

Time Curves: Folding Time to Visualize Patterns of Temporal Evolution in Data

Benjamin Bach, Conglei Shi, Nicolas Heulot, Tara Madhyastha, Tom Grabowski, Pierre Dragicevic

▶ **To cite this version:**

Benjamin Bach, Conglei Shi, Nicolas Heulot, Tara Madhyastha, Tom Grabowski, et al.. Time Curves: Folding Time to Visualize Patterns of Temporal Evolution in Data. IEEE Transactions on Visualization and Computer Graphics, 2016, 22 (1), 10.1109/TVCG.2015.2467851 . hal-01205821

HAL Id: hal-01205821
https://inria.hal.science/hal-01205821

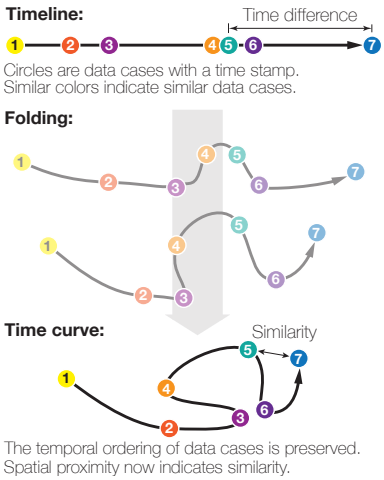
Submitted on 29 Nov 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

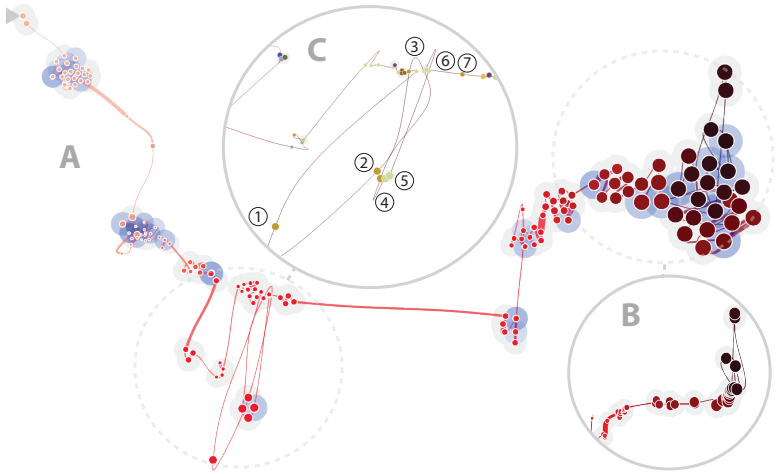
L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Time Curves: Folding Time to Visualize Patterns of Temporal Evolution in Data

Benjamin Bach, Conglei Shi, Nicolas Heulot, Tara Madhyastha, Tom Grabowski, Pierre Dragicevic



(a) Folding time



(b) History of the Wikipedia article on Palestine

Fig. 1. The time curve principle: a) a timeline is folded into itself in such a way that similar time points end up being close to each other; b) Example: a time curve showing the evolution of a Wikipedia article.

Abstract—We introduce *time curves* as a general approach for visualizing patterns of evolution in temporal data. Examples of such patterns include slow and regular progressions, large sudden changes, and reversals to previous states. These patterns can be of interest in a range of domains, such as collaborative document editing, dynamic network analysis, and video analysis. Time curves employ the metaphor of folding a timeline visualization into itself so as to bring similar time points close to each other. This metaphor can be applied to any dataset where a similarity metric between temporal snapshots can be defined, thus it is largely datatype-agnostic. We illustrate how time curves can visually reveal informative patterns in a range of different datasets.

Index Terms—Temporal data visualization, information visualization, multidimensional scaling

1 Introduction

A large portion of the information we produce is temporal: video recordings, revision histories, meteorological records, brain scans, or any digital collection that contains entities recorded at different times. All such information artefacts reflect dynamic processes with possibly complex patterns of evolution. For example, an article being written can stagnate or progress quickly, or can undergo reversals in case of a disagreement between multiple authors. Brain activity can vary between different states, reflecting changing external stimuli and cognitive processes. Weather is chaotic in the short run but follows steady cyclic patterns on a larger scale (seasons), and trends on an even larger scale (climate). All such temporal patterns that can be of great interest to domain experts or to a more general audience.

Many temporal data exploration tools have been developed that can help better understand such patterns (for reviews see [2, 5, 8]), but they are typically domain-specific or assume a particular data structure, such as multidimensional tabular data. Yet information artefacts are diverse and many of them are largely unstructured (e.g., plain text or photos). Developing specialized visualization tools for each possible domain and type of dataset can be costly and impractical. Thus we need to develop more visual representations of temporal data that can be applied to a range of datasets. Such visual representations can not only help to reduce production costs, but can also be learned once for all and become part of the repertoire of charts routinely used in public communication. By introducing *time curves*, we show that while each temporal dataset is different, many such datasets share similar high-level patterns of temporal evolution that do not necessarily require elaborate and specialized techniques to be seen.

The time curve technique is a generic approach for visualizing temporal data based on self-similarity. It only assumes that the underlying information artefact can be broken down into discrete *time points*, and that the *similarity* between any two time points can be quantified through a meaningful metric. For example, a Wikipedia article can be broken down into *revisions*, and the *edit distance* can be used to quantify the similarity between any two revisions. A time curve can be seen

- *Benjamin Bach is with Microsoft Research-Inria Joint Centre. E-mail: benj.bach@gmail.com.*
- *Conglei Shi is with the IBM T.J, Watson Research Center, Yorktown Height, NY. E-mail: shiconglei@gmail.com.*
- *Nicolas Heulot is with IRT SystemX: nicolas.heulot@irt-systemx.fr*
- *Pierre Dragicevic is with Inria: pierre.dragice@gmail.com*
- *Tara Madhyastha is with the Department of Radiology at University of Washington. E-mail: madhyt@u.washington.edu.*
- *Tom Grabowski is with the Department of Radiology and Neurology at University of Washington. E-mail: tgrabow@u.washington.edu.*

Manuscript received 31 Mar. 2015; accepted 1 Aug. 2015; date of publication xx Aug. 2015; date of current version 25 Oct. 2015. For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

as a timeline that has been folded into itself to reflect self-similarity (see Figure 1(a)). On the initial timeline, each dot is a time point, and position encodes time. The timeline is then stretched and folded into itself so that similar time points are brought close to each other (bottom). Quantitative temporal information is discarded as spacing now reflects similarity, but the temporal ordering is preserved.

Figure 1(b), curve *A*, shows a concrete example: the evolution of a Wikipedia article on Palestine. Each dot is an article revision, where dot brightness encodes revision time (the darker the more recent) and dot size encodes article length. Here, time goes from left to right. At first, the article evolves slowly and irregularly, then it goes through a controversy phase after which it undergoes a large change. This change allows to reach a consensus and the article grows dramatically, after which it remains stable and experiences only minor revisions.

Time curves employ multi-dimensional scaling (MDS) [46] to embed time points in 2D space. Crucially, time curves add the temporal component that so far has been missing in standard MDS visualizations. Instead of being shown separately, data points are *laid out on a curve*, which conveys temporal ordering and high-level progression patterns through its shape. It also brings additional design and implementation challenges, such as how to draw the curve in such a way that the visualization remains legible. We also introduce a range of improvements over traditional MDS visualizations, including overlap removal and highlighting of identical time points. Besides the generic technique, we also suggest dataset-specific adaptations that can optimize time curves for well-identified families of datasets.

Time curves are not meant to replace domain-specific temporal visualizations, as in many cases such visualizations may be required for a deeper examination of the data once interesting patterns have been identified. The goal of time curves is to offer a generic way of producing simple visual overviews for a range of temporal datasets. Throughout this article, we go through several examples to illustrate how informative and useful such overviews can be.

After reviewing previous work on time and similarity visualization, we consider several use cases involving different types of datasets (document edits, videos, weather data, and dynamic networks) and discuss the insights that can be gained by using time curves. We then report on a collaboration with neuroscientists where time curves were used to visualize dynamic brain connectivity data. We then provide a typology of time curve patterns, discuss implementation considerations, and finally conclude with a discussion of the strengths and limitations of time curves, as well as possible future work.

2 Related Work

Time curves relate to both the visualization of time and the visualization of similarity in datasets.

2.1 Visualizing Temporal Data

Temporal data is ubiquitous and often considered as requiring dedicated visualization tools [44] and a remarkably large variety of temporal visualization techniques have been proposed (for reviews see [2, 8, 5]), many of which are domain-specific. For example, History Flow [48] gives an overview of Wikipedia edit histories, and is very effective at conveying changes over time and revealing typical patterns such as edit wars. While very useful for most text edit histories, the visualization is not applicable to other datasets.

Other temporal visualization techniques are more generic and can be applied to a wider range of datasets. Among them are a wide class of visualizations that can be derived from a space-time cube representation [5],such as small multiples, time-flattened views (e.g., connected scatterplots), animations and 3D space-time cubes. Such techniques are generally effective, but they assume that each time point has an inherent 2D spatial representation (e.g., a map or a 2D scatterplot). Another common approach is the use of parallel line charts [2], but this technique assumes multidimensional scalar data in tabular form. Many datasets are not that neatly structured, Wikipedia edit histories being one example. Other examples are covered in Section 3.

By only requiring a similarity or distance metric to be defined between time points, time curves cover a wide and complementary spectrum of datasets and use cases. Conversely, time curves are naturally not optimized for specific datasets, nor are they optimal for multidimensional temporal data since they do not show multiple attributes at a time. However, they can possibly simplify the visualization of high-dimensional datasets by reducing them to a planar curve.

2.2 Visualizing Similarity Data

Some datasets such as subjective similarity judgments from psychology studies only consist in a set of similarity measurements between nominal entities [24]. Shaded matrix displays [17] are widely used for visualizing such data, but they can be hard to read [19].

A popular alternative is multidimensional scaling (MDS). Similar to spring based network layouts (e.g. [22]), MDS lays out data points on a low-dimensional space so that Euclidian distances reflect similarities between data points [46]. Despite the large body of theoretical work on MDS, conveying time has not been a concern. Time curves show how MDS visualizations can be extended to show temporal information, thus opening up a range of new possibilities for the analysis of dynamic processes.

2.3 Visualizing Both Time and Similarity

There has been some work on visually conveying both time and similarity. With only two time points, difference views can be used where color typically encodes changes. They have been successfully used with text [10] and graphs [21], but each implementation is domain-dependent. Also, difference views do not scale up to many time points.

In biology and linguistics, similarity data is routinely used to build phylogenetic trees [29]. The temporal structure conveyed is hierarchical rather than linear [2], and it is reconstructed from the data.

Arc Diagrams [49] show repetition patterns over a linear timeline, but the notion of similarity is binary: time points are either similar or dissimilar. MultiPiles [6] is a system for visualizing dynamic networks that can group time points, represented as adjacency matrices, by piling them according to their similarity. It also assume a dichotomous view of similarity. None of these systems is able to convey complex evolution patterns involving different magnitudes of change.

Pless [37] introduced "video trajectories", i.e., 2D and 3D curves that convey neighborhood information between video frames. While video trajectories share many similarities with time curves, they are targeted at video analysis. In in this article we focus on being data-agnostic and aim at illustrating how such an approach can benefit many areas besides video analysis. We delve deeper into the human factor aspects by reporting on a real case study, analyzing time curve patterns, and discussing the many design issues. In Section 6 we further highlight the technical differences between Pless' approach and ours.

Finally, some visualization techniques such as LineSets [4] visually resemble time curves, but they convey neither time nor similarity and are thus outside the scope of this work.

3 Use Case Scenarios for Time Curves

We implemented several interactive prototypes to generate and interactively explore time curves (listed in Section 6). The time curves we discuss here were generated with a simplified web app.[1] The web app takes similarity data in the form of a distance matrix, with optional labels (time stamps or otherwise) for each data point. Support for interaction is minimal: labels are shown when hovering the dots, and dense areas can be enlarged with a fisheye lens. Here we focus on the visualization aspects. Technical details about the distance metrics employed are given in Section 6.

3.1 Exploring Document Histories

Wikipedia articles are collaborative documents whose evolution over time can carry a deep meaning. Such evolutions reflect diversities of opinions as well as controversies occurring at a societal level, possibly under the influence of the media. For someone who is interested in the history of an article, questions include: "is this article active or stable?", "is it controversial?", or "did it go through different stages?".

[1] www.aviz.fr/~bbach/timecurves

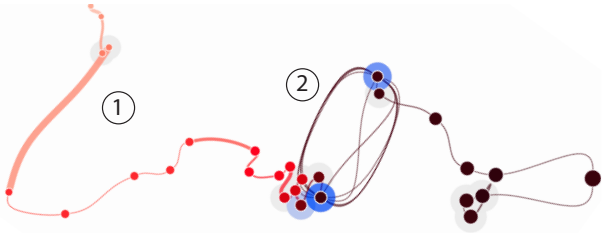


Fig. 2. Wikipedia article on *Chocolate* showing an edit war in stage (2). Blue halos indicate identical revisions.

Progress and Stagnation—Every Wikipedia article has a unique story, and thus a unique time curve. For example, the time curve in Figure 1(b) reveals that the article on *Palestine* underwent three stages, including turbulences in the form of zig-zag patterns suggesting a controversial stage. The controversy is then resolved and revisions become large and clustered, suggesting maturity. The time curve in Figure 2 (*Chocolate*) is stable overall, except for a stage where the curve alternates between the exact same revisions (blue halos), suggesting a so-called "edit war".

A time curve can give Wikipedia readers a rapid overview of an article's writing process, and possibly give them cues as to whether the article can be trusted [11]. As articles never stop being edited, time curves can also benefit contributors who monitor specific articles over time. Time curves make it easy to spot patterns that can be further examined using "detail-on-demand" techniques. Our web app does not support such techniques, thus for our scenarios we used Wikipedia's history system to examine differences between revisions of interest.

Similar and Identical Revisions—Clusters are groups of highly similar revisions which only differ by minor edits. In clusters, dots can overlap significantly. To improve legibility and clearly show cluster cardinality, time curves implement an overlap removal mechanism. This can be seen in the final stage of Figure 1. Dots which have been spread apart are displayed with a gray halo, reinforcing the visual impression of a cluster. If exact positions are important, overlap removal can be disabled globally or through an interactive lens. For example, the inset B on Figure 1(b) shows that the final stage of *Palestine* undergoes a gradual progression with a dense cluster at the end suggesting a stabilization.

Overlap removal is not applied to identical revisions. Instead, these are superimposed and shown with a blue halo – the darker the blue, the more points have been superimposed. In Figure 2, the two blue halos are rather dark, suggesting a long edit war. One of the opponents finally won and the article continued to progress. The oscillation pattern at the bottom left of Figure 1(b) shows an "informal edit war". No explicit revert was employed, but instead portions of the article were repeatedly changed back to previous versions. Thus, there is still a progression, but a very inefficient one. Finally the community did not give way and the article mostly went back to where it was. This article contains occasional blue halos, which correspond to minor formatting changes not captured by our similarity metric.

User Contributions—The Wikipedia version of time curves can allocate colors to users and show who is responsible for each change. Inset C in Figure 1(b) is a zoomed-in view of the informal edit war in the middle of the curve. Mostly two users were involved: Brown and Green. The history on Wikipedia reveals that ① Brown changed a specific paragraph; ② Brown then reintroduced part of the original text but kept his insertions; ③ Green then removed most changes made by Brown and ④ Brown reintroduced them; ⑤ Perhaps without noticing Brown's edits, Green made a minor change elsewhere in the article; ⑥ After noticing, Green removed Brown's edits again; Finally, ⑦ Brown gives up and inserts a single sentence stating that the topic is still debated.

Figure 3(a) on *Erich Honecker* shows another example. Initially, the article was edited almost exclusively by user Blue, in a cumulative fashion. The final edits are minor clarifications and rewordings by other contributors. The time color encoding (Figure 3(b)) reveals that Blue's edits were made in a very short amount of time (all points are bright pink), while the later minor edits spanned a longer time period. Thus this is a non-controversial article that quickly stabilized.

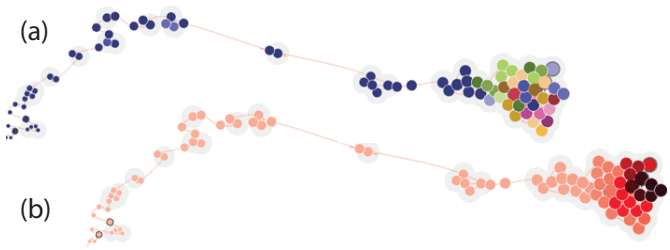


Fig. 3. Wikipedia article about former leader of German Democratic Republic *Erich Honecker*, using different color encodings for nodes.

Vandalism—In cases of "vandalism", parts or the entire article are removed or replaced by irrelevant content. Acts of vandalism are visible as extreme outliers on a time curve. Figure 4 shows the curve for *Crimea*, where a user deleted the entire article and inserted a redirect to *Putin*.



Fig. 4. Vandalism on the Wikipedia article *Crimea*. The time point on the right (very small) is a revision that contains a single link only.

Visual Signatures—Figure 5 shows time curves for multiple Wikipedia articles. Even though the curves are scaled down, their main visual characteristics are maintained. Time curves can thus serve as visual signatures or thumbnails when navigating across multiple datasets. We can see that the majority of the depicted Wikipedia articles contains clusters of minor revisions, while approximately half of them exhibit a monotonic progression. The other half involves vandalism or edit wars. Such visualizations could, for example, help Wikipedia administrators navigate collections of articles within a given category and monitor anomalies.

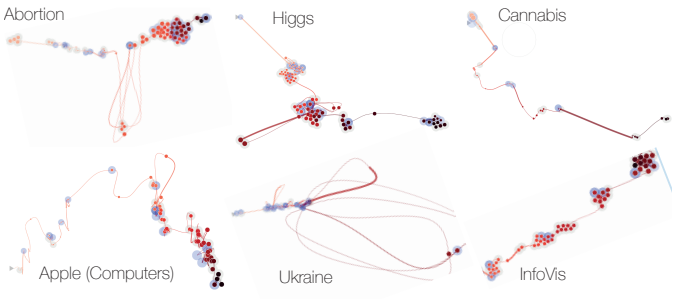


Fig. 5. Time curve signatures of different Wikipedia articles.

3.2 Video Recordings

Video recordings seem to have little in common with document revision histories, yet they are also information artefacts reflecting changes over time. Thus video time curves can be produced where *time points* are *video frames* or groups of adjacent videos frames, and their *similarity* is estimated through *image similarity* computation. Questions related to video analysis include: "when do sudden changes happen in this security camera footage?" or "how is this movie structured?".

Surveillance Videos—Surveillance videos often consist in mostly static scenes, or scenes with constant motion such as highways, sidewalks or counters. Interesting moments typically appear as outliers on a time curve. Figure 6 shows a one-minute footage from a security camera.[2] In this figure and all following figures, annotations on time points (labels and photos) are by us. In the central cluster (e.g., time

[2] https://www.youtube.com/watch?v=L8WV9wLBzdg

point 86), only minor changes happen, such as slight changes in illumination or moving leaves. Outlier dots indicate frames where people cross the scene (e.g., time points 42 and 44). The original footage shows a complex scene, recorded in bad quality and in black and white. We found that the time curve shows outlying frames more clearly than the video stills.

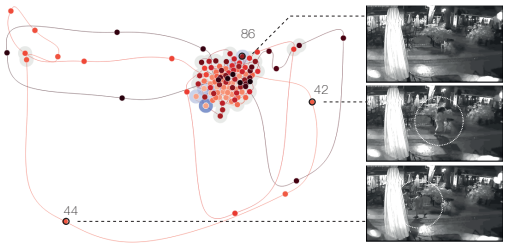


Fig. 6. A surveillance video of a street. Outliers are passing pedestrians.

Movie Analysis—Figure 7 shows an eight-minute animated film.[3] Although the curve is much harder to follow, a structure can be seen. Different scenes appear as clusters. The movie mostly employs scene cuts, without camera motion or transition effects. Some scenes are visited twice, such as at time points 55 and 105. The large-scale structure of the movie can be inferred from dot colors. The brightest and darkest dots are clumped together on the top left, suggesting little action at the beginning and at the end of the movie. In contrast, red dots undergo large changes, with frequent scene cuts. This seems to follow a common pattern in dramaturgy. See Pless [37] for other examples of video analysis using a similar method. Video time curves can give an initial rough overview of the dynamic structure of a movie. They may also be useful as visual signatures (see Figure 5), or serve as mental maps to navigate videos. Compared to a straight timeline or seeker bar, a time curve could facilitate video navigation by providing recognizable visual landmarks that can help seek, memorize and revisit scenes of interest [14].

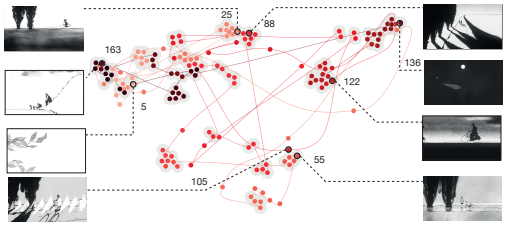


Fig. 7. Animated movie. Scenes appear as clusters. Video stills are not ordered chronologically, but placed next to the corresponding time point on the curve, showing similar scenes and scenes that get revisited.

3.3 Analyzing Dynamic Visualizations

There exist temporal datasets that are hard to access or visualize, but for which dynamic visualizations exist and are easily available. One example is weather data, for which animated map visualizations are regularly created and made available to the public. By treating such visualizations as videos, it is possible to create time curves that uncover patterns and features in the data that are not necessarily visible on the visualizations themselves. Such an approach essentially consists of generating overviews of complex dynamic datasets by using existing animated visualizations as proxies to the underlying data.

Precipitation Patterns—The time curve in Figure 8 summarizes a video of cloud coverage and precipitations over one year.[4] It reveals large-scale changes across the entire year, with oscillations of about a week. January and August are two extremes. On December, the weather does not come back to where it was on January, but instead to where it was on April. Although the video itself provides much more spatial and temporal details, variations occurring at multiple scales are hard to see (right of Figure 8).

[3] https://www.youtube.com/watch?v=pePhP-qRzSc
[4] http://tinyurl.com/colorado-global

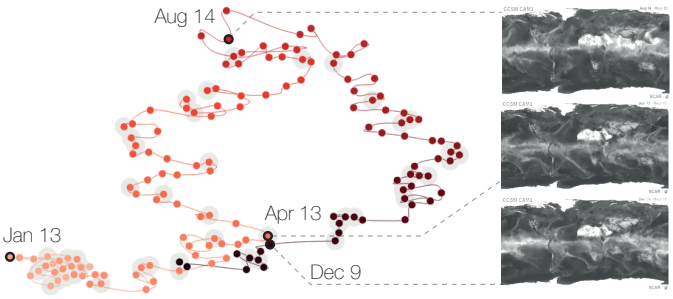


Fig. 8. Worldwide cloud coverage and precipitation over one year, as shown by an animated map visualization.

Figure 9 similarly shows precipitation across the United States over the course of one year (averaged 1981-2010).[5] The curve closes itself at the end of the year, suggesting a yearly cycle. It also crosses itself, revealing that geographical precipitation patterns around October/November were the same as in March/April. In contrast to the previous example, the curve does not show local oscillations. We can also see three extrema: (i) low precipitation in the center of the country but high on the coasts (Nov to March), (ii) generally high precipitation (May to June), and (iii) low precipitation on the West part, and high on the East part (July to Sep). From the video alone, it is hard to spot these three extrema, as well as the relatively smooth transitions between them.

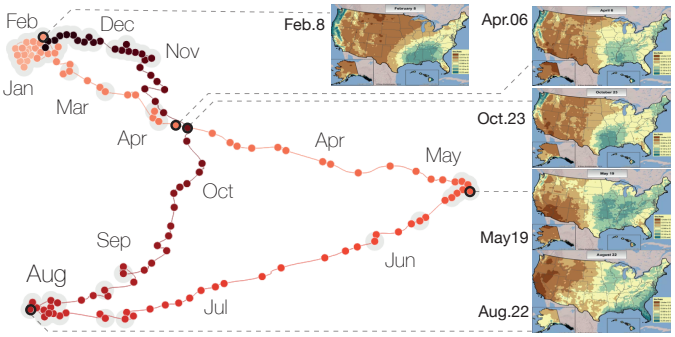


Fig. 9. Precipitation in the US across one year.

Temperature Patterns—Figure 10 shows data on a much larger scale, i.e., annual temperature world maps from 1884 to 2012.[6] The time curve shows three major stages with rapid transitions between them. The first two stages cover roughly 50 years each (1884-1935 and 1942-1991). The first stage is clearly less stable than the second, which suggests that world temperature underwent multiple alternations [18] before starting its current monotonic progression; the last stage (1991-2012) shows a rapid progression during a relatively short period of time (21 years).

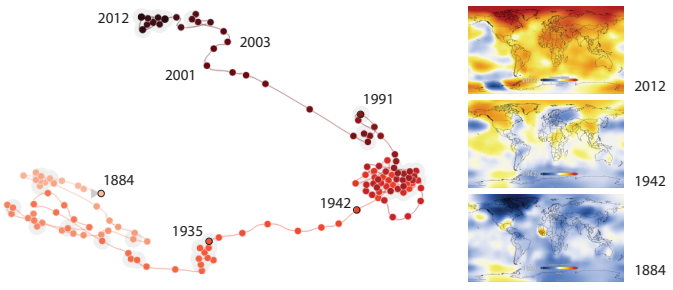


Fig. 10. Evolution of global temperature between 1884 and 2012. The backtracking is an artefact of video analysis (see stills on the right).

[5] https://www.youtube.com/watch?v=pz2DsQeF_UM
[6] http://tinyurl.com/nasa-temperatures

Surprisingly, the curve seems to suggest that trends are somehow reverting to the state of affairs in the 19th. This is however not true, as shown by the video stills in Figure 10: the map is dark blue in 1884 and dark red in 2012. Since our frame matching algorithm operates on grayscale images, the frames are wrongly interpreted as similar. This illustrates a general pitfall with time curves, i.e., the importance of the choice of similarity metrics. However, no similarity metric is perfect, thus time curves are best used in combination with other visualizations and details-on-demand interactions. Here, they could be used to facilitate navigation in the dynamic visualizations by revealing patterns that may not be obvious in the visualizations themselves.

4 Time Curves in Neuroscience Research

Besides exploring simple use case scenarios for time curves, we collaborated with a neuroscientist (T.M.) over the course of two months in order to investigate whether time curves can help domain experts to explore and understand functional brain connectivity data.

Functional brain connectivity refers to the network of correlation between activity in different regions of the brain (regions of interest, ROI). Activity is measured by the blood-oxygen-level dependent (BOLD) signal, obtained using functional magnetic resonance imaging (fMRI). Changes in these networks reflect cognitive task engagement, presence of psychoactive drugs, or neurological diseases [23] and analyses of connectivity may be performed on datasets that include anywhere from a few to hundreds of individual ROIs and several hundreds of time points, acquired 2-3 seconds apart.

The tasks involved in such an analysis range from identifying noise in data, which can dramatically change analysis results [38, 39], to identifying specific states of brain connectivity [40] (i.e., commonly reoccuring network configurations) to, most importantly, comparing patterns of connectivity across multiple individuals [3]. Currently, making sense of such datasets is essentially based on obtaining different statistical measures. No visualization exists yet that scales to that amount of data, while remaining simple to interpret. In this section we briefly report on the preliminary but promising results of our collaboration. We focus on two cases that illustrate how time curves helped our neuroscience collaborator formulate hypotheses to achieve a more targeted analysis in her research.

4.1 Time Curves for Individual fMRI Scans

Figure 11 shows examples of time curves obtained from elderly subjects while at rest and with eyes closed. These curves include 150 time points and for each time point the connectivity network can be represented as a correlation matrix (Figure 12, right). The distance metric we used to calculate the curve, is the Euclidean distance between correlation matrices [33].

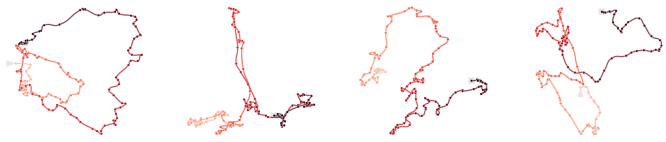


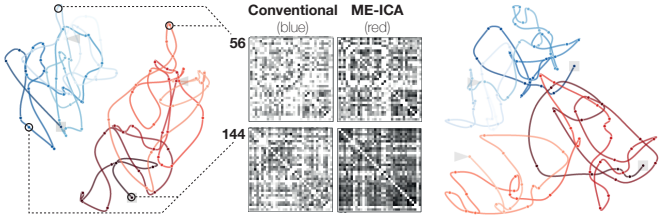
Fig. 11. Time curves obtained from fMRI scans of different subjects.

The curves in Figure 11 show that connectivity is constantly changing over time. We can see that the shapes for individual curves are different, which is expected because the subjects were not given a specific task. However, because each of those curves has been embedded in its own MDS space, curves can not accurately be compared.

4.2 Comparing Denoising Techniques

Comparing brain connectivity is important, for example, to examine the effect of data denoising techniques, since any processing can change the correlation structure of the data. Our collaborator T.M. was interested in comparing the Multi-echo Independent Components Analysis (ME-ICA) algorithm [28] to a conventional denoising method [33], applied to the same raw data from the same individual.

To be able to compare the resulting two data sets (ME-ICA, conventional), we calculated similarities between any pair of time points *across* the two data sets. Thus, we could project all time points (each time point appears twice) into the same 2D space and drew two time curves (Figure 12(a)): one connecting the timepoints from ME-ICA (red), and the other one connecting the time points from the data set denoised conventionally (blue).



(a) Subject A (b) Subject B

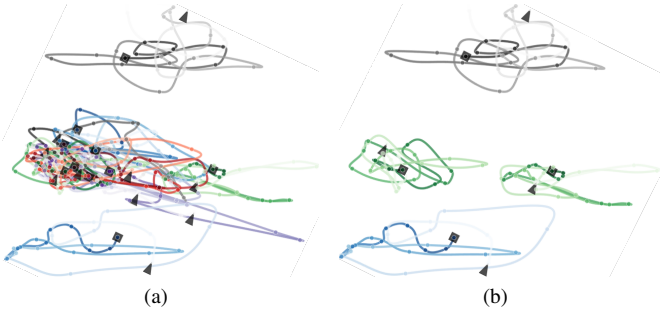
Fig. 12. Two curve pairs (subject A and subject B) showing offsets between conventionally denoised scans (blue) and scans denoised using ME-ICA (red). Matrix representations for timepoints 56 and 144 show an increase in signal strength through ME-ICA denoising.

Our main observation was that, while the red and blue curves in each example are offset, their shape remains similar across denoising techniques. A second observation was that the MC-ICA curves (red) where consistently larger. This was observed for all of the 21 subjects for whom denoised time curves were generated (cf. Figure 12(a)). T.M. hypothesized that the ME-ICA denoising method may strengthen correlations by removing more noise (larger curves) than traditional methods, without radically changing the temporal patterns (similar curve shapes). To the right of Figure 12(a) are snapshots of the correlation matrices corresponding to the same timepoints on both curves, showing that correlation indeed increased (darker cells and higher contrast). These observations, across all 21 individuals, where very informative, since obtaining the same information with regular statistics may have taken several days of analysis and a conventional MDS visualization would not have been able to show that the general temporal patterns, i.e. the curve shapes, were similar.

4.3 Comparing Connectivity Across Individuals

Since we were successfully able to compare time curves, T.M further wanted to determine whether there were differences in the individual brain connectivity across individuals: drawn into the same MDS space, overlapping time curves would indicate similar brain states, whereas larger or smaller curves would indicate more or less variance.

Figure 13 shows time curves from ten individuals, each displayed with a different color. In Figure (a) most of the curves are superimposed but we can immediately identify two outliers (gray and blue). We then turned the visibility for individual curves off in order to identify all non-overlapping curves. Shown in Figure (b), four curves are entirely disjoint. These differences may reflect meaningful differences in the individuals' physiological function, and thus data from these subjects deserve to be investigated by follow-up statistical analyses.



(a) (b)

Fig. 13. (a) Time curves from ten individuals embedded in the same MDS space; (b) a selection of four non-overlapping curves (pictures rotated to save space).

4.4 Informal User Feedback and Future Directions

Our investigations have shown that time curves seem to hold promise in neuroscience and may become a powerful addition to the neuroscientist's visualization toolbox. During our collaboration with T.M., we received encouraging feedback regarding the usability and usefulness of time curves. When we initially showed and explained time curves to her, she appeared to immediately understand the metaphor. Before even loading her datasets, she explained to us which patterns she was expecting to see, and what they would mean to her. While more extensive studies are needed before we can establish the utility of time curves in this particular domain, we consider our experience as anecdotal evidence that time curves can deliver quick insights on complex datasets to domain experts who are initially not used to the method.

In the future, we plan to use time curves to examine and compare different measures for brain connectivity and its states, and which are more likely to sensitively distinguish the physiological status of individuals. Time curves may also be particularly useful for visualizing data from various subjects in longitudinal studies. In situations where, e.g., subjects are imaged every month or once a year. Metrics could then be computed from their scans, yielding a single connectivity matrix for each time point (scan). One might thus imagine comparing a group of subjects having a neurodegenerative disease with a group of matched controls.

5 Time Curve Characteristics and Patterns

Across several examples, we showed how an analyst can read visual characteristics from a time curve and found that some visual patterns kept reoccurring. Here, we provide a more systematic typology of these characteristics and patterns and discuss their possible interpretations. Such a typology is necessarily incomplete and subjective, as time curves are continuous by nature and cannot be fully described in discrete terms. Our goal is *not* to provide a formal procedure to extract time curve patterns and characteristics, and map them to features in data, but rather to offer an informal descriptive terminology that can help novices learn how to read time curves, and analysts to capture and communicate insights. This also gives a better idea of the expressiveness and generality of time curves.

5.1 Time Point Distances

Any two time points geometrically relate in three different ways. For any pair of (not necessarily adjacent) time points *A* and *B*, we refer to **rank distance** as the number of time points between *A* and *B* along the curve, plus 1; **curvilinear distance** as the length of the curve segment between *A* and *B* (see Figure 14); and **spatial distance** as the 2D Euclidean distance between *A* and *B*.

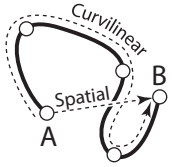


Fig. 14. Distances between time points.

All three distances are informative. Rank distance gives information on how far *A* and *B* are in the sequence of time points (e.g., how many revisions apart in the edit history), and is roughly linked to **temporal distance**. *Curvilinear distance* reflects the cumulated amount of changes, or activity, irrespective of how efficient the process is (e.g., all cumulated edits between two revisions). Since curvilinear distance is greater than cumulative pairwise distance and sensitive to the curve drawing algorithm (see Section 6.4), their relationship is necessarily approximative. *Spatial distance* reflects effective changes or activity (e.g., the difference between any two revisions), and is closely linked to **data distance**, i.e., the distance in the data's similarity matrix.

How the three distances relate can be very informative. A temporal process is perfectly linear (or maximally effective) when spatial distance equals curvilinear distance, i.e., when the curve is straight. The larger the curvilinear distance compared to the direct distance, the more non-linear and ineffective the process is. If the spatial distance between two points is null and the rank distance is 1, no change occurred (Figures 1(b), 6); If the rank distance is higher, there was a reversion to a previous state, such as in an edit war (Figure 2).

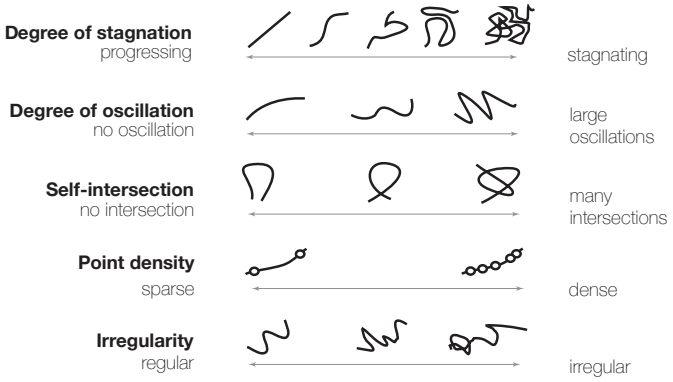


Fig. 15. Five geometric characteristics of time curves.

5.2 Geometric Characteristics of Curves

For any two (possibly remote) points connected with a curve whose length is greater than their spatial distance, geometric characteristics of the curve can convey information on the nature of the non-linear process. Such **characteristics** include (Figure 15):

- **Degree of stagnation**. A straight or smooth curve indicates progression (e.g., Figure 10), whereas a curve that does not exhibit any long-term change in location is indicative of a stagnating process.
- **Degree of oscillation**. A curve with no oscillation suggests a stable process, while a high degree of oscillation suggests a process that is unstable or alternates between states. Examples include edit wars in Wikipedia (Figure 1(b)) or weather patterns (Figure 8).
- **Self-intersection**. A curve with many self-intersections is indicative of an ineffective or highly non-linear process with many reversals (e.g., Figure 12). Although self-intersection is correlated with stagnation, it is possible for stagnating curves to have no self-intersection, and for progressing curves to have self-intersections. Since the exact number of self-intersections is sensitive to the curve drawing algorithm used (see Section 6.4), this characteristic is not meant to be interpreted literally but rather taken as an indication.
- **Point density**. Point density refers to the ratio between the number of time points along a curve and the curve's length: high density indicates series of small changes, while low density indicates few major changes. Variations in density are indicative of either a change in the process' speed or a change in its sampling rate.
- **Irregularity**. Regular curves have predictable characteristics, e.g., an oscillation of fixed periodicity and amplitude, or consistent changes in point density. They suggest lawful processes. Irregular curves are unpredictable and suggest chaotic processes.

These are just a few examples of time curve characteristics that cover common cases present in the examples discussed in Sections 3 and 4. Any such typology is necessarily incomplete, since possible variations in curve shape are numerous and their interpretation can differ depending on the choice of distance metric, dimensionality reduction method, and curve drawing algorithm.Still, the five characteristics we discussed so far allow us to offer a (rough) definition of a time curve's **complexity**: generally, the higher a time curve is on any of the five axes of Figure 8, the more complex it is. A time curve is all the more complex if it is high on several axes simultaneously.

5.3 Combinations of Characteristics

Figure 16 shows how two characteristics can be expressed simultaneously. The left panel shows combinations of oscillation and irregularity. The middle panel shows how stagnation can combine with point density, and the last panel shows how it can combine with oscillation. The curve on the top left oscillates but is still progressing, suggesting a trend of progression on a large scale, but minor fluctuations on a small scale. In contrast, the curve on the top right does not progress because it alternates between the same versions.

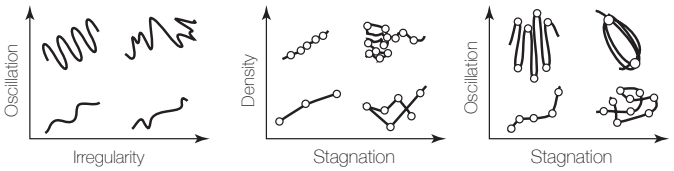


Fig. 16. Examples of combinations of curve characteristics.

A time curve can also combine several geometric characteristics in sequence, and such a curve can be thought of as segmented into separate **stages**. For example, in Figure 1(b) we can identify a stage of fast and high-amplitude oscillations, that visually differs from the previous and next curve segments by its degree of oscillation. Similarly, the curve in Figure 3 can be visually segmented according to point density, i.e., broken down into separate "clusters".

Although some of this segmentation process could be in principle delegated to computer algorithms, we see more benefits in showing the time curve as is, and letting users visually inspect and interpret the meaning of visual patterns using their domain knowledge and externally available contextual information.

5.4 Patterns

Extreme characteristics or specific combinations of characteristics can yield visually recognizable **patterns** (Figure 17). While characteristics are on a continuous scale, patterns are discrete and recognizable.

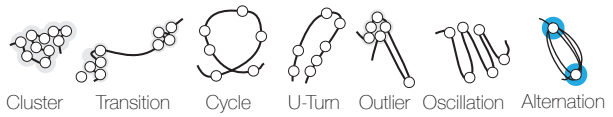


Fig. 17. Examples of visual patterns in time curves.

Clusters appear if a curve segment has a significantly denser set of points than its neighborhood, or when it has a significantly higher degree of stagnation. Examples can be found in Figure 6 (a video scene where nothing happens), and Figure 3 (a period of constant minor revisions). Related to clusters are **transitions**, i.e., curve segments between clusters with a high degree of progression. Transitions can be sparse in time points (Figure 1(b)) or dense (Figure 10). The presence of multiple clusters and transitions evoke a dynamic process with different states. A **cycle** refers to the situation where a time curves comes back to a previous point after a long progression. Figure 8 shows a single cycle, while Figure 2 shows multiple cycles. Complex patterns of cycles can be seen in Figures 7 and 6. Other examples of prominent patterns include **u-turns** indicating reversal in the process, **outliers** indicating anomalies such as acts of Wikipedia vandalism (Figure 4), periods of **high oscillation** that indicate informal edit wars (Figure 1(b)), or periods of **alternation** that can indicate literal edit wars (Figure 2).

5.5 Multiscale Characteristics

Time curves can exhibit different characteristics at different scales. For example, the curve in Figure 8 is cyclical on a large scale, is progressing on a medium scale, and is oscillating on a smaller scale. Similarly, in Figure 12 the entire curve is stagnating, but locally, it is progressing. Two different time curves can exhibit opposite multiscale characteristics. For example, a curve can be progressing globally but stagnating locally, in contrast with the previous example. If no restriction is imposed on the number of time points, a curve can in principle possess an arbitrary number of scales and even exhibit fractal characteristics.

6 Implementing Time Curves

We now discuss implementation considerations by drawing from our past experience in building prototypes over the course of four years. We implemented four time curve prototypes (see Figure 18):

- *Prototype #1* is an early Java version using force-directed MDS,
- *Prototype #2* is an interactive d3 implementation based on classical MDS and with support for alternative views (matrix and timeline),
- *Prototype #3* is an extension supporting thumbnails and animations,
- *Prototype #4* is the simplified Web version featured in Section 3.

We first define key terms, including the notion of *temporal similarity dataset*, that defines the class of datasets compatible with time curves. We then discuss how to construct distance matrices, position time points, draw the curve, and enrich or improve the legibility and the usability of time curves.

6.1 Definitions

Consistently with previous frameworks [34], we define a *temporal dataset* as a set of *time points* $P = p_0, p_1, ..., p_n$, where each time point consists in a timestamp $t_i \in \mathbb{R}$ and a data *snapshot* $s_i \in \mathbb{S}$ at time t_i: $p_i = (t_i, s_i), 0 \leq i \leq n$. The data snapshot domain $\mathbb{S}$ can be of any type, and thus s_i can be a single quantitative value, a high-dimensional vector, a bitmap, a network or adjacency matrix, a tree, a text, etc.

From a temporal dataset P one can define a *distance matrix* $D = [d_{ij}]$ that contains all pairwise distances between snapshots. Such a matrix can be constructed from a well-formed distance metric $d : \mathbb{S}^2 \rightarrow \mathbb{R}^+$ or derived from similarity measurements [43, 24]. A *temporal similarity dataset* P_D is a temporal dataset P with a distance matrix D.

Assuming D is ordered such that $\forall(i, j), i > j \Rightarrow t_i \geq t_j$, D is enough to construct a time curve. $t_i, \ldots t_n$ can be further used to convey quantitative time, e.g., through dot coloring.

6.2 Obtaining Distance Matrices

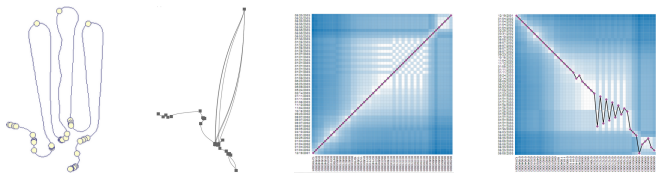
There are many ways the distance matrix D can be obtained, and the right method depends on the data type considered, its meaning, and the questions at hand. This issue is long known in the MDS community and has been extensively discussed [43, 24, 46]. In this section, we report on the methods we used for our scenarios in Section 3:

- **Wikipedia histories:** We retrieved document revisions in plain text format using the Wikipedia API. We then computed a diff between all possible pairs of revisions and measured the edit distance as the number of characters inserted or deleted [10]. More elaborate text distance metrics can be used [26].
- **Videos:** For videos, we first sampled video frames on regular intervals, e.g., one frame per second. We then estimated pairwise frame distance by computing normalized absolute pixel difference [1]. This naive approach does not account for large camera motions but already yields informative time curves, as could be seen in Section 3. A wide range of far more elaborate approaches exist, from color histogram comparison [30, 31] to local feature matching [32]. For his "video trajectories", Pless [37] experimented with a variety of metrics – including our naive pixel difference approach – and found that most of them generally produced satisfactory results.
- **Dynamic networks:** For the brain connectivity application, we calculated the Euclidean distance between adjacency matrices, i.e., the square root of the sum of squared cell differences. This method is effective when node labelling is stable across time. When this is not the case, algorithms involving inexact graph matching need to be employed [12, 41].

6.3 Positioning Time Points

There are many ways to position high-dimensional data points in a two-dimensional space while trying to preserve their metric structure [47]. MDS approaches are well-suited to time curves since they take distance matrices as input [46]. In our early prototype #1, we implemented a force-directed MDS by using the JBox2D physics engine and connecting all pairs of time points with springs whose ideal length was given by the matrix D. This method gives more control over the results (as we will later see) but is computationally ineffective and hard to tune. In our prototypes #2 to #4, we used the so-called "classical" MDS algorithm, which converges much faster [46].

More sophisticated multidimensional projection methods exist, but their downside is that they often make more assumptions about the data and have free parameters. For example, the ISOMAP method [25] used by Pless [37] assumes that topological neighborhood information is sufficient to describe the data and requires setting a neighborhood parameter. In addition, many of Pless' examples involve 3D curves, while we chose to stick to 2D projections for usability reasons [45, 42].



(a) Prototype #1 (b) Proto. #2 (c) Distance matrix in Proto. #3 (d) time ordering on x axis, Proto. #3

Fig. 18. Screenshots from earliers prototypes, all showing the Wikipedia article *Chocolate* from Figure 2.

6.4 Drawing Curves

As the curve indicates ordering, it needs to be easy to follow. We found that simply joining time points with segments is not sufficient. In our force-based prototype #1, we modelled the curve as a string (see Figure 18(a)). The string is initially straight, and collision detection prevents it from intersecting itself when folded. Thickness was added so that the curve repels itself, and rigidity was simulated with angular springs so that the curve looks smooth. Since the placement of time points was subject to these constraints, the MDS embedding was not accurate. In particular, identical data snapshots could not overlap.

Switching to classical MDS in prototype #2 made point embedding faster and more accurate, but drawing the curve became a challenge. Many of the interpolation methods we tried produced excessive swings and self-intersections, or made the curve look too angular (see Figure 18(b) for monotone cubic interpolation). In prototypes #3 and #4, we converged to a variant of Catmull-Rom interpolation [20]:

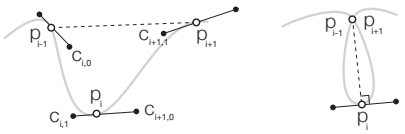


Fig. 19. Placement of control points for (a) normal cases, and (b) if the predecessor and successor of a time point are confounded.

We join all time points using Bézier curves with control points p_{i-1}, $c_{i,0}$, $c_{i,1}$ and p_i (see Figure 19). Thus a time point p_i is adjacent to two control points: $c_{i,1}$ and $c_{i+1,0}$. These control points are positioned so that the line joining them is parallel to (p_{i-1}, p_{i+1}). The distance of $c_{i,1}$ (*resp.* $c_{i+1,0}$) to p_i is set to the distance between p_i and p_{i-1} (*resp.* p_{i+1}) multiplied by a a smoothing parameter σ. We found $\sigma = 0.3$ to yield good results. In the special case where p_{i-1} and p_{i+1} are confounded, we place $(c_{i,1}, c_{i+1,0})$ orthogonally to (p_{i-1}, p_{i+1}). We finally add small random perturbations to the slope of Bézier tangents so that multiple alternations appear more clearly (see Figure 2).

Other interpolation approaches are possible. Overall the challenge is to find the right trade-off between producing legible curves and remaining faithful to the data.

6.5 Removing Time Point Overlap

Since MDS can place very similar points arbitrarily close to each other, another approach for improving legibility is by eliminating point overlap. Such techniques are commonly employed in infovis [16, 7]. Prototype #1 supported overlap removal through collision detection. In Prototypes #3–4, we used a simple iterative approach that moves pairs of overlapping nodes apart until there is no more overlap, or a limit has been reached. More elaborate techniques exist for arbitrary rectangular shapes [15]. Removing overlap emphasizes clusters (see Section 5), which can improve legibility but can also mislead about data. Thus in Prototype #4 we highlight displaced points with a halo, as seen in Section 3. Overlap removal can be activated on demand.

6.6 Rotating Curves

MDS embeddings do not have meaningful axes and are thus rotation invariant. Since the convention in Western cultures is that time goes from left to right, a proper curve orientation can facilitate reading. In our prototypes #3 and #4, we rotate time curves so as to align the initial and final time points horizontally and bring the initial time point to the left (see examples in Section 3). Thus, curves with a linear progression can be read from left to right, while in more complex curves, reversals in direction can indicate backtrackings and non-monotonic progress.

6.7 Additional Features

In our different prototypes we implemented and tested a variety of features. We discuss a few of them here.

Extra Visual Encodings—A time curve leaves many free visual variables that can be used to encode extra information. Since the only temporal information conveyed by default in time curves is the ordering of time points, it can be sometimes useful to convey $(t_i, \ldots, t_n)$ more explicitly. As seen in Section 3, Prototype #4 maps this information to the color of time points to give a rough indication. We also encode duration $t_{i+1} - t_i$ between two time points by varying the curve's thickness. Thicker curve segments indicate long time intervals, while thinner segments indicate shorter intervals. For example, the middle segment in Figure 1(b) is thick, indicating that the edit on the right end occurred after a relatively long period of inactivity. The segment is also long, indicating major changes. Alternatively, duration could be conveyed by curvilinear length using meandering curve segments [9], although the artificial geometrical patterns introduced may interfere with the interpretation of the time curve. Time points or curve segments can also be used to convey information on snapshots $(s_i, \ldots, s_n)$. In Section 3 we gave an example of using time point color to encode Wikipedia contributors.

Extra Views—Since a time curve does not show all the information available in a dataset P_D, complementary views can be useful.

Being a dimensionality reduction technique, MDS necessarily discards information from the distance matrix D. To help address this, Prototype #3 included a shaded matrix display [17]. In Figure 18(c), time points are on rows and columns, in chronological order, and cell darkness encodes distance. Series of minor edits appear as white squares on the diagonal, and edit wars appear as checkered patterns.

While information-rich, distance matrices can be hard to understand, especially by novices. To help address this, we overlaid a time curve whose points are positioned at the intersection of the corresponding row and column (purple dots in Figure 18(c)), and added support for animated transitions from and to the original MDS curve. We also added an option to reorder rows by similarity so that the curve reads like a line chart (with time on the x axis, see Figure 18(d)).

To convey time information $(t_i, \ldots, t_n)$ more accurately, we additionally implemented a time curve variant where x-coordinates are mapped to time, and y-coordinates are mapped to either a projection of the original 2D MDS on the y-axis, or to a one-dimensional MDS embedding. The resulting curve is, again, analogous to a line chart. As part of our efforts to unify matrix and curve representations and to support animated transitions, we implemented the same visualizations on the matrix view. The result is similar to Figure 18(c) with the difference that rows and columns have irregular width and height.

We tested layouts involving both multiple coordinated views and animated transitions within the same view. Views supported synchronized highlighting and details-on-demand (i.e., clicking a time point brings up the corresponding Wikipedia article). Despite our efforts to make a fully functional and "user-friendly" system, all views involving matrix representations turned out to be hard to read and less useful than expected, while 1D MDS turned out to be much less informative than 2D MDS. Thus we focused our recent efforts on improving and enriching the rendering of the 2D MDS time curve itself.

Finally, we experimented with animated transitions between regular timelines and time curves (see Figure 1(a)). We first implemented this by introducing time as a weighed parameter in the MDS and by animating the weight. However, this approach was computationally expensive (the MDS had to be computed for each animation step), and often resulted in jarring animations suffering from jitter and sometimes abrupt jumps. Since the intermediate stages between a regular timeline and time curve are not particularly meaningful, we finally decided to use a simple linear geometrical interpolation (see website).

7 Limitations and Challenges

Here we summarize the limitations of time curves and discuss the key issues that remain to be addressed. In particular, time curves inherit all benefits from MDS (i.e., its generality) but also all its weaknesses.

7.1 Information not Conveyed by Time Curves

As we already mentioned, time curves cannot — and are not meant to — convey *all* information in temporal datasets. Time curves convey *some* information on *i)* time and *ii)* self-similarity. More specifically, time curves convey the *ordinal* aspect of time, but not its *quantitative* aspects (durations, relative times). Although we discussed approaches for conveying quantitative temporal information through other means, time curves are best employed when quantitative time is not important. Furthermore, time curves do not convey all information on similarity either, and are only as powerful and informative as MDS or any other dimensionality reduction method. We simply see time curves as a useful addition to the repertoire of existing time visualization techniques [2, 5] that fills a gap by providing a different type of overview, and by being rather generic and rather straightforward to learn.

7.2 Choosing and Tuning Distances Metrics

As mentioned before, the quality of a time curve is highly dependent on the quality of the distance metric chosen. In addition, users may not necessarily know how distances were computed, and therefore how to interpret a time curve. This can be especially problematic when a time curve is used for communicating data to large audiences. While some metrics may be easy to grasp intuitively (e.g., edit distance), others may not be (e.g., brain connectivity distance). However, we expect that in scenarios such as in the brain connectivity example, analysts will take part in defining and tuning their own distance metrics.

Our collaboration with T.M. also suggests that there are cases where a single metric does not capture all characteristics of the data, and several curves with different distance metrics need to be compared. Such situations would benefit from tools to explore and tune distance metrics while curves are dynamically updated. This may pose challenges regarding the efficient calculation and rendering of time curves.

7.3 Scalability

From a computational perspective, a major bottleneck is the 2D embedding stage. In using classical MDS (whose complexity is $O(n^3)$) and on standard hardware, time point embedding took us 9 seconds for 50 time points, 20 seconds for 100 time points and 500 seconds for 500 time points, on average. This stage will most likely become faster as computer hardware and MDS algorithms improve.

From an infovis perspective, perceptual scalability is perhaps a more important concern. The legibility of a time curve does not only depend on the mere number of time points, but also on its *complexity* (see Section 5). A few points are enough to make a curve look messy. Complexity depends on the data, and on the structure of the matrix D. Some datasets produce trivial curves (i.e, a straight line) while others produce curves that are too complex to be useful. Time curves are most useful for datasets that are in the middle of this continuum.

Assuming a dataset of reasonable complexity, several thousands of time points would likely be too small or too dense to be legible, but the curve itself would remain visible. Nevertheless, such a curve may exhibit multiscale features, requiring zooming tools. Classical zooming or space distortion tools may be useful, but the one-dimensional nature of time curves could also be exploited. For example, one could imagine a one-dimensional window that can slide along the time curve, and that would magnify the curve segment on a separate view.

Perceptual scalability issues can be partly alleviated through downsampling, i.e., aggregating or decimating time points in the dataset. We used downsampling in our video, weather and brain connectivity examples. However, there are many ways downsampling can be done (e.g., uniform vs. adaptive, smoothed vs. raw), and the choices made can greatly affect the results. Thus in cases where downsampling is desirable, sampling tools are best coupled with a dynamic time curve representation. This is, again, a feature that appeared to be potentially useful in the context of neuroscience data analysis.

7.4 Reproducibility, Stability and Robustness

It is desirable that the same dataset produces the same curve across different runs of the layout algorithm (*reproducibility*). Also, similar datasets should ideally produce similar time curves [27], for example when a dataset is perturbed by noise (*robustness*) or updated with additional time points such as in Wikipedia articles (*stability*). Since none of these is guaranteed by MDS, we conducted preliminary tests using our classical MDS implementation. More results can be found online.

We tested reproducibility on several datasets by creating multiple time curves for each dataset. Although MDS did produce a slightly different curve each time, the curve's visual characteristics and patterns remained mostly unchanged. Some curves were mirrored, but in Section 6.6 we explained how we lock curve orientation, and a similar approach may be devised for eliminating mirroring.

We then assessed robustness by adding Gaussian noise to existing datasets (5%, 10%, and 15%). As expected, noise affected the curves, sometimes occulting the original visual patterns. However, curves remained mostly unaffected below 10% noise. Finally, to assess stability we appended new data points to several datasets. Figure 20 shows two typical outcomes of our tests: although the shape of the curve gets deformed over time, the main geometrical patterns are generally preserved and there is no evidence of sudden jumps.

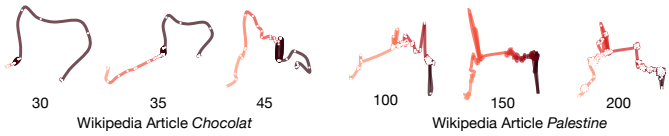


Fig. 20. Evolution of time curves (Wikipedia articles) as more ancient revisions are included. Numbers indicate the number of revisions.

Our initial tests suggest that our particular implementation of time curves is reasonably reproducible and stable. However, more extensive tests are needed to understand how changes in data sets are reflected by changes on the time curve. Such tests will need to involve other types of dynamically changing datasets, and will need to assess the influence of the dimensionality reduction algorithm employed.

8 Conclusion

We presented *time curves*, a simple visual overview technique that can reveal patterns in temporal datasets of many different types. We illustrated time curves with a range of examples, and explored its potential in neuroscience research. We also introduced a framework to characterize visual patterns that are frequently exhibited by time curves and how they may be interpreted, discussed the technique's implementation, as well as its limitations and pending challenges.

Future work includes exploring a wider range of similarity metrics (e.g., to support the extraction of semantic color scales in videos) and a wider range of dimensionality reduction algorithms. Of particular interest are real-time MDS algorithms that would allow analysts to interactively explore the effect of different similarity metrics, and incremental MDS algorithms that are able to guarantee visual stability as datasets grow over time. More work is also needed to understand to what extent time curves are faithful to the data, i.e., what crucial information is thrown away and in how far the visual patterns we identified convey reliable information about the data [27].

Other datasets and applications of time curves also remain to be explored. Time curves could be used as "curved sliders" to navigate in audiovisual media [14, 13], or to visualize user interaction histories and provenance in infovis applications [35]. Time curves may also be used as visual signatures or "dataset thumbnails" to label and identify a multitude of datasets in personal media libraries, collaborative platforms, or in clinical settings for labelling and retrieving patients. Similar to Sparklines or Sportslines [36], time curves could be embedded in text to convey information on, e.g., oscillating, stagnating or diverging processes, or to report on sets of similar data cases. Like Sparklines, miniature time curves curves would not support the precise reading of values, but could come useful to visually convey simple or complex temporal patterns to a wide audience.

REFERENCES

[1] Imabsdiff. http://www.mathworks.fr/fr/help/images/ref/imabsdiff.html. online, accessed Jan. 21, 2015.

[2] W. Aigner, S. Miksch, H. Schumann, and C. Tominski. *Visualization of Time-Oriented Data*. Springer, 2011.

[3] B. Alper, B. Bach, N. Henry Riche, T. Isenberg, and J.-D. Fekete. Weighted graph comparison techniques for brain connectivity analysis. In *Proc. of ACM CHI*, pages 483–492, 2013.

[4] B. Alper, N. Riche, G. Ramos, and M. Czerwinski. Design study of linesets, a novel set visualization technique. *IEEE TVCG*, 17(12):2259–2267, Dec 2011.

[5] B. Bach, P. Dragicevic, D. Archambault, C. Hurter, and S. Carpendale. A Review of Temporal Data Visualizations Based on Space-Time Cube Operations. In *Proc. of EuroVis*, 2014.

[6] B. Bach, N. Henry-Riche, T. Dwyer, T. Madhyastha, J.-D. Fekete, and T. Grabowski. Small MultiPiles: Piling Time to Explore Temporal Patterns in Dynamic Networks. *Computer Graphics Forum*, 2015.

[7] A. Bezerianos, F. Chevalier, P. Dragicevic, N. Elmqvist, and J.-D. Fekete. Graphdice: A system for exploring multivariate social networks. In *Computer Graphics Forum*, volume 29, pages 863–872, 2010.

[8] R. Borgo, M. Chen, B. Daubney, E. Grundy, G. Heidemann, B. Höferlin, M. Höferlin, H. Leitte, D. Weiskopf, and X. Xie. State of the art report on video-based graphics and video visualization. *Computer Graphics Forum*, 31(8):2450–2477, Dec. 2012.

[9] K. Buchin, A. van Goethem, M. Hoffmann, M. van Kreveld, and B. Speckmann. Travel-time maps: Linear cartograms with fixed vertex locations. In M. Duckham, E. Pebesma, K. Stewart, and A. Frank, editors, *Geographic Information Science*, volume 8728 of *LNCS*, pages 18–33. Springer, 2014.

[10] F. Chevalier, P. Dragicevic, A. Bezerianos, and J.-D. Fekete. Using text animated transitions to support navigation in document histories. In *Proc. of ACM CHI*, pages 683–692, 2010.

[11] F. Chevalier, S. Huot, and J.-D. Fekete. Wikipediaviz: Conveying article quality for casual wikipedia readers. In *Proc. of PacificVis*, pages 49–56, 2010.

[12] D. Conte, P. Foggia, C. Sansone, and M. Vento. Thirty years of graph matching in pattern recognition. *International journal of pattern recognition and artificial intelligence*, 18(03):265–298, 2004.

[13] P. Dragicevic, G. Ramos, J. Bibliowitcz, D. Nowrouzezahrai, R. Balakrishnan, and K. Singh. Video browsing by direct manipulation. In *Proc. of ACM CHI*, pages 237–246, 2008.

[14] P. Dragicevic and K. Singh. Audio curves. [Online; accessed Mar 2015] http://www.lri.fr/~dragice/audiocurve/, 2006.

[15] T. Dwyer, K. Marriott, and P. J. Stuckey. Fast node overlap removal: Correction. In *Proc. of Graph Drawing*, pages 446–447, Berlin, Heidelberg, 2007. Springer-Verlag.

[16] G. Ellis and A. Dix. A taxonomy of clutter reduction for information visualisation. *IEEE TVCG*, 13(6):1216–1223, 2007.

[17] J. Foote. Visualizing music and audio using self-similarity. In *Proc. of Multimedia*, pages 77–80, New York, NY, USA, 1999. ACM.

[18] U. C. for Atmospheric Research. How much has the global temperature risen in the last 100 years? https://www2.ucar.edu/news/how-much-has-global-temperature-risen-last-100-years, 2015.

[19] M. Ghoniem, J.-D. Fekete, and P. Castagliola. A comparison of the readability of graphs using node-link and matrix-based representations. In *Proc. of InfoVis*, pages 17–24, 2004.

[20] A. Hardy and W.-H. Steeb. *Mathematical tools in computer graphics with C# implementations*. World Scientific, 2008.

[21] M. Hascoët and P. Dragicevic. Interactive graph matching and visual comparison of graphs and clustered graphs. In *Proc. of AVI*, pages 522–529, 2012.

[22] K. Hinum, S. Miksch, W. Aigner, S. Ohmann, C. Popow, M. Pohl, and M. Rester. Gravi++: Interactive information visualization to explore highly structured temporal. *j-jucs*, 11(11):1792–1805, nov 2005.

[23] R. M. Hutchison, T. Womelsdorf, E. A. Allen, P. A. Bandettini, V. D. Calhoun, M. Corbetta, S. Della Penna, J. H. Duyn, G. H. Glover, J. Gonzalez-Castillo, D. A. Handwerker, S. Keilholz, V. Kiviniemi, D. A. Leopold, F. de Pasquale, O. Sporns, M. Walter, and C. Chang. Dynamic functional connectivity: Promise, issues, and interpretations. *NeuroImage*, 80:360–378, Oct. 2013.

[24] N. Jaworska and A. Chupetlovska-Anastasova. A review of multidimensional scaling (MDS) and its utility in various psychological domains. *Tutorials in Quantitative Methods for Psychology*, 5(1):1–10, 2009.

[25] J. C. L. Joshua B. Tenenbaum, Vin de Silva. A global geometric framework for nonlinear dimensionality reduction. *Science*, pages 2319–2323, 2000.

[26] D. Jurafsky and J. H. Martin. *Speech and language processing*. Pearson, 2014.

[27] G. Kindlmann and C. Scheidegger. An algebraic process for visualization design. *IEEE TVCG*, 20(12):2181–2190, Dec 2014.

[28] P. Kundu, S. J. Inati, J. W. Evans, W.-M. Luh, and P. A. Bandettini. Differentiating BOLD and non-BOLD signals in fMRI time series using multi-echo EPI. *NeuroImage*, 60(3):1759–1770, Apr. 2012.

[29] M. Li, X. Chen, X. Li, B. Ma, and P. M. Vitányi. The similarity metric. *IEEE Transactions on Information Theory*, 50(12):3250–3264, 2004.

[30] R. W. Lienhart. Comparison of automatic shot boundary detection algorithms. In *Proceedings of Electronic Imaging*, pages 290–301, 1998.

[31] Y. Liu, D. Zhang, G. Lu, and W.-Y. Ma. A survey of content-based image retrieval with high-level semantics. *Pattern Recognition*, 40(1):262–282, 2007.

[32] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[33] T. M. Madhyastha, M. K. Askren, P. Boord, and T. J. Grabowski. Dynamic Connectivity at Rest Predicts Attention Task Performance. *Brain Connectivity*, July 2014.

[34] W. Muller and H. Schumann. Visualization methods for time-dependent data - an overview. In *Proceedings of the International Conference on Machine Learning and Cybernetics*, pages 737–745. IEEE, 2003.

[35] C. North, R. Chang, A. Endert, W. Dou, R. May, B. Pike, and G. Fink. Analytic provenance: process+ interaction+ insight. In *CHI'11 Extended Abstracts on Human Factors in Computing Systems*, pages 33–36. ACM, 2011.

[36] C. Perin, R. Vuillemot, and J.-D. Fekete. Soccerstories: A kick-off for visual soccer analysis. *IEEE TVCG*, 19(12):2506–2515, 2013.

[37] R. Pless. Image spaces and video trajectories: Using isomap to explore video sequences. In *Proc. of IEEE ICCV*, pages 1433 – 1440, 2003.

[38] J. D. Power, K. A. Barnes, A. Z. Snyder, B. L. Schlaggar, and S. E. Petersen. Spurious but systematic correlations in functional connectivity MRI networks arise from subject motion. *Neuroimage*, 59(3):2142–2154, Feb. 2012.

[39] J. D. Power, A. Mitra, T. O. Laumann, A. Z. Snyder, B. L. Schlaggar, and S. E. Petersen. Methods to detect, characterize, and remove motion artifact in resting state fMRI. *NeuroImage*, 84:320–341, Jan. 2014.

[40] M. Rabinovich, R. Huerta, P. Varona, and V. Afraimovich. Transient cognitive dynamics, metastability, and decision making. *PLoS Computational Biology*, (4), 2008.

[41] M. Roy, S. Schmid, and G. Tredan. Modeling and measuring graph similarity: The case for centrality distance. In *Proceedings of ACM International Workshop on Foundations of Mobile Computing*, FOMC '14, pages 47–52, New York, NY, USA, 2014. ACM.

[42] M. Sedlmair, M. Brehmer, S. Ingram, and T. Munzner. Dimensionality reduction in the wild: Gaps and guidance. Technical Report TR-2012-03, UBC Computer Science Technical Report, 2012.

[43] R. N. Shepard. The analysis of proximities: Multidimensional scaling with an unknown distance function. *Psychometrika*, 27(2):125–140, 1962.

[44] B. Shneiderman. The eyes have it: a task by data type taxonomy for information visualizations. In *IEEE Symposium on Visual Languages*, pages 336–343, Sep 1996.

[45] B. Shneiderman. Why not make interfaces better than 3d reality? *Computer Graphics and Applications, IEEE*, 23(6):12–15, 2003.

[46] W. Torgerson. Multidimensional scaling: I. Theory and method. *Psychometrika*, 17(4):401–419, 1952.

[47] L. van der Maaten, E. O. Postma, and H. J. van den Herik. Dimensionality reduction: A comparative review. http://www.iai.uni-bonn.de/~jz/dimensionality_reduction_a_comparative_review.pdf, 2008.

[48] F. B. Viégas, M. Wattenberg, and K. Dave. Studying cooperation and conflict between authors with history flow visualizations. In *Proc. of ACM CHI*, pages 575–582, New York, NY, USA, 2004. ACM.

[49] M. Wattenberg. Arc diagrams: Visualizing structure in strings. In *Proc of IEEE InfoVis*, pages 110–116, Washington, DC, USA, 2002.