This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2016.2598876, IEEE Transactions on Visualization and Computer Graphics

Temporal Summary Images: An Approach to Narrative Visualization via Interactive Annotation Generation and Placement

Chris Bryan, *Student Member, IEEE*, Kwan-Liu Ma, *Fellow, IEEE*, and Jonathan Woodring

Abstract— Visualization is a powerful technique for analysis and communication of complex, multidimensional, and time-varying data. However, it can be difficult to manually synthesize a coherent narrative in a chart or graph due to the quantity of visualized attributes, a variety of salient features, and the awareness required to interpret points of interest (POIs). We present Temporal Summary Images (TSIs) as an approach for both exploring this data and creating stories from it. As a visualization, a TSI is composed of three common components: (1) a temporal layout, (2) comic strip-style data snapshots, and (3) textual annotations. To augment user analysis and exploration, we have developed a number of interactive techniques that recommend relevant data features and design choices, including an automatic annotations workflow. As the analysis and visual design processes converge, the resultant image becomes appropriate for data storytelling. For validation, we use a prototype implementation for TSIs to conduct two case studies with large-scale, scientific simulation datasets.

Index Terms—Narrative visualization, storytelling, annotations, comic strip visualization, time-varying data.

1 Introduction

Visualization may be used as both an exploratory and explanatory tool for those who routinely need to analyze and extract essential information from their datasets and then communicate findings with others. While many visual analysis methods have been developed, there is less support for creating narrative visualizations. In particular, as data becomes large, complex, multidimensional, and sometimes heterogeneous, manually sifting through, identifying, and highlighting the essential aspects of a chart or graph becomes a laborious task. It would be desirable if, over the process of exploration and analysis, the visualization system suggests important areas and features to select and label, for the purpose of deriving a data story to use in subsequent tasks or presentations.

To help address this problem, we present Temporal Summary Images (TSIs), a framework to create narrative visualizations of multivariate, time-varying datasets. Visually, a TSI is composed of three common components: (1) a temporal layout view such as a line chart or storyline, (2) data snapshots appended at relevant timesteps, and (3) anchored textual annotations. Figure 1 shows an example TSI. It succinctly tells a story about immigration in the United States using a stacked graph for its temporal layout, five cartographic maps as the set of data snapshots, and six descriptive annotations.

The focus of this paper is not solely the finalized, "presentation-style" images that a TSI designer can produce. Rather, we emphasize the convergence of the analysis and design processes. To augment exploration, we contribute a number of interactive "under the hood" techniques. These assist data interaction and visualization creation by performing two tasks: (1) selection of relevant timesteps for data snapshots and (2) providing a user-in-the-loop, automatic annotations workflow to recommend data points of interest (POIs) on the display. A large portion of this paper focuses on this novel annotation support, which automatically creates, scores, ranks, and appends data-driven annotations onto the display. During analysis, these alert the user both to salient visual regions and to significant data features. If desired, recommended annotations can be saved and subsequently used to convey key data observations about the data when the TSI is presented to a general audience.

Furthermore, as the overall exploration and building process happens, a designer searches, filters, and edits the data to display while tweaking its overall visual appearance. As these two tasks synthesize together, a unified, summary data story effectively emerges that emphasizes the important aspects and trends of the underlying dataset. Further stylizing the components results in an image appropriate for presentation or public display according to the author's purpose.

We have created a prototype TSI application based on the framework and techniques we describe in this paper. For validation, we conduct two case studies using large-scale, scientific simulation datasets (a disease and cosmological model, respectively). Based on domain participant feedback, our approach is effective for both analyzing and summarizing datasets.

2 Background and Related Work

Related prior work falls into two main categories: (1) narrative visualization and storytelling as methods for communicating data, and (2) the three specific visual components of a TSI: time-varying techniques, small multiples (also called comic strips), and annotations.

2.1 Narrative Visualization and Storytelling

Segel and Heer categorize and review narrative visualization in [35]. They describe seven specific genres for this, including annotated graphs/charts and comic strip visualizations. A narrative visualization can be framed to tell a *data story* by prioritizing a specific interpretation or perception of the data [19].

The concept of data storytelling itself has been highlighted in the InfoVis, SciVis, and business communities [15, 23, 25, 28]. Here, the focus is on how film, literary, and theatrical narrative conventions can be used to adapt visualizations for communication to broad demographic audiences. In [26], Lee *et al.* argue for explicitly defining the scope of what should be called a data story, and propose a three-step process for visual storytelling based on the following tasks: (1) find insights, (2) create a story, and (3) tell the story. We focus the TSI process on the first two points: exploring the data and creating a presentation-quality visualization.

In the broader context of visualization design for communication, Moere and Leuven argue in [32] that *aesthetics* make up an important third constraint for visualization (in addition to *soundness* and *utility*). Recent papers have focused on the workflow aspects of creating infographic or presentation-style visualizations [7, 34, 39, 46], though most require fully manual design with no consideration for analytic tasks. While some tools combine both goals [16, 36], their design is for a different set of tasks that what the TSI framework addresses.

- *Chris Bryan and Kwan-Liu Ma are with the University of California, Davis. E-mail: {cjbryan, klma}@ucdavis.edu.*
- *Jonathan Woodring is with Los Alamos National Laboratory. Email: woodring@lanl.gov.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org.
Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2016.2598876, IEEE Transactions on Visualization and Computer Graphics

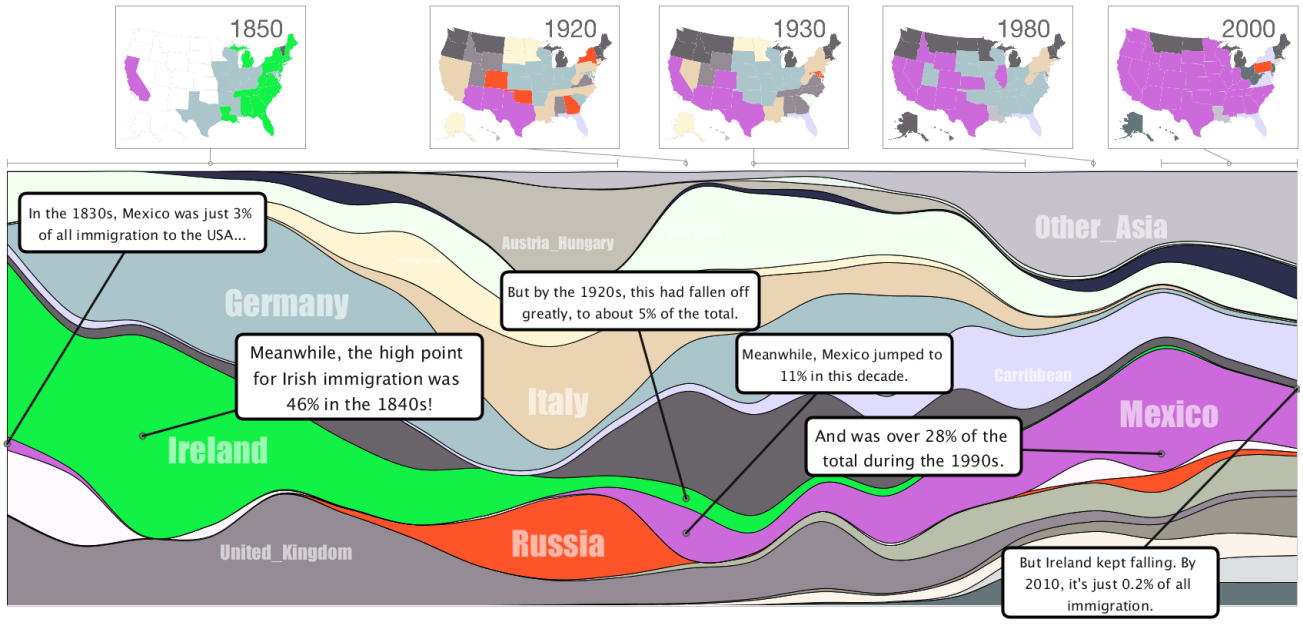


Fig. 1: A built TSI using a stacked graph to show immigration to the United States, 1830-2010. Each layer represents the total percent of immigrants based on country (or region) of origin. Data snapshots show how the distribution of immigrants has changed over time. Annotations juxtapose how Ireland was once a significant portion of total immigration but now is only a fraction, while for Mexico the opposite has happened.

2.2 Displaying Time-Oriented Data

The largest visual component in a TSI is its temporal layout for displaying time-varying data. The chosen layout depends on the author's discretion; the four options we discuss in the current framework are line charts and streamgraphs [10] for purely numeric data, and storylines [40] and alluvial diagrams [33] for flow-based, categorical data. While these are widely-known, conventional techniques (we choose them for this reason), there are many more potential ways to show time varying data [5] that have been designed with specific datasets or aesthetics in mind.

In addition to straightforward visual plotting, a multi-component system can augment a temporal view via linked displays or appended visual components, usually to enable data analytics. For example, the STAC [42] and PieceStack [43] systems focus on analysis of stacked graphs. ChronoLenses [44] are a lens-based, data-transformation pipeline for line charts. The SemanticTimeZoom system [6] supports data analysis by combining both qualitative and quantitative visuals in a chart. These techniques support detailed interactions and understanding with the underlying data, but do so at the expense of having to focus on specific visual layouts (i.e., only streamgraphs) and do not consider data storytelling or presentation. In contrast, TSIs can display multiple types of temporal views and use text annotations as a way to communicate qualitative data observations without requiring a training period for TSI authors/viewers to interpret their meanings.

2.3 Small Multiples and Comic Strips

Small multiples use a set of views (or frames) at discrete data increments to show change across one or more dimensions [41]. When this change tends to follow a strictly linear data path (even if it involves zooming and filtering), the technique can be defined as a comic strip-style of narrative visualization [35].

Prior work has used comic strip visualizations as a way to summarize or present data [12, 45, 46]. Conversely, the VizPattern system uses comic strips as an interface for creating visual queries to generate data charts [21]. A recent paper called Graph Comics [7] uses comic strips to summarize network change over time. Frames are stylized by appending text captions, labels, and annotations to highlight specific aspects of the temporal data evolution. However, all design and building is manually performed by the system user.

The comic strip technique is used for the TSI data snapshots component. To assist choosing which snapshots to show, we present three algorithms for timestep selection, see Section 4.2. This is similar to some prior systems (such as [45]) in that timesteps are chosen using distance and clustering heuristics. A TSI author selects both the desired timestep selection technique and the desired data attributes to segment on, and can manually tweak results or select a different algorithm until an acceptable result is found.

2.4 Annotating Visualization

Perceptual understanding of salient features is important for graph and chart comprehension [18]. Text-based annotations help this process by "graphically pointing" a viewer's attention to regions of interest, and can be used to suggest conclusions and provide data context [35].

In [20], the authors define annotations that specifically reference visualized data as *observational*, while *additive* annotations provide extra information not shown in the view itself. Annotations created by sketching are defined as *freeform*, and are especially important in asynchronous, collaborative settings and in journalism/infographic design [11, 17]. Alternatively, *data-driven* annotations are generated and placed by querying the underlying dataset and referencing the visual layout [20, 22].

Data-driven annotations that are automatically created attempt to identify and label a dataset/visualization's most interesting features or overall themes. Google Drive recently launched "verbalizations" [1] for their spreadsheets application, which creates a chart of the data with a descriptive caption. In [22], Kandogan introduces a system to annotate clusters, outliers, and trends in point-based data visualizations. Kong and Agrawala created an observational annotation approach in [24] that labels an already-created chart's features and dimensions without referencing the underlying raw data values.

A particularly relevant work to this paper is by Hullman *et al.* [20]. They annotate stock market timelines by matching price extremas with temporally-relevant news stories retrieved from a database. This lets them create context-aware, additive annotations. A similar approach is used in [14] to annotate geographic maps. In contrast, the TSI framework can create both additive and observational types of annotations, with a focus on placement. Annotations can also be applied to different types of time-varying visual techniques (not only line charts).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2016.2598876, IEEE Transactions on Visualization and Computer Graphics

3 Design Requirements and Workflow

The motivation for TSIs came from discussion with the EpiSimS disease simulation team (see Section 6.1.1 for a case study). The members of this group, though well-versed in epidemic research, were not visualization experts and had limited experience in using complex visual analytic and design tools. Based on their usual analysis needs and the steps they take to create images for review, we defined the following specific set of tasks for the team:

T1 Results along spatiotemporal dimensions. EpiSimS simulation output data has two main dimensional axes. (1) Disease spread happens over a time period as the epidemic grows, peaks, and decays. (2) This spread happens over a geographic region, usually initially via hotspots and then to full diffusion.

T2 Data analysis by querying for features. EpiSimS scientists have a high familiarity with their domain data and this directs their exploration. Their main focus is understanding how simulation input parameters and mitigation strategies affect epidemic lifecycle behavior for specific points of interest (POIs), such as an outbreak's peak or its distribution throughout a population's demographics. This is done by querying the underlying dataset via SQL or table-based spreadsheet functions.

T3 Presentation with conventional tools. To present results to collaborators or general audiences (such as at a conference or in a paper), static plots are used (line charts, maps, etc.), created with tools like R or Python's matplotlib library. They are combined and captioned using image editing software.

Though these tasks as written are specific to the EpiSimS team, they can easily be generalized. In a broad sense, researchers and chart designers may first wish to quickly review, analyze, and explore their data for relevant features or POIs. They then summarize results with a set of visual elements for presentation or storytelling. As opposed to doing these tasks manually and with no guidance, the TSI framework combines them into a single workflow and provides techniques to augment the analysis and the image building processes. To formally justify the design components and interaction techniques discussed throughout the rest of this paper, we first define a set of guidelines that the TSI framework should adhere to:

DG1 Temporal-plus data views. The dataset should be visualized on (at least) two primary dimensional axes: the time-varying domain plus one or more "other" dimensions. In generalizing from the EpiSimS-specific tasks, the spatial domain is abstracted; it now only needs to be orthogonal to the temporal axis.

DG2 Highlight important elements. Important dataset features and POIs should be highlighted to initially call attention to authors creating a TSI and then to viewers observing a completed TSI.

DG3 Succinct view for presentation. A completed or "built" TSI should be suitable for data presentation or storytelling as a single, connected, static figure. During creation, this implies an emphasis on styling and configuring the graphical components of a TSI to the author's preferred liking, and then exporting the image to a format suitable for display. In the context of visualization theory, this suggests that conventional or widely understood visual techniques be used, since they will be more easily understood by a general population of viewers.

Three visual components were chosen to fulfill these requirements. For DG1, an author-selected *temporal layout* displays a time-varying data view. In consideration of DG3, we use conventional time-varying techniques. One or more comic-strip sets of *data snapshots* show the "other," orthogonal dimensions. The reasoning for using comic strips (as opposed to animation or similar techniques) is due to the static image aspect of DG3. To ensure that components are linked together (forming a single, connected overall image), snapshots are appended above the temporal layout along a track at their corresponding timesteps. To help TSI authors choose the "best" snapshots to show, we contribute a set of automatic timestep selection techniques.

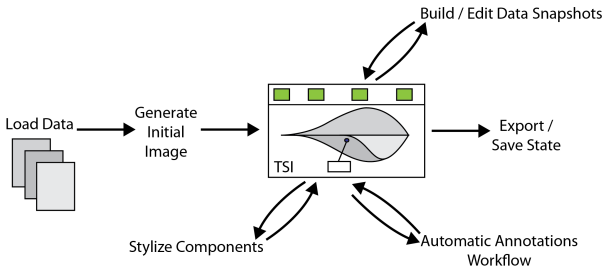


Fig. 2: An overview of the TSI analysis and design workflow. The automatic annotations workflow is illustrated in Figure 4.

To address DG2, we use textual annotations to perform graphical pointing of important elements on the temporal layout. Data-driven annotations are automatically created and appended to the display in a novel workflow; they act as a form of guided exploration for a TSI designer by alerting him/her to important data features or salient regions in the view. When familiar with the data, an author can quickly interact with a list of created annotations to search for relevant attributes, extremas, and POIs, or manually query the data to create new annotations. For communicating key observations to viewers in a built TSI, selected annotations can be pinned and saved. In this way the annotations workflow serves a dual purpose, by enriching analysis and exploration for authors and by communicating findings to viewers. Since annotations are text-based, they have an advantage over more abstract visual techniques (such as [6]) of requiring no training for interpretation, since the relevant POI or feature is explicitly described by the note's text.

3.1 Workflow

Besides simply defining a set of visual components, the design guidelines imply that there must be a process for TSI exploration and image building. We show this workflow in Figure 2, which notes the specific steps and interactions that a TSI author performs.

The user first loads a set of files. In our implementation, besides the raw data these can include configuration options like choice of temporal layout, color palettes, and pre-saved annotations. This generates an initial view of the TSI by creating the temporal layout, selecting a set of data snapshots based on the default (or user-specified) timestep selection options, and loading any saved annotations.

There are many interactions the user can now perform. Data snapshots can be edited by selecting a different set of timesteps (choosing a new heuristic for timestep selection, changing the number to show, etc.) or dragging them to different timesteps along their track. A second set of data snapshots can additionally be loaded and appended above the first set (see Figure 10).

Regarding annotations, the author can select the attributes and POI types that are important to him/her and start the automatic annotations workflow. This creates a set of data-driven annotations and tries to place the most important ones on the display based on a relevancy ranking. We contribute two algorithms for annotation placement. Annotations can individually be interacted with: dragged, edited, deleted and filtered, or the user can edit the underlying attribute/POI scores to wholesale re-rank, filter, select, and place annotations on the display.

In addition to this, our system provides an interface for manual data querying (via SQL) and annotation creation. We also let designers stylize and tweak the visual components: setting colors, font styles, editing default text and labels, adjusting component sizes, etc. When an author is satisfied with the visual output, s/he can export or save the finished TSI as an image file or save its current state for later reuse. An exported TSI is meant to be a summarization of the dataset, as it includes views along important dimensional axes (via the temporal layout and data snapshots). The set of saved annotations acts as a way to help narrate trends or highlights to TSI viewers.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2016.2598876, IEEE Transactions on Visualization and Computer Graphics

3.2 Example TSI: US Immigration 1830-2010

We illustrate a built TSI through an example use case, shown in Figure 1. This TSI tells a story about immigration trends in the United States from 1830-2010 (data from [3, 4]). The temporal layout shows the percent of immigrants for each decade based on country (or region) of origin. Each timestep sums to 100%, which is intuitively communicated by the use of a stacked graph. The layers for Ireland, Russia, and Mexico have been highlighted with bright colors. Above this, data snapshots are cartographic maps; each state is colored according to its dominant immigrant population.

The combination of the stacked graph and map snapshots show immigration has dramatically evolved in two ways: the countries that people are coming from, and the states they are moving to. To emphasize that the trends of Ireland and Mexico have mirrored each other, selected annotations describe these two data layers with playful text. They direct the viewer's attention: Irish immigrants once made up almost half of all immigrants to the United States but have presently dwindled to only a small fraction. Meanwhile, Mexican immigration has gone the opposite direction.

4 Temporal Layout and Data Snapshots

We now give a high-level overview of a TSI's first two visual components: the temporal layout and data snapshots.

4.1 Temporal Layout

The temporal layout shows a time-varying view of the data; it is centrally placed and oriented horizontally from left-to-right. The choice of visualization here depends on a TSI author's discretion, and so should be carefully considered. For independent, numeric time series (such as stock prices), a line chart is a straightforward choice. A streamgraph can be utilized if showing the temporal data magnitudes is an important consideration, such as in the immigration example (Figure 1). For categorical or flow data, a technique like storylines or alluvial diagrams should be used.

Our current implementation uses these four techniques as options for the temporal layout. As they are well-known conventional techniques, they should be effective when used for presentation to a general audience. In theory however, any time-dependent visualization technique could be used for this component, though more complex or abstract views introduce potential interpretation issues.

4.2 Data Snapshots

Data snapshots are a comic strip set of visual frames, showing a view of the dataset that is orthogonal to the temporal layout. In the immigration example, this is a cartographic view of the data. Snapshots are appended to a track above the temporal layout, with placement corresponding to their associated timesteps. They are meant to provide additional insight into the temporal evolution of the dataset and help give a holistic summary of the data. Like the temporal layout, the choice of visualization technique used in the frames is left to the author. Our current TSI prototype stores data snapshots as sets of image files; relevant ones are retrieved and appended to the display. In the case studies (Section 6.1), spatial and volume renderings are used, but more abstract or projectional mappings might be appropriate depending on the context. These can include scatter plots, bar charts, heatmaps, video stills, and other dimensionality reduction techniques.

The set of data snapshots shown is dependent on a set of chosen timesteps. To assist a TSI author in selecting appropriate snapshots, we provide three techniques for automatic timestep selection: modulo timestep indexing, entropy threshold selection, and hierarchical segmentation. To use one of these, a user first selects one or more temporal attributes in the dataset. The heuristic is applied to the set of attributes and returns a set of timesteps. Figure 3 shows how the techniques would select timesteps for an example attribute: a numerical data vector in a line chart. For each chosen timestep, the relevant snapshot image is retrieved and appended to the TSI.

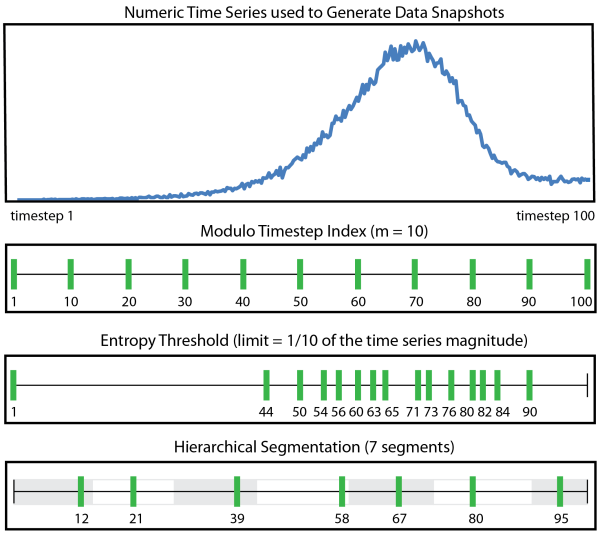


Fig. 3: Automatic timestep selection for a line chart based on three techniques. Modulo timestep indexing generates a snapshot every m steps. Entropy threshold selection generates snapshots when the data change passes a denoted limit. Hierarchical segmentation generates a user-defined number of segments (the alternating gray-white bars) and picks a single step to "represent" each.

Modulo Timestep Index This is the typical timestep selection process employed in simulations where data is saved at periodic intervals; i.e., a snapshot is selected for every m number of steps. This returns a linearly discretized selection of the data snapshots with constant step size.

Entropy Threshold The entropy is calculated between each successive timestep (or set of timesteps). When overall entropy exceeds a defined limit, that timestep is selected. This technique emphasizes timestep selection based on large-scale data changes, as opposed to generally flat or stable data.

Hierarchical Segmentation The *delta* (change) at each timestep is calculated. A hierarchical clustering is applied to the deltas to generate a hierarchical segmentation over all timesteps, then a cut is made for the desired number of segments to return. A "representative" timestep is selected for each segment based on a desired statistical metric (such as mean delta value or standard deviation). We currently use mean delta value to determine this.

We note here that these three techniques are based on a data abstraction; that is, snapshot selection is not linked to the chosen temporal layout technique or the currently displayed set of annotations. The reasoning for this is that an author might choose snapshot timesteps based on attributes disconnected to (independently of) these components. The used data attributes might not be shown in the temporal layout and are only included in the dataset specifically for snapshot selection.

Once loaded, appended snapshots can be interactively dragged, removed, or reset based on designer preferences. If a heuristic gives insufficient results then another may be tried or timesteps may be fully manually selected.

5 Annotations

Annotations visually point to salient data features and/or elements. They are the third component of a TSI, overlaid and anchored onto the temporal layout.

A major aspect of the TSI framework is its workflow for automatic annotation support, shown in Figure 4. This section describes this workflow. We start by explaining what data POIs are and how they are leveraged for annotation creation. Once created, annotations are scored and ranked. Before display, they must be properly positioned.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2016.2598876, IEEE Transactions on Visualization and Computer Graphics

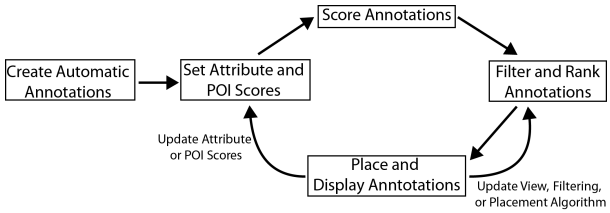
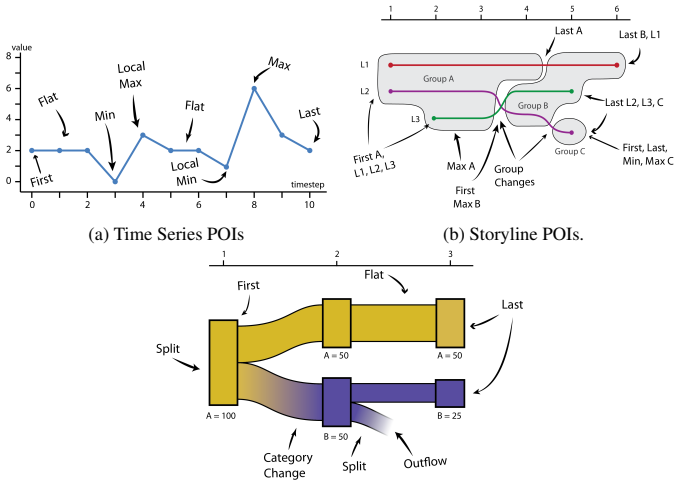


Fig. 4: The workflow for automatic annotations supports creation, placement, and interaction. Annotations are initially created from the POIs of data attributes. Based on POI type and attribute scores, the annotations are scored, filtered, ranked, placed, and displayed. User interaction triggers a prior step in the process.



(a) Time Series POIs (b) Storyline POIs.

(c) Alluvial Diagram POIs.

Fig. 5: Data aspects (termed POIs, or points of interest) that can be extracted from three types of time-dependent visualizations.

This entails filtering and selecting only the most highly ranked and relevant annotations, for which we introduce two placement algorithms. Interacting with the system, such as zooming, panning, deleting an undesired annotation, or updating an attribute's score triggers a prior step in the annotations workflow, forcing annotation re-ranking and potentially causing new annotations to be shown and old ones removed. This forms a user-in-the-loop cycle, where ranked annotations can be reviewed and used to guide analysis and exploration. Desired annotations can be pinned to the display, to explain relevant results or highlight aspects of the data.

5.1 Data Points of Interest (POIs)

An annotation describes a single aspect of the underlying dataset, which we refer to as a point of interest (POI). Conversely stated, a POI can be described using a single annotation. POIs are properties of data attributes like value extremas or changes, starts and stops, categorical, state, or flow changes, stable regions, and more. Though a single POI can apply to multiple elements of a data attribute (a flat region may span multiple timesteps, for example), it still refers to a single feature of the data. Hence, we use the term POI only and not ROI (region of interest) for description.

As noted in Section 2.4, annotations are either additive or observational depending on whether they reference visualized data or add additional information to the view. Our system allows for additive annotations to be manually created or queried for against background data attributes not shown in the temporal layout; they can also be loaded in the initial set of data used to build the TSI.

5.1.1 Types of POIs

Figure 5 displays examples of observational POIs that can be extracted from three types of data attributes: numerical vectors, storylines, and alluvial diagrams. Each shown POI corresponds to a single annotation.

Numerical Vector POIs A time series can be shown using a line chart (as in Figure 5a) or a stacked graph. The data used to show each line (or stacked graph layer) in the chart is an independent array of numbers (also called a numerical vector). By examining the vector's value features, numerical POIs can be extracted: the first and last instances of numbers, minimas and maximas, flat regions, and slope changes (increases and decreases over a set of timesteps).

Storyline POIs Figure 5b shows examples of POIs in storylines. Line-specific POIs are categorical, in that they describe the current state of a line (or set of lines). For example, line L1 starts at step 1 in group A, changes to group B at step 5, and ends at step 6. Alternatively, numerical POIs describe the groups that lines enter and leave. Group A starts at step 1 with a size of 2 (it contains two lines), has a maximum of 3 at steps 2 and 3, and so forth. Group A's numeric size vector would thus be [2, 3, 3, 1, *null*, *null*].

Alluvial Diagram POIs Figure 5c notes examples POIs in alluvial diagrams, which show data flow between groups over time. Flow size can change by having a part of a stream split off and go to another group, or by having flow from another group merge into the current stream. These POIs indicate the group's size has changed. They are both numeric and categorical because the flow changes a numeric amount and the groups that the flow goes between are qualitatively distinct. When flow is constant between steps, it is either in a stable region (i.e., no splitting or merging to other groups) or because the entire flow changes groups together. Numeric POIs based on group size can also be extracted, as is done for storyline groups.

POIs from Combined Attributes For each of the example visualizations, we note there are POIs that can be extracted from "combo attributes." A combo attribute is when two or more data attributes are combined via logical operators. In the storyline example, the maximum size of group [B AND C] is 3 at step 5. Line [L1 AND L3] first enters group B at step 5; line [L1 OR L3] does so at step 4. Lines can also be combined with groups to form combo attributes, for example, Line [L1 AND A] is the subset of line L1 when it is part of group A.

5.1.2 Manually Queried and Additive POIs

Since data attributes are either numeric or categorical vectors, they can be stored in a manner that permits SQL querying. Our TSI implementation stores data attributes using an HSQLDB cache [2] and includes a query interface. This allows for searching of more complex POIs that are not automatically extracted, as well as access to background data attributes not shown in the temporal display. In the cosmology case study (Section 6.1.2), one participant noted for his datasets he would use this feature to identify timesteps where data attributes are 50% of their maximum value. When a query is run, it creates a single annotation based on the timesteps(s) and attribute(s) it applies to.

If a manual query references only background data, then an additive annotation is created. This type of annotation is anchored to the timestep(s) to which it refers. Additive annotation can also be loaded at TSI initialization time, and used to refer to "outside" or contextual information beyond the scope of the immediate dataset. For examples of this, see the Appendix.

5.2 Automatically Created Annotations

In Figure 4, the first step in the annotations workflow is "Create Annotations." This only needs to be done one time, by parsing through each data attribute (and combo attribute) and identifying its POIs. Each POI creates one annotation. However, there is a problem in that this generates an overwhelming number of annotations. There might be far too many to simply append to the TSI. Our solution is only to show the most relevant or important ones.

To determine which annotations are "important," we use the notion that each created annotation has a score, represented as a single, numerical value. More than this, POIs and data attributes have their own

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2016.2598876, IEEE Transactions on Visualization and Computer Graphics

scores. To calculate an annotation's overall score, the scores of its relevant attribute and POI are multiplied together:

$$score_{annotation} = score_{attribute} \cdot score_{POI}$$

In this way, every annotation has its own score. Once the full set of created annotations has been scored, they can simply be sorted into a list. The highest ranked annotations are deemed the most important. (We give both manually-queried and loaded additive annotations the highest possible score/rank since they are assumed *a priori* to be important to the user, as they were manually created/loaded.)

As an example of scoring and ranking two automatic annotations, consider a data attribute A with a score of 50 ($score_A = 50$), and an attribute B with a score of 25 ($score_B = 25$). The POI denoting an attribute's global maximum has a score of 10 ($score_{max}$=10). With these values, the annotation denoting A's maximum will have a score of 500 while for B it will be 250. Ranking is by overall score, so the annotation Max_A will be ranked above Max_B.

Before display, the list of ranked annotations is filtered. Some annotations are discarded because they cannot currently be shown on the view. If the user has zoomed in on part of the display then annotations outside of the current window cannot be shown. Alternatively, an applied filter can hide some POIs types from display. When annotations are removed as candidates for display, lower-ranked annotations in the list move up. Annotations are selected and placed on the temporal layout based on the remaining list. First however, we discuss how automatic annotations scores are determined.

5.2.1 Scoring Data Attributes

Our implemented system includes an interface where a user sets the score for both data attributes and POI types. Combo attributes may be created in this dialog by combining two currently selected attributes. Attribute scores can also be saved to a configuration file to be applied at load time and then tweaked as needed. While our current system only supports manual attribute scoring, it is possible to automatically derive attribute scores and create combo attributes based on the dataset properties. See Section 7.2 for discussion of this.

5.2.2 Determining POI Scores

Like data attributes, POI types can be manually scored by the user. In the scoring example from the start of Section 5.2, the POI type "maximum" had a score of 10 ($score_{max} = 10$). This POI is considered "global" because each attribute only has one maximum value. (This is true even if the maximum happens over multiple timesteps. The value is the same.) Global POIs include features like "maximum," "minimum", "first," and "last."

However, what about local maximas? For a numerical attribute, these might occur multiple times; each instance is a POI. Intuitively, smaller extremas should be given less weight than larger ones, and their specific POI and subsequent annotation scores should be lower. Some time series compression algorithms use this assumption to weight extremas. Local extrema weights are determined using a distance function (such as the absolute distance between other extremas); very low-weighted points are then discarded from the compressed line. To weight local POIs, we use a modified version of the compression algorithm from [13] which has been extended to weight slopes and flat regions (where larger slopes and flat regions have higher weight).

To turn a local POI weight into a POI score, we normalize over the overall set of local POI weights for that attribute. For example, if the POI score for local maxima is set to 5 ($score_{local\_max} = 5$) and an attribute has three local maximas with compression weights of 28, 120, and 200, the corresponding POI scores are .7, 3, and 5.

Local POIs also exist in categorical data attributes. As an example of local POIs in storylines, first consider that a line can stay inside a group for multiple timesteps. In Figure 5b, line L1 goes from group A to B at step 5, a "group change" POI. The weight for this is based on how long L1 resided in group A before leaving; a longer time means a higher weight. Alluvial diagram POIs can be similarly weighted, except that the weight must also be scaled by the group's overall flow at the relevant timestep(s) of the POI.

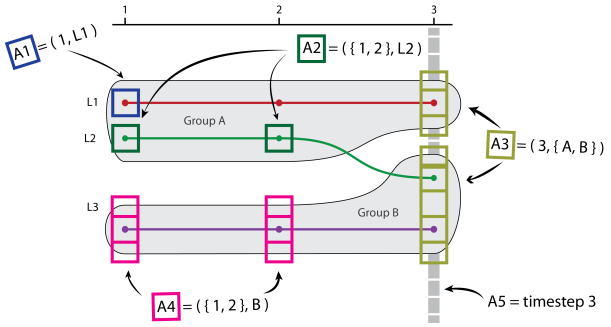


Fig. 6: There are five possible ways to place an annotation, depending on the timestep(s) and attribute(s) it maps to: ($A1$) a single timestep and vertical position. ($A2$ and $A3$) a single timestep and multiple vertical positions, or vice versa. ($A4$) a range of both timesteps and vertical positions. ($A5$) a timestep directly.

5.3 Annotation Placement

After annotations are created, scored, filtered, and ranked, they can be placed onto the temporal layout. Depending on an annotation's type, whether it is additive or observational, there are five ways it can be anchored to the view (Figure 6). These are determined by the timestep(s) and attribute(s) the annotation maps to, and the choice of the temporal layout. Observational annotations can map to a single (x,y) spot, to a spot fixed in one direction but with a range of positions in the other, or with a range in both dimensions. Additive annotations refer to a timestep (or set of timesteps) directly.

To determine how annotations are positioned on the temporal layout, we provide two placement algorithms (pseudocode is provided in the Appendix). The first, called the top-n ranked placement algorithm, chooses annotations for display based solely on rank. If a user is concerned only with seeing the highest-scored annotations (perhaps to focus his/her analysis on specific attributes or POIs types), this algorithm works well by always appending them to the display.

The second algorithm, called the density-based placement algorithm, is designed to distribute annotations over the display while keeping it from becoming cluttered. It does this by applying a weight field above the temporal layout and using a greedy placement strategy to try and place annotations. Prior-placed annotations exert a weight on the field, so lower-ranked annotations may not be able to be placed if they are below the threshold at their possible anchor spot(s).

5.3.1 Top-n Ranked Placement Algorithm

This algorithm takes as input the ranked list of annotations and a number n. It does a cut at the nth position, and the remaining set of annotations are then placed at their most "prominent positions."

An annotation's most prominent position is determined by the following logic: For additive annotations that map to a timestep only, the annotation is anchored at that timestep's x-position and halfway between the top of the temporal layout and the top of the view window. If the annotation maps to a single (x,y) position, that spot is used. If it maps to more than one timestep, the middle one is chosen for the x-position. Determining the y-spot depends on the data attribute the annotation maps to. If a single line (such as in a line chart or storyline), then the y-position of the line is used. For a set of lines, the middle-positioned one is used. If the annotation maps to a layer or group with a vertical height (such as a streamgraph layer or storyline group), the group with the largest vertical height at that timestep is selected and its middle y-position used. If the attribute maps to a combo attribute that contains both lines and groups, the line's middle-most y-position is used.

The top-n ranked placement algorithm has the advantage of showing only the highest scored (i.e., the "most important") annotations and doing so in their most prominent locations. However, this can sometimes cause occlusion when multiple annotations anchor to the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2016.2598876, IEEE Transactions on Visualization and Computer Graphics

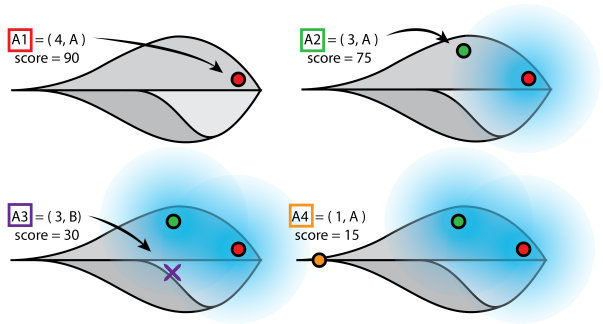


Fig. 7: An example of trying to place four annotations on a streamgraph using the density-based placement algorithm. *A*1 is placed at timestep 4 in layer A. Due to *A*1's score affecting the weight field, *A*2 finds the lowest-weighted vertical position in layer A at timestep 3. *A*3 cannot be placed, because its score is under the density field's weight at all positions in layer B at timestep 3. *A*4 has a lower score than *A*3, but can be placed because the density field has no weight there.

same (x,y) position. In this bothers the user, s/he can drag annotation anchors to other available positions to resolve this issue.

5.3.2 Density-Based Placement Algorithm

In the density algorithm, each annotation placed on the view applies a weight to its surrounding region. More important annotations (ones with a higher score) have a larger influence on their immediate area. If a region has no annotations near it, a lower-scored annotation can be appended if its score is above the field's density weight in that area. An area with multiple, highly-scored annotations will exert a strong weight on the surrounding region, keeping that part of the display from showing nearby, lower-scored annotations and becoming cluttered. Figure 7 shows an example of using this algorithm to try and place four annotations.

This algorithm works as follows: Starting with the highest ranked annotation, place it according to its "prominent position." Apply a kernel density estimate [37] using its score to the surrounding area. The distribution function can be user-defined, but we find both a normal and linear distribution work well. Next, begin iterating down the list of annotations. For each, of the potential (x,y) spots it is allowed to be placed, choose the one that has the lowest weight in the density field. If it's score is greater than the field's weight at this position, place the annotation on the view and apply its score to the density field. If the annotation can be equally placed at multiple (x,y) positions, choose the one closest to its "prominent position." Continue iterating until reaching the end of the list or until all remaining annotations are below the field's lowest threshold score (a number defined by the user).

When a user updates the view window, the list of possible annotations for display is reset and refiltered, so an annotation that was previously removed due to the weight of another might now be shown. This allows for annotations to "pop up" and fill in new areas as the user zooms and pans around the temporal layout. Another feature of this technique is that annotations not fixed to a single (x,y) position are allowed to "slide" to less weighted parts of the TSI when interaction happens. If the user doesn't like an annotation, deleting it allows for lower ranked annotations to appear in its vacated space.

6 Implementation and Evaluation

We have implemented a prototype application for TSIs, written in Java and Processing. For temporal layouts, it allows display with line charts, streamgraphs, and storylines. Data snapshots are stored as image files which are retrieved based on selected timesteps.

Annotations are shown as text boxes connected to their anchors by springs. A force directed layout ensures the text boxes achieve a nice distribution and do not occlude each other. For stylizing a TSI, force direction can be disabled and text boxes dragged to preferred positions. For each POI type, we use default "pretty print" sentences when generating annotations, but these can be edited.

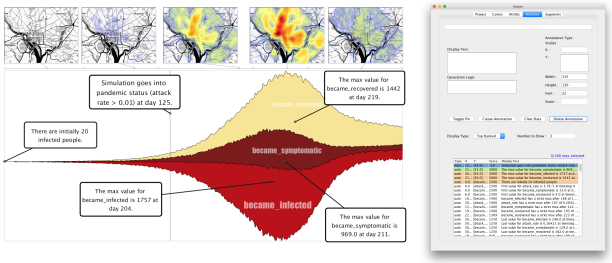


Fig. 8: Our TSI implementation's interface (viewer window and Helper panel) showing EpiSimS data with a streamgraph. The Helper panel's currently selected tab is used for interacting with annotations.

Manually-created annotations default to the highest rank, so they are always placed (unless outside the current view window). Annotations can also be "pinned" to the display. Our system lets users drag an annotation's anchor between its available (x,y) positions. Weighting parameters for the density-based placement algorithm can be adjusted, including the weight distribution radius and the field's lowest threshold score. POIs and attributes deemed unimportant by a user can either be deleted or have their scores set to zero. This ensures any referencing annotations will be ranked last in the sorted list and higher-scored annotations shown first.

Our prototype includes a separate Helper panel in addition to the main TSI view (shown in Figure 8). This panel handles most system interactions such as selecting a placement algorithm or editing data snapshot logic. It also displays the list of all created annotations (even ones not shown on the display) to allow for inspection and searching of specific annotation types that can be pinned to view.

6.1 Evaluation

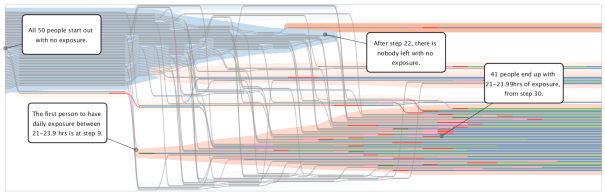
To validate our framework and demonstrate how TSIs can be used "in the wild," we performed two case studies with scientific datasets. The first was done over two sessions with three members of the EpiSimS team: a programmer and statistician both with 20+ years of experience, and a third-year postdoctoral physicist. The second case study was conducted with six cosmology researchers: a single session with three graduate students and one postdoctoral scholar, and two separate online interviews performed with graduate students. In both studies, participants interacted with our system and built TSIs (though the online sessions were demonstration only). Based on feedback and observations, we feel that TSIs are effective in assisting domain users analyzing their data and creating succinct, narrative visualizations.

6.1.1 Case Study One: EpiSimS Simulation

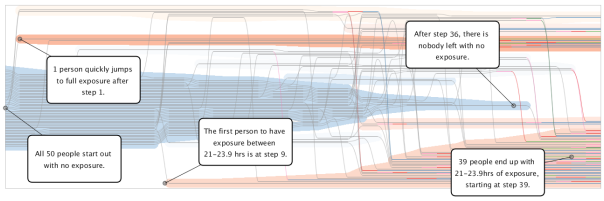
EpiSimS is a scalable, stochastic, agent-based model for infectious disease within the United States [30]. Based on input parameters like transmission rate, incubation period, mosquito population, and response mechanisms, a disease is introduced into a susceptible population where it spreads through agent-to-agent interaction. In a normal (non-TSI) workflow, team members select a set of input parameters and run an instance of the simulation. To examine the output data attributes, they usually create line charts and/or spatial views using a conventional tool like R or matplotlib.

With a TSI, these views are combined. Figure 8 shows a built TSI from the study's first session (data from [31]). A streamgraph shows three important temporal disease attributes. Data snapshots show the spatial disease evolution. With the top-*n* ranked placement algorithm, the team identified the maximum extremas of the three temporal attributes and pinned their annotations to the view, along with an annotation noting the initial infection seeding value (the POI type of "First" for the *became_infected* attribute). Finally, a manually-queried annotation marks this simulation's transition into "pandemic" status.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2016.2598876, IEEE Transactions on Visualization and Computer Graphics

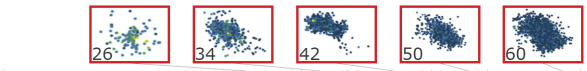


(a) Susceptibility parameter = 1.0.

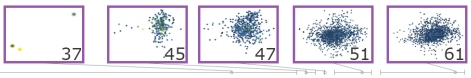


(b) Susceptibility parameter = 0.5.

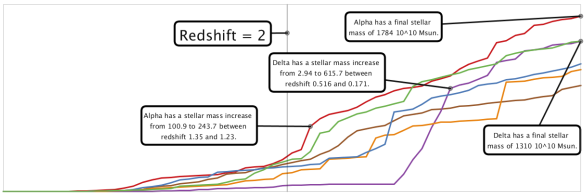
Fig. 9: Comparing two EpiSimS runs using annotated storylines (with data snapshots hidden) gives insight into how varying a susceptibility parameter between two simulation runs affects disease saturation rates in a population (where every person has moved into red groups).



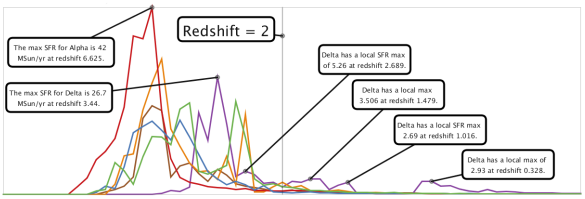
(a) Data snapshots for the Alpha merger tree.



(b) Data snapshots for the Delta merger tree.



(c) Temporal layout and annotations showing stellar mass over redshift.



(d) Temporal layout and annotations showing star formation rate over redshift.

Fig. 10: TSI for showing galaxy properties in a cosmological simulation. Annotations highlight the juxtaposition of the star formation rate and stellar mass properties by describing two of the merger trees.

In Figure 9, we show two storyline-based TSIs. These were created to show how using the density placement algorithm can help explain the visual features in a complex plot. In this case, two simulation outputs are compared to each other. Due to space constraints, discussion of these TSIs can be found in the Appendix.

Based on discussion with EpiSimS team members during the two conducted sessions of building TSIs and using the system, we received feedback about many aspects of the TSI workflow, summed below:

For data summarization and presentation: "*I like these Temporal Summary Images. They are easy to get feedback. For flu we might be interested in one set of lines, but for the mosquito-borne diseases something else. Having the snapshots up at the top gives you a feel for the geography, where the disease is going.*" "*There is a lot of benefit to this. These are the pictures I want to put in my publications.*"

For the annotations workflow: "*The annotations help point to what I care about. I need to know, how many people are infected at this time? How many are dead?*" and "*The automatic generation of annotations is very helpful to quickly see when significant changes occurred.*"

For the streamgraph-based TSIs, the team used the top-*n* ranked placement algorithm to explore the data. "*It's good to quickly see the highlights because you can narrow down what you want to further explore. Really good idea, really useful.*" The density placement algorithm was not seen as useful for the streamgraph example (Figure 8), due to the relatively consistent sloping behavior of the disease. "*It's too simple for the other [density] placement technique.*" Its utility was noted instead for the storyline figures, which were much more visually complex. "*Nice to see when and what significant changes occurred, to note when behaviors happened. Easier to explore this way.*"

6.1.2 Case Study Two: Millennium Cosmological Simulation

ΛCDM cosmological models such as the Millennium simulation [27, 38] are concerned with the structure and formation of the universe. For the second case study, we analyze galaxy properties that occur [8]. For a dataset, we retrieved galaxies attached to the main progenitor branch nodes of the six largest merger trees. For these galaxies, we view two important properties: stellar mass (SM) and star formation rate (SFR), which can be used to show structure and formation behavior trends. They occur in the context of an important universe-wide "astronomical POI," which marks the transition from the matter-dominated era to the dark energy-dominated era at approximately redshift $= 2$.

Figure 10 shows a TSI from this study. Line charts show the SM and SFR over redshift for the six selected merger trees. Two sets of data snapshots highlight the spatial formation of the Alpha and Delta merger trees. For both line charts, an additive annotation at redshift $= 2$ marks the "astronomical POI." Other annotations highlight salient extremas of the Alpha and Delta trees (the red and purple lines).

Juxtaposing the SM and SFR line charts highlights how the galaxy properties exhibit important behavior shifts on opposite sides of the redshift $= 2$ annotation. For SM, most growth happens in the dark energy-dominated era (to the right of redshift $= 2$). Annotations mark both the rapid increases and maximums for the Alpha and Delta trees. The Delta tree shows a particularly interesting rapid slope SM growth (i.e., slope change) from redshift 0.516 to 0.171. In the SFR chart, most activity happens to the left of redshift $= 2$. Annotations on the Alpha tree highlight that it has a large peak but then drops off for the rest of the simulation. Delta's maximum peak also occurs to the left of redshift $= 2$, but annotations note multiple local peaks to the right of it, including one well into the dark-energy dominated era. This happens at redshift 0.171, which coincides with its time period of rapid SM increase. This odd behavior could warrant further investigation of Delta's raw merger tree data (as noted by study participants).

Feedback and discussion from the cosmology participants was more mixed than the EpiSimS team. While most liked the TSI framework and saw its techniques as an improvement to their normal workflow, one online session participant stated he did not believe the TSI approach would be useful for his needs. Referring to the observational annotations, he felt that, "*labeling these only clutters the image.*" Other users felt differently though; their comments are curated below:

Automatic annotations were able to give an initial sense of the data: "*The notes are nice for giving a quick order of things like order of magnitude estimate.*" For more subtle or derived POIs, one participant would switch to querying. "*[Then] the manual queries are very useful. The things I look for depending on the graph change so much it might be unreasonable for a single system to cover all the criteria.*" This was echoed in two other users, who felt that annotations might have trouble showing the underlying significance or reasoning of a data POI:

One participant liked having the ability to switch between the two placement algorithms: "*So I like having flexibility for annotation placements. I think top-ranked annotations especially is useful if there is a certain event in time you want to mark, and take stock of what the numbers or features are there.*"

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2016.2598876, IEEE Transactions on Visualization and Computer Graphics

In comparing TSIs to their domain tools (mainly R and Python): "*We have trouble in the community coming up with good ways of showing graphs of things that evolve. This would definitely be useful for displaying data in papers, rather than what we normally use.*" In particular, one advantage TSIs provide is their focus on storytelling: "*I like the idea of being creative in adding snapshots and/or in-figure annotations. These things can help tell a story depending on what specifically you're trying to show or understand.*"

7 Discussion

Based on the design process and feedback from case studies, there are a number of points that can be discussed about the current state of the TSI framework and our prototype implementation.

7.1 Advantages of the TSI Framework

TSIs are designed to summarize data that have at least two strong dimensional axes, where one is always assumed to be *temporal*. While this bounds the framework to time-dependent datasets (using the four currently implemented temporal layout options), there exist a wide range of both general- and domain-specific datasets that can leverage our workflow for visualization creation and analysis.

Participants in both case studies noted they felt annotations guided their analytical perception of the data, and that the "presentation" aspect of TSIs was very visually appealing and persuasive. While Lee *et al*. have questioned if a single tool should combine both analysis and design processes [26], based on the set of design requirements for the TSI framework and feedback from case study participants, we feel it is justified for this problem space. In this respect, a TSI is more powerful than simple plotting like R or Python by because it leverages an interactive and analytic workflow that recommends snapshots and annotations to the user. With an end result of building visual data stories, it has advantages over a highly complex, technical, or multi-component approach designed purely for data analysis.

The intentional use of conventional (and even simple) visual components in a TSI is an additional strength. By not introducing novel visual representations, an author building a TSI for presentation knows that there is no inherent learning curve for the potential audience, especially since annotations are text-based, and can focus instead on analyzing and summarizing the data.

7.2 Automatically Scoring Data Attributes

As noted in Section 5.2.1, our current system only supports manual attribute scoring. However, it is possible to automatically derive attribute scores based on dataset properties. This could be especially useful when the number of attributes scales to an amount that makes individual scoring difficult or time-consuming. To automatically score time series, a distance-based metric such as Euclidean, Minkowski, or Manhattan distance can measure the similarity between each vector; there are similar techniques categorical data [9]. Each data attribute can be scored based on its overall similarity to other attributes in the dataset. A byproduct of this is that attributes found to be especially similar can be combined to form combo attributes. Another approach for automatic scoring is to use a data entropy or magnitude measure to measure the dataset's overall change or volatility and rank attributes based on this. Combo attributes can additionally be formed by combining attributes with similar scores.

7.3 Going Deeper, Wider, and into Snapshot Annotations

We currently are planning to expand our annotation creation process in three ways beyond the temporal, POI-based tagging currently used by the system. We can do this by going "deeper," "wider," and by integrating annotations with data snapshots. To go deeper means that we will include informational metrics or cues that help explain each created annotation's significance (the lack of this was noted by a participant in the cosmology case study). Going wider means including more types of POIs that the system recognizes such as statistical or derived metrics. We also plan to allow TSI authors to save manual POI queries that can later be retrieved to generate custom annotations, like a stored procedure in SQL.

Finally, there are plans to integrate annotations more closely with data snapshots. One way to do this is by allowing "data snapshot annotations," where a data snapshot frame can be directly appended to the temporal layout. This type of annotation can act as a "zoomed-in," data-orthogonal view for a specific data attribute on the display.

7.4 Current Design Limitations

While TSIs recommend annotations and data snapshot timesteps, for most other system interactions there is little user guidance. That is, the designer must explicitly choose options like which technique to use for the temporal layout and data snapshots, and what the appropriate color palette should be.

However, an assumption here is that TSI authors have a familiarity with the underlying dataset. As such, they can simply choose the same views they currently use in other plotting tools and leverage the advantages the TSI framework provides. While a recommendation system like Show Me [29] can be effective in suggesting a new visual projection to a user, the authors of that paper note that once a user has "settled" on a set of preferred views that usage of this recommendation feature dramatically drops. Therefore, the lack of recommendations for design choices like this becomes negligible if the user knows what general types of views work well for their data.

TSIs also provide no guidance for issues like choosing the optimal number of data snapshots to show. Obviously, choosing too many will clutter the display (our implementation shrinks snapshots based on the number displayed, but there is a minimum size limit). This same cluttering can happen if too many annotations are appended to the display. We note however, that while issues like this can be addressed with different timestep selection and placement algorithms, our system easily allows a user to interactively modify constraints for these components. If the view is cluttered, they can change the necessary settings to clear the display and fix this type of problem.

While aesthetically our current TSI editing options are mostly considered "*up to the task*" via case study feedback, we are working on ways to expand how a user can tweak and style the view. A recent paper describing the GraphCoiffure system [39] presents a set of techniques for improving user workflow to create presentation-style images of graph networks; approaches like this can be integrated into our system to improve flexibility and speed up the design process.

Finally, though (as summarized by one case study participant) the annotation placement algorithms, "*seem to work well*," our current results and feedback for these are colloquial. A lack of formal evaluation is a current limitation, and we plan to perform a full usability study to determine optimal usage practices and strategies.

8 Conclusions

We present Temporal Summary Images, a new approach to making explanatory visualization through the process of interactive exploration. By leveraging "under the hood" techniques to assist user analysis and design, we ease the process of creating narrative visualizations for complex or multidimensional datasets, helping bridge the gap between data exploration and storytelling.

A TSI is an apt analytics and summarization technique for many datasets, and we provide examples of its use in both general and scientific domains, along with domain user feedback validating our approach. Future work will focus on maturing and improving our implemented framework and expanding it to new features and techniques.

Acknowledgments

The authors wish to thank Annie Preston (apreston@ucdavis.edu) of VIDi Labs at the University of California, Davis, for assistance with the cosmological case study. This research was sponsored in part by INGVA/LANL, US National Science Foundation via grants DRL-1323214, IIS-1528203, and IIS-1320229, and U.S. Department of Energy via grant DE-FC02-12ER26072. The Millennium Simulation databases used in this paper and the web application providing online access to them were constructed as part of the activities of the German Astrophysical Virtual Observatory (GAVO).

REFERENCES

[1] Google docs: Quickly get insights on a spreadsheet using explore. https://support.google.com/docs/answer/6280499. Accessed February 02, 2016.

[2] HSQLDB. http://hsqldb.org/. Accessed January 20, 2016.

[3] Pew research center. http://www.pewresearch.org. Accessed January 04, 2016.

[4] Publications: Department of homeland security. https://www.dhs.gov/publications/. Accessed January 04, 2016.

[5] W. Aigner, S. Miksch, H. Schumann, and C. Tominski. *Visualization of Time-Oriented Dsata*. Springer Science & Business Media, 2011.

[6] W. Aigner, A. Rind, and S. Hoffmann. Comparative evaluation of an interactive time-series visualization that combines quantitative data with qualitative abstractions. In *Computer Graphics Forum*, volume 31, pages 995–1004. Wiley Online Library, 2012.

[7] B. Bach, N. Kerracher, K. Hall, S. Carpendale, J. Kennedy, and N. Riche. Telling stories about dynamic networks with graph comics. In *Proceedings of the Conference on Human Factors in Information Systems (CHI)*. ACM, New York, United States, 2016.

[8] S. Bertone, G. De Lucia, and P. Thomas. The recycling of gas and metals in galaxy formation: predictions of a dynamical feedback model. *Monthly Notices of the Royal Astronomical Society*, 379(3):1143–1154, 2007.

[9] S. Boriah, V. Chandola, and V. Kumar. Similarity measures for categorical data: A comparative evaluation. *red*, 30(2):3, 2008.

[10] L. Byron and M. Wattenberg. Stacked graphs–geometry & aesthetics. *Visualization and Computer Graphics, IEEE Transactions on*, 14(6):1245–1252, 2008.

[11] A. Cairo. *The Functional Art: An introduction to information graphics and visualization*. New Riders, 2012.

[12] W.-T. Chu, C.-H. Yu, and H.-H. Wang. Optimized comics-based storytelling for temporal image sequences. *Multimedia, IEEE Transactions on*, 17(2):201–215, 2015.

[13] E. Fink and H. S. Gandhi. Compression of time series by extracting major extrema. *Journal of Experimental & Theoretical Artificial Intelligence*, 23(2):255–270, 2011.

[14] T. Gao, J. Hullman, E. Adar, B. Hecht, and N. Diakopoulos. Newsviews: An automated pipeline for creating custom geovisualizations for news. In *Proceedings of the 32nd Annual ACM Conference on Human Factors in Computing Systems*, pages 3005–3014. ACM, 2014.

[15] N. Gershon and W. Page. What storytelling can do for information visualization. *Communications of the ACM*, 44(8):31–37, 2001.

[16] S. Gratzl, A. Lex, N. Gehlenborg, N. Cosgrove, and M. Streit. From visual exploration to storytelling and back again. In *Eurographics Conference on Visualization (EuroVis) 2016*. ACM, 2016.

[17] J. Heer, F. Viégas, and M. Wattenberg. Voyagers and voyeurs: supporting asynchronous collaborative information visualization. In *Proceedings of the SIGCHI conference on Human factors in Computing Systems*, pages 1029–1038. ACM, 2007.

[18] D. Hoffman and M. Singh. Salience of visual parts. *Cognition*, 63(1):29–78, 1997.

[19] J. Hullman and N. Diakopoulos. Visualization rhetoric: Framing effects in narrative visualization. *Visualization and Computer Graphics, IEEE Transactions on*, 17(12):2231–2240, 2011.

[20] J. Hullman, N. Diakopoulos, and E. Adar. Contextifier: automatic generation of annotated stock visualizations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2707–2716. ACM, 2013.

[21] J. Jin and P. Szekely. Interactive querying of temporal data using a comic strip metaphor. In *Visual Analytics Science and Technology (VAST), 2010 IEEE Symposium on*, pages 163–170. IEEE, 2010.

[22] E. Kandogan. Just-in-time annotation of clusters, outliers, and trends in point-based data visualizations. In *Visual Analytics Science and Technology (VAST), 2012 IEEE Conference on*, pages 73–82. IEEE, 2012.

[23] C. N. Knaflic. *Storytelling with Data: A Data Visualization Guide for Business Professionals*. John Wiley & Sons, 2015.

[24] N. Kong and M. Agrawala. Graphical overlays: Using layered elements to aid chart reading. *Visualization and Computer Graphics, IEEE Transactions on*, 18(12):2631–2638, 2012.

[25] R. Kosara and J. Mackinlay. Storytelling: The next step for visualization. *Computer*, (5):44–50, 2013.

[26] B. Lee, N. Riche, P. Isenberg, and S. Carpendale. More than telling a story: Transforming data into visually shared stories. *IEEE Computer Graphics and Applications*, 35(5):84–90, Sept 2015.

[27] G. Lemson et al. Halo and galaxy formation histories from the millennium simulation: Public release of a vo-oriented and sql-queryable database for studying the evolution of galaxies in the lambdacdm cosmogony. *arXiv preprint astro-ph/0608019*, 2006.

[28] K.-L. Ma, I. Liao, J. Frazier, H. Hauser, and H.-N. Kostis. Scientific storytelling using visualization. *Computer Graphics and Applications, IEEE*, 32(1):12–19, 2012.

[29] J. Mackinlay, P. Hanrahan, and C. Stolte. Show me: Automatic presentation for visual analysis. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1137–1144, 2007.

[30] S. Mniszewski, S. Del Valle, P. Stroud, J. Riese, and S. Sydoriak. Episims simulation of a multi-component strategy for pandemic influenza. In *Proceedings of the 2008 Spring Simulation Multiconference*, pages 556–563. Society for Computer Simulation International, 2008.

[31] S. Mniszewski, C. Manore, C. Bryan, S. Del Valle, and D. Roberts. Towards a hybrid agent-based model for mosquito borne disease. In *Proceedings of the 2014 Summer Simulation Multiconference*, page 10. Society for Computer Simulation International, 2014.

[32] A. Moere and H. Purchase. On the role of design in information visualization. *Information Visualization*, 10(4):356–371, 2011.

[33] M. Rosvall and C. T. Bergstrom. Mapping change in large networks. *PLOS ONE*, 5(1):e8694, 2010.

[34] A. Satyanarayan and J. Heer. Authoring narrative visualizations with ellipsis. In *Computer Graphics Forum*, volume 33, pages 361–370. Wiley Online Library, 2014.

[35] E. Segel and J. Heer. Narrative visualization: Telling stories with data. *Visualization and Computer Graphics, IEEE Transactions on*, 16(6):1139–1148, 2010.

[36] Y. Shrinivasan and J. van Wijk. Supporting the analytical reasoning process in information visualization. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 1237–1246. ACM, 2008.

[37] B. Silverman. Density estimation for statistics and data analysis. In *Mono. on State and Appl. Probability*. Chapman & Hall, 1992.

[38] V. Springel, S. D. White, A. Jenkins, C. S. Frenk, N. Yoshida, L. Gao, J. Navarro, R. Thacker, D. Croton, J. Helly, et al. Simulations of the formation, evolution and clustering of galaxies and quasars. *Nature*, 435(7042):629–636, 2005.

[39] A. Spritzer, J. Boy, P. Dragicevic, J.-D. Fekete, and C. Dal Sasso Freitas. Towards a smooth design process for static communicative node-link diagrams. In *Computer Graphics Forum*, volume 34, pages 461–470. Wiley Online Library, 2015.

[40] Y. Tanahashi and K.-L. Ma. Design considerations for optimizing storyline visualizations. *Visualization and Computer Graphics, IEEE Transactions on*, 18(12):2679–2688, 2012.

[41] E. Tufte. *The visual display of quantitative information*. Number v. 914 in The Visual Display of Quantitative Information. Graphics Press, 1983.

[42] Y. Wang, T. Wu, Z. Chen, Q. Luo, and H. Qu. Stac: Enhancing stacked graphs for time series analysis. In *Pacific Visualization Symposium (PacificVis), 2016 IEEE*, pages 234–238. IEEE, 2016.

[43] T. Wu, Y. Wu, C. Shi, H. Qu, and W. Cui. Piecestack: Toward better understanding of stacked graphs. *Visualization and Computer Graphics, IEEE Transactions on*, 22(6):1640–1651, 2016.

[44] J. Zhao, F. Chevalier, E. Pietriga, and R. Balakrishnan. Exploratory analysis of time-series with chronolenses. *Visualization and Computer Graphics, IEEE Transactions on*, 17(12):2422–2431, 2011.

[45] Z. Zhao, W. Benjamin, N. Elmqvist, and K. Ramani. Sketcholution: Interaction histories for sketching. *International Journal of Human-Computer Studies*, 82:11–20, 2015.

[46] Z. Zhao, R. Marr, and N. Elmqvist. Data comics: Sequential art for data-driven storytelling. Technical report, Technical Report. Human Computer Interaction Lab, University of Maryland, 2015.