. The condensed results are reported in Table 1, and the complete scores are available as supplemental material. Within the ICE-T protocol, our evaluation can be considered successful.

`DynTrix` scored above 5 on all the components, highest on the Insights component (i.e., the ability to discover insights about the data) with 6.04 and lowest on Confidence (i.e., ability to generate confidence and trust about the data) with 5.42. Time (i.e., the ability to minimize time to find information and answers) also scored 5.88 and can be considered a strong point of our proposed system. On the other hand, Confidence (i.e., the ability to generate trust about the data) got both the lowest average (while still above minimum)

	Avg.	σ
`Insight`	6,04	0,91
`Confidence`	5,42	1,40
`Essence`	5,83	0,99
`Time`	5,88	0,96

Table 1: *Condensed results of the ICE-T Evaluation.*

and highest standard deviation. We believe this to be due to the fact that assessing data quality was not within `DynTrix` design tasks. Essence component (i.e., ability to convey the overall *essence* or takeaway sense of the data w.r.t. overview and context) scored 5.83, hence suggesting that `DynTrix` enables users to get a higher-level understanding of the data and its context.

Moving from the high-level components from the individual heuristics, the lowest score, with an average of 4.17, concerned confidence and data wrangling: "*If there were data issues like unexpected, duplicate, missing, or invalid data, the visualization would highlight those issues*" [WAM*19]. The heuristic "*The interface supports using different attributes of the data to reorganize the visualization's appearance*" [WAM*19] got an almost perfect average score of 6.83, as the study participants recognized the value of the different interactions methods in `DynTrix` to extract insights.

6. Discussion

In this section, we discuss our impressions and lessons learned from the ICE-T evaluation and the most notable limitations of `DynTrix` we identified with our experts.

During our experiments, we could observe how the different participants tackled the tasks and, in turn, how they used the system features to achieve their analysis objective. While we introduced large graphs in our evaluation to encourage the participants to create and use matrices, we found that three experts preferred to use the node-link visualization as much as possible. For example, in the Dialogs task (ii – finding relationships between families over time), which was designed so that participants would take advantage of matrices, they pinned and dragged the nodes belonging to the different families close together, rather than creating matrices out of their selection. One of the participants stated that *Pride and Prejudice* was their favorite book: while we did not observe a difference in their behavior while using the system compared to others, they were immediately able to match specific circumstances of the book (e.g., the ball in chapter 2), grouping characters in matrices according to their knowledge of the book. Similar behavior was also found once for the more complex VisPub tasks (i – identify institutions' members) and (ii – explore institutions' relationships). In this case, the volatility encoding in node-link (i.e., node size) allowed one user to discover a specific pattern in the VisPub data. Larger, more stable nodes (typically representing the institution head) created star-shaped patterns with several more volatile (thus smaller) actors. Enabling the in-/out-going node highlighting, further emphasized these patterns. Nonetheless, significantly reducing the use of matrices generally meant a much slower analysis, compared to the other participants who used them as we predicted when preparing the tasks. Matrices indeed require greater concentration effort to unpack

compared to a node-link representation, which is the reason why we believe some of our study participants followed different workflows.

Volatility played a major role during all interviews and was a feature that was greatly appreciated by the participants. It was also one of the few elements that was computed across all timeslices and therefore gave participants a way to predict the behavior of the nodes. All the experts however did confuse the concept of volatility we introduced (the number of times a node/edge appears or disappears, see Section 4.2) with "persistence", that is the *time* interval an element is present. They thought that larger nodes remained visible for more timeslices (more *persistent*) rather than being the nodes with the least number of enter/exit events (or less *volatile*). For sure, persistence is an easier concept to convey, but once the users understood the difference they could make use of the feature profitably. All experts could identify the protagonists of the novel in Dialogs (i.e., Pride and Prejudice) with little to no knowledge of the plot with most using volatility. This provides evidence that volatility can unveil relevant parts of the network temporal structure and supports serendipitous discovery.

Our design decisions concerning layout and group stability were positively received by the experts. While aggregation is not the most sophisticated way to compute dynamic layouts [BM11], when combined with node and matrix pinning it allowed the users to "draw" their mental map of the graph by rearranging interesting nodes manually. Subsequently, they could navigate time without losing orientation. Similarly, keeping matrices stable across time (regardless of node presence) was also an appreciated design decision – this made observing the evolution in the relationships between and within an arbitrarily small group of nodes over time way easier. However, one expert was concerned that this could be confusing: while we explicitly mark in matrices nodes that are not present during that timeslice (see Figure 1) we don't remove them from visualization. However, nodes that are not present disappear completely from the node-link view. We justify our choice by saying that creating a matrix assumes that a specific group of vertices is somehow "special" to the user, and the fact that we encode absent nodes in matrices differently should minimize the risk of confusion.

Limitations. All experts in our evaluation agreed on the potential of the use of hybrid representations for dynamic graphs, but also discussed with us relevant limitations of our system. We discuss and report the ones that were shared by the majority of experts, aiming at building a knowledge base for future research on this topic.

Missing Overview. The system is designed to support the exploration of the individual timeslice, but lacks an overview of the graph dynamics. In turn, you would have to explore the whole time sequence to get a complete understanding of the changes in the graph: something that can prove difficult with several timeslices (as in the Dialogs dataset, for example). Volatility mitigates this problem.

Matrix Generation. While the interaction design was generally positively reviewed by our study participants, there were some concerns about some "slippery" selections, especially with very crowded node-link visualizations. Two experts also suggested introducing a control to select a node and all of its neighbors (while now only the full cluster can be selected), with one suggesting a control to create matrices from all visible clusters at once.

Clustering. In our design, we used the Louvain [BGLL08] clustering algorithm. It is a time efficient algorithm with a readily available implementation, making it a fitting candidate for inclusion in `Dyn-Trix`. However, Louvain is meant for *static* community detection. While we mitigate this issue via its implementation (see Section 4.2) other solutions in literature address this problem directly, introducing methods that perform a full dynamic community detection, identifying and visualizing the movement of nodes across clusters over time [VBAW15, VBW16, LTPR17].

Visual Scalability. Generally speaking, the combination of matrices and node-link should reduce the overall clutter, as the densest parts of the graph are grouped together into matrices. However, as we tested this metaphor with graphs with thousands of nodes, we found that matrices cannot be larger than a few tens of elements: the quadratic space requirement easily fills out the available screen real estate and also makes the individual cells smaller as the size increases (to fit the matrix in the view). For this reason, several smaller matrices should be created, but this would make the whole point of the hybrid representation weaker. To increase visible scalability, methods such as the "untangling" approach by Nocaj et al. [NOB15] could be used. Graph simplification measures (e.g., filtering or aggregation) could improve the readability of the final visualization. We could also investigate edge bundling methods to increase the visual scalability of the method.

Time Scalability. We tested `DynTrix` both on large graphs with few timeslices but also on a reasonably small graph (in terms of number of nodes and edges) with many (Dialogs). Our simple "list" of timeslices showed its limits in this case, coupled with the lack of an overview to provide some guidance to the user (as discussed before). Nonetheless, users could still identify, in Dialogs, some of the major events – however mostly in the first 10 chapters. In one case, one participant decided to compare the characters in the first 3 chapters with the ones present in the last one (60) to solve the task. Large time sequences navigation is still an open problem [BBDW17, LAN19].

7. Conclusions and Future Work

In this paper we presented `DynTrix`, a hybrid representation for dynamic graphs. We introduce visual metaphors and interactions specifically designed to take advantage of the peculiarities of the node-link plus matrix representation in a dynamic scenario. We show the efficacy of the system through case studies and conduct a qualitative evaluation with experts to evaluate the system performance in a realistic analysis scenario. We encourage further research on this topic, for example by investigating overlapping or fuzzy groupings, more sophisticated dynamic graph drawing methodologies (such as anchoring or linking [BM11]), and by introducing graph simplification measures to tackle even larger graphs. We believe that `DynTrix` could be a first step in this direction, with our current limitations being opportunities for future researchers.

References

[ADM*22] ANGORI L., DIDIMO W., MONTECCHIANI F., PAGLIUCA D., TAPPINI A.: Hybrid graph visualizations with ChordLink: Algorithms, experiments, and applications. *IEEE Transactions on Visualization and Computer Graphics 28*, 2 (2022), 1288–1300. `doi:10.1109/TVCG.2020.3016055`. 1, 2

[AMA22] ARLEO A., MIKSCH S., ARCHAMBAULT D.: Event-based dynamic graph drawing without the agonizing pain. In *Computer Graphics Forum* (2022), Wiley Online Library. `doi:10.1111/cgf.14615`. 2

[AP13a] ARCHAMBAULT D., PURCHASE H. C.: Mental map preservation helps user orientation in dynamic graphs. In *Graph Drawing & Network Visualization* (2013), Springer, pp. 475–486. `doi:10.1007/978-3-642-36763-2\_42`. 2

[AP13b] ARCHAMBAULT D., PURCHASE H. C.: The "map" in the mental map: Experimental results in dynamic graph drawing. *International Journal of Human-Computer Studies 71*, 11 (2013), 1044–1055. `doi:10.1016/j.ijhcs.2013.08.004`. 2

[AP16] ARCHAMBAULT D., PURCHASE H. C.: Can animation support the visualisation of dynamic graphs? *Information Sciences* (2016), 495–509. `doi:10.1016/j.ins.2015.04.017`. 2

[APP11] ARCHAMBAULT D., PURCHASE H., PINAUD B.: Animation, small multiples, and the effect of mental map preservation in dynamic graphs. *IEEE Trans. on Visualization and Computer Graphics* (2011). `doi:10.1109/TVCG.2010.78`. 2

[ATS16] AGARWAL S., TOMAR A., SREEVALSAN-NAIR J.: NodeTrix-Multiplex: Visual analytics of multiplex small world networks. In *Proceedings of the 5th International Workshop on Complex Networks and their Applications* (2016), Springer, pp. 579–591. `doi:10.1007/978-3-319-50901-3\_46`. 2

[Aus93] AUSTEN J.: Pride and Prejudice (1813). *New York* (1993). 6

[BB99] BEDERSON B. B., BOLTMAN A.: Does animation help users build mental maps of spatial information? In *Proc. IEEE Sym. on Information Visualization* (1999), pp. 28–35. `doi:10.1109/INFVIS.1999.801854`. 2

[BBDW17] BECK F., BURCH M., DIEHL S., WEISKOPF D.: A taxonomy and survey of dynamic graph visualization. *Computer Graphics Forum 36*, 1 (2017), 133–159. `doi:10.1111/cgf.12791`. 1, 2, 10

[BBHR*16] BEHRISCH M., BACH B., HENRY RICHE N., SCHRECK T., FEKETE J.-D.: Matrix reordering methods for table and network visualization. In *Computer Graphics Forum* (2016), vol. 35, Wiley Online Library, pp. 693–716. `doi:10.1111/cgf.12935`. 3, 4

[BGLL08] BLONDEL V. D., GUILLAUME J.-L., LAMBIOTTE R., LEFEBVRE E.: Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment 2008*, 10 (2008), P10008. `doi:10.1088/1742-5468/2008/10/P10008`. 5, 10

[BM11] BRANDES U., MADER M.: A quantitative comparison of stress-minimization approaches for offline dynamic graph drawing. In *International Symposium on Graph Drawing* (2011), Springer, pp. 99–110. `doi:10.1007/978-3-642-25878-7\_11`. 2, 5, 10

[BOH11] BOSTOCK M., OGIEVETSKY V., HEER J.: D³ data-driven documents. *IEEE Trans. Vis. Comput. Graph. 17*, 12 (2011), 2301–2309. `doi:10.1109/TVCG.2011.185`. 5

[BPF14] BACH B., PIETRIGA E., FEKETE J. D.: GraphDiaries: Animated transitions and temporal navigation for dynamic networks. *IEEE Transactions on Visualization and Computer Graphics 20*, 5 (2014), 740–754. `doi:10.1109/TVCG.2013.254`. 2

[BtBC*21] BURCH M., TEN BRINKE K. B., CASTELLA A., KARRAY G., PETERS S., SHTERIYANOV V., VLASVINKEL R.: Dynamic graph exploration by interactively linked node-link diagrams and matrix visualizations. *Vis. Comput. Ind. Biomed. Art 4*, 1 (2021), 23. `doi:10.1186/S42492-021-00088-8`. 2

[BVB*11] BURCH M., VEHLOW C., BECK F., DIEHL S., WEISKOPF D.: Parallel edge splatting for scalable dynamic graph visualization. *IEEE Transactions on Visualization and Computer Graphics 17*, 12 (2011), 2344–2353. `doi:10.1109/TVCG.2011.226`. 2

[BW04] BRANDES U., WAGNER D.: Analysis and visualization of social networks. In *Graph Drawing Software*, Jünger M., Mutzel P., (Eds.). Springer, 2004, pp. 321–340. `doi:10.1007/978-3-642-18638-7\_15`. 2

[CCM17] CRNOVRSANIN T., CHU J., MA K.: An incremental layout method for visualizing online dynamic graphs. *J. Graph Algorithms Appl. 21*, 1 (2017), 55–80. `doi:10.7155/JGAA.00406`. 2

[dyn] DynTrix github. `https://github.com/franklyn4000/DynTrix`. Online, Accessed: 2024-02-15. 6

[EHK*03] ERTEN C., HARDING P. J., KOBOUROV S. G., WAMPLER K., YEE G.: Graphael: Graph animations with evolving layouts. In *International Symposium on Graph Drawing* (2003), Springer, pp. 98–110. `doi:10.1007/978-3-540-24595-7\_9`. 2

[FABM23] FILIPOV V., ARLEO A., BÖGL M., MIKSCH S.: On network structural and temporal encodings: A space and time odyssey. *IEEE Transactions on Visualization and Computer Graphics* (2023), 1–14. `doi:10.1109/TVCG.2023.3310019`. 2, 3

[FAM23] FILIPOV V., ARLEO A., MIKSCH S.: Are we there yet? a roadmap of network visualization from surveys to task taxonomies. *Computer Graphics Forum 42*, 6 (2023), e14794. `doi:https://doi.org/10.1111/cgf.14794`. 1

[Fek15] FEKETE J.-D.: Reorder.js: A javascript library to reorder tables and networks. IEEE VIS 2015, Oct. 2015. Poster. URL: `https://hal.inria.fr/hal-01214274`. 4, 5

[FHQ11] FARRUGIA M., HURLEY N. J., QUIGLEY A. J.: Exploring temporal ego networks using small multiples and tree-ring layouts. In *International Conference on Advances in Computer-Human Interaction* (2011). URL: `https://api.semanticscholar.org/CorpusID:55520867`. 2

[FKN*04] FORRESTER D., KOBOUROV S. G., NAVABI A., WAMPLER K., YEE G. V.: Graphael: A system for generalized force-directed layouts. In *Graph Drawing, 12th International Symposium* (2004), vol. 3383 of *LNCS*, Springer, pp. 454–464. `doi:10.1007/978-3-540-31843-9\_47`. 2

[FR91] FRUCHTERMAN T. M., REINGOLD E. M.: Graph drawing by force-directed placement. *Software: Practice and experience 21*, 11 (1991), 1129–1164. `doi:10.1002/spe.4380211102`. 5

[FWSL12] FENG K., WANG C., SHEN H., LEE T.: Coherent time-varying graph drawing with multifocus+context interaction. *IEEE Trans. Vis. Comput. Graph. 18*, 8 (2012), 1330–1342. `doi:10.1109/TVCG.2011.128`. 2

[GBPD04] GÖRG C., BIRKE P., POHL M., DIEHL S.: Dynamic graph drawing of sequences of orthogonal and hierarchical graphs. In *Graph Drawing, 12th International Symposium* (2004), vol. 3383 of *LNCS*, Springer, pp. 228–238. `doi:10.1007/978-3-540-31843-9\_24`. 2

[GDL*22] GIACOMO E. D., DIDIMO W., LIOTTA G., MONTECCHIANI F., TAPPINI A.: Comparative study and evaluation of hybrid visualizations of graphs. *IEEE Transactions on Visualization and Computer Graphics* (2022), 1–13. `doi:10.1109/TVCG.2022.3233389`. 1, 2, 7

[GHGH09] GODIYAL A., HOBEROCK J., GARLAND M., HART J. C.: Rapid multipole graph drawing on the GPU. In *Graph Drawing: 16th International Symposium* (2009), Springer, pp. 90–101. `doi:10.1007/978-3-642-00219-9\_10`. 5

[GWMG16] GRAYSON S., WADE K., MEANEY G., GREENE D.: The sense and sensibility of different sliding windows in constructing co-occurrence networks from literature. In *International Workshop on Computational History and Data-Driven Humanities* (2016), Springer, pp. 65–77. `doi:10.1007/978-3-319-46224-0\_7`. 6

[HB03] HARROWER M., BREWER C. A.: ColorBrewer.org: An online tool for selecting colour schemes for maps. *The Cartographic Journal 40*, 1 (2003), 27–37. `doi:10.1179/000870403235002042`. 5

[HFM07] HENRY N., FEKETE J.-D., MCGUFFIN M. J.: NodeTrix: a hybrid visualization of social networks. *IEEE transactions on visualization and computer graphics 13*, 6 (2007), 1302–1309. doi:10.1109/TVCG.2007.70582. 1, 2, 3

[HJ05] HACHUL S., JÜNGER M.: An experimental comparison of fast algorithms for drawing general large graphs. In *Graph Drawing, 13th International Symposium* (2005), vol. 3843 of *LNCS*, Springer, pp. 235–250. doi:10.1007/11618058_22. 2

[HSS11] HADLAK S., SCHULZ H.-J., SCHUMANN H.: In situ exploration of large dynamic networks. *IEEE Transactions on Visualization and Computer Graphics 17*, 12 (2011), 2334–2343. doi:10.1109/TVCG.2011.213. 2

[Hu05] HU Y.: Efficient, high-quality force-directed graph drawing. *Mathematica journal 10*, 1 (2005), 37–71. 5

[inf] CiteVis: Visualizing citations among infovis conference papers. https://sites.cc.gatech.edu/gvu/ii/citevis/infovis-citation-data.txt. Online, Accessed: 2023-11-22. 6

[jlo] jLouvain github repository. https://github.com/upphiminn/jLouvain. Accessed: 2023-06-02. 5

[KKCG15] KERRACHER N., KENNEDY J., CHALMERS K., GRAHAM M.: Visual techniques to support exploratory analysis of temporal graph data. In *EuroVis - Short Papers* (2015), The Eurographics Association. doi:10.2312/eurovisshort.20151133. 3

[LAN19] LEE A., ARCHAMBAULT D., NACENTA M.: Dynamic network plaid: A tool for the analysis of dynamic networks. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (2019), pp. 1–14. doi:10.1145/3290605.3300360. 2, 10

[LAN21] LEE A., ARCHAMBAULT D., NACENTA M. A.: The effectiveness of interactive visualization techniques for time navigation of dynamic graphs on large displays. *IEEE Trans. on Visualization and Computer Graphics 27*, 2 (2021), 528–538. doi:10.1109/TVCG.2020.3030446. 2

[LHS*15] LIU Q., HU Y., SHI L., MU X., ZHANG Y., TANG J.: EgoNet-Cloud: Event-based egocentric dynamic network visualization. In *2015 IEEE Conference on Visual Analytics Science and Technology (VAST)* (2015), IEEE, pp. 65–72. doi:10.1109/VAST.2015.7347632. 2

[LRT21] LIOTTA G., RUTTER I., TAPPINI A.: Simultaneous FPQ-ordering and hybrid planarity testing. *Theoretical Computer Science 874* (2021), 59–79. doi:10.1016/j.tcs.2021.05.012. 3

[LTPR17] LINHARES C. D., TRAVENÇOLO B. A., PAIVA J. G. S., ROCHA L. E.: DyNetVis: a system for visualization of dynamic networks. In *Proceedings of the symposium on applied computing* (2017), pp. 187–194. doi:10.1145/3019612.3019686. 10

[Mic17] MICHOTTE A.: *The perception of causality*, vol. 21. Routledge, 2017. 5

[mok] Generate palettes of optimally distinct colors. https://mokole.com/palette.html. Accessed: 2023-06-06. 5

[NOB15] NOCAJ A., ORTMANN M., BRANDES U.: Untangling the hairballs of multi-centered, small-world online social media networks. *J. Graph Algorithms Appl. 19*, 2 (2015), 595–618. doi:10.7155/JGAA.00370. 10

[OJK17] OKOE M., JIANU R., KOBOUROV S. G.: Revisited experimental comparison of node-link and matrix representations. In *Graph Drawing and Network Visualization - 25th International Symposium* (2017), vol. 10692 of *LNCS*, Springer, pp. 287–302. doi:10.1007/978-3-319-73915-1_23. 2

[OJK19] OKOE M., JIANU R., KOBOUROV S. G.: Node-link or adjacency matrices: Old question, new insights. *IEEE Trans. Vis. Comput. Graph. 25*, 10 (2019), 2940–2952. doi:10.1109/TVCG.2018.2865940. 2

[Pur98] PURCHASE H.: The effects of graph layout. In *Proceedings 1998 Australasian Computer Human Interaction Conference. OzCHI'98* (1998), pp. 80–86. doi:10.1109/OZCHI.1998.732199. 2

[RM13] RUFIANGE S., MCGUFFIN M. J.: DiffAni: Visualizing dynamic graphs with a hybrid of difference maps and animation. *IEEE Trans. Vis. Comput. Graph. 19*, 12 (2013), 2556–2565. doi:10.1109/TVCG.2013.149. 2

[RMF12] RUFIANGE S., MCGUFFIN M. J., FUHRMAN C. P.: TreeMatrix: A hybrid visualization of compound graphs. *Computer Graphics Forum 31*, 1 (2012), 89–101. doi:10.1111/j.1467-8659.2011.02087.x. 2

[SA06] SHNEIDERMAN B., ARIS A.: Network visualization by semantic substrates. *IEEE transactions on visualization and computer graphics 12*, 5 (2006), 733–740. doi:10.1109/TVCG.2006.166. 2

[SAK18] SIMONETTO P., ARCHAMBAULT D., KOBOUROV S.: Event-based dynamic graph visualisation. *IEEE Transactions on Visualization and Computer Graphics 26*, 7 (2018), 2373–2386. doi:10.1109/TVCG.2018.2886901. 2

[Shn92] SHNEIDERMAN B.: Tree visualization with tree-maps: 2-d space-filling approach. *ACM Transactions on graphics (TOG) 11*, 1 (1992), 92–99. doi:10.1145/102377.115768. 2

[Sta14] STASKO J. T.: Value-driven evaluation of visualizations. In *Proceedings of the Fifth Workshop on Beyond Time and Errors: Novel Evaluation Methods for Visualization* (2014), ACM, pp. 46–53. doi:10.1145/2669557.2669579. 7

[VBAW15] VEHLOW C., BECK F., AUWÄRTER P., WEISKOPF D.: Visualizing the evolution of communities in dynamic graphs. *Comput. Graph. Forum 34*, 1 (2015), 277–288. doi:10.1111/CGF.12512. 10

[VBW16] VEHLOW C., BECK F., WEISKOPF D.: Visualizing dynamic hierarchies in graph sequences. *IEEE Trans. Vis. Comput. Graph. 22*, 10 (2016), 2343–2357. doi:10.1109/TVCG.2015.2507595. 10

[WAM*19] WALL E., AGNIHOTRI M., MATZEN L., DIVIS K., HAASS M., ENDERT A., STASKO J.: A heuristic approach to value-driven evaluation of visualizations. *IEEE Transactions on Visualization and Computer Graphics 25*, 1 (2019), 491–500. doi:10.1109/TVCG.2018.2865146. 2, 6, 7, 8, 9

[WB04] WARE C., BOBROW R.: Motion to support rapid interactive queries on node–link diagrams. *ACM Transactions on Applied Perception (TAP) 1*, 1 (2004), 3–18. doi:10.1145/1008722.1008724. 5

[WB05] WARE C., BOBROW R.: Supporting visual queries on medium-sized node–link diagrams. *Information Visualization 4*, 1 (2005), 49–58. doi:10.1057/palgrave.ivs.9500090. 5

[ZMC05] ZHAO S., MCGUFFIN M., CHIGNELL M.: Elastic hierarchies: combining treemaps and node-link diagrams. In *IEEE Symposium on Information Visualization (InfoVis 05)* (2005), pp. 57–64. doi:10.1109/INFVIS.2005.1532129. 2