

Appendix A

Survey of Visualization Techniques

This appendix is dedicated to surveying existing visualization techniques for time and time-oriented data. As there are different types of time and time-oriented data as well as many different questions one may ask about the data, there are numerous visualization techniques taking the peculiarities of data and tasks into account. Some of them are specific to a particular application, others are more general with potential applicability in various domains.

The following pages list the techniques on a per-page basis. Each page briefly describes a particular technique's background, explains its main idea and concepts, and indicates its application. Each page also includes relevant references to publications that originally proposed or substantially extended a technique. Illustrating figures demonstrate the techniques in use or their conceptual construction. Additionally, each page has a side-bar for categorizing the described visualization technique. Here we use the simplified categorization schema introduced in Chapter 7. Accordingly, the side-bar information follows this pattern:

- **Time**
 - *Primitives* – points vs. intervals
 - *Arrangement* – linear vs. cyclic
- **Data**
 - *Number of variables* – single vs. multiple
 - *Frame of reference* – abstract vs. spatial
- **Visual representation**
 - *Mapping of time* – static vs. dynamic
 - *Dimensionality* – 2D vs. 3D

Where possible, a distinct classification will be given. However, this is not always achievable, particularly for more general and flexible visualization approaches. In such cases, we will indicate that a technique exhibits multiple characteristics per category.

While the list of techniques included in our survey is extensive, it is clear that we cannot claim completeness. This is due to the fact that visualization of time-oriented data is a hot research area constantly yielding new techniques. Moreover, visualization solutions can be highly application-dependent, and it is virtually impossible to dig out every tiny variation of existing visualization approaches that might be hidden in the vast body of scientific literature across application domains. Therefore, we took care to include a wide spectrum of key techniques, both classic ones with proven usefulness and contemporary ones with potential impact.

In what follows, you will find the one-page descriptions for all the techniques listed in Table 7.1 from page 201 of Chapter 7. Below all techniques are listed alphabetically with their corresponding page numbers for easy reference. Visit <https://browser.timeviz.net> for an online version of the survey.

A.1 List of Techniques

3D ThemeRiver	p. 317	Flocking Boids	p. 333
3D TimeWheel	p. 324	Flow Map	p. 371
Anemone	p. 251	Flowstrates	p. 366
Animated Scatter Plot	p. 330	FluxFlow	p. 306
Arc Diagrams	p. 239	Gantt Chart	p. 253
Bar Graph, Spike Graph	p. 234	GeoTime	p. 383
BinX	p. 291	Gravi++	p. 327
Braided Graph	p. 301	Great Wall of Space-Time	p. 369
CareCruiser	p. 300	GROOVE	p. 271
Chro-Ring	p. 388	Growth Matrix	p. 241
ChronoLenses	p. 303	Growth Ring Maps	p. 374
CircleView	p. 328	Helix Icons	p. 389
Circos	p. 352	history flow	p. 294
Circular	p. 363	Horizon Graph	p. 277
CiteSpace II	p. 302	Icons on Maps	p. 379
ClockMap	p. 275	InfoBUG	p. 326
CloudLines	p. 329	Interactive Parallel Bar Charts	p. 248
Cluster and Calendar-Based Visualization	p. 273	Intrusion Detection	p. 357
Co-Bridges	p. 346	Intrusion Monitoring	p. 250
Connected Scatterplot	p. 304	Kaleidomaps	p. 353
Continuum	p. 338	KAVAGait	p. 354
Cycle Plot	p. 268	Kiviat Tube	p. 319
Data Tube Technique	p. 318	KNAVE II	p. 337
Data Vases	p. 385	KronoMiner	p. 358
DateLens	p. 257	Layer Area Graph	p. 289
Decision Chart	p. 237	LifeLines	p. 341
DimpVis	p. 305	LifeLines2	p. 295
Dynamic Word Clouds	p. 252	Lin-spiration	p. 283
Enhanced Interactive Spiral	p. 274	Line Density Plot	p. 307
Event-Flow Visualization	p. 345	Line Plot	p. 233
EventRiver	p. 342	LiveGantt	p. 347
EventViewer	p. 360	LiveRAC	p. 299
FacetZoom	p. 336	Matrix-Based Comparison	p. 308

Midgaard	p. 340	TACO	p. 314
MOSAN	p. 316	Temporal Focus+Context	p. 387
MoSculp	p. 370	Temporal Mosaic	p. 265
Multi Scale Temporal Behavior	p. 270	Temporal Star	p. 320
Multi-Resolution Vis. of Time Series	p. 242	TextFlow	p. 344
MultiComb	p. 292	ThemeRiver	p. 293
Multiple Temporal Axes Model	p. 384	ThermalPlot	p. 362
MultiStream	p. 309	Tile Maps	p. 269
netflower	p. 310	Time Annotation Glyph	p. 261
Optimized Stream Graphs	p. 311	Time Curves	p. 279
Paint Strips	p. 259	Time Line Browser	p. 334
Parallel Glyphs	p. 351	Time Maps	p. 246
PatternFinder	p. 335	Time-Oriented Polygons on Maps	p. 373
Pencil Icons	p. 386	Time-Ray Maps	p. 381
PeopleGarden	p. 348	Time-tunnel	p. 321
Perspective Wall	p. 256	Time-Varying Hierarchies on Maps	p. 368
Pinus View	p. 243	TimeDensityPlots	p. 247
Pixel-Oriented Network Visualization	p. 350	TimeHistogram 3D	p. 249
PlanningLines	p. 260	Timeline	p. 258
Point Plot	p. 232	Timeline Trees	p. 288
PostHistory	p. 349	TimeNets	p. 263
Process Visualization	p. 331	TimeRider	p. 332
RankExplorer	p. 312	TimeSearcher	p. 290
Recursive Pattern	p. 282	TimeSearcher 3, River Plot	p. 287
Ring Maps	p. 361	TimeSets	p. 255
Ripple Graph	p. 244	TimeSlice	p. 280
Sankey Diagram, Alluvial Diagram	p. 313	TimeTree	p. 238
SentiCompass	p. 355	TimeWheel	p. 298
Set Streams	p. 254	Traffigram	p. 367
Silhouette Graph	p. 281	Train Delay Uncertainty	p. 266
Similan	p. 296	Trajectory Wall	p. 375
Small MultiPiles	p. 245	TreeRose	p. 356
Small Multiples	p. 359	TrendDisplay	p. 236
Software Evolution Analysis	p. 323	Triangular Model	p. 267
SolarPlot	p. 272	Value Flow Map	p. 365
SOPO Diagram	p. 262	Vanishing-Point Plot	p. 325
Space-Time Cube	p. 377	ViDX	p. 364
Space-Time Path	p. 378	VIE-VISU	p. 297
SparkClouds	p. 240	VIS-STAMP	p. 380
Sparklines	p. 235	Visits	p. 372
Spatio-Temporal Event Visualization	p. 376	VisuExplore	p. 339
SpiraClock	p. 276	VizTree	p. 278
Spiral Display	p. 285	Wakame	p. 382
Spiral Graph	p. 284	WireVis	p. 315
Stacked Graphs	p. 286	Worm Plots	p. 322
Story Curves	p. 343		
Storyline Visualization	p. 264		

time

primitives: points
arrangement: linear

data

number of variables: single
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

A.2 Techniques for Abstract Time-Oriented Data

Point Plot

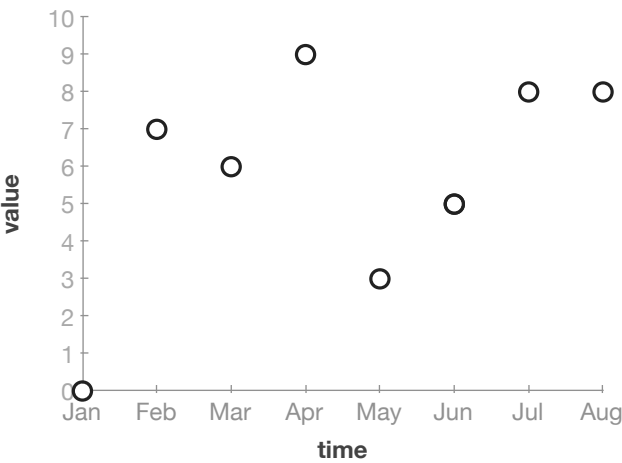


Fig. A.1: Data are displayed as points in a Cartesian coordinate system where time and data are mapped to the horizontal axis and the vertical axis, respectively. © The authors.

One of the most straightforward ways of depicting time-series data is using a Cartesian coordinate system with time on the horizontal axis and the corresponding value on the vertical axis. A point is plotted for every measured time-value pair. This kind of representation is called point plot, point graph, or scatter plot, respectively. Harris (1999) describes it as a 2-dimensional representation where quantitative data aspects are visualized by distance from the main axis. Many extensions of this basic form such as 3D techniques (layer graph) or techniques that use different symbols instead of points are known. This technique is particularly suited for emphasizing individual values. Moreover, depicting data using position along a common scale can be perceived most precisely by the human perceptual system.

Relevant references: Harris (1999)

Line Plot

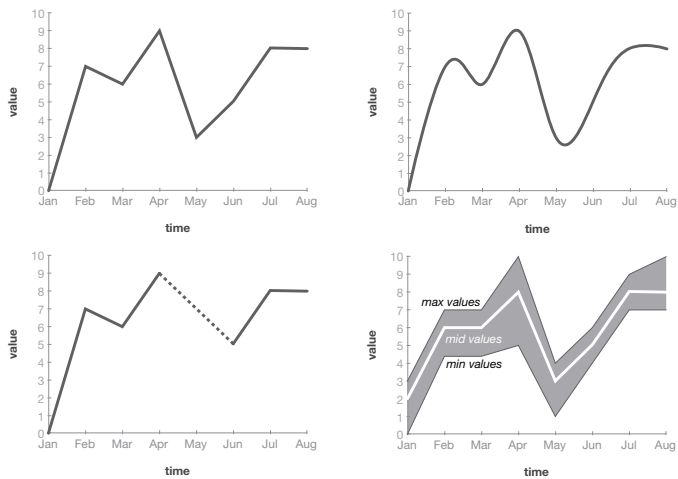


Fig. A.2: Successive data points are connected with lines to visualize the overall change over time. Top-left: straight lines; top-right: Bézier curves; bottom-left: missing data; bottom-right: band graph. © The authors.

The most common form of representing time series are line plots. They extend point plots (\leftrightarrow p. 232) by linking the data points with lines, which emphasizes their temporal relation. Consequently, line plots focus on the overall shape of data over time. This is in contrast to point plots where individual data points are emphasized. As illustrated in the figure, different styles of connections between the data points such as straight lines, step lines (instant value changes), or Bézier curves can be used depending on the phenomenon under consideration. However, what has to be kept in mind is that one can not be sure in all cases about the data values in the time interval between two data points and that any kind of connection between data points reflects only an approximation. A further point of caution is missing data. Simply connecting subsequent data points might lead to false conclusions regarding the data. Therefore, this should be made visible to the viewer, for instance by using dotted lines (bottom left). Harris (1999) lists many extensions and subtypes of line plots, including fever graphs, band graphs (bottom right), layer line graphs, surface graphs, index graphs, and control graphs.

Relevant references: Harris (1999)

time

primitives: points
arrangement: linear

data

number of variables: single
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

time

primitives: points
arrangement: linear

Bar Graph, Spike Graph

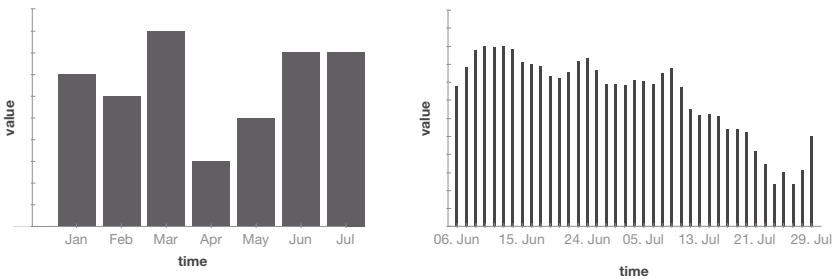


Fig. A.3: Time-dependent data values are represented as bars of different lengths. Left: regular bars; right: spike graph where bars are narrowed to spikes. © The authors.

data

number of variables: single
frame of reference: abstract

Bar graphs are a well-known and widely used type of representation where bars are used to depict data values. This makes comparisons easier than with point plots. As bar length is used to depict data values, only variables with a ratio scale (having a natural zero) can be represented. Consequently, the value scale also has to start with zero to allow for a fair visual comparison. In contrast to line plots, bar graphs emphasize individual values as do point plots. A variant of bar graphs often used for graphing larger time series (e.g., stock market data such as price or volume) are spike graphs. As illustrated in right-hand figure, the vertical bars are reduced so as to appear as spikes, where spike height is again used to encode data values. This way, a good visual balance is achieved between focusing on individual values and showing the overall development of a larger number of data values.

Relevant references: Harris (1999)

vis

mapping of time: static
dimensionality: 2D

Sparklines

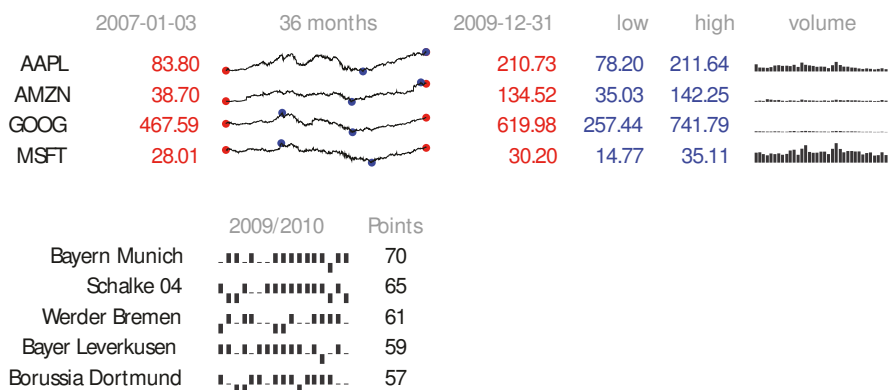






Fig. A.4: Sparklines are miniature charts to be integrated into text. Top: Sparklines visualizing stock market data; bottom: Soccer results visualized using ticks (up = win, down = loss, base = draw). © The authors. Generated with the sparklines package for L^AT_EX.

Tufte (2006) describes sparklines as simple, word-like graphics intended to be integrated into text. This adds richer information about the development of a variable over time that words themselves could hardly convey. The visualization method focuses mainly on giving an overview of the development of values for time-oriented data rather than on specific values or dates due to their small size and the omission of axes and labels. Sparklines can be integrated seamlessly into paragraphs of text, can be laid out as tables, or can be used for dashboards. They are increasingly adopted to present information on web pages (such as usage statistics) in newspapers (e.g., for sports statistics), or in finance (e.g., for stock market data). Usually, miniaturized versions of line plots (\hookrightarrow p. 233)  and bar graphs (\hookrightarrow p. 234)  are employed to represent data. For line plots, the first and last values can be emphasized by showing red dots and printing the associated values to the left and right of the sparkline (top figure). Additionally, the minimum and maximum values can be marked by blue dots. Besides this, colored bands in the background of the plot can be used to show normal value ranges 4.8  8.3. For the special case of binary or three-valued data, special bar graphs can be applied that use ticks extending up and down a horizontal baseline . One use for this kind of representation is the wins and losses of sports teams where the history of a whole season can be presented using very little space (bottom figure).

Relevant references: Tufte (2006)

time

primitives: **points**
arrangement: **linear**

data

number of variables: **single**
frame of reference: **abstract**

*vis*mapping of time: **static**
dimensionality: **2D**

time

primitives: points
arrangement: linear

TrendDisplay

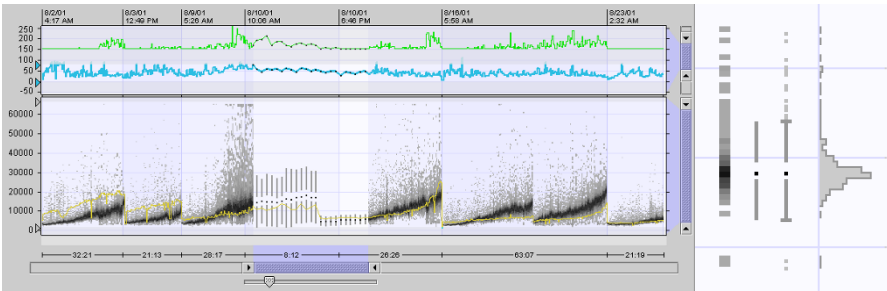


Fig. A.5: TrendDisplay shows data (center) and derived statistics (top) at four different levels of visual abstraction. Depending on the available screen space, the visual representation is dynamically chosen to be either color-coded density distributions, thin box plots, box plots plus outliers, or bar histograms (right). © 2003 IEEE. Reprinted, with permission, from Brodbeck and Girardin (2003).

data

number of variables: single
frame of reference: abstract

The TrendDisplay technique by Brodbeck and Girardin (2003) allows the analysis of trends in larger time series. The technique is used for the drug discovery process and in quality control. Basically, the TrendDisplay window is composed of two panels. The main panel in the center shows the measured (raw) data and the top panel depicts derived statistical values. Four different levels of detail (right) are used in order to cope with large numbers of time points: color-coded density distributions, thin box plots, box plots plus outliers, and bar histograms (from low to high level of detail). In the temporal dimension, bifocal focus+context functionality is used for enlarging areas of interest without losing context information about neighboring data. The different levels of detail are chosen automatically depending on the available screen space. Moreover, brushing & linking as well as smooth transitions complete the highly interactive interface.

vis

mapping of time: static
dimensionality: 2D

Relevant references: Brodbeck and Girardin (2003)

Decision Chart

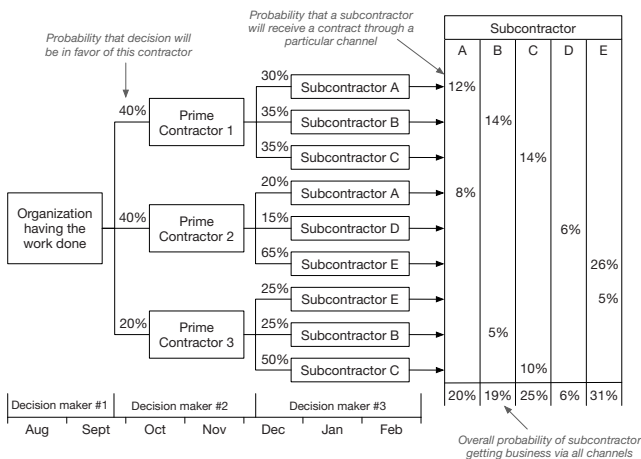


Fig. A.6: Future decisions and corresponding alternative outcomes along with their probabilities are depicted over time in a decision chart. © The authors. Adapted from Harris (1999).

Harris (1999) describes decision charts as a graphical representation for depicting future decisions and potential alternative outcomes along with their probabilities over time. Decision charts are one of the very few techniques for time-oriented data that use the *branching time* model. Decision charts use a horizontal time axis along which information elements (decisions and probabilities) are aligned. Multiple decisions for a particular time interval are stacked on top of each other, indicating that they are possible alternatives for that interval. However, the temporal context itself is not of prime interest and is just indicated by a simple time scale at the bottom of the chart. The main advantage of the decision chart is that it allows planners to investigate possible outcomes and implications before decisions are made.

Relevant references: Harris (1999)

time

primitives: points
arrangement: linear

data

number of variables: single
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

TimeTree

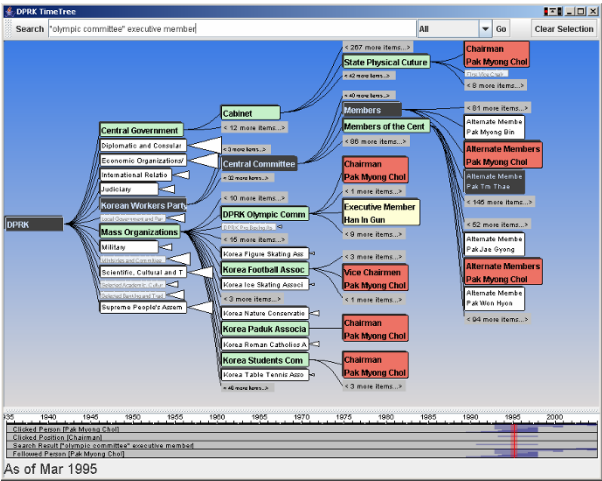


Fig. A.7: A TimeTree showing changes in the organizational structure of officials in the DPR Korea with focused and important nodes highlighted. The time slider (currently set to March 1995) shows selected personal events and serves for interactive navigation. © 2006 IEEE. Reprinted, with permission, from Card et al. (2006).

TimeTree by Card et al. (2006) is a visualization technique to enable the exploration of changing hierarchical organizational structures and of individuals within such structures. The visualization consists of three parts: a time slider (bottom), a tree view(center), and a search interface (top). The time slider's main purpose is to allow users to navigate to any point in time. Additionally, it shows information strips with events for a selected set of individuals, and thus, provides insight into where in time interesting things have happened. The tree view shows the snapshot of the organizational structure corresponding to the selected time point. The tree visualization uses a degree-of-interest (DoI) approach to highlight important information. To this end, a specific color scheme is used to indicate certain data characteristics, as for instance, important nodes, nodes matching with search queries, or recently clicked nodes. Low-interest nodes, with regard to the user's current focus and search query, are ghosted, and entire sub-trees may be represented as triangular abstractions in order to de-clutter the display and maintain the readability of important nodes. The search interface supports a textual search for individuals in the represented organization.

Relevant references: Card et al. (2006)

Arc Diagrams

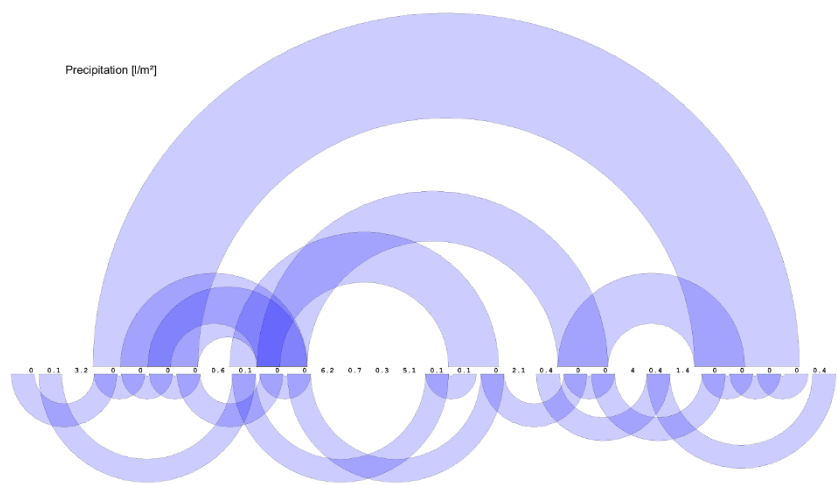


Fig. A.8: A sequence of data values is shown along the horizontal axis. Matching subsequences are connected by arcs, where arc thickness and height encode subsequence size and occurrence distance, respectively. © Courtesy of Michael Zornow.

Patterns in sequences of data values can be visualized using arc diagrams. They were introduced as an interactive visualization technique by Wattenberg (2002). Given a sequence of values, the goal is to extract significant subsequences that occur multiple times in the original sequence. The visualization displays the sequence of data values in textual form along the horizontal (time) axis. Occurrences of significant subsequences are visually connected by spanning arcs. The arcs’ thickness represents the size of the subsequence, that is, the number of data values in the subsequence. The height of an arc indicates the distance between two successive occurrences of the subsequence. To express data-specific aspects, one can separately use the space above or below the data sequence. This also helps to reduce the overlap of arcs. Additionally, transparency is used to allow users to see through overlapping arcs. The visualization can be controlled interactively via several parameters (e.g., minimum size of subsequences or tolerance threshold for fuzzy pattern extraction) to keep the number of arcs at an interpretable level.

Relevant references: Wattenberg (2002)

time

primitives: points
arrangement: linear

data

number of variables: single
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

time

SparkClouds

primitives: points
arrangement: linear

data

number of variables: single
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D



Fig. A.9: Display of important keywords in time-varying data. Font size encodes the keyword importance. Keywords being less relevant for the studied point in time are attenuated by using dimmed color and smaller font size. © 2010 IEEE. Reprinted, with permission, from Lee et al. (2010).

Tag clouds visualize a set of keywords weighted by their importance. To this end, a layout of the keywords is computed. By varying font size, color, or other visual variables important keywords are emphasized over less-important keywords. Classic tag clouds, however, are incapable of representing the evolution of keywords. Lee et al. (2010) integrate sparklines (↔ p. 235) into tag clouds in order to visualize temporal trends in the development of keywords. The idea is to visually combine a keyword (or tag) and its temporal evolution. The keyword’s importance is encoded with the font size used to render the text, where the size can correspond either to the overall importance of the keyword for the entire time series or to the importance at a particular point in time. Attached to the keyword is a sparkline that represents the keyword’s trend. A color gradient is shown in the background of each keyword-sparkline pair to make this design perceivable as a visual unit. Lee et al. (2010) conducted user studies with sparkclouds and could confirm that sparkclouds are useful and have advantages over alternative standard methods for visualizing text and temporal information.

Relevant references: Lee et al. (2010)

Growth Matrix

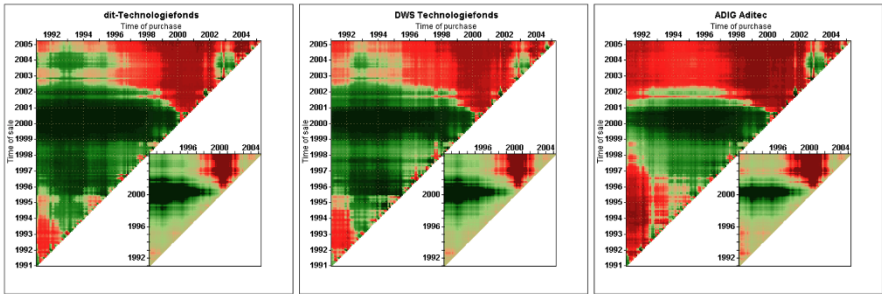


Fig. A.10: Growth Matrices are used for the analysis of financial data. For each triangle, all possible subintervals between time of purchase (x-axis) and time of sale (y-axis) are shown by a single pixel. Each pixel’s color indicates financial gain (green) or loss (red) for each subinterval. The example shows the performance of three different funds (large triangles) compared to the market performance of similar assets (small triangles). © ⓘ ⓘ ⓘ ⓘ *Growth Matrix* by Keim et al., also see Keim et al. (2006b).

Analysis of the performance of assets is a core task in financial analytics. To support this, Keim et al. (2006b) developed Growth Matrix, a heatmap-based visualization technique for analyzing asset return rates over all possible subintervals in a given time frame. It is based on the Return Triangle visualization technique in financial analytics and extends it to a dense, pixel-based visualization approach. In the triangle, the horizontal and vertical dimensions represent time from left to right and bottom to top, respectively. Color-coding is applied to show growth rate quantities. Growth rates are normalized to achieve a normalized color-mapping scheme using red hues for losses and green hues for gains. Smaller Growth matrices might be added to allow for comparisons, e.g., against the overall market performance of similar assets (see figure for an example). In subsequent work, Ziegler et al. (2007) extended the approach by adding Performance Matrices that directly depict performance values for different holding periods in a rectangular matrix by mapping time of sale on the x-axis and holding periods on the y-axis. Moreover, improvements of color mappings are introduced by extending the original approach to a 2-dimensional color map that is able to depict both, inter-asset and intra-asset performance.

Relevant references: Keim et al. (2006b) • Ziegler et al. (2007)

time

primitives: points
arrangement: linear

data

number of variables: single
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

time

primitives: points
arrangement: linear

Multi-Resolution Visualization of Time Series

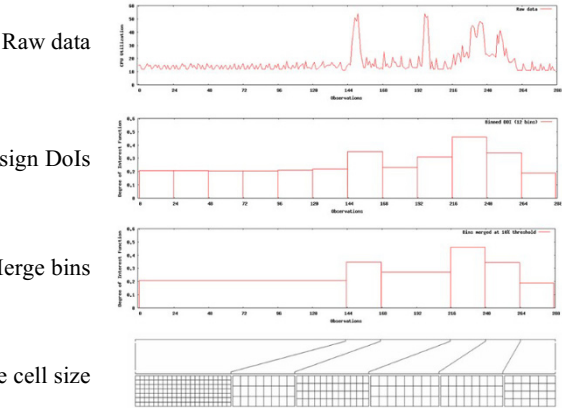


Fig. A.11: Multi-resolution visualization of time-series data is based on the idea of assigning more display space to relevant portions of the data and less space to less relevant parts. © Courtesy of Tobias Schreck.

data

number of variables: single
frame of reference: abstract

Hao et al. (2007) address the problem of visualizing large time series with many time points. Extending earlier work on importance-driven layouts of time series (see Hao et al. (2005)), the authors propose a degree-of-interest (DoI) approach to generate a multi-resolution visualization. First, the raw data are binned and a DoI score is assigned to each bin. Optionally, neighboring bins with similar scores can be merged. The DoI scores are then used to determine the cell size for the different parts of the visualization. Data with low DoI will be represented at lower resolution (smaller cells), whereas interesting data with high DoI will be shown at higher resolution (larger cells). In this way, less relevant data take up less display space and relevant data will be assigned more display space. This makes it easier for the user to analyze the relevant parts in more detail. While the authors propose a matrix-like color-based visualization, the multi-resolution approach is generally applicable to other kinds of visual representations of time-oriented data as well.

vis

mapping of time: static
dimensionality: 2D

Relevant references: Hao et al. (2007) • Hao et al. (2005)

Pinus View

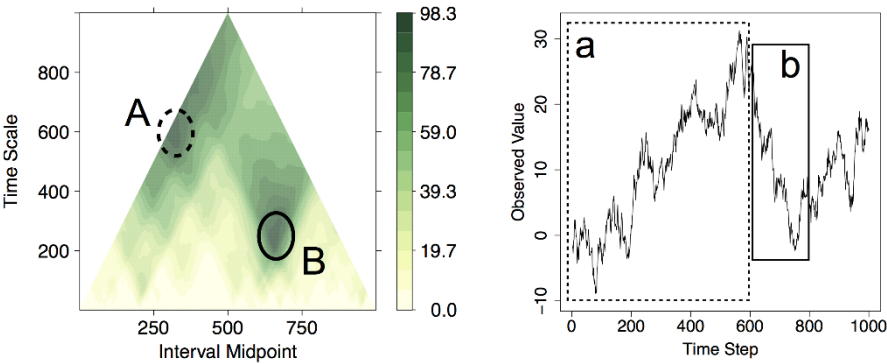


Fig. A.12: The matrix-based Pinus view. Users can identify regions A and B with variance values in the Pinus view and their corresponding temporal patterns a and b in the time series. Color indicates variance values of subsequences. © 2012 IEEE. Reprinted, with permission, from Sips et al. (2012).

Detecting interesting patterns in numerical time series according to various time scales and starting points an analyst is looking at is an important tasks, in particular when exploring environmental time series. Sips et al. (2012) propose a visual analytics approach to tackle this research problem, by providing (1) an algorithm to compute statistical values for all possible time scales and starting positions of intervals, (2) visual identification of potentially interesting patterns in a matrix visualization, and (3) interactive exploration of detected patterns. The core matrix-based visualization is *Pinus* view, which presents the variation of a user-chosen statistical measure across all possible time scales and starting positions of intervals. As such, the pinus view is similar to the triangular model (↔ p. 267). The utility of the pinus view was demonstrated with two use cases (regime changes of the earth’s climate system and ocean modeling).

Relevant references: Sips et al. (2012)

time

primitives: points, intervals
arrangement: linear

data

number of variables: single
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

time

Ripple Graph

primitives: points
arrangement: linear

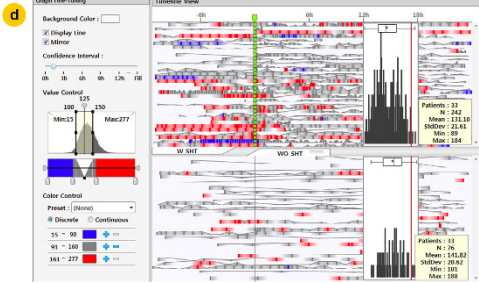
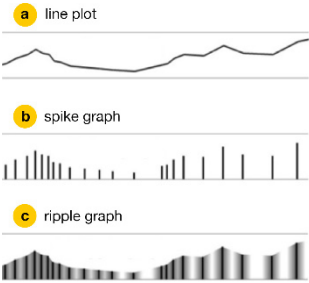


Fig. A.13: Ripple graphs visualize measurement frequency as well as uncertainties of values between measurements. The left part of the figure shows how ripple graphs are constructed: starting from a line plot (a), vertical lines are drawn at the time of measurements resulting in a spike graph (b), finally, color gradients are added to represent uncertainties between measurements (c). On the right, ripple graphs are shown in the context of the Stroscope visualization system. © 2014 IEEE. Reprinted, with permission, from Cho et al. (2014).

data

number of variables: single
frame of reference: abstract

Although most modeling, visualization, and automated methods expect time-series data at regular time intervals, measurements at irregular time intervals are frequently present in many domains. Irregular measurements are particularly challenging due to varying (un)certainty of what can be said about the values between measurements. With Ripple graphs, Cho et al. (2014) developed a visualization technique that makes the aspects of measurement frequency as well as uncertainty between points of measurement explicit. Ripple graphs are based on spike graphs (\leftrightarrow p. 234) which are spaced according to the time of measurements. To represent the level of uncertainty, color gradients are added with decreasing opacity for increasing distances from the points of measurement. Ripple graphs are applied within the Stroscope system, a multi-scale visualization approach for irregularly measured time-series data. In addition to the ripple graphs, color-coding is applied to highlight values above or below user-defined regions of interest (d). On the left side of the Stroscope interface (d), a control panel allows for the configuration of the ripple graphs. Here, the user can adjust the temporal confidence intervals, i.e., the extent of the color gradients displayed around the spikes, and define regions of interest for the data values along with a user-defined color-coding.

vis

mapping of time: static
dimensionality: 2D

Relevant references: Cho et al. (2014)

Small MultiPiles

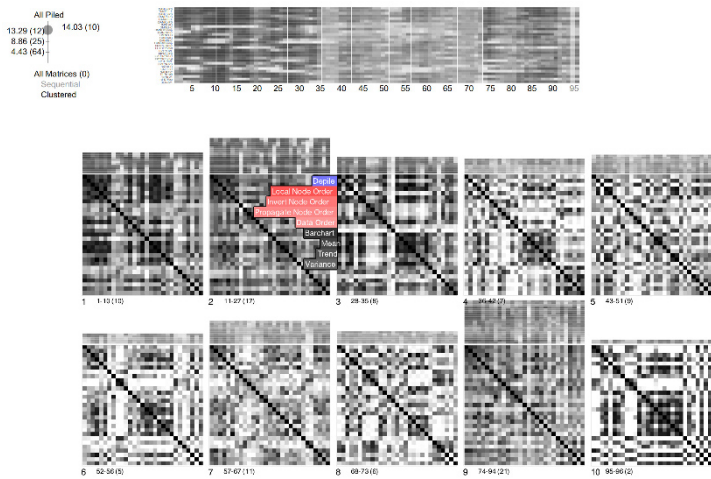


Fig. A.14: The Small MultiPiles approach shows dense networks as piles of matrices. A slider can be used to adjust the number of piles. An overview at the top represents node connectivity over time. © The authors. Generated with the *Small MultiPiles* software by Benjamin Bach.

A common approach to visualize dense networks that vary over time is to show them as matrices in a small multiples arrangement (↔ p. 359). Each matrix shows the network for a particular time point and each matrix cell color-codes the weight of an edge between two network nodes. However, when the number of time points is large, the matrices consume much display space and scrolling is required to get an overview of the data. The Small MultiPiles (note the exact spelling) by Bach et al. (2015) are a technique for generating compact overviews by creating piles of matrices. The piles are formed by means of hierarchical clustering, where the number of visible clusters (i.e., piles) can be adjusted via a slider in the interface (top left in the figure). If necessary, the piles can be edited manually using simple drag and drop gestures. With the Small MultiPiles approach, each pile groups similar matrices, and because there are fewer piles than original time steps, the arrangement of piles is compact and grants an overview of the data. Hovering the matrix preview that is attached to the top of each pile grants access to the details of each individual matrix. Moreover, for each pile, the cover matrix (i.e., the topmost matrix of a pile) can show different aggregated information about the matrices in a pile, including summary, trend, difference, and variability. An overview at the top represents node connectivity over time.

Relevant references: Bach et al. (2015)

time

primitives: points
arrangement: linear

data

number of variables: single
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

time

Time Maps

primitives: points
arrangement: linear

data

number of variables: single
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

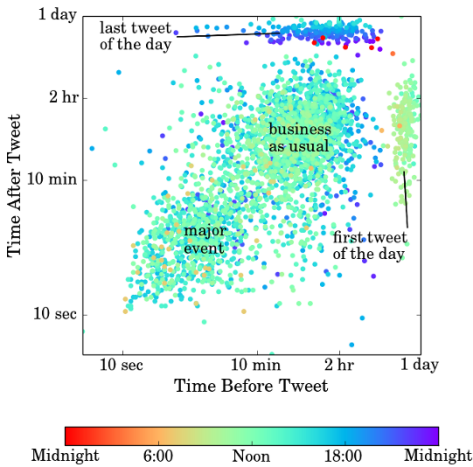


Fig. A.15: Time map visualization of tweets. The x-position of a point corresponds to a tweet’s temporal distance to its preceding tweet. The y-position shows the distance to a tweet’s subsequent tweet. Color visualizes the time of the day a tween was posted. © *Courtesy of Max C. Watson.*

Watson (2015) proposes *Time Maps* to visualize discrete event data. A time map is basically a point plot (↪ p. 232), where each point corresponds to an event in time. Time maps are special in that an event’s x-coordinate is the time between the event itself and the preceding event. The y-coordinate is the time between the event itself and the subsequent event. Plotted this way, time maps allow the viewer to identify critical features, no matter if they occur on a timescale of milliseconds or months. Each point in a time map can also be colored to encode additional information, such as the time of day as in the figure.

Relevant references: Watson (2015)

TimeDensityPlots

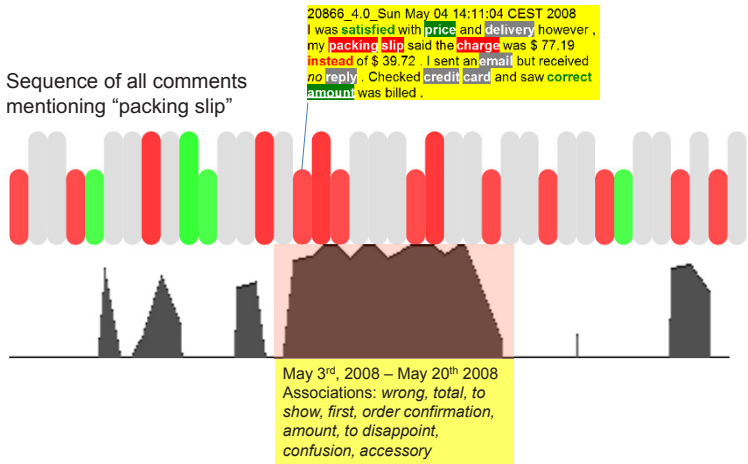


Fig. A.16: TimeDensityPlots combine the visualization of a qualitative sequence of events (colored bars) with a density plot (dark silhouette curve) providing some quantitative information of the temporal distribution of data items. © Courtesy of Christian Rohrdantz.

In the context of text stream visualization, Rohrdantz et al. (2012) developed a technique called TimeDensityPlots. It is an point-based 2D visualization. Data items (here documents) are shown as a sequence of colored bars (red, gray, green in the figure). The order of the bars corresponds to their qualitative position in time, meaning we know that a data item is before another one, but we do not know the distance between the two. In order to maintain some quantitative information about the temporal distribution of data items, a density plot is aligned with the bar panel. The density plot visualizes the density of data items with respect to the time scale in the bars panel as a silhouette graph (↔ p. 281). The higher the curve below the bars of two items, the closer they are in time. In the figure, the highlighted interval represents a relatively dense part, which means that the associated bars represent data items being close together in time.

Relevant references: Rohrdantz et al. (2012)

time

primitives: points
arrangement: linear

data

number of variables: single
frame of reference: abstract

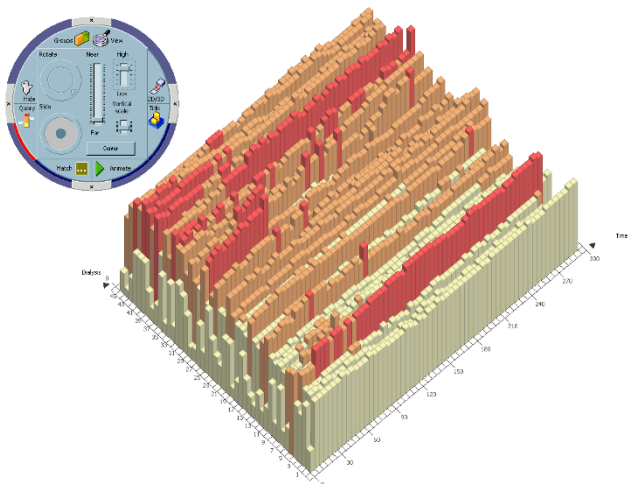
vis

mapping of time: static
dimensionality: 2D

time

primitives: points
arrangement: linear

Interactive Parallel Bar Charts



data

number of variables: single
frame of reference: abstract

Fig. A.17: Visualization of clinical time-dependent data where one axis represents different hemodialysis sessions and the other axis represents the series of time steps. One time-dependent variable (e.g., blood pressure) is encoded to the height of the bars. © 2003 Elsevier. Reprinted, with permission, from Chittaro et al. (2003).

vis

mapping of time: static
dimensionality: 3D

Chittaro et al. (2003) present a technique for visualizing time-dependent hemodialysis data. To keep the visualization of multiple hemodialysis sessions simple and easy to use for physicians, the design is based on common 3D bar charts, where the height of bars encodes individual data values as in regular 2D bar graphs (↔ p. 234). Multiple bar charts (one per hemodialysis session) are arranged on a regular grid in a parallel fashion. This visual display is easy to interpret despite the 3D projection. Visual exploration and analysis are facilitated through various interaction tools. Dynamic filtering combined with a color-coding mechanism supports visual classification. To manage occlusions, interactive features are provided, such as flattening individual bars or groups of bars, or leaving only colored squares in the grid. The water level interaction is particularly helpful for comparison tasks: virtual water engulfs all bars with a height below a user-defined threshold. This eases the assessment of similarities and differences with respect to dialysis sessions and time. Additional visual cues support physicians in detecting anomalies or special events in the data and enable them to take necessary actions quickly. For multivariate analysis, an integration with parallel coordinate plots is supported.

Relevant references: Chittaro et al. (2003)

TimeHistogram 3D

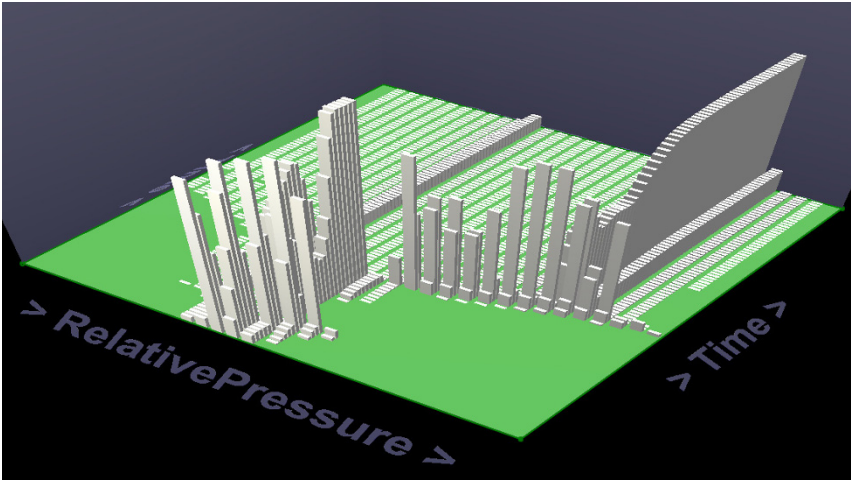


Fig. A.18: Time is encoded along the x-axis, while a time-dependent variable (RelativePressure) is represented along the y-axis. For each time-value pair in the resulting grid in the x-y plane, the height of a cuboid represents the frequency of data items per grid cell. © 2004 IEEE. Reprinted, with permission, from Kosara et al. (2004).

TimeHistogram 3D is an interactive extension of well-known histograms. The Time-Histogram 3D is specially designed for time-oriented data. It has been developed to give an overview of complex data in the application context of computational fluid dynamics (CFD). A design goal of this technique was to show temporal information in static images while maintaining the easy readability of standard histograms. In the figure, the x-axis encodes time and the y-axis encodes a time-dependent variable (RelativePressure), effectively creating a grid in the x-y plane, where each grid cell corresponds to a unique time-value pair. In order to visualize the number of data items (i.e., their frequency) per time-value pair, cuboids are shown for each cell, where the height of a cuboid encodes the frequency. This way, the user can see where in time and in which value range data items accumulate. Several interactive features such as brushing, scaling, and a 2D context display, which is shown in the background of the histogram, are part of this technique.

Relevant references: Kosara et al. (2004)

time

primitives: points
arrangement: linear

data

number of variables: single
frame of reference: abstract

vis

mapping of time: static
dimensionality: 3D

time

primitives: points
arrangement: linear

Intrusion Monitoring

data

number of variables: single
frame of reference: abstract

vis

mapping of time: dynamic
dimensionality: 2D

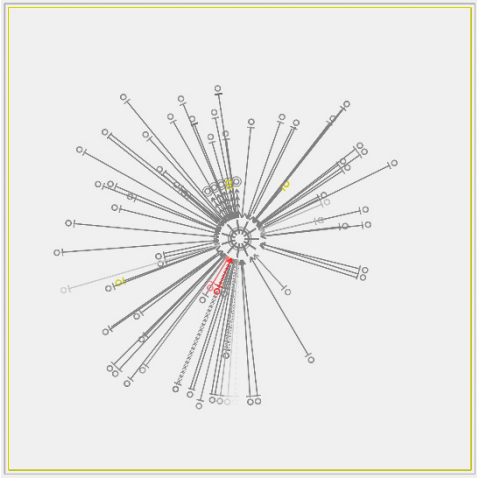


Fig. A.19: The glyph in the center represents system being monitored. Connections to remote hosts are depicted as radially arranged lines. Critical and suspicious connections are visualized by red and yellow color, respectively. © Courtesy of Robert F. Erbacher.

Erbacher et al. (2002) describe a system that visualizes time-stamped network-related log messages that are dynamically generated by a monitored system. These messages correspond to events in a linear continuous time domain. The visualization shows the monitored server as a central glyph encoding the number of users and the server’s load. Events are shown as radially arranged lines at whose end the remote host is shown as a smaller glyph. Regular network activities are drawn with a shade of gray. Unexpected or suspicious activities result in a change of color: Hosts that try to open privileged connections are colored in red, hosts that fail to respond turn yellow, lines representing timed-out connections or connections that failed the authentication procedure are shown in red, and connections that have been identified as intrusions are represented with even brighter red. To preserve a history of connections that have been terminated, the corresponding lines are faded out gradually. This kind of visual representation helps administrators in observing network communication. Presenting colored (red or yellow) lines among gray lines attracts the attention of administrators to suspicious activity and actions can be taken quickly to counter network attacks from remote hosts.

Relevant references: Erbacher et al. (2002)

Anemone

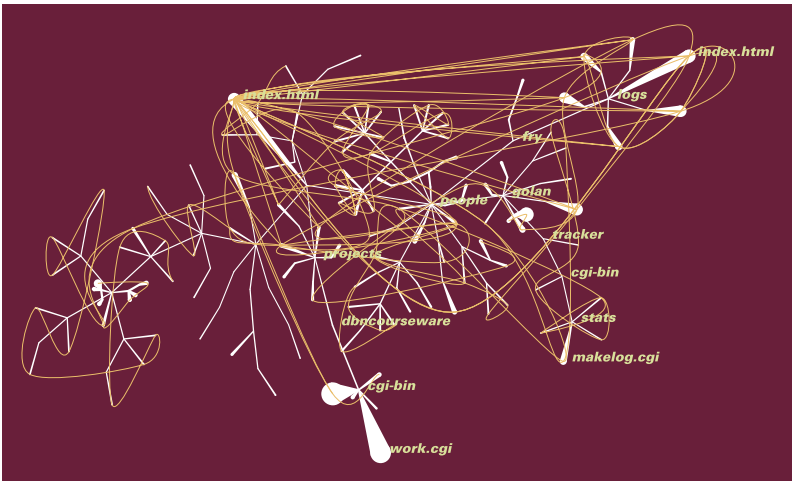


Fig. A.20: Snapshot of Anemone showing traffic patterns of people visiting the web site of the Aesthetics & Computation Group at the M.I.T. Media Lab. The structure of the web site is shown as a node-link representation. Nodes vary in size, depending on how frequently a page is visited by users. Rarely-visited parts fade out slowly. © Courtesy of Ben Fry, MIT Media Laboratory, Aesthetics + Computation Group.

Anemone by Fry (2000) is a technique related to the visualization of structured information. It is a dynamic, organic representation designed to reveal not only the static structure of a website, which is based on its organization into folders and files, but also to reveal dynamic usage patterns. To this end, a classic node-link representation is visually enriched with dynamically updated usage statistics to form a living representation that truly reflects the restless nature of a website. The static structure is shown as nodes that are connected via straight branches. At the tip of a branch resides the actual web page. Additional labels can be used to identify nodes by their corresponding page’s name. Nodes dynamically change size depending on how often they are visited by users. When a user follows a link from one page to another, a thin curved line is drawn connecting both pages. If parts of a site have not been visited for a long time, they shrink in size and slowly fade out. To allow users to concentrate on particular items of interest, it is possible to select nodes and lock them to a dedicated position. This is quite useful because the dynamic character of the technique implies that visual representation constantly changes its appearance.

Relevant references: Fry (2000)

time	primitives: points
	arrangement: linear
data	number of variables: single
	frame of reference: abstract
vis	mapping of time: dynamic
	dimensionality: 2D

time

Dynamic Word Clouds

primitives: points
arrangement: linear

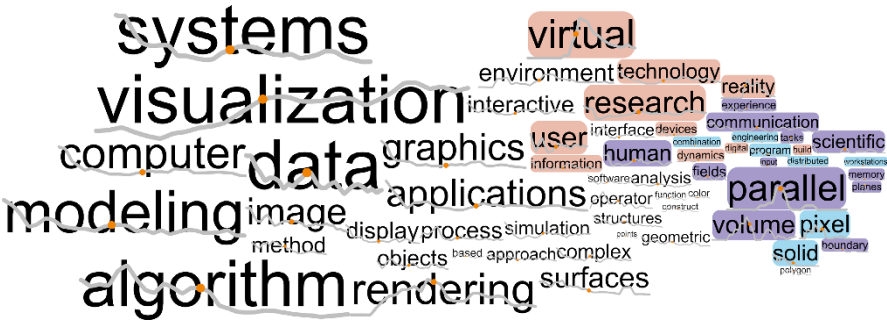


Fig. A.21: Dynamic word clouds are primarily concerned with keeping the layout of words coherent between subsequent time steps of a dynamic visualization. Addition visual cues (e.g., sparklines and color) can be used to show additional temporal information. © Courtesy of Weiwei Cui.

data

number of variables: single
frame of reference: abstract

Word clouds are visual summaries of text where important words are shown in larger font sizes, while context words are smaller. Word clouds are popular for static texts. However, standard algorithms are not suitable for generating word clouds for texts that change over time. The reason is that the word clouds are generated independently for each time step, which may cause word positions to change drastically from one time point to the next. This makes it difficult to follow words when studying a sequence of word clouds. Dynamic word clouds tackle this problem in different ways. A key concern is to lay out the words coherently between subsequent time steps. To this end, one can use the adapted force-directed algorithm by Cui et al. (2010) or the spiral-based layout strategy in combination with collision detection by Seyfert and Viola (2017). In addition to coherent word positions, further pieces of information can be encoded visually. The figure shows sparklines (↔ p. 235) attached to the words, which is similar to SparkClouds (↔ p. 240). Moreover, colors indicate if words are appearing, disappearing or unique. It is also possible to tilt words to indicate words with increasing or decreasing importance with respect to previous time steps.

vis

mapping of time: dynamic
dimensionality: 2D

Relevant references: Cui et al. (2010) • Seyfert and Viola (2017)

Gantt Chart

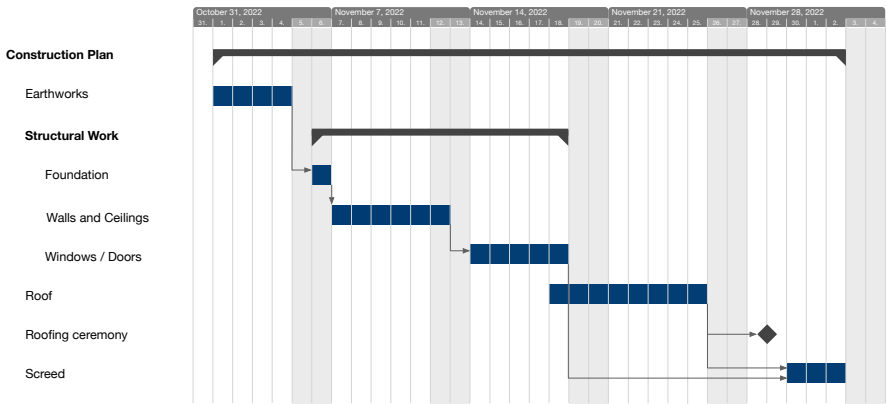


Fig. A.22: The Gantt chart shows a project plan for construction works. To the left, the chart provides an indented list of tasks. In the main panel, timelines show position and duration of tasks in time, where black and blue bars stand for groups of tasks and individual tasks, respectively. Additionally, diamonds indicate milestones. © The authors.

Planning activities, people, and resources is a task that is particularly important in the field of project management. One of the common visualization techniques used for such tasks is Gantt charts. This kind of representation was originally invented by Gantt (1910) who studied the order of steps in work processes. Mainly work tasks with their temporal location and duration as well as milestones are depicted. The tasks are displayed as a textual list in the left part of the diagram and might be augmented by additional textual information such as resources, for example. Related tasks can be grouped to form a hierarchy, which is reflected by indentation in the task list. For displaying the position and duration of tasks in time, timelines (\longleftrightarrow p. 258) are drawn at the corresponding vertical position of the task list. This leads to an easily comprehensible representation of information from the past, present, and future. Hierarchically grouped tasks can be expanded and collapsed interactively. Summary lines are used to maintain an overview of larger plans. Sequence relationships are represented by arrows that connect tasks (e.g., an arrow from the end of task A to the beginning of task B shows that task B may start only after task A is finished). Milestones indicating important time points for synchronization within a project plan are visually represented by diamonds. The fact that tasks are mostly ordered chronologically, typically leads to a diagonal layout from the upper left to the lower-right corner of the display.

Relevant references: Gantt (1910)

time

primitives: points, intervals
arrangement: linear

data

number of variables: single
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

Set Streams

time

primitives: points, intervals
arrangement: linear

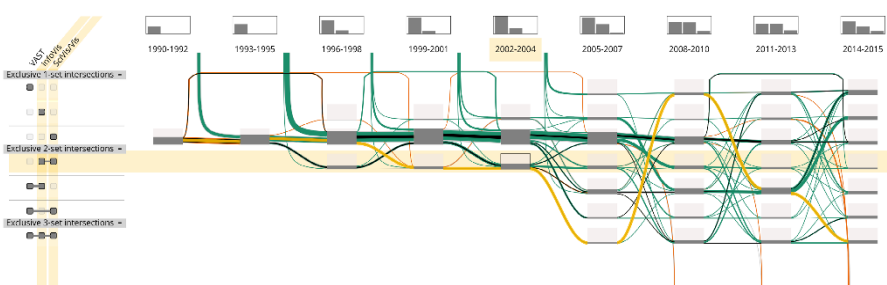


Fig. A.23: Set Streams visualize changes in set membership over time. Here, elements are authors, and sets are the three big visualization venues SciVis, InfoVis, and VAST. The time axis has been segmented into three-year intervals. © The authors. Generated with the *Set Streams* software by Shivam Agarwal.

data

number of variables: single
frame of reference: abstract

Set Streams by Agarwal and Beck (2020) is a technique for visualizing how set memberships of elements change over time. In the case of Set Streams, elements can be members of multiple sets, that is, the sets may overlap. Therefore, set intersections and their affiliated members are of particular interest. The visualization is based on a grid. Each row corresponds to a so-called exclusive set intersection, where intersections involve one, two, or three sets. The grid columns correspond to the time axis, where individual columns may represent time points or segmented time intervals, as shown in the figure. The fill level of the grid cells indicates the number of elements per corresponding exclusive set intersection and time primitive. Changes in set membership are visualized by curved streams between adjacent columns similar to Sankey diagrams (↪ p. 313). The streams may branch and merge depending on the underlying data changes. These streams make it possible for users to trace the path of elements through time and the various set intersections. On top of that, interaction techniques can be used to investigate the data in detail. For example, the grid rows can be sorted according to different criteria and a selected element can be highlighted across the entire grid. Moreover, two groups of elements can be defined using a dynamic query mechanism. The two groups are then distinctly colored in the visualization to facilitate their detailed comparison.

vis

mapping of time: static
dimensionality: 2D

Relevant references: Agarwal and Beck (2020)

TimeSets

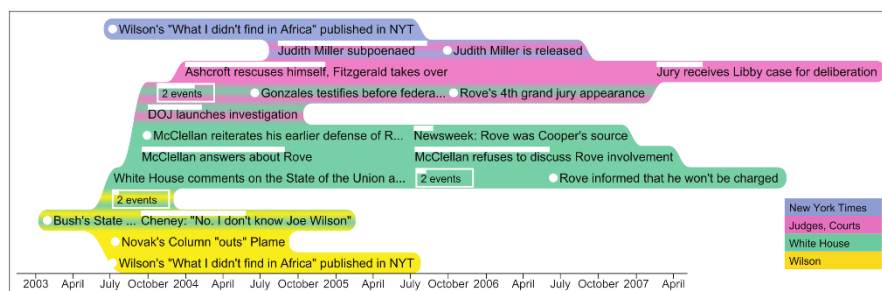


Fig. A.24: TimeSets visualization using the CIA leak case. The timeline contains events based on time points or intervals from 2002 to 2007. Each event has a label and topics. Events are positioned along the horizontal time axis based on time points and are vertically grouped by topics. A time-point event is shown with a white circle to its left and an interval-based event with a horizontal bar on top showing its duration. Each topic has a unique color, and events shared by two topics have gradient backgrounds. © 2015 SAGE. Reprinted, with permission, from Nguyen et al. (2016).

TimeSets by Nguyen et al. (2016) visualize set relationships among events using a timeline design (\hookrightarrow p. 258). Following two Gestalt principles of grouping, namely proximity and uniform connectedness, the TimeSets technique groups temporal events vertically with colored backgrounds according to their set memberships. Events shared by two sets are visualized using layers with a color gradient background transitioning between the colors of the two topics. TimeSets also dynamically adjust the event labels between three levels of detail to scale with the number of events. The number of displayed event labels can be reduced to easily follow events chronologically, which is controlled by a traceability layout algorithm. Xu et al. (2020b) incorporated additional analysis capacities in the TimeSets visualization to support open-source intelligence analysis with Twitter data, particularly the challenge of finding the right questions to ask, and to facilitate uncertainty analysis involving fake news. In a controlled experiment, TimeSets were significantly more accurate, and the participants preferred TimeSets for aesthetics and readability.

Relevant references: Nguyen et al. (2016) • Xu et al. (2020b)

time

primitives: points, intervals
arrangement: linear

data

number of variables: **single**
frame of reference: **abstract**

*vis*mapping of time: **static**
dimensionality: **2D**

time

Perspective Wall

primitives: points, intervals
arrangement: linear

data

number of variables: single
frame of reference: abstract

vis

mapping of time: static
dimensionality: 3D

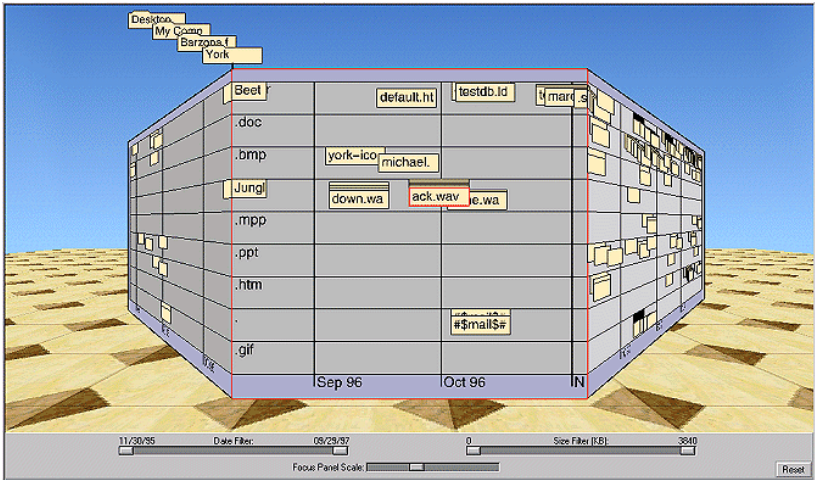


Fig. A.25: A perspective wall representing time-related information of a file system for a period of several months. The focus (currently set to September/October 1996) shows detailed text labels for files, whereas the context regions only indicate files as yellow boxes. © Inxight Federal Systems. Used with permission.

Time-oriented data that are linked to a longer time axis (i.e., wide span in time or many time primitives) are usually difficult to represent visually because the image becomes very wide and exhibits an aspect ratio that is not suited for common displays. The perspective wall by Mackinlay et al. (1991) is a technique that addresses this problem by means of a focus+context approach. The key idea is to map time-oriented data to a 3D wall. For a user-selected focus, full detail is provided in the center of the display. Two context representations show the data in the past (to the left) and in the future (to the right) with regard to the current focus. The context is bent perspectivevly to reduce the display space occupied by these regions, effectively allowing for better space utilization in the central focus. Interaction methods are provided to enable users to navigate in time in order to bring different time spans into focus. The actual data representation on the wall may vary across applications; the only requirement is that time is mapped linearly from left to right. For example, one can use bars as in the figure or more advanced visual representations such as the ThemeRiver (↪ p. 293).

Relevant references: Mackinlay et al. (1991)

DateLens

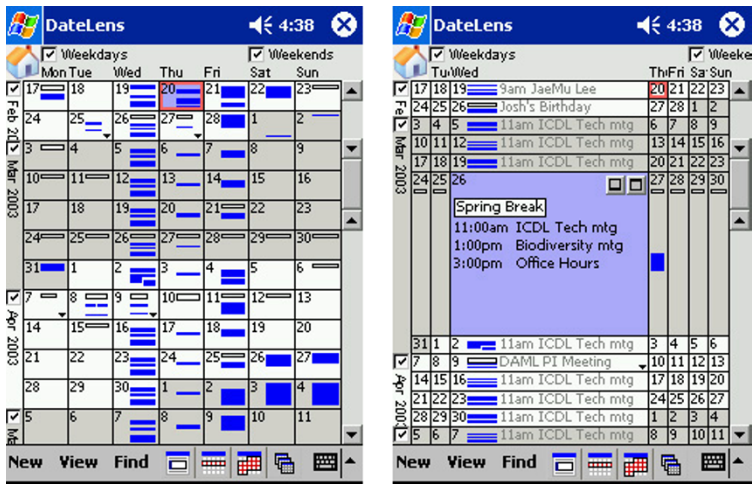


Fig. A.26: A calendar grid shows the items of one’s personal schedule as colored bars (left). Fisheye distortion is applied to show detailed textual information at the point the user is focusing on, and to maintain the context with less graphical detail (right). © *Courtesy of Ben Bederson.*

Most people use calendars to plan their daily life, for instance, to maintain a list of appointments or bookmark future events. Bederson et al. (2004) developed the DateLens to make it easier to work with a personal schedule on small displays. According to Langner et al. (2021), the DateLens is a classic technique for mobile data visualization. Because display space is usually limited on mobile devices (compared to common desktop displays), focus+context mechanisms are applied to present temporal information at different levels of detail. Based on a common tabular representation of a calendar (left), the DateLens magnifies selected table cells (right) so as to provide more display space for important information that is currently in the user’s focus. The fisheye distortion magnifies the focus and reduces graphical detail in the context of the display. If sufficient display space is available, calendar entries are shown in textual form. Otherwise, temporal intervals of calendar entries are indicated by bars that visualize the starting point and temporal extent of appointments and events stored in the calendar. Various interaction mechanisms allow users to view the calendar at different temporal granularities and to navigate forward and backward in time.

Relevant references: Bederson et al. (2004) • Langner et al. (2021)

time

primitives: intervals
arrangement: linear

data

number of variables: single
frame of reference: abstract

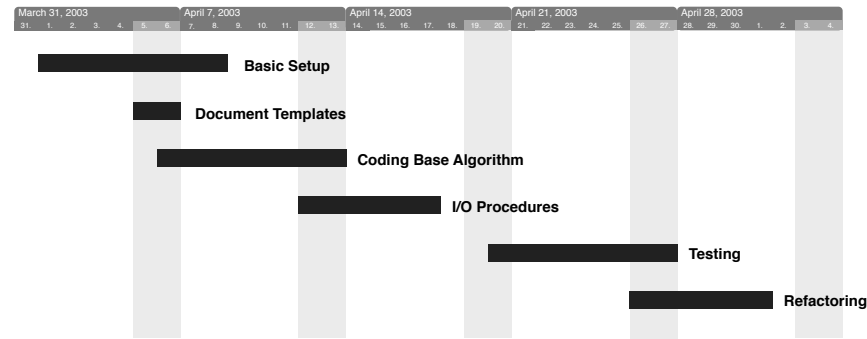
vis

mapping of time: static
dimensionality: 2D

time

Timeline

primitives: intervals
arrangement: linear



data

Fig. A.27: Bars are arranged relative to a time axis to visualize both the location and duration of intervals. One can also see how intervals are related to each other. © The authors.

number of variables: single
frame of reference: abstract

When the time primitives of interest are not points but intervals, the visualization has to communicate not only where in time a primitive is located, but also how long it is. A simple and intuitive way of depicting incidents with a duration is by marking them visually along a time axis. This form of visualization is called a timeline. Most commonly, a visual element such as a line or a bar represents an interval's starting point and duration (and consequently its end). The figure shows an example with bars. If multiple intervals share a common time axis, as in this example, it is even possible to discern how the various intervals are related to each other. Timelines are a very powerful visualization technique that, according to Tufte (1983), had been used long before computers even appeared. Many different variants of timelines exist in diverse visualization tools, where contemporary timelines come with a number of design options that go beyond the original approach (see Brehmer et al., 2017). Additional interaction techniques often allow users not only to view time intervals, but also to create and edit them. Prominent examples are LifeLines (↔ p. 341) and Gantt charts (↔ p. 253).

vis

mapping of time: static
dimensionality: 2D

Relevant references: Tufte (1983) • Brehmer et al. (2017)

Paint Strips

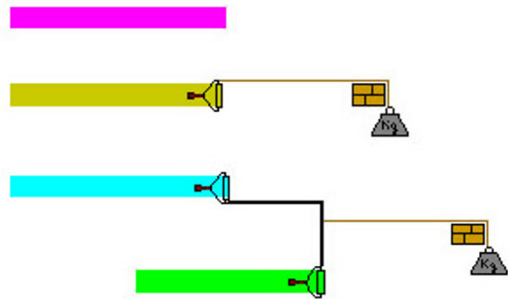


Fig. A.28: Paint strips indicate the location and duration of time intervals, effectively allowing users to assess relationships of intervals. Temporal indeterminacy of intervals is indicated by paint rollers that can move flexibly within certain constraints, which are represented by wall elements. © Courtesy of Luca Chittaro.

Chittaro and Combi (2003) designed paint strips to represent relations between time intervals for visualizing queries on medical databases. The technique is strongly related to timelines (\leftrightarrow p. 258), but here paint strips are used as equivalents of bars to indicate time intervals, and optionally, the indeterminacy of intervals is communicated by placing paint rollers at either end of the paint strips. A paint roller with a weight attached to it means this interval can possibly extend in time. Graphical depictions of wall elements represent constraints on the extension. This way, the maximum duration and earliest start or latest end of intervals are defined, depending on which end of the painting strip the paint rollers are attached to. It is also possible to link strips, which means if one strip moves, the other one moves to the same extent as well. This relationship is indicated graphically by connecting the involved paint rollers before attaching them to the rope that holds the weight. Paint strips were specially developed for medical applications but can be used anywhere where indeterminate time intervals have to be visualized. Thanks to the simplicity of the paint strip metaphor, there is room for application-dependent enhancements, such as textual annotations for start and end points as well as for durations of intervals.

Relevant references: Chittaro and Combi (2003)

time

primitives: intervals
arrangement: linear

data

number of variables: single
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

PlanningLines

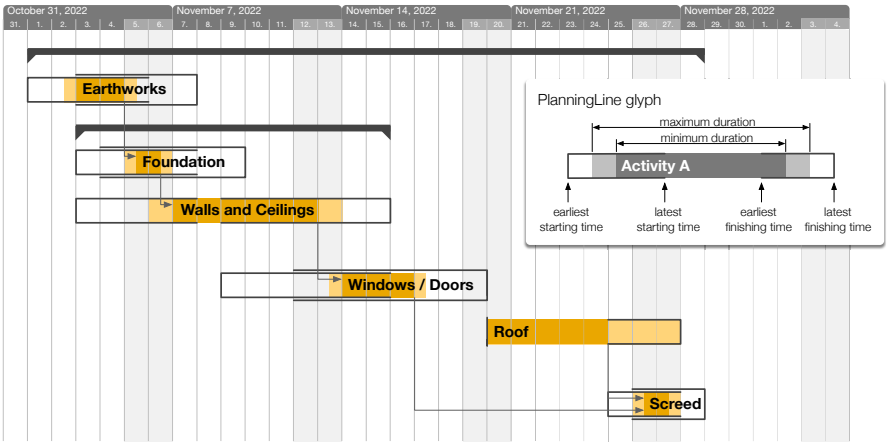


Fig. A.29: Project plan of construction works that represents temporal uncertainties via PlanningLines. A PlanningLine glyph consists of two encapsulated bars that represent minimum and maximum duration. The bars are bounded by two caps encoding the start and end intervals. © The authors. Adapted from Aigner et al. (2005).

Since the future is always inherently connected with possible uncertainties, delays, and the unforeseen, these issues need to be dealt with in many domains like project management or medical treatment planning. PlanningLines by Aigner et al. (2005) allow the representation of temporal uncertainties, thus supporting project managers in their difficult planning and controlling tasks. PlanningLines have been designed to be easily integrated into well-known timeline-based visualization techniques such as Gantt charts (↔ p. 253). As indicated in the figure, a single glyph provides a visual representation of the temporal indeterminacy of a single activity, facilitates the identification of (un)defined attributes, supports in maintaining logical constraints (e.g., bars may not extend caps), and gives a visual impression of the individual and overall uncertainties. Uncertainties might be introduced by explicit specifications, usually connected with future planning (e.g., “The meeting will start between 12 p.m. and 2 p.m.” – which might be any point in time between noon and 2 p.m.) or is implicitly present in cases where data are given with respect to multiple temporal granularities (e.g., data given on a granularity of days and shown on an hourly scale).

The original concept of PlanningLines has been extended by Höhn et al. (2022) to allow for visualizing both, variable and fixed activities with possibly open start and end. Moreover, the probability distribution of the activities can be defined and input methods for mouse and keyboard as well as stylus interaction are offered.

Relevant references: Aigner et al. (2005) • Höhn et al. (2022)

Time Annotation Glyph

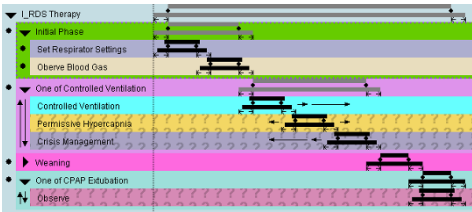
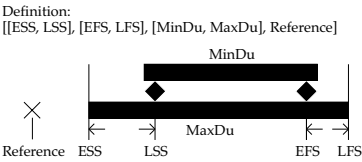


Fig. A.30: The time annotation glyph was designed to represent temporal constraints. It uses the metaphor of bars that lie on pillars. Left: single glyph and associated parameters; right: glyphs being applied in a tool for representing medical treatment plans. © *Courtesy of Robert Kosara.*

The time annotation glyph by Kosara and Miksch (2001) uses the simple metaphor of bars that lie on pillars to represent a complex set of time attributes. Four vertical lines on the base specify the earliest and the latest starting and ending times. Supported by these pillars lies a bar that is as long as the maximum duration. On top of the maximum duration bar, a bar that represents the minimum duration lies upon two diamonds indicating the latest start and the earliest end. Furthermore, undefined parts and different granularities are indicated visually. Because of this metaphor, a few simple time parameter constraints can be understood intuitively. For example, the minimum duration cannot be shorter than the interval between the latest start and earliest end – if it was, the minimum duration bar would fall down between its supports. All of the parameters might be defined relative to a reference point that is also represented graphically. In summary, the following parameters are shown: earliest starting shift (ESS), latest starting shift (LSS), earliest finishing shift (EFS), latest finishing shift (LFS), minimum duration (MinDu), and maximum duration (MaxDu). The technique is used to represent the time annotations of medical treatment plans within the AsbruView application.

Relevant references: Kosara and Miksch (2001)

time

primitives: intervals
arrangement: linear

data

number of variables: single
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

time

SOPO Diagram

primitives: intervals
arrangement: linear

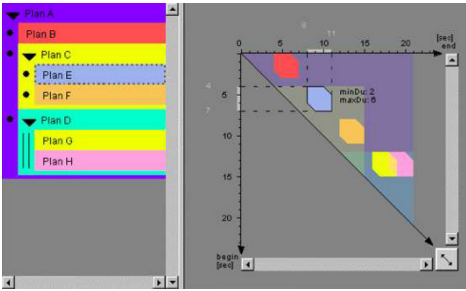
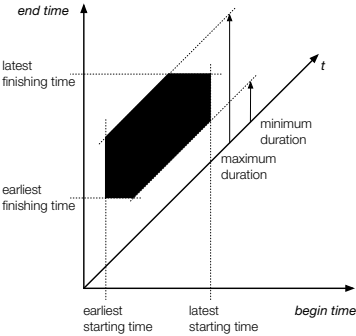


Fig. A.31: A SOPO diagram (left) shows the possible configurations of the begin and end times of an event via a constrained polygonal shape. SOPOView (right) is an interactive visualization tool for working with SOPOs applied for medical treatment plans. © *Courtesy of Robert Kosara.*

data

number of variables: single
frame of reference: abstract

For planning and scheduling, the temporal extents of events can be characterized by sets of possible occurrences (SOPOs), i.e., a set of possible begin and end times during which an event may happen. Rit (1986) defined a theoretical model for the definition and propagation of temporal constraints for scheduling problems. A graphical representation of SOPOs was introduced as a visual aid for understanding and solving such problems. In this representation, the extent of temporal uncertainty is expressed via a polygonal shape. The axes of a SOPO diagram represent begin time (x-axis) and end time (y-axis). Similar to the triangular model (\leftrightarrow p. 267), points in this diagram do not represent points in time, but complete intervals specified by their begin (x-coordinate) and end time (y-coordinate). Hence, the extent of an interval is represented by its position, not its visual extent. The area an item covers reflects all intervals that fit the specification given by means of earliest start, latest start, earliest end, latest end, minimum, and maximum duration. Moreover, the exact occurrence of an event may be constrained by other related events which further modify the sets of possible occurrences. The propagation of such constraints is aided graphically, e.g., via overlaps of individual SOPOs. Later, this idea was interactively enhanced and further developed to be applied for visualizing medical treatment plans in the tool SOPOView by Kosara and Miksch (2002).

vis

mapping of time: static
dimensionality: 2D

Relevant references: Rit (1986) • Kosara and Miksch (2002)

TimeNets

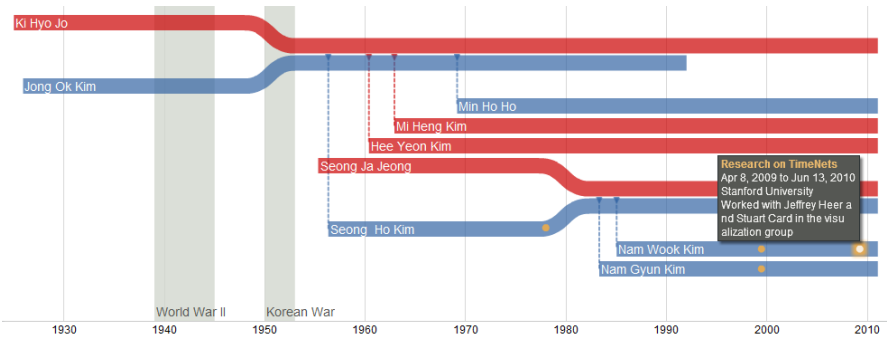


Fig. A.32: TimeNets visualize temporal and structural aspects of genealogical data. Bands that extend along the horizontal time axis visualize individuals. Marriage and divorce are indicated by converging and diverging bands, respectively. Children are connected to their parents via drop lines. Labels are shown for the persons’ names as well as for historical and personal events. © *Courtesy of Jeffrey Heer and Nam Wook Kim.*

Genealogical data are an interesting source of time-oriented information. In such data, not only family structures are of interest, but also temporal relationships. Kim et al. (2010) propose the TimeNets approach, which aims to visualize both of these aspects. TimeNets are similar to storylines (↔ p. 264) and represent persons as individual bands that extend horizontally along a time axis from left to right. Each band shows a label of the person’s name and different colors are used to encode sex: red is reserved for females, and males are shown in blue. Marriage of persons is visualized by converging the corresponding bands, while divorce is indicated by diverging bands. When a child is born, a new band is added to the display. A so-called drop line connects the band of the child to the parents’ bands to convey the parent-child relationship. In order to allow users to focus on relevant parts of the data, a degree-of-interest (DoI) algorithm is applied. Bands below the DoI threshold are filtered out or smoothly fade in where they are linked to bands of relevant persons. Users can select multiple persons to focus on. On each change of the focus, the visualization shows a smooth transition of the display to keep users oriented.

Relevant references: Kim et al. (2010)

time

primitives: intervals
arrangement: linear

data

number of variables: single
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

time

Storyline Visualization

primitives: intervals
arrangement: linear

data

number of variables: single
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

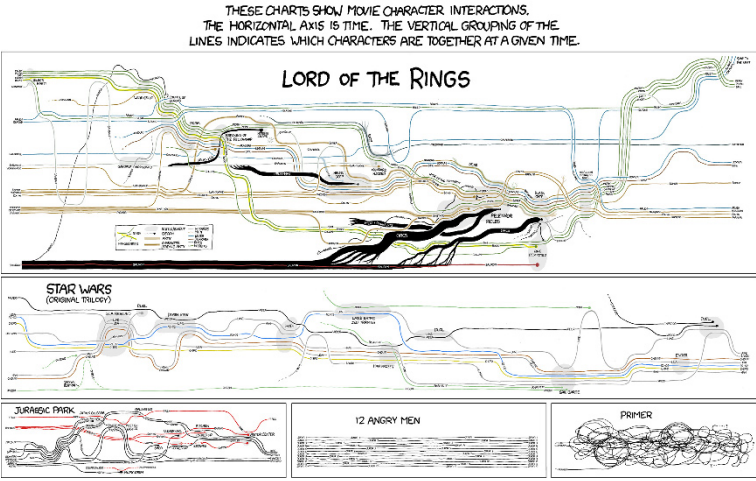


Fig. A.33: Storyline visualization of character co-occurrence in movies. Time is shown from left to right and each band represents a movie character. The proximity of bands indicates character co-occurrence. © 2012 “Movie Narrative Charts” from xkcd comics by Randall Munroe.

Storylines have become a popular means to visualize the relationships of objects over time (see Tang et al., 2019b). They are similar to timelines (↔ p. 258) in that each object is represented as a band that extends from left to right along the horizontal time axis. Yet, for storylines, individual bands can bend to change their vertical position in the visualization. This way, the bands can be bundled to converge to denser clusters, or they can be thinned out to create distance from other bands. This basic encoding is useful in a variety of scenarios. The figure shows storyline visualizations of character co-occurrence in movies. Here, bands running closely and in parallel represent times when the corresponding characters appear together. Storyline visualizations can also be useful for comparing the similarity of data objects over time. In this case, bands representing similar objects run in clusters. Changes in cluster affiliation, that is, individual bands turning away from their original cluster to a new one, may indicate significant events in the development of data objects over time. The TimeNets (↔ p. 263) technique employs the idea of storylines to visualize genealogical data. Despite their simplicity, storyline visualizations are not trivial to generate. The key difficulty is to find a layout for the bands with minimal crossings, bend angles, and display space. Finding a globally optimal band layout is a computationally complex problem. Tanahashi and Ma (2012), Liu et al. (2013), Kostitsyna et al. (2015), and Tanahashi et al. (2015) investigate optimization and computational aspects of storyline visualizations in detail.

Relevant references: Tanahashi and Ma (2012) • Liu et al. (2013) • Kostitsyna et al. (2015) • Tanahashi et al. (2015) • Tang et al. (2019b)

Temporal Mosaic

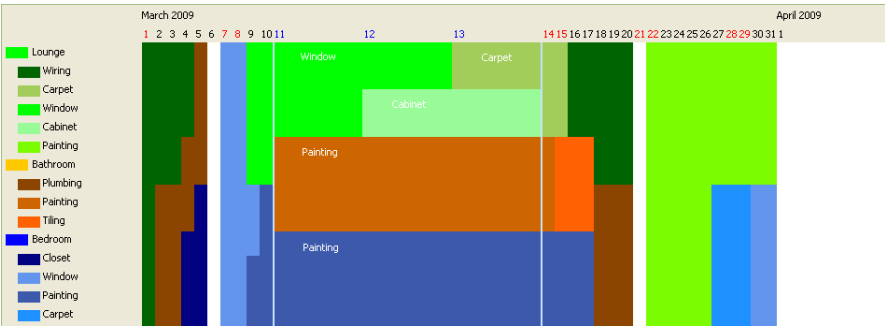


Fig. A.34: Temporal mosaic visualizing concurrent activities of a house renovation plan. Time is mapped from left to right, concurrent activities are stacked along the vertical axes and colors indicate the type of activity. © *Courtesy of Saturnino Luz.*

Temporal mosaic is a technique for the visualization of parallel time-based streams. It provides a compact way of representing concurrent events by allocating a fixed drawing area to time intervals and partitioning that area according to the number of events that co-occur in that time interval. The figure shows a temporal mosaic representation of a house renovation schedule with hierarchical event dependencies. Time is mapped along the horizontal axis. Differently colored regions in the mosaic indicate at which time a particular renovation is scheduled. One can see that some renovations can be executed in parallel, while others depend on each other and need to be carried out one after the other. The temporal mosaic technique has also been evaluated in the context of meeting browsing tasks.

Relevant references: Luz and Masoodian (2004) • Luz and Masoodian (2007) • Luz and Masoodian (2010)

time

primitives: intervals
arrangement: linear

data

number of variables: single
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

time

Train Delay Uncertainty

primitives: intervals
arrangement: linear

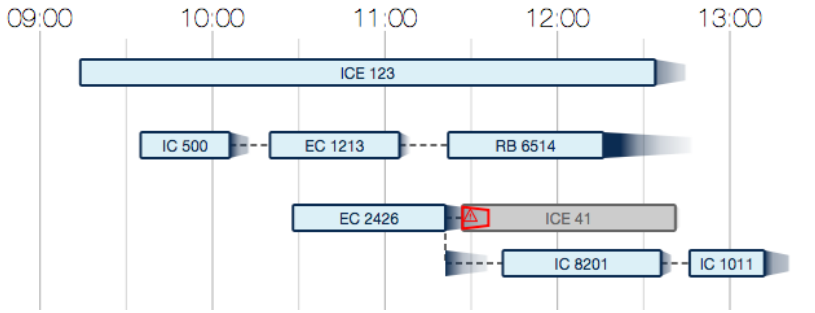


Fig. A.35: Train itinerary planning under consideration of uncertain train delays. Trains are shown by rectangles at whose right end a color gradient indicates potential delays. ©¹ Courtesy of Tatiana von Landesberger.

data

number of variables: single
frame of reference: abstract

While train delays are minimal in some countries, they can be a critical issue in other countries, especially when a train connection is missed due to a delayed train. While itinerary planning already takes some buffer into account for travelers to reach connecting trains, the consequences of delayed trains are usually not considered. Wunderlich et al. (2017) describe a visual solution to help travelers in planning and understanding itineraries under the influence of uncertain train delays. As illustrated in the figure, the solution uses a horizontal time axis and labeled rectangles representing train trips. The trains are positioned according to their scheduled departure and arrival times. Train delay, more specifically, the expected cumulative train delay is indicated via a color gradient at the arrival (right side of the rectangles). The cumulative delay represents the probability that a train is delayed no longer than a certain amount of minutes. Situations, where a train’s expected cumulative delay could lead to missing a train connection, are marked with a small exclamation sign and the potentially missed train is shown in gray. If possible, the system automatically suggests alternative connections that would avoid missed transfers.

vis

mapping of time: static
dimensionality: 2D

Relevant references: Wunderlich et al. (2017)

Triangular Model

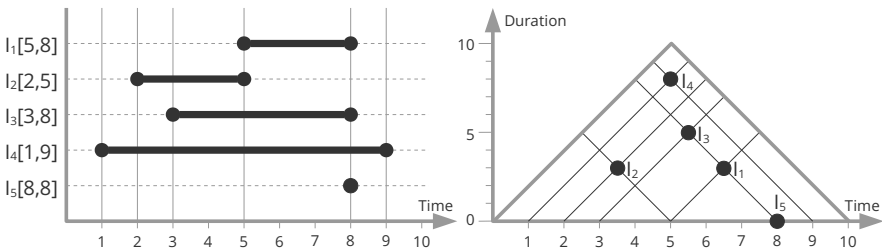


Fig. A.36: In contrast to common timelines (left), which represent intervals as lines or bars, intervals become points in the triangular model representation (right), which encodes interval start and end as well as interval duration. © The authors. Adapted from Qiang et al. (2012).

Qiang et al. (2012) popularize the original idea by Kulpa (1997) and Kulpa (2006) of visualizing time intervals using a triangular model. In this model, an interval is represented as a dot with two attached arms. The dot is placed so that the arms connect the horizontal time axis exactly at the start and the end of the represented interval. The point's height corresponds to the interval's duration, which is mapped along the vertical axis. This representation is useful in many scenarios, in particular, when it comes to reasoning about properties and the relationships of multiple intervals. In that case, the triangular model generates easily distinguishable visual patterns for all possible interval relations. The figure compares a standard representation (intervals shown as bars) with a corresponding representation using the triangular model (intervals shown as dots).

Relevant references: Qiang et al. (2012) • Qiang et al. (2014) • Kulpa (1997) • Kulpa (2006)

time

primitives: intervals
arrangement: linear

data

number of variables: single
frame of reference: abstract

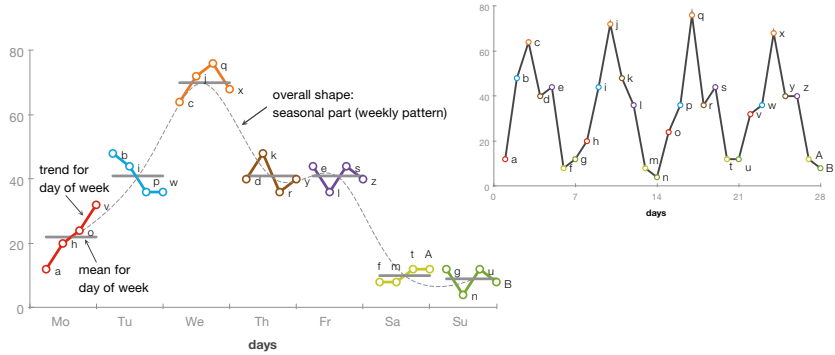
vis

mapping of time: static
dimensionality: 2D

time

Cycle Plot

primitives: points
arrangement: linear, cyclic



data

Fig. A.37: Cycle plots make it easier to discern seasonal components and general trends from time-series data (left), which is considerably more difficult when using line plots (right). © The authors. Adapted from Cleveland (1993).

number of variables: single
frame of reference: abstract

Time-series data may contain seasonal components as well as general trends, which is also reflected in many statistical models. Cleveland (1993) describes cycle plots as a technique to make seasonal components and trends visually discernable. This is achieved by showing individual trends as line plots embedded within a plot that shows the seasonal pattern. For constructing a cycle plot, one has to define the time primitives to be considered for the seasonal component. The horizontal axis of the cycle plot is then subdivided accordingly. The left part of the figure demonstrates this using weekdays. We are interested in the trend for each weekday and the general weekly pattern. The data for a particular weekday are visualized as a separate line plot (e.g., data of the 1st, 2nd, 3rd, and 4th Monday). This allows the identification of individual trends for each day of the week. In the figure, we see an increasing trend for Mondays, but a decreasing trend for Tuesdays. Additionally, the cycle plot shows the mean value for each weekday (depicted as gray lines). Connecting the mean values as a line plot (dashed line in the figure) reveals the seasonal pattern, which in this case is a weekly pattern that clearly shows a peak on Wednesday.

vis

mapping of time: static
dimensionality: 2D

Relevant references: Cleveland (1993)

Tile Maps

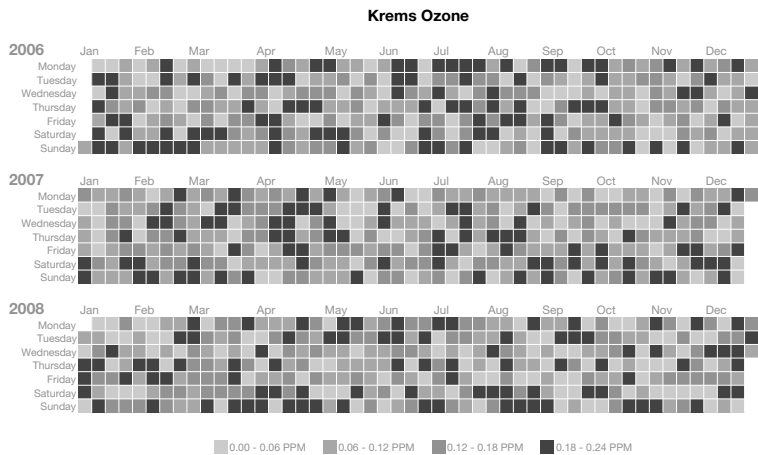


Fig. A.38: The matrices show ozone measurements made in the course of three years, where data values for individual days are encoded to the brightness of matrix cells. © The authors. Adapted from Mintz et al. (1997).

Tile maps as described by Mintz et al. (1997) represent a series of data values along a calendar division. The idea behind this heatmap-based technique is to arrange data values according to different temporal granularities. For example, data values measured on a daily basis are displayed in a matrix where each cell (or tile) corresponds to a distinct day, a column represents a week, and a row represents all data values for a particular weekday. One additional level of granularity can be integrated by stacking multiple matrices as shown in the figure. Data values are visualized by varying the lightness of individual tiles. A visual representation constructed this way can be interpreted quite easily, because it corresponds to our experience of looking at calendars. The arrangement as a two-dimensional matrix allows users to identify long-term trends by considering the matrix as a whole, to discern individual trends for Mondays, Tuesdays, and so forth by looking at the matrix rows, and to derive weekly patterns by investigating matrix columns. For example, the U.S. Environmental Protection Agency provides a web tool that generates tile maps automatically for specified air pollutants and locations.

Relevant references: Mintz et al. (1997)

time

primitives: points
arrangement: linear, cyclic

data

number of variables: single
frame of reference: abstract

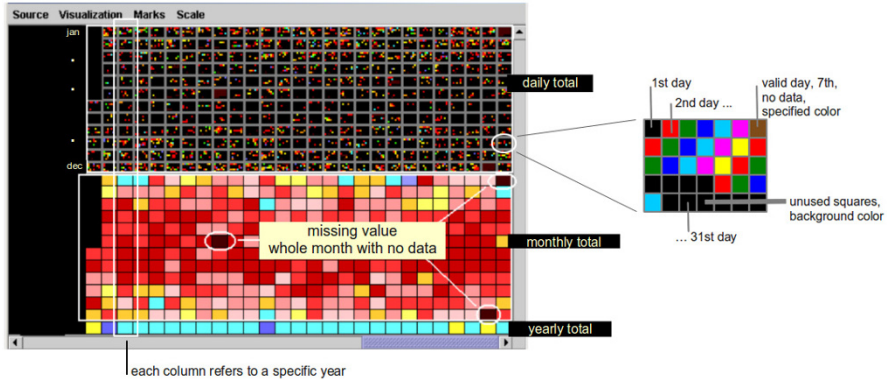
vis

mapping of time: static
dimensionality: 2D

time

primitives: points
arrangement: linear, cyclic

Multi Scale Temporal Behavior



data

number of variables: single
frame of reference: abstract

Fig. A.39: Different levels of granularity are shown simultaneously in one display to allow users to explore patterns in precipitation data from Brazil at different temporal levels. © Courtesy of Milton Hirokazu Shimabukuro.

vis

mapping of time: static
dimensionality: 2D

The Multi Scale Temporal Behavior technique by Shimabukuro et al. (2004) comprises different levels of granularity and aggregation to explore patterns at different temporal levels. The basis for the visualization is a matrix that is divided vertically into three regions, one for each of the three scale levels: daily data, monthly data, and yearly data (top to bottom in the figure). Each column of the matrix represents a year worth of data. The cells in the topmost region represent months. They show full detail by color-coding individual pixels within a cell according to daily total values. The middle region shows aggregated data. Here cells are no longer subdivided into pixels, but are colored uniformly, where the color represents the aggregated monthly total value. The same principle is applied to the bottom region (in fact, the bottom row). Twelve monthly values are aggregated into a single value for the year, which can again be represented by color. A significant and non-trivial problem in dealing with real-world datasets are missing data values. This issue is tackled here by preprocessing the data and marking missing values as such in the display.

Relevant references: Shimabukuro et al. (2004)

GROOVE

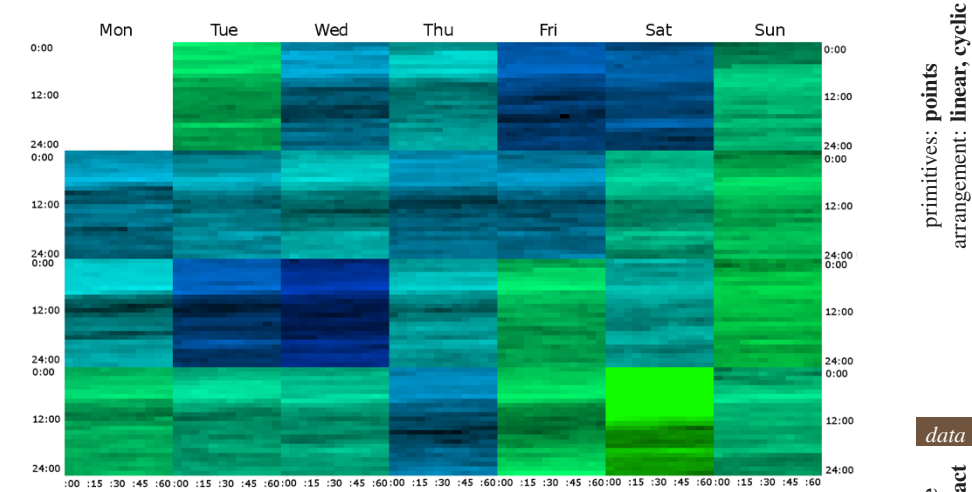


Fig. A.40: GROOVE visualizations combine detail and overview readings and use regular layouts based on time granularities. Here, police assignments are shown over the course of one month. For each 5-minute interval, the number of units assigned to a duty is given. Each block shows a day, rows of blocks represent weeks, columns of blocks show the day of the week, and within each block, a row of pixels displays one hour. Color components are used to show detail values for pixels (lightness – bright: low, dark: high) and average values per block (hue – green: low, blue: high). © The authors.

GROOVE (Granularity Overview OVERlay) visualizations as presented by Lam-marsch et al. (2009) utilize a user-configurable set of four time granularities to partition a dataset in a regular manner. A recursive layout shows columns and rows of larger blocks and a pixel arrangement within blocks for the details of the structure, where different arrangements might be chosen (e.g., row-by-row or back-and-forth) similar to the recursive patterns approach (↔ p. 282). GROOVE visualizations combine overview (aggregated values) and details in one place using one of three kinds of overlays. This allows micro and macro readings and avoids eye movements between the overview and detail representations. First, color components can be employed with color-based overlay as shown in the figure. Second, opacity overlay applies interactive crossfading between the overview and the detail display. Third, spatial overlay can be used for viewing the data selectively at different levels of aggregation by expanding and collapsing areas.

Relevant references: Lam-marsch et al. (2009)

time

primitives: points
arrangement: linear, cyclic

data

number of variables: single
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

SolarPlot

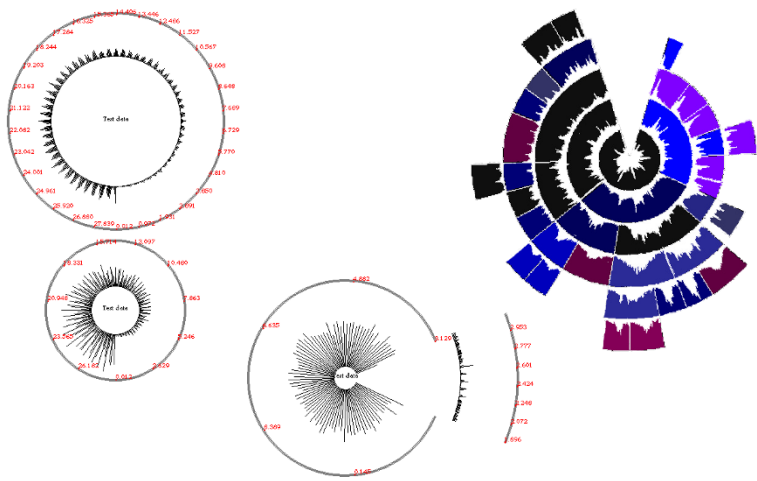


Fig. A.41: Daily values of ticket sales data over a period of 30 years are plotted around the circumference of a resizable circle (left). For a selected interval, a more detailed arc can be shown (center). The technique can be enhanced so as to combine the representation of data and a hierarchical structure, in this case, email traffic and company organization (right). © 1998 IEEE. Reprinted, with permission, from Chuah (1998).

With the SolarPlot technique introduced by Chuah (1998), values are plotted around the circumference of a circle as shown left in the figure. Much like in a circular histogram, the first step is to partition the data series into a number of bins. Each bin is represented by a sunbeam whose length encodes the frequency of data items in the corresponding bin. The SolarPlot determines the number of bins dynamically depending on the size of the circle. Users are allowed to expand or contract the circle in order to get more or fewer bins, or in other words, to get a more or less detailed representation of the data. This way it is possible to explore the data at different levels of abstractions and to discern patterns globally across aggregation levels. The SolarPlot also supports switching locally to a more detailed plot for a user-selected focus interval as shown in the center of the figure. Chuah (1998) further suggests a variation of the SolarPlot called SolarPlot + Aggregate TreeMap, where the display of data is combined with a visual representation of a hierarchical structure as shown to the right.

Relevant references: Chuah (1998)

Cluster and Calendar-Based Visualization

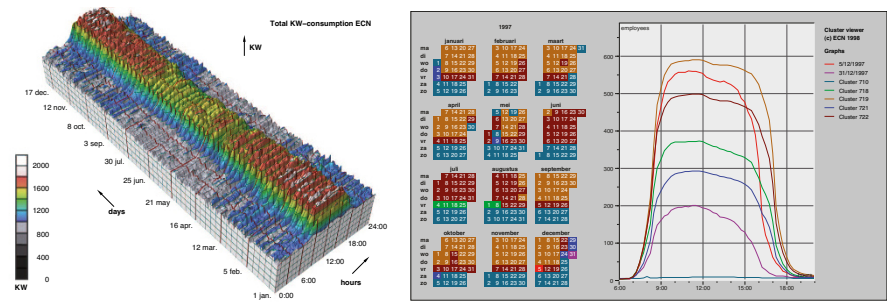


Fig. A.42: The 3D visualization (left) represents daily power consumption patterns for several weeks of a year. The calendar representation (center) color-codes cluster affiliation with respect to daily patterns of employees’ office hours as shown in the line plots (right). © 1999 IEEE. Reprinted, with permission, from van Wijk and van Selow (1999).

Temporal patterns can indicate at which time of the day certain resources are highly stressed. Relevant applications can be found in computing centers, traffic networks, or power supply networks. The cluster and calendar-based approach by van Wijk and van Selow (1999) allows analysts to find such temporal patterns at different temporal granularities. The starting point of the approach is to consider the course of a day as a line plot (↔ p. 233) covering the 24 hours of a day. Multiple daily courses of this kind are visualized as a three-dimensional height field (left part of the figure), where the hours of the day are encoded along one axis, individual days are encoded along the second axis, and data values are encoded as height (along the third axis). This allows users to detect short-term daily patterns and long-term trends of the data at higher temporal granularity. To assist in the analysis of the data, the approach further groups similar daily courses into clusters. The data belonging to a particular cluster are aggregated to define a representative for that cluster, i.e., the average line plot for all days belonging to a cluster. Cluster affiliation of individual calendar dates is then color-coded into a calendar (center) and individual or clustered daily patterns are shown as line plots (right). The user can adjust the number of clusters to be shown so as to find the level of abstraction that suits the data and the task at hand. The combination of analytical and visual methods as applied here is useful for identifying days of common and exceptional daily behavior.

Relevant references: van Wijk and van Selow (1999)

time

primitives: points
arrangement: linear, cyclic

data

number of variables: single
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D, 3D

time

primitives: points
arrangement: cyclic

Enhanced Interactive Spiral

data

number of variables: single
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

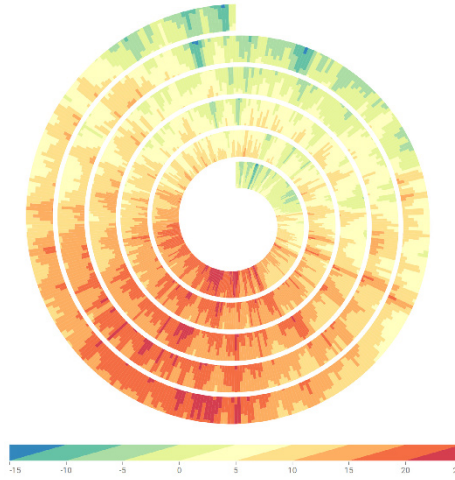


Fig. A.43: The spiral shows the daily temperature measured in the city of Rostock from 2006 to 2010. The blue and green colors in the upper part of the spiral represent the colder winters of 2009 and 2010. © The authors. Generated with [EnhancedSpiral.js](#) by Christian Tominski.

Tominski and Schumann (2008) apply the enhanced two-tone color-coding by Saito et al. (2005) to visualize time-dependent data along a spiral. Each time primitive is mapped to a unique segment of the spiral. Each segment is subdivided into two parts that are colored according to the two-tone coloring method. The advantage of using the two-tone approach is that it realizes the overview+detail concept by design. The two colors used per spiral segment allow users to quickly recognize the value range of that segment (quick overview). If a perceived value range is of interest to the user, the proportion of the two colors indicates the particular data value more precisely (detailed inspection). The enhanced spiral can be adjusted interactively in various ways. Setting a suitable number of segments per spiral cycle is crucial for being able to see cyclic patterns. The search for such a suitable parametrization can be supported by guidance methods (see Ceneda et al., 2018). Further parameters can be tuned to adjust the visualization, including the number of time primitives, the number of spiral cycles, the direction of the mapping of time (inward or outward), the color scale, and the number of color classes. Navigation in time is possible via direct manipulation of a time slider.

Relevant references: Tominski and Schumann (2008) • Ceneda et al. (2018) • Saito et al. (2005)

ClockMap

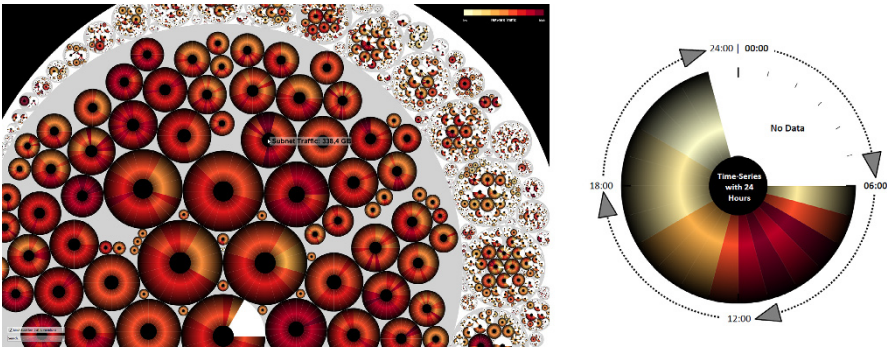


Fig. A.44: ClockMap is a visualization technique that supports the user in exploring and finding patterns in hierarchical time-series data. Left: Example shows 24 hours of network traffic of a large number of subnets in its hierarchical decomposition. Right: Clockeye glyph. Each glyph circle represents a 24-hour time series where color is mapped to the amount of network traffic in bytes. © Fabian Fischer. <https://ff.cx/clockmap/>

Hierarchical structures are commonplace for many datasets in a variety of application domains, such as IT networks, budget data, file systems, or regional decompositions. A well-known visualization technique for hierarchical data with quantitative attributes is the treemap. A treemap is a space-filling visualization technique based on nesting, i.e., an iterative subdivision of rectangular areas. However, in their basic form, treemaps focus on time-invariant data and are not suitable to represent changes over time. ClockMaps aim to fill this gap by combining circular glyphs (clockeyes) using a circular treemap layout. Individual glyphs (clockeyes) are based on the metaphor of an analog clock and represent a time series in a circle, whereas color is used to represent a quantity for each radial line. In doing so, the temporal changes of each node as well as the hierarchical structure of the data are combined in a single visualization technique. The size of a clockeye glyph is determined by the amount of items that are nested within and does not represent a data attribute. Moreover, interactive features support the user in exploring and finding patterns in hierarchical time-series data. Users are able to drill-down, and apply semantic zoom, as well as details-on-demand.

Relevant references: Fischer et al. (2012)

time

primitives: points
arrangement: cyclic

data

number of variables: single
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

time

SpiraClock

primitives: intervals
arrangement: cyclic

data

number of variables: single
frame of reference: abstract

vis

mapping of time: dynamic
dimensionality: 2D

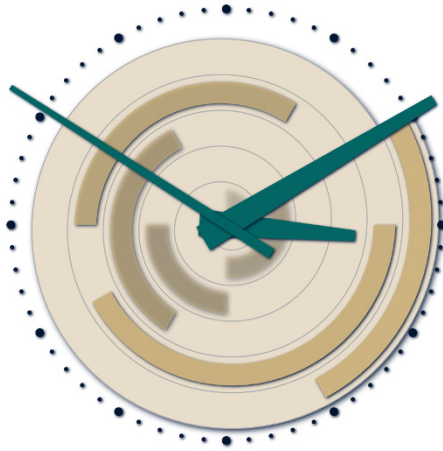


Fig. A.45: The SpiraClock combines the classic analog clock with a spiral display of future events. The minute hand currently points to a meeting that has just started. Future events are aligned along a spiral within the clock face. © The authors. Generated with [SpiraClock.js](#) by Christian Tominski.

The SpiraClock invented by Dragicevic and Huot (2002) visualizes time by using the clock metaphor. The visual representation consists of a clock face and clock hands indicating the current time. The interior of the clock shows a spiral that extends from the clock’s circumference inward toward its center. Each cycle of the spiral represents one hour, with the current hour being shown at the outermost cycle and future hours displayed in the center (about five hours overall in the figure). Time intervals (e.g., meetings or appointments) are represented as spiral segments embedded into the spiral cycles exactly where the intervals start and end. This kind of representation makes it easy to see if the intervals are overlap-free and if the intervals are uniformly distributed in time (e.g., to confirm that there are sufficient breaks and no conflicting appointments). As time advances, the spiral is constantly updated and future intervals gradually move outward until they are current. Past intervals gradually fade out. Overall, the SpiraClock enhances classic clocks with a preview of the near future and a brief view of the past. The SpiraClock allows users to drag the clock handles to visit different points in time. Moreover, intervals of interest can be highlighted and corresponding textual annotations can be displayed.

Relevant references: Dragicevic and Huot (2002)

Horizon Graph

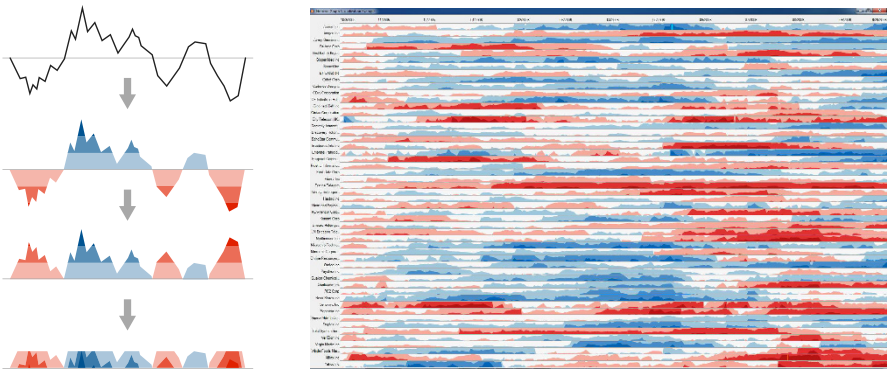


Fig. A.46: Horizon graphs require only little screen space and are very useful for comparing multiple time-dependent variables. Left: construction of a horizon graph from a line chart; right: stock market data visualized as stacked horizon graphs. *Left: © The authors. Adapted from Reijner (2008). Right: © Courtesy of Hannes Reijner.*

Reijner (2008) describes horizon graphs as a visualization technique for comparing a large number of time-dependent variables. Horizon graphs are based on the two-tone pseudo coloring technique by Saito et al. (2005). The left part of the figure demonstrates the construction of horizon graphs (from top to bottom). Starting from a common line plot, the value range is divided into equally sized bands that are discriminated by increasing color intensity toward the maximum and minimum values while using different hues for positive and negative values. Then, negative values are mirrored horizontally at the zero line. Finally, the bands are layered on top of each other. This way, less vertical space is used, which means data density is increased while the resolution is preserved. A study by Heer et al. (2009) has shown that mirroring does not have negative effects and that layered bands are more effective than the standard line plot (\leftrightarrow p. 233) for charts of small size. To improve the basic concept of horizon graphs, several extensions have been proposed since their introduction. Perin et al. (2013) extend horizon graphs with dedicated interaction techniques for adjusting the zero line and the number of colored bands. With their work on qualizon graphs, Federico et al. (2014) combine quantitative data with qualitative abstractions. Moreover, Dahnert et al. (2019) present further variations called collapsed horizon graphs, where the idea of horizon graphs is extended to two dimensions. That is, apart from the vertical quantitative variable, also the horizontal temporal dimension is being collapsed using bivariate color maps.

Relevant references: Reijner (2008) • Saito et al. (2005) • Heer et al. (2009) • Perin et al. (2013) • Federico et al. (2014) • Dahnert et al. (2019)

time

primitives: points
arrangement: linear

data

number of variables: single, multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

time

primitives: points
arrangement: linear

data

number of variables: single, multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

VizTree

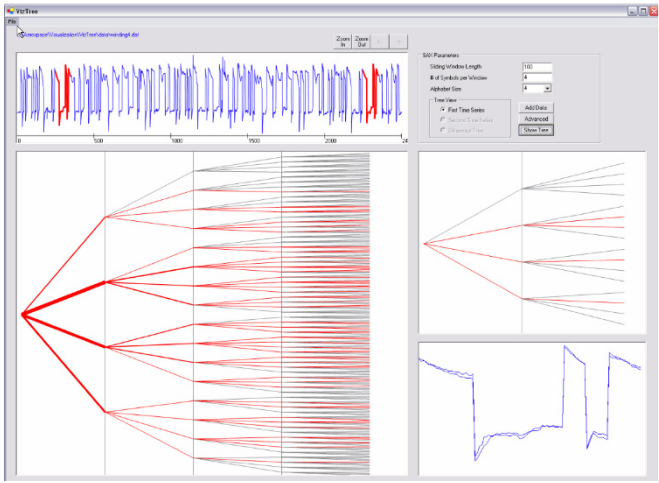


Fig. A.47: Time series of arbitrary lengths are represented as fixed-length subsequence trees. The figure shows a winding dataset that records the angular speed of a reel. Top left: input time series; top right: parameter setting area for discretization and subsequence length; center left: subsequence tree for the time series; center right: detail view of the tree shown in the left panel; bottom right: subsequences matching a particular string representation (e.g., subsequences starting with ‘ab’) whereas positions of matched subsequences are highlighted in the top-left panel. © *Courtesy of Eamonn Keogh.*

VizTree by Lin et al. (2005) is a time-series pattern discovery and visualization system for massive time-series datasets. It uses the time-series abstraction method SAX (Symbolic Aggregate approXimation) to discretize numeric time series into sequences of abstract symbols like ‘abacacc’ (see Lin et al., 2003; Lin et al., 2007). Subsequences (patterns) are generated by moving a sliding window along the sequence. These subsequences are combined and represented by a horizontal tree visualization where the frequency of a pattern is encoded by the thickness of a branch (or light gray if the frequency is zero). The VizTree interface consists of multiple coordinated views that show the input time series along with the subsequence tree as well as control and detail-on-demand panels. VizTree can be used to accomplish different pattern discovery tasks interactively: finding frequently occurring patterns (i.e., motif discovery) and surprising patterns (i.e., anomaly detection), query by content, and the comparison of two time series by calculating a difference tree.

Relevant references: Lin et al. (2005) • Lin et al. (2003) • Lin et al. (2007)

Time Curves

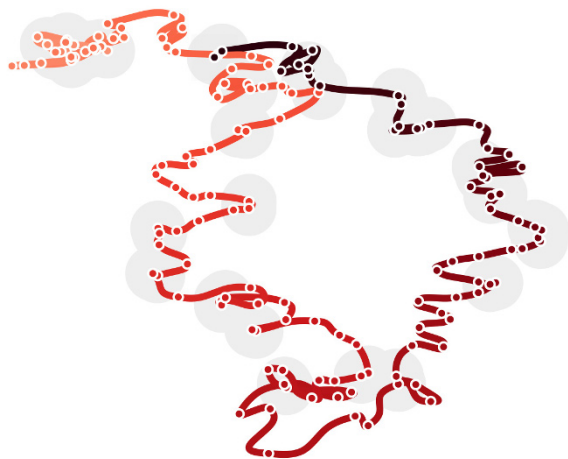


Fig. A.48: Time Curve showing global cloud circulation data. Time points are positioned based on their similarity, that is, similar points are close to each other, and distant points are dissimilar. Subsequent time points are connected to form a contiguous curve spanning the entire timeline. © ⓘ The authors. Generated with the *Time Curves* software by Benjamin Bach.

Bach et al. (2016) describe a technique called Time Curves. The metaphor behind this technique is to fold a line plot 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 time points can be defined. Technically, each time point is assigned a two-dimensional position based on the similarity metric. This can, for example, be implemented by dimensionality reduction via multi-dimensional scaling (MDS). Once positioned, successive time points are connected by a curve so as to visualize their temporal order. Time curves are a general approach to visualize patterns of evolution in temporal data, such as progression and stagnation, sudden changes, regularity and irregularity, reversals to previous states, temporal states and transitions, reversals to previous states, and others. It is important to mention that the interpretability of a Time Curves visualization heavily depends on the employed similarity metric and inherits all advantages and disadvantages of the implemented dimensionality reduction method.

Relevant references: Bach et al. (2016)

time

primitives: points
arrangement: linear

data

number of variables: single, multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

TimeSlice

time

primitives: points, intervals
arrangement: linear

data

number of variables: single, multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

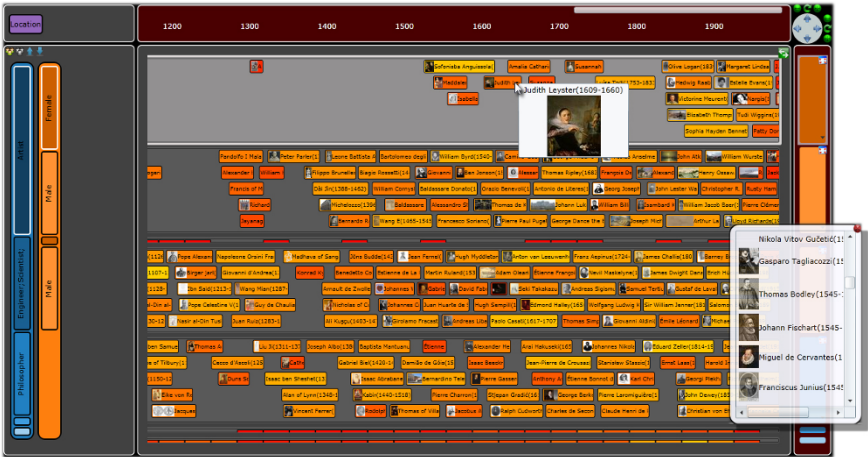


Fig. A.49: TimeSlice interface (left) for faceted browsing of a timeline visualization (center) of the lives of famous people. © Courtesy of Jian Zhao.

TimeSlice is an interactive faceted browsing tool (↔ p. 336) for time-oriented data. It provides a flexible approach for constructing, comparing, and manipulating multiple queries over faceted timelines (↔ p. 258). The main view shows time intervals (the life span of famous people in the figure) as vertical colored bars. Queries are organized in a dynamic filtering tree structure to the left of the interface. It displays the currently focused queries and also their contexts (such as queries that share the same attribute on one facet but that differ on another). Tree nodes (representing attributes) and tree levels (representing facets) can be manipulated directly, which offers efficient navigation across different perspectives of the data.

Relevant references: Zhao et al. (2012)

Silhouette Graph

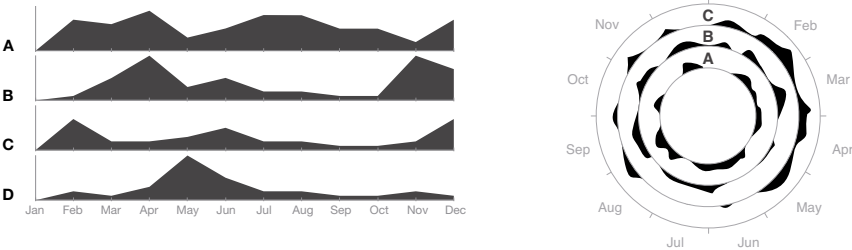


Fig. A.50: Silhouette graphs are filled line plots that can be used for enhancing the comparison of multiple time series that are shown side by side. © ⓘ The authors. Adapted from Harris (1999).

Silhouette graphs emphasize the visual impression of time series by filling the area below the plotted lines (see Harris, 1999). This leads to distinct silhouettes that enhance perception at wide aspect ratios of long time series compared to line plots (↔ p. 233) and allow an easier comparison of multiple time series. In the left figure, time is mapped to the horizontal axes and multiple time series are stacked upon each other. Other layouts of the axes might be used for reflecting different time characteristics. One example is circular silhouette graphs as shown in the right figure, where silhouette graphs are drawn on concentric circles in order to emphasize periodicity in time. The ideas of stacking graphs and showing them along concentric circles are not restricted to silhouette graphs, but can be found in several other techniques, for example, in horizon graphs (↔ p. 277) and in enhanced interactive spirals (↔ p. 274).

Relevant references: Harris (1999)

time

primitives: points
arrangement: linear, cyclic

data

number of variables: single, multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

time

primitives: points
arrangement: linear, cyclic

data

number of variables: single, multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

Recursive Pattern

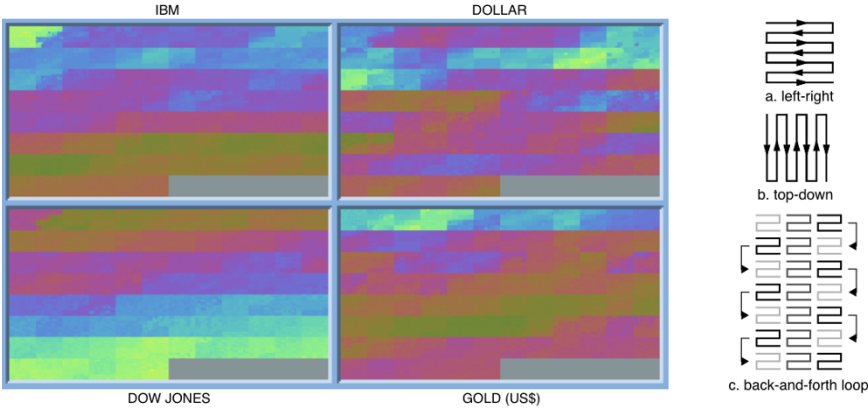


Fig. A.51: Four time series are shown as recursive patterns, where the color of a pixel represents the stock price. Pixels are arranged to semantically match the hierarchical structure of the data: A row represents a year with its 12 months, and each month is further subdivided into weeks, which in turn consist of five workdays, each of which represents nine data values for a single day. Examples of possible pixel arrangements are shown to the right. © 1995 IEEE. Reprinted, with permission, from Keim et al. (1995).

The most space-efficient way of visualizing data is to represent them on a per-pixel basis (see Lévy et al., 2007). Keim et al. (1995) propose a variety of pixel-based visualization approaches of which the recursive pattern technique is particularly suited to display large time series. The key idea behind the recursive pattern technique is to construct an arrangement of pixels that corresponds to the inherently hierarchical structure of time-oriented data given at multiple granularities. The figure shows financial data as a pixel-based visualization. The initial step is to map nine data values collected per day to a 3×3 pixel group. This group is then used to form a larger group for a week of workdays containing 5×1 day groups. Groups for months, years, and decades can be created by recursively arranging groups of the next lower granularity in a semantically meaningful way (e.g., 12 months are grouped into a year). In the resulting pattern, each pixel is color-coded with regard to a single data value in the time series. Multiple dense pixel displays of this kind can be combined to generate overviews of large multivariate datasets.

Relevant references: Keim et al. (1995) • Lévy et al. (2007)

Lin-spiration

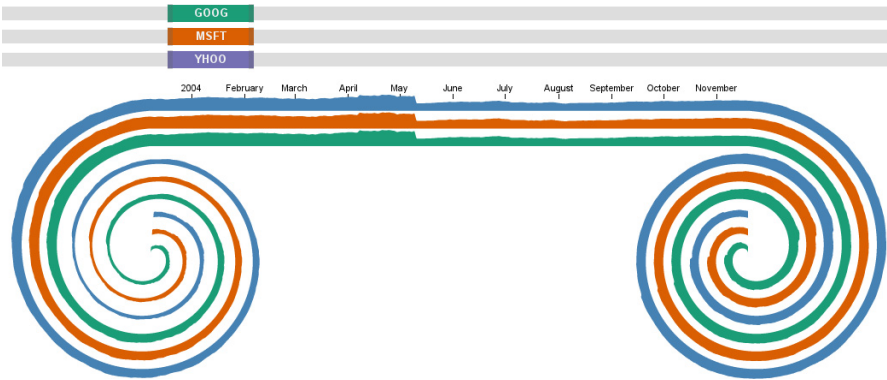


Fig. A.52: Lin-spiration combines linear (center) and cyclic (left and right) representations of multivariate time series. Time sliders (top) allow users to navigate the time axis of each variable independently. © 2012 ACM. Reprinted, with permission, from Graells and Jaimes (2012).

Graells and Jaimes (2012) combine linear silhouette graphs (↔ p. 281) and spiral graphs (↔ p. 285) to a hybrid interactive visualization technique they call *Lin-spiration*. The basic idea is to show a focused subset of the data as multiple stacked linear silhouette graphs at full temporal resolution in the center of the display. To each side of the linear silhouette graphs, spiral graphs show in radial form the data preceding and succeeding the selected subset. The spirals’ purpose is to provide context information about how the data looked like in the past and how it will look like in the future (with respect to the currently selected subset). To accommodate very large time-oriented data, the spirals show the time axis in compressed form. The resulting figure metaphorically resembles film rolls in classic photo cameras as already suggested by Tominski et al. (2003) and also bears resemblance with the Perspective Wall (↔ p. 256). The Lin-spiration approach is combined with linear time sliders (top) that allow users to navigate the time axis of each visualized variable independently and to compare the data not only for the same time period, but for different periods of interest.

Relevant references: Graells and Jaimes (2012) • Tominski et al. (2003)

time

primitives: points
arrangement: linear, cyclic

data

number of variables: single, multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

time

primitives: points
arrangement: cyclic

data

number of variables: single, multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

Spiral Graph

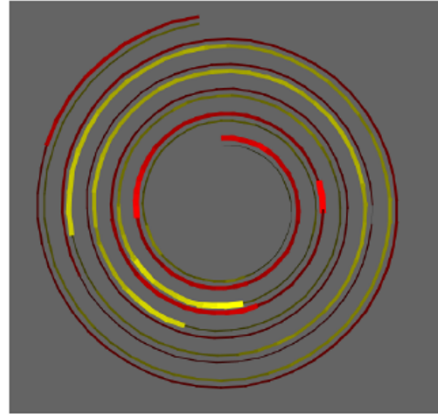
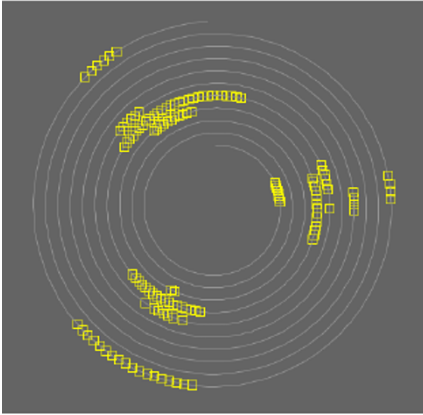


Fig. A.53: A spiral graph encodes time-series data along a spiral, thus emphasizing the cyclic character of the time domain. Actual data values are visualized using symbols for nominal data (left) as well as color and line thickness for quantitative data (right). © 2001 IEEE. Reprinted, with permission, from Weber et al. (2001).

The spiral graph developed by Weber et al. (2001) is a visualization technique that focuses on cyclic characteristics of time-oriented data. To this end, the time axis is represented by a spiral. Time-oriented data are then mapped along the spiral path. While nominal data are represented by simple icons, quantitative data can be visualized by color, line thickness, or texture. One can also visualize multivariate time series by intertwining several spirals as shown for the example of two variables in the spiral on the right. In this case, a distinct hue is used per spiral so that individual variables can be discerned. Weber et al. (2001) further envision extending the spiral to a three-dimensional helix, in order to cope with larger time series. The main purpose of the spiral graph is the detection of previously unknown periodic behavior of the data. The user can interactively adjust the spiral's cycle length (i.e., the number of data values mapped per spiral cycle) to explore the data for cyclic patterns. As an alternative, it is also possible to smoothly animate through possible cycle lengths. In this case, the periodic behavior of the data becomes immediately apparent by the emergence of a pattern. When such a pattern is spotted, the user stops the animation and an interesting cycle length has been found.

Relevant references: Weber et al. (2001)

Spiral Display

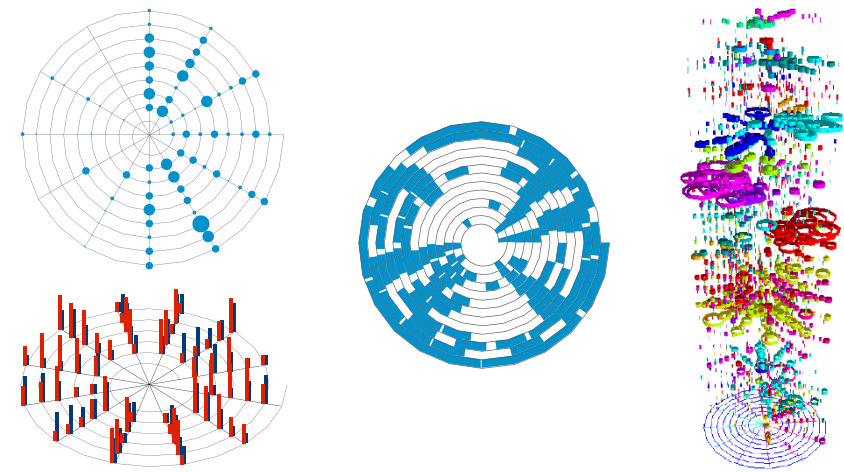


Fig. A.54: Data can be visualized along a spiral in different ways: by the area of circular elements (top left), by the sizes of multiple spikes (bottom left), as bars marking start and end of intervals (center), or by the volume of hollow cylinders aligned at different layers along the vertical axis (right). © 1998 ACM. Reprinted, with permission, from Carlis and Konstan (1998).

The interactive spiral display by Carlis and Konstan (1998) uses Archimedean spirals to represent the time domain. Data values at particular time points are visualized as filled circular elements whose area is proportional to the data value (top left). In the case of interval-based data, filled bars are aligned with the spiral shape to indicate the start and end of intervals (center). If multivariate data are given at time points, the spiral is tilted and data values are visualized as differently colored spikes, where spike color indicates variable affiliation and spike height encodes the corresponding data value (bottom left). Alternatively, one can use the vertical z-axis to separate the display of multiple variables (right). In this case, each time-dependent variable has its own layer along the z-axis and is represented with a unique color. Within a layer, data values are encoded to the volume of cylinders, which are hollow to prevent occlusion. The system implemented by Carlis and Konstan (1998) allows users to display multiple linked spirals to perform comparison tasks. The cycle lengths of spirals can be adjusted interactively and can also be animated automatically for discovering periodic patterns.

Relevant references: Carlis and Konstan (1998)

time

primitives: points, intervals
arrangement: cyclic

data

number of variables: single, multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D, 3D

time

Stacked Graphs

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

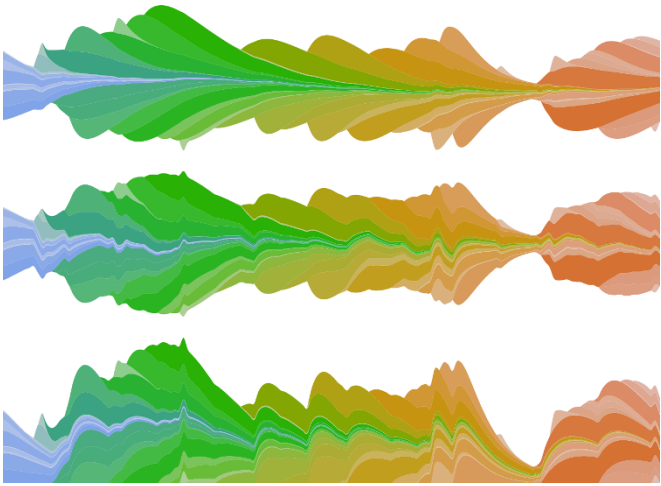


Fig. A.55: Multivariate time series visualized as stacked graphs with different designs: stream graph design (top), ThemeRiver layout (center), and traditional stacking (bottom). © The authors. Generated with the `streamgraph_generator` code base by Lee Byron.

Stacking multiple graphs on top of each other is a suitable approach to visualizing multiple time-dependent variables (see Harris (1999)). Elaborate variants of stacked graphs have been investigated in detail by Byron and Wattenberg (2008). To visualize the evolution of an individual variable, data values are encoded to the height of a so-called layer that extends along the horizontal time axis. A special color map is applied to visualize additional data variables and to make individual layers distinguishable. Several layers are then stacked on top of each other, effectively creating an overall graph that represents the visual sum of the entire dataset. Layout and sorting of layers can be done in various ways, resulting in quite different designs such as the so-called stream graph design, the ThemeRiver layout (↔ p. 293), or traditional layer area graphs (↔ p. 289), illustrated from top to bottom in the figure. The stream graph design (top) is notable because it received quite positive feedback when it appeared on the New York Times website as a visual representation of box office revenues. In that version, individual layers were also outfitted with text labels.

Relevant references: Harris (1999) • Byron and Wattenberg (2008)

TimeSearcher 3, River Plot

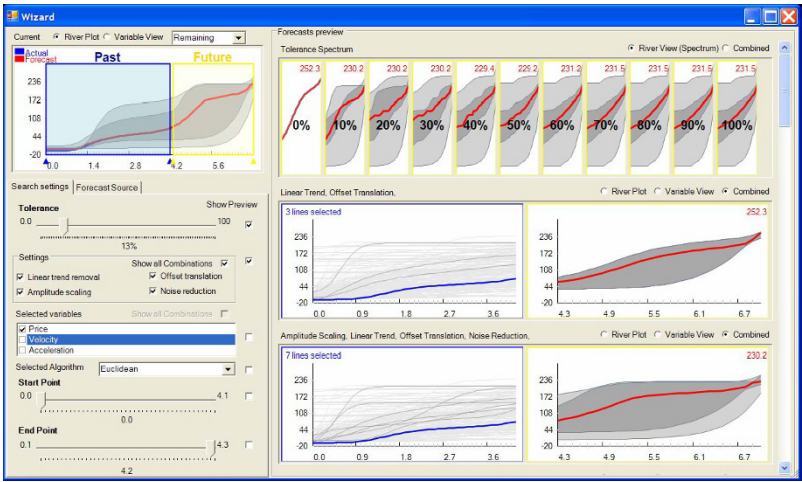


Fig. A.56: Forecasting of online auction data. Top left: selection of time interval for similarity search; bottom left: selection of variables to consider and parameters to vary in the previews; right: preview area that assists users in understanding the impact of parameters – varying tolerance levels as river plots and different combinations of applied transformations as line plots and river plots. © *Courtesy of Paolo Buono.*

TimeSearcher 3 is a tool to support similarity-based forecasting of multivariate time series. Similarity-based forecasting is a data-driven method using the similarity to a set of historical data for predicting future behavior. The outcome of the algorithm is affected by a number of options and parameters, for instance, the transformations applied or the tolerance threshold used for matching. As a result, the median of the matched subsets becomes the forecast and descriptive statistics measures reflect the uncertainty associated with the forecast. This is displayed graphically as a simplified, continuous box plot, called a river plot. It uses superimposed, colored regions, for which light gray indicates the range between the minimum and maximum and dark gray the range between the 25% and 75% percentiles, and a line in the center, where red indicates the forecast, brown shows the median during the matching period, and black is the median before this period. TimeSearcher 3 builds upon TimeSearcher (↔ p. 290) and adds a preview interface to allow users to interactively explore the effects of adjusting algorithm parameters and to see multiple forecasts simultaneously.

Relevant references: Buono et al. (2007)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

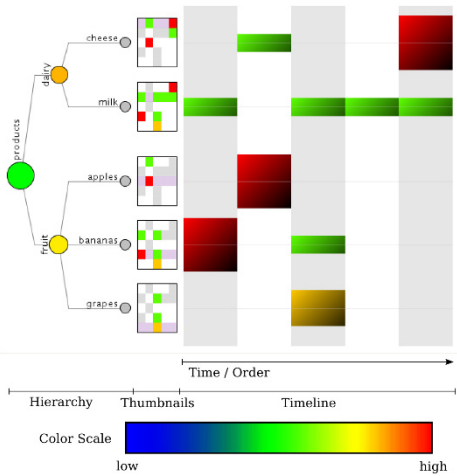
mapping of time: static
dimensionality: 2D

time

primitives: points
arrangement: linear

Timeline Trees

Day **Market basket and money spent**
Mon: milk \$1, bananas \$3
Tue: cheese \$1, apples \$3
Wed: milk \$1, bananas \$1, grapes \$2
Thu: milk \$1
Fri: milk \$1, cheese \$3



data

number of variables: multiple
frame of reference: abstract

Fig. A.57: A smaller set of products in a market basket is visualized using timeline trees. One can see that milk is bought regularly (green boxes for all but one day), and that cheese, apples, and bananas are more expensive (higher red-colored boxes). © Courtesy of Michael Burch.

vis

mapping of time: static
dimensionality: 2D

Data that describe items being related to each other are quite common. An example of such data is transactions in online shopping systems where products being bought together are considered to be related. Burch et al. (2008) visualize temporal sequences of transactions by means of the so-called timeline trees. The visual representation consists of three parts: a display of an information hierarchy, a timeline representation of temporal sequences, and thumbnail pictures. The information hierarchy is a static hierarchical categorization of data items (e.g., a system of product groups), where groups can be expanded or collapsed interactively to view the data at different levels of detail. The timeline view shows multiple sequences of boxes for the current level of detail, where color and box size are used to encode data values (e.g., product price) of an item (or group) at a particular point in time. Thumbnails for each leaf of the information hierarchy show an overview of transactions masked by the corresponding leaf node. Enhanced with several interaction facilities, timeline trees help users to understand trends in the data and to find relations between different levels of abstractions (e.g., product groups and specific products).

Relevant references: Burch et al. (2008)

Layer Area Graph

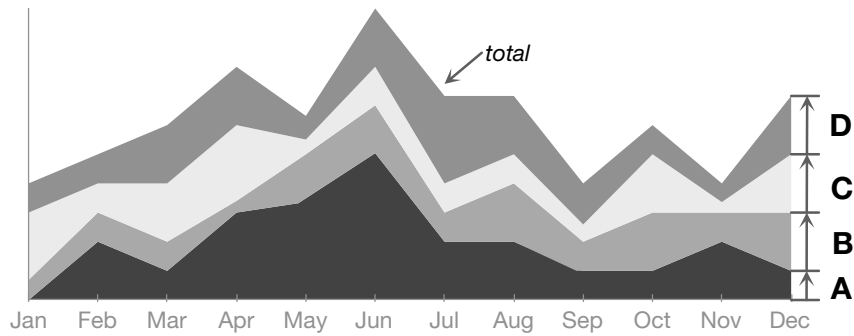


Fig. A.58: Layered area graphs show individual variables as bands stacked on top of each other emphasizing the total sum of all variables. © The authors. Adapted from Harris (1999).

Layer area graphs might be used when comparing time series that share the same unit and can be summed up (see Harris, 1999). A layer area graph is a stacked visualization where time-series plots are drawn upon each other as layered bands. Caution needs to be exercised for this kind of representation because it is sensitive to the order of the layers (see Mathiesen and Schulz, 2021). Different orders influence the visual appearance of the individual layers because only the bottommost layer has a straight baseline. All subsequent layers are drawn relative to the layers below. An advantage of layer area graphs is the fact that they emphasize the total sum of values while providing information about the parts that constitute it. More advanced visualization techniques such as the ThemeRiver (↔ p. 293) or stacked graphs (↔ p. 286) build upon the basic principle of layer area graphs.

Relevant references: Harris (1999) • Mathiesen and Schulz (2021)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

TimeSearcher

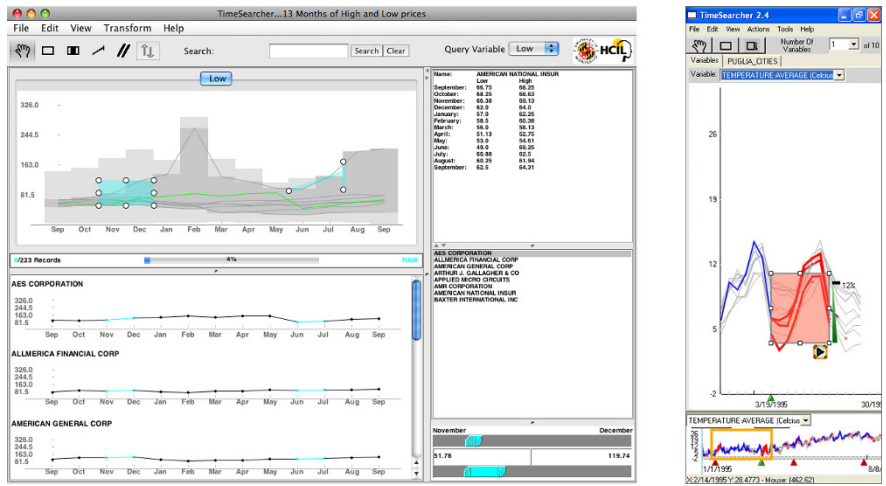


Fig. A.59: Left: TimeSearcher showing stock price data (top left: multiple line plots and query boxes; bottom left: line plots for individual stocks matching the query). Right: TimeSearcher 2 with query-by-example using a searchbox (light red background) and matching items in red. © The authors. Generated with the TimeSearcher software from the University of Maryland Human-Computer Interaction Lab.

Hochheiser and Shneiderman (2004) implemented TimeSearcher as a visual exploration tool for multiple time series. While employing a straightforward visual representation using line plots (↔ p. 233), its main objective is to enable users to identify and find patterns in the investigated data. To this end, the so-called *time-box query model* has been developed. It allows the specification of a rectangular query region that defines both a time interval and a value range of interest. Those time series that comply with a query (i.e., overlap with the timebox) are displayed, whereas all others are filtered out. Users can combine multiple timeboxes to refine the query further and other query functionalities such as leaders and ladders, angular queries, and variable timeboxes are also part of TimeSearcher. To provide contextual information, the data envelope and the query envelope can be displayed. Buono et al. (2005) extended these features in TimeSearcher 2 by allowing the representation of heterogeneous datasets and providing a *searchbox query model* that effectively implements a query-by-example functionality. Here, occurrences of a brushed portion of the time series are searched, whereas the similarity threshold of matches can be adjusted.

Relevant references: Hochheiser and Shneiderman (2004) • Buono et al. (2005)

BinX

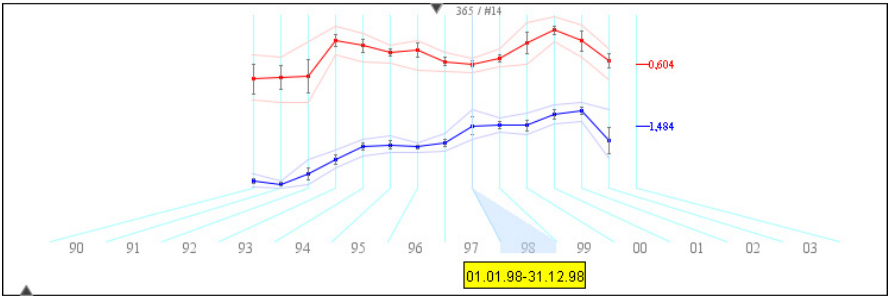


Fig. A.60: Exchange rates for two currencies are compared using the BinX tool. Each bin aggregates the daily rates for a whole year. A selected bin is highlighted and its position on the global time scale is marked accordingly. © The authors. Generated with the BinX tool by Lior Berry.

Large time series require the application of abstraction methods in order to reduce the number of time points to be displayed, thus keeping visualization costs at a manageable level. Finding a suitable degree of abstraction, however, is not an easy task. The BinX tool developed by Berry and Munzner (2004) is interesting in that it supports the exploration of different aggregations of a time series. The aggregation is based on constructing bins, each of which holds a user-defined number of time points. Easy-to-use interaction is offered to quickly try out different bin sizes. BinX visualizes one or two quantitative time-dependent variables using common chart elements. An overview of the time axis is preserved at all times at the bottom of the BinX representation. The central chart view displays the two time series in an aggregated fashion according to the currently chosen bin size. In order to faithfully represent aggregated information, line plots (↔ p. 233), box plots, and a min-max band are used in combination. The correspondence between a point in the chart and a time span (bin) on the time axis is represented upon user request. BinX supports the clustering of bins as an additional mechanism for analytic abstraction. In this case, the cluster affiliation of bins is encoded via color.

Relevant references: Berry and Munzner (2004)

time
primitives: points
arrangement: linear

data
number of variables: multiple
frame of reference: abstract

vis
mapping of time: static
dimensionality: 2D

MultiComb

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

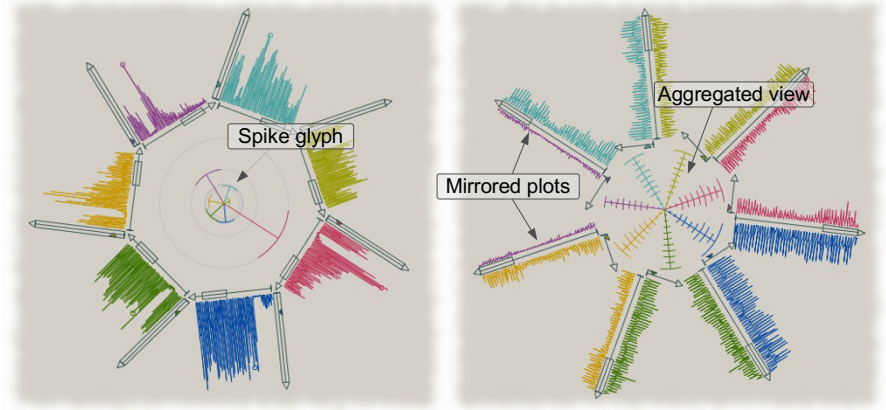


Fig. A.61: Two MultiComb representations visualize seven time-dependent variables. In the left MultiComb, line plots are arranged around the display center, in the right one, they extend outwards. The interior of the MultiCombs can be used to display additional information via a spike glyph or an aggregated view. © The authors. Generated with the [VisAxes](#) software by Christian Tominski.

Line plots (↪ p. 233) are expressive visual representations for univariate data. The rationale behind the MultiComb visualization is to utilize this expressiveness for representing multiple time-dependent variables. Tominski et al. (2004) describe the MultiComb as a visual representation that consists of multiple radially arranged line plots. Two alternative designs exist: time axes are arranged around the display center (left figure) or time axes extend outwards from the MultiComb’s center (right figure). In the latter case, optional mirror plots duplicate the plots of neighbor variables to ease visual comparison. To maintain a certain aspect ratio for the separate plots, the axes do not start in the very center of the MultiComb. The screen space in the interior can thus be used to provide additional views: a spike glyph can be shown to allow a detailed comparison of data values for a selected time point, or an aggregated view might display the history of a temporal data stream in an aggregated fashion. Various possibilities for interaction allow users to browse in time, to zoom into details of the time axes, as well as to add, remove, and reorder plots, and to rotate the MultiComb.

Relevant references: Tominski et al. (2004)

ThemeRiver

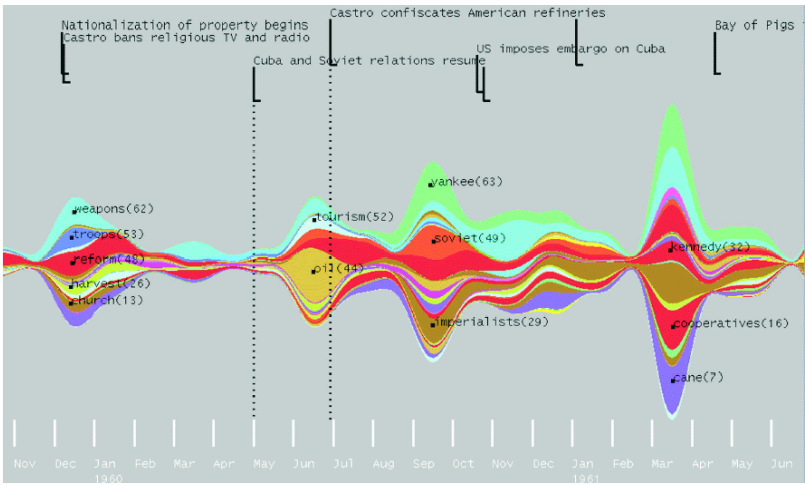


Fig. A.62: The ThemeRiver representation uses the metaphor of a river that flows through time. Colored currents within the river reflect thematic changes in a document collection, where the width of a current represents the relevance of its associated theme. © 2002 IEEE. Reprinted, with permission, from Havre et al. (2002).

The ThemeRiver technique developed by Havre et al. (2000) represents changes in news topics in the media. Each topic is displayed as a colored current whose width varies continuously as it flows through time. The overall image resembles a river composed of multiple currents. The ThemeRiver provides an overview of the topics that were important at certain points in time. Hence, the main focus is directed toward establishing a picture of an easy to follow evolution over time using interpolation and approximation. Moreover, ThemeRiver representations can be annotated, e.g., with related major historical events, and raw data points with exact values can be shown. Even though the ThemeRiver was originally invented to visualize thematic changes in document collections, it is also suited to represent other multivariate, quantitative data. Because perception of data differs depending on where in the river individual variables are shown, it is important to provide interaction techniques to allow users to rearrange the horizontal position of variables. Generalizations of the ThemeRiver technique became known as stacked graphs or stream graphs (↪ p. 286). The ThemeRiver can also be used in combination with other views. Hashimoto and Matsushita (2012) combined the ThemeRiver technique with a heat map view, which illustrates the quantities at a given time point. Jiang et al. (2016) combines the ThemeRiver technique with spirals and scatter plots, to a so-called spiral theme plot.

Relevant references: Havre et al. (2000) • Havre et al. (2002) • Hashimoto and Matsushita (2012) • Jiang et al. (2016)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

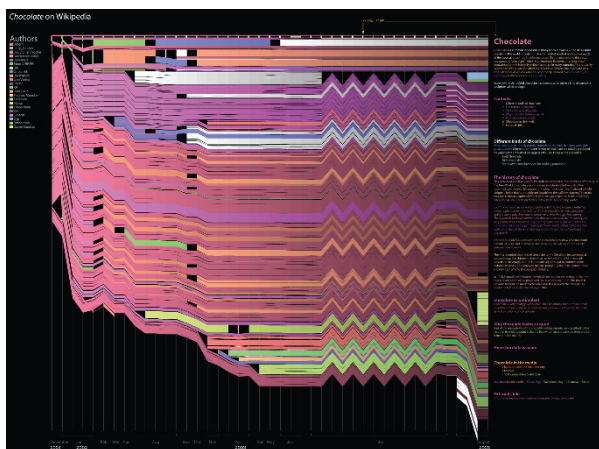
mapping of time: static
dimensionality: 2D

history flow

version 1 version 2 version 3 version 4



Fig. A.63: history flow shows vertical revision lines, one for each revision, where colored sections reflect the different authors of a document as schematically depicted on the left. This method is applied to the visualization of the Wikipedia entry on chocolate as shown on the right. © *Courtesy of Fernanda B. Viégas.*



Viégas et al. (2004b) designed history flow as an exploratory wiki article analysis tool for finding author collaboration patterns, showing relations between document versions, revealing patterns of cooperation and conflict, as well as making broad trends immediately visible. The basis for the representation is the so-called revision lines. These top-aligned, vertical lines are displayed for every version of a document. The length of revision lines is proportional to the document length. Individual sections of a revision line are colored differently to visualize which authors worked on which parts of a document. The sections associated with a particular author are visually connected from one revision to the next. One can discern stable sections and splits of sections. Gaps in connections clearly indicate deletions and insertions. Two different layouts can be used for spacing revision lines: uniform spacing (space by occurrence/event-based) or spacing according to time (space by date/time-based). The first layout shows each document change equally spaced without showing time intervals between versions proportionally. The second alternative additionally gives information about the exact timing.

Relevant references: Viégas et al. (2004b)

mapping of time: **static**
dimensionality: **2D**

LifeLines2

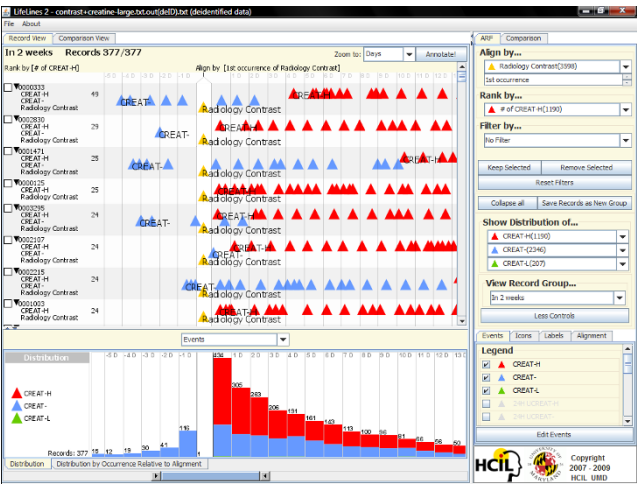


Fig. A.64: LifeLines2 show patient records in stacked rows, where triangles indicate health-related events. A histogram view visualizes the number of events over time. Interaction operators (i.e., align, rank, filter) support visual exploration. © Courtesy of Taowei David Wang.

LifeLines2 is an interactive visual exploration interface for instantaneous events based on categorical, health-related data (e.g., high, normal, or low body temperature). Events are displayed as triangles along a horizontal time axis, where color indicates event categories and data of different patient records are stacked vertically. An aggregation of events is represented as a histogram showing the number of occurrences over time. LifeLines2 introduces three powerful operators for interactive exploration: align, rank, and filter. The align operator can be used to arrange all records along a specific event type in temporal order, for example, to align a group of patients with regard to their first heart attack. Additionally, the time axis switches from an absolute time representation to relative time originating from the specified event (e.g., one week before, or two weeks after the first heart attack). The rank operator is useful for ordering records according to the number of occurrences of a specified event type. The filter operator allows users to search for particular sequences of events including both the presence of events and the absence of events (e.g., patients having had a heart attack but no stroke following it).

Relevant references: Wang et al. (2009)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

Similan

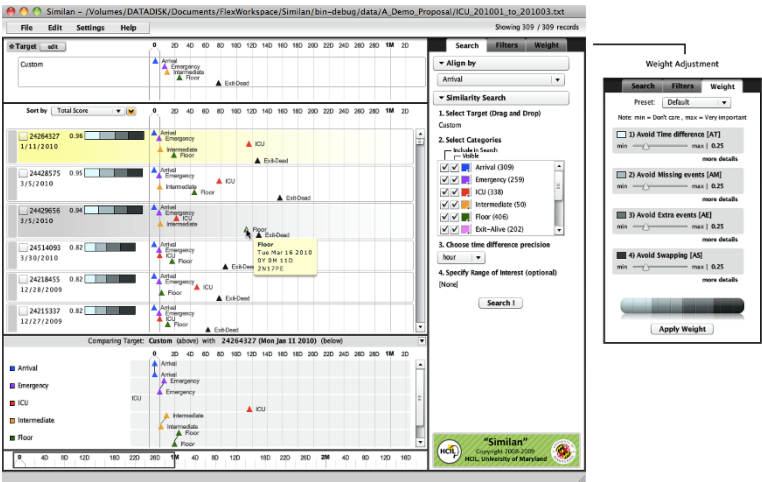


Fig. A.65: Similan ranks patient records (center) according to their similarity to a target record (top). Individual records can be compared directly with the target (bottom). Various interaction operators, including adjustment of similarity weights (right), can be used to refine the visualization. © Courtesy of Krist Wongsuphasawat.

Wongsuphasawat and Shneiderman (2009) describe Similan as a system for exploring patient records. Patient records are stacked upon each other and show health-related events as triangles, where color indicates event categories (e.g., arrival, emergency, ICU). Similan uses the same visual representation as LifeLines2 (↔ p. 295) but provides a different approach to data exploration. Instead of interactive filtering, records are ranked according to their similarity to a given event sequence (query-by-example). In the figure, the topmost record is the one that is most similar to the user-specified event sequence. This can be used to search for groups of patients who share similar temporal patterns. A dedicated view is provided to allow a direct comparison of the target query record with any particular record in the data. Another scenario is to search for an event sequence that the user is not certain whether it exists in the data. In this way, the tool can give the most similar results if the exact event sequence does not exist. For determining the similarity of event sequences a similarity measure (M&M measure) has been developed. The weights of factors that determine the similarity measure can be adjusted interactively by the user.

Relevant references: Wongsuphasawat and Shneiderman (2009)

time
primitives: points
arrangement: linear

data
number of variables: multiple
frame of reference: abstract

vis
mapping of time: static
dimensionality: 2D

VIE-VISU

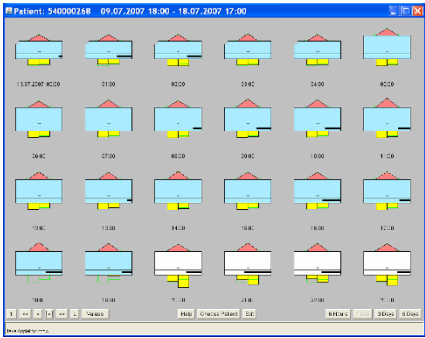
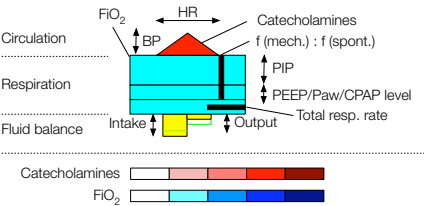


Fig. A.66: VIE-VISU encodes fifteen health-related patient parameters to different visual attributes of a glyph (left). The example on the right shows neonatal patient information on an hourly basis for the course of the day. Left: © The authors. Adapted from Horn et al. (2001). Right: © Courtesy of Werner Horn.

Paper-based analysis of patient records is hard to conduct because many parameters are involved and an overall assessment of the patient’s situation is difficult. Therefore, Horn et al. (2001) developed VIE-VISU, an interactive glyph-based visualization technique for time-oriented patient records. The glyph consists of three parts that represent circulation, respiration, and fluid balance parameters. All in all, 15 parameters are visualized using different visual attributes (i.e., length, width, color) as illustrated in the left part of the figure. For example, the circulation parameter heart rate (HR) is encoded to the width of the triangle on top of the glyph and the triangle’s color encodes catecholamines (color legend is given at the bottom). Each glyph represents a one hour period and 24 glyphs are combined in a small multiples display (↔ p. 359) as shown in the right part of the figure. Interaction controls support navigation in time and switching to different periods for the small multiples view. VIE-VISU helps users to combine different measurements, maintain their relationships, show their development over time, and make specific, possibly life-threatening situations easy to spot.

Relevant references: Horn et al. (2001)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

time

primitives: points
arrangement: linear

TimeWheel

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

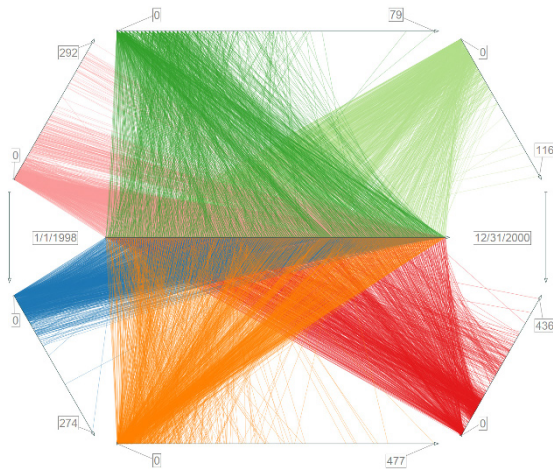


Fig. A.67: The TimeWheel’s central axis represents time. The axes in the periphery represent time-dependent variables. Here we see the number of cases for eight diagnoses. © The authors. Generated with the VisAxes software by Christian Tominski.

Tominski et al. (2004) describe the TimeWheel as a technique for visualizing multiple time-dependent variables. Similar to the Parallel Coordinates technique for general multivariate data, the TimeWheel consists of several axes that are connected via lines. There is one time axis and multiple data axes for the data variables. The time axis is placed in the center of the display to emphasize the temporal character of the data. The data axes are associated with individual colors and are arranged circularly around the time axis. In order to visualize data, lines emanate from the time axis to each of the data axes to establish a visual connection between points in time and associated data values. These lines form visual patterns that allow users to identify positive or negative correlations with the time axis, trends, and outliers. Such patterns can be best discerned for those data axes that are parallel to the time axis. To bring data axes of interest into this focus, users can rotate the TimeWheel. Focused data axes are further emphasized by stretching them, effectively providing them with more drawing space. Data axes that are perpendicular to the time axis are more difficult to interpret and are, therefore, attenuated using color fading and shrinking. Interactive exploration, including navigation in time, is supported through different types of interactive axes. The idea of using linked interactive axes has been generalized by Claessen and van Wijk (2011). They propose a general model for linked axes that can be used to define a variety of axes-based visualization techniques of which the TimeWheel is only one example.

Relevant references: Tominski et al. (2004) • Claessen and van Wijk (2011)

LiveRAC

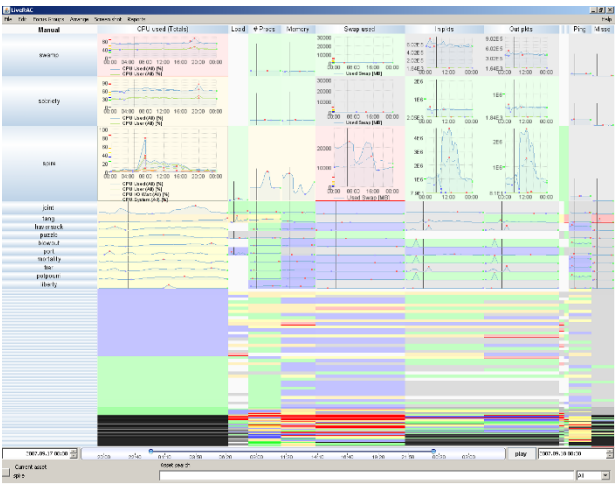


Fig. A.68: A full day of system management time-series data showing more than 4000 devices in rows and 11 columns representing groups of monitored parameters. Representations within cells adapt to the available screen space by using different representations with more or less detail. © 2008 ACM. Reprinted, with permission, from McLachlan et al. (2008).

LiveRAC is a system for analyzing system management time-series data. It scales to dozens of parameters collected from thousands of network devices. Familiar representations such as line plots (↔ p. 233), bar graphs (↔ p. 234), and sparklines (↔ p. 235) appear as the cells of a spreadsheet-like matrix. Rows and columns of the matrix are associated with monitored network devices and monitored parameters, respectively. Each cell contains an area-aware chart showing time on the horizontal axis and parameters on the vertical axis. To ensure that all cells remain visible at all times (i.e., to avoid scrolling), LiveRAC uses a so-called *stretch and squish* layout, which dynamically compresses and expands cells according to user interaction. Moreover, the individual charts adapt to the available screen space. This semantic zoom functionality ranges from charts with detailed labels, to smaller charts with fewer curves and less labeling, and ultimately to colored blocks for the smallest view. The cell background color represents changeable thresholds of minimum, maximum, or average values of the displayed parameters. Aggregation is applied if cells would overlap due to space restrictions, which is reflected in color intensity.

Relevant references: McLachlan et al. (2008)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

CareCruiser

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

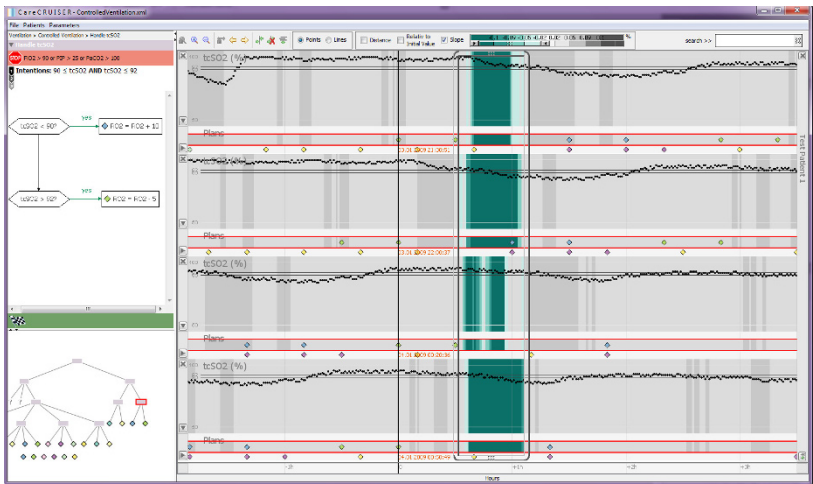


Fig. A.69: A patient’s parameters are displayed together with the applied clinical actions. In the selected area on the right, a delayed drop of the patient’s $tcSO_2$ values after applying a specific clinical action is revealed. Contextual views are shown on the left – top left: flow-chart like representation of the treatment plan logic; bottom left: hierarchical decomposition of the treatment plan. © The authors. Generated with the CareCruiser software.

CareCruiser by Gschwandtner et al. (2011) is a visualization system for exploring the effects of clinical actions on a patient’s condition. It supports exploration via aligning, color highlighting, filtering, and providing focus and context information. Aligning clinical treatment plans vertically supports the comparison of the effects of different treatments or the comparison of different effects of one treatment plan applied to different patients. Three different color schemes are provided to highlight interesting portions of the development of a parameter: highlighting the distance of the actual values to the intended value helps to identify critical values; highlighting the progress of the actual values relative to the initial values shows to what extent the applied treatment plan has the intended effect; and highlighting the slope of a value helps to explore the immediate effects of applied clinical actions. A range slider is provided (at the top of the interface) to filter the color highlighting for selected events, and a focus window that grays out the color information outside its borders is used to support a focused investigation of a region of specific interest.

Relevant references: Gschwandtner et al. (2011)

Braided Graph

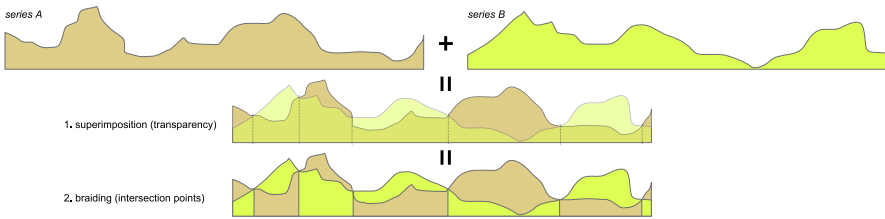


Fig. A.70: Construction scheme of a braided graph for two variables. Top: individual variables as silhouette graphs; center: superimposed silhouette graphs using transparency (dashed lines show intersection points); bottom: braided graph (segments between intersection points are sorted to ensure visibility of all fragments). © 2010 IEEE. Reprinted, with permission, from Javed et al. (2010).

Braided graphs are a technique for superimposing silhouette graphs to show multivariate data. They take advantage of the expressiveness of silhouette graphs (↔ p. 281) and at the same time avoid the disadvantage of varying baselines of layered graphs (↔ p. 289). Simply drawing silhouette graphs on top of each other would lead to occlusion problems where a silhouette for larger data values occludes silhouettes for smaller values. The solution to this problem is to identify the points at which silhouettes intersect and to adapt the drawing order in between two intersections individually so that smaller silhouettes are always in front of larger ones. This ensures that all segments of all variables remain visible for the complete time series. In a user study, Javed et al. (2010) compared line plots (↔ p. 233), silhouette graphs (↔ p. 281), horizon graphs (↔ p. 277), and braided graphs along the three tasks of determining local maxima, comparing global slopes, as well as locating and comparing values at specific time points. Besides the visualization type, the number of displayed time series and the height of the representation were varied. Interestingly, the type of visualization was not found to have a significant effect on task correctness in all conditions. However, subjects using line plots and braided graphs were significantly faster when searching for local maxima. For value and slope comparison tasks this was not the case. In general, higher numbers of time series caused decreased correctness and increased completion time. Decreasing display space had a negative effect on correctness but little impact on completion time.

Relevant references: Javed et al. (2010)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

CiteSpace II

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

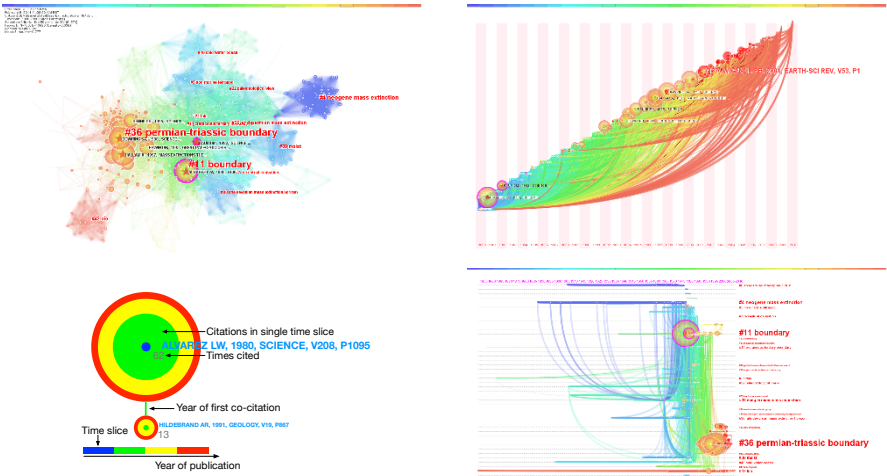


Fig. A.71: Visualization of a network of 750 most cited articles on mass extinction (1980–2010). Top left: cluster view; bottom left: legend – node size reflects overall amount of citations and colored rings show citations per time slice; top right: timezone view; bottom right: timeline view. © Courtesy of Chaomei Chen.

CiteSpace II by Chen (2006) is a system that supports the visual exploration of bibliographic databases. It combines rich analytic capabilities to analyze emerging trends in a knowledge domain with interactive visualization of co-citation networks. Three complementary views are provided for the visual representation: a cluster view, a time-zone view, and a timeline view. The cluster view represents a network as a node-link diagram using a force-directed layout. Node size shows how often an article or cluster was cited overall and citation tree rings of a node display the citation history from the center outward. The color of a ring represents a time period and its thickness is proportional to the number of citations in this period. The colors of links represent the time slice of the first co-citation. The time-zone view displays a network by arranging its nodes along vertical strips representing time zones using a modified spring-embedder layout that controls only the vertical positions of nodes freely. In the timeline view, time is mapped to the horizontal position and clusters are arranged along horizontal lines. Users can adjust a complex set of parameters to control the analysis process as well as interact and manipulate the visualization of a knowledge domain. CiteSpace II also provides clustering and labeling functions to help the user interpret various structural and temporal patterns.

Relevant references: Chen (2006)

ChronoLenses

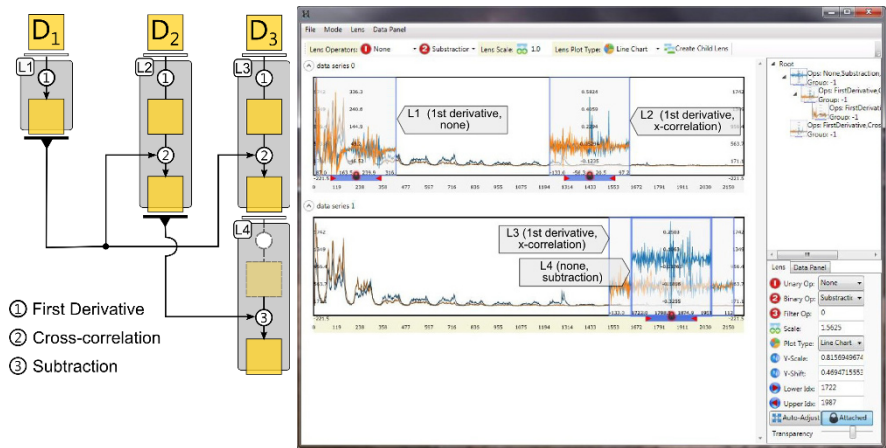


Fig. A.72: ChronoLenses define pipelines of transformations (left) to be applied to selected intervals of time-series data (right) in order to support various exploratory visual analysis tasks. © *Courtesy of Jian Zhao.*

ChronoLenses by Zhao et al. (2011b) is domain-independent time-series visualization technique supporting users in exploratory visual analysis tasks, such as visualizing derived values, identifying correlations, or discovering anomalies beyond obvious outliers. Such tasks often require carrying out elaborate transformations on the original time series. Tightly integrating visual analysis with interaction based on direct manipulation, ChronoLenses perform such transformations on-the-fly on an existing line plot (↔ p. 233) of the data. Users can build pipelines (shown left in the figure) composed of lenses performing various transformation functions, effectively creating flexible and reusable time-series visual analysis interfaces. At any moment, users can change the parameters of already created lenses, with the modifications instantaneously propagating down through the pipeline, providing immediate visual feedback that supports the iterative exploration process. Applied to the visualization, ChronoLenses locally enhance the original time series with the results of the transformation (intervals marked in blue in the main interface) allowing users to gain a deeper insight into the data.

Relevant references: Zhao et al. (2011b)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

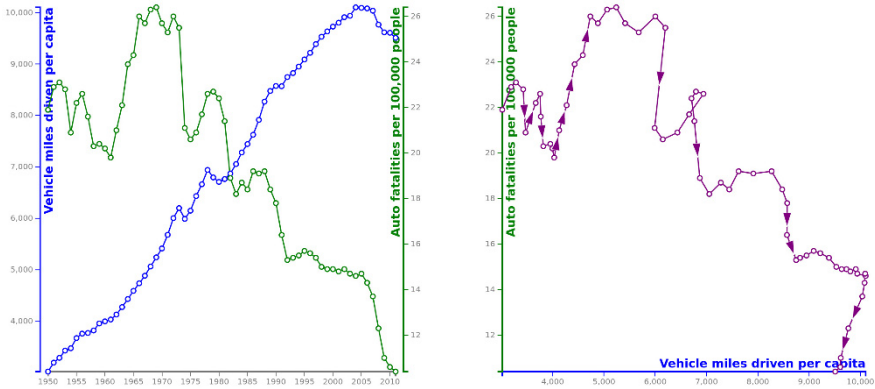
vis

mapping of time: static
dimensionality: 2D

time

Connected Scatterplot

primitives: points
arrangement: linear



data

number of variables: multiple
frame of reference: abstract

Fig. A.73: Comparison of a regular line plot (left) with a connected scatterplot (right). Both plots show the same data, but lead to different visual representations, which has an impact on how the data are interpreted. © The authors. Generated with the *connected scatterplot* software by Steve Haroz.

vis

mapping of time: static
dimensionality: 2D

Connected scatterplots, also known as scatter line graphs (see Harris, 1999), are a classic visualization technique for time-oriented data and have been used widely in news media for the purpose of storytelling. Connected scatterplots are a variant of standard plot techniques such as point plot (\hookrightarrow p. 232) and line plot (\hookrightarrow p. 233). Traditional time-series plots show time on the horizontal axis and time-dependent variables on the vertical axes (see left-hand plot). In contrast, a connected scatterplot maps time-dependent variables to the vertical axis and the horizontal axis, which corresponds to classic scatterplots (which do not include time per se). For connected scatterplots, the dots in the plot are connected via lines or arrows based on the temporal order of the data points. That is, one can see the order of data points (qualitative information), but one cannot tell the temporal distance between them (quantitative information). The figure compares the line plot (left) and connected scatterplot (right) of two variables related to driving safety. Haroz et al. (2016) conducted several studies to test how well connected scatterplots can be interpreted. They found that plots of data with low complexity can be interpreted quite easily. Yet they also found issues with misinterpretations of connected scatterplots and suggest guidelines to circumvent or mitigate them. One such guideline is to clearly communicate the order of time, for example, by including further visual cues in addition to arrows.

Relevant references: Harris (1999) • Haroz et al. (2016)

DimpVis

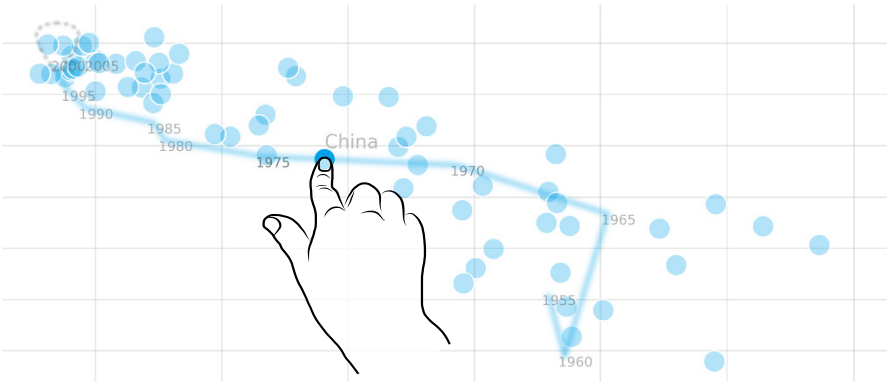


Fig. A.74: The DimpVis technique makes it possible to explore the temporal dimension even when time is not present in the original visual representation. This is achieved by dragging data items along hint paths that indicate the data item’s position at different points in time. © The authors. Generated with the *DimpVis* software by Kondo and Collins (2014).

DimpVis by Kondo and Collins (2014) is special in that it is not really a visualization technique for time-oriented data, but an interaction technique that makes the time dimension explorable even when time is not present in the visual representation. To this end, data items are visualized as graphical objects that can be interacted with via direct manipulation. Once a data item is picked, visual hint paths are displayed to indicate where the data item would be located at different points in time. The hint path can be a timeline connecting time points in sequence or a flashlight providing links to spatially adjacent time points. The figure shows how a timeline path can look like for a scatterplot visualization. Performing a drag gesture along the hint path allows the user to explore the dimension of time while the entire visualization (not just the dragged data item) is updated automatically. In this sense, the DimpVis approach is similar to animated scatter plots (↔ p. 330) where, however, the animation is replaced by the interactive dragging. DimpVis is applicable not only to scatterplots but to a variety of visualization techniques. Kondo and Collins (2014) demonstrate DimpVis for bar charts, pie charts, and heatmap visualizations. Their user study found that DimpVis is a suitable solution for a variety of visualization tasks and may even be preferable over standard solutions such as time sliders or small multiples (↔ p. 359).

Relevant references: Kondo and Collins (2014)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

time

FluxFlow

primitives: points
arrangement: linear

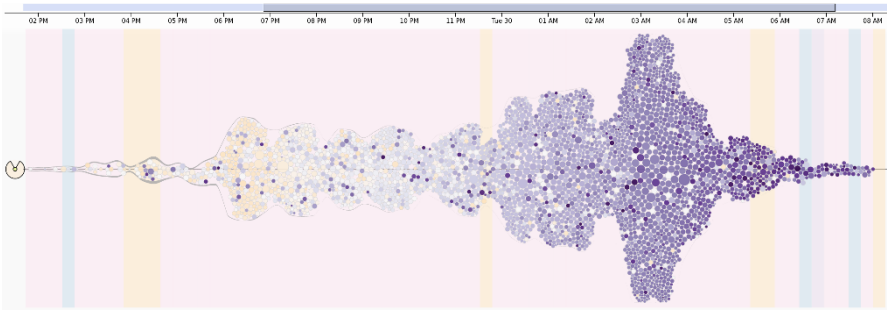


Fig. A.75: FluxFlow’s thread timeline visualization (called volume circle view) represents the top 100 ranked anomalous retweeting threads during the Hurricane Sandy in 2012. The background colors illustrate the eight hidden state variables generated by the model. The circles indicate the participating Twitter users in the thread using a circular glyph to visually summarize important aspects of a retweeting thread. © The authors. Generated with the FluxFlow software by Jian Zhao.

data

number of variables: multiple
frame of reference: abstract

FluxFlow by Zhao et al. (2014) is an interactive visual analysis system for revealing and analyzing anomalous information spreading in social media. The challenge is to distinguish anomalous information behavior, such as the spread of rumors or misinformation, from normal behavior, such as popular topics and newsworthy events. FluxFlow incorporates machine learning algorithms to detect such behavior and novel visualization techniques for in-depth analysis. Consequently, FluxFlow provides (1) a thread glyph, which visually aggregates important aspects of a retweeting thread, (2) a volume chart, (3) a linear circle view, and (4) a volume circle view. Various interactively coordinated user interface components ease the exploration process. Through quantitative evaluation of the model and qualitative interviews with three domain experts, study results indicated that FluxFlow’s anomaly detection algorithm is efficient in identifying misinformation, and the visualization is useful for analysts to discover insights and comprehend the model.

vis

mapping of time: static
dimensionality: 2D

Relevant references: Zhao et al. (2014)

Line Density Plot

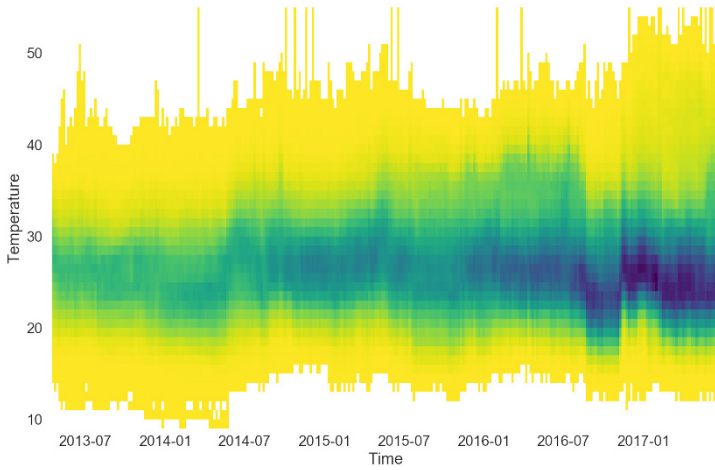


Fig. A.76: Line density plot of the temperature of more than 100.000 hard drives over a period of four years. Yellow to violet colors indicate low to high density of the underlying time series. © Courtesy of Dominik Moritz.

Datasets that consist of very many time series are difficult to visualize using the standard line plot (\leftrightarrow p. 233) technique. The reason is that for each time series a separate curve must be drawn in the plot. For large numbers of time series, this not only drains the rendering performance, but also leads to cluttered visual representations that are difficult to decipher. The idea of line density plots is to visualize not individual lines, but the density of lines on the display. Moritz and Fisher (2018) describe the steps that are necessary to construct a line density plot. First, the time axis and the value axis of the plot are subdivided into bins. As a result, the plot area is defined as grid of discrete cells. Then a time series is rendered as a polyline onto the grid, but only virtually so. Instead of actually drawing onto the grid, the grid cells just store the value of 1 if the polyline passes through them. The stored values are then normalized on a per column basis. This is done for all time series. In the end, one obtains a grid where cells that are passed by many lines contain a large density value, whereas cells passed by only a few lines have a lower value stored in them. The density values are then mapped to colors using an appropriate color scheme. The figure shows an example where low density is indicated by yellow, medium density is shown in green, and high density values are represented in violet. From the density plot, one can easily see where the majority of data values lie. Also, trends can be discerned, for example, the increasing accumulation of high density around the mid-twenty degrees temperature toward the right end of the time axis.

Relevant references: Moritz and Fisher (2018)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

Matrix-Based Comparison

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

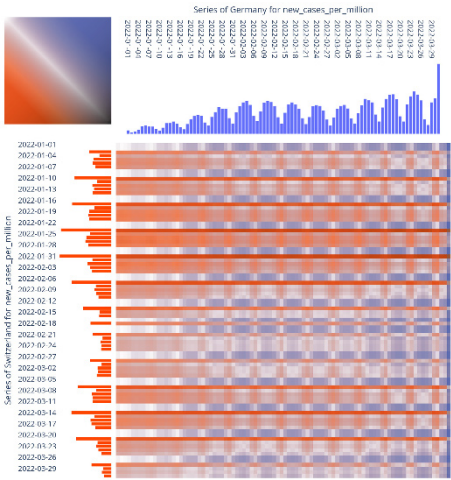


Fig. A.77: Matrix-based visual comparison of time-series data. The example shows data about CoViD cases over three months in Germany (horizontal axis) and Switzerland (vertical axis).
© Courtesy of Sarah Clavadetscher.

Behrisch et al. (2012) and Beck et al. (2016) introduce a matrix-based visualization approach for comparing two time series. The idea is to map the two time series to the vertical and horizontal axes of a matrix. The figure shows an example of CoViD data for three months from 2022. We see the number of cases of CoViD in Germany along the horizontal axis, whereas the vertical axis shows the data of Switzerland. Each cell of the matrix corresponds to a pair of time points and their associated data values. The cells are color-coded to communicate two pieces of information: a difference measure δ representing the difference between two time points and an aggregation measure σ representing the combined value of two data points. In the example, δ is encoded by varying the hue from red via neutral hue-less gray to blue. Color brightness encodes σ , where brighter colors correspond to smaller aggregated values and darker colors stand for larger values. A non-linear mapping function is used to make colors better distinguishable. The resulting two-dimensional color map is shown in the top-left corner. The advantage of this color-coding is that it can directly visualize the difference between two values and also provide an indication of the overall magnitude of two values. In the figure, dark and saturated red cells represent pairs of time points where Switzerland had a higher number of cases, while the cases in both Switzerland and Germany were generally high. Brighter less saturated blue cells indicate that Germany had a slightly higher number of cases, while the numbers were generally low in both countries.

Relevant references: Behrisch et al. (2012) • Beck et al. (2016)

MultiStream

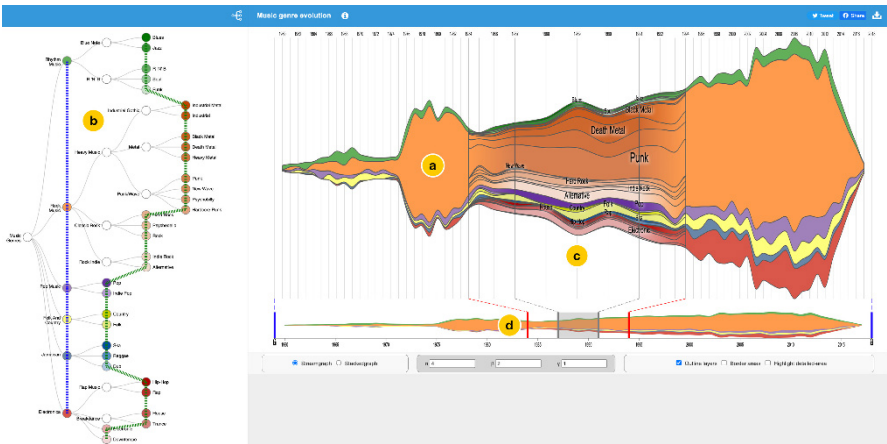


Fig. A.78: MultiStream is a streamgraph-based visualization approach (a) that incorporates the hierarchical structure of multiple time series (b) and adds a focus+context view (c) as well as overview+detail navigation (d). The example shows the evolution of different music genres over time. © The authors. Generated with the *MultiStream* prototype by Erick Cuenca.

Streamgraph (↔ p. 286) representations are well suited for representing multiple time series that can be stacked, i.e., with a meaningful sum. However, they do not scale very well when a large number of time series or longer time intervals need to be displayed. MultiStream by Cuenca et al. (2018) has been designed to address both of these challenges. The approach makes use of the hierarchical organization of the time series, e.g., music genres and sub genres, in order to provide less cluttered overviews that still convey the most important information. The MultiStream interface is split into multiple components: a streamgraph (a), a hierarchy manager (b), a multi-resolution focus+context view (c), and an overview+detail view with navigation (d). The hierarchy manager on the left allows navigating through the hierarchical structure of the data. The multi-resolution view on the top right represents the data at three different levels of detail – full temporal and hierarchical information in the central focus area as well as less detailed and more compressed information in the two context areas to the left and right. Finally, the overview+detail view on the bottom right shows a coarse overview of the whole dataset while allowing one to navigate in time and select the time interval shown in the multi-resolution view.

Relevant references: Cuenca et al. (2018)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

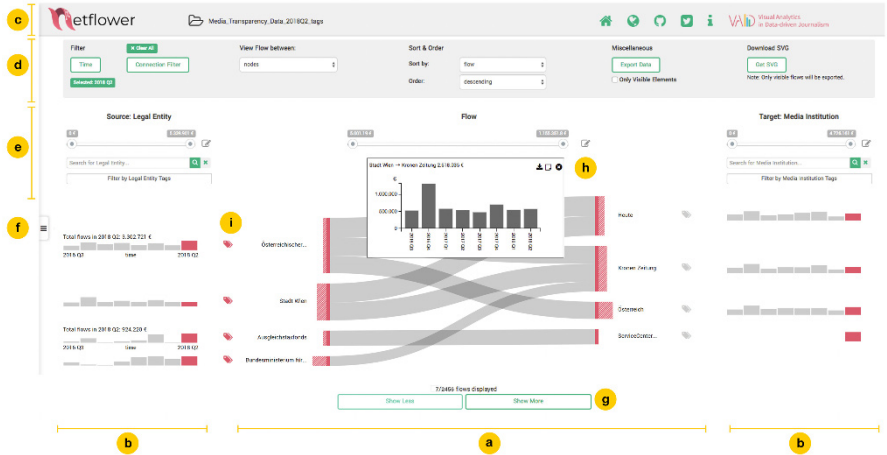
vis

mapping of time: static
dimensionality: 2D

time

netflower

primitives: points
arrangement: linear



data

number of variables: multiple
frame of reference: abstract

Fig. A.79: Netflower allows exploring the temporal dynamics of flows in bipartite networks. Its central visual metaphor is based on Sankey diagrams. The example shows quarterly money flows between government institutions (nodes on the left) and media outlets (nodes on the right). © The authors. Generated with the *netflower* prototype by Stoiber et al. (2019).

vis

mapping of time: static
dimensionality: 2D

Netflower supports the visual exploration of flows in dynamic networks, such as money flows between organizations or migration flows between countries. The central visualization (a) is a Sankey diagram (↔ p. 313) whereas the left side shows the originating nodes and the right side the destination nodes of a bipartite network. The width of each connecting line is proportional to the flow quantity as sum over the selected time interval. Netflower particularly aims for scalability to large graphs and supporting the exploration of the temporal dynamics of flows using details-on-demand, filters and tags, as well as auxiliary views. To keep the initial amount of elements shown on the screen manageable when loading new data, only the first dozen of elements are represented in full detail while more elements are shown upon user request (g). To signify that only a part of the total flows for each node is shown, a node's area is split into a filled and a hatched area, whereas the latter represents the portion of flows not currently displayed on the screen. Netflower provides functions for filtering, sorting and ordering, as well as a provenance notebook (d-f, i). The time interval to be shown in the Sankey view can be selected using a filter (d). Upon selection of a flow, a bar graph (↔ p. 234) shows the flow values for each time point in the data (h). To the left and right of the nodes, bar graph sparklines (↔ p. 235) on both sides (b) give a temporal overview of outgoing (source nodes) and incoming flows (target nodes).

Relevant references: Stoiber et al. (2019)

Optimized Stream Graphs

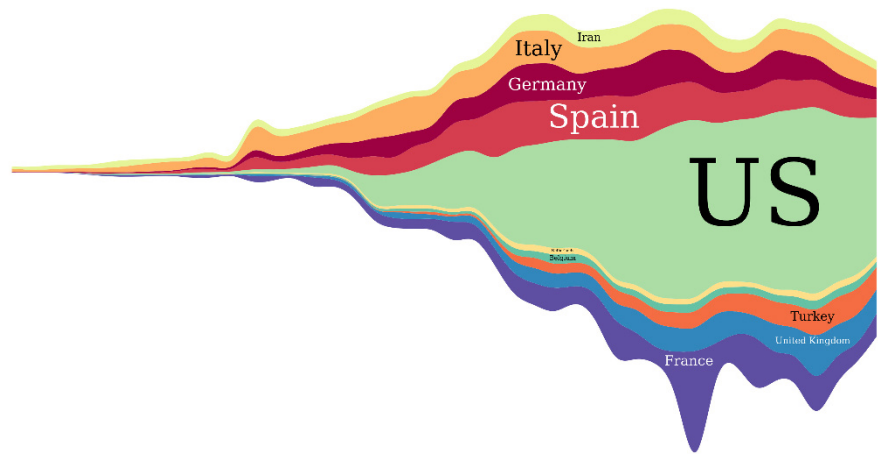


Fig. A.80: Optimized stream graphs adjust the baseline and the stacking order of stacked graphs to reduce sine illusion when interpreting the data. © The authors. Generated with the *SineStream* software by Chuan Bu.

Visualization techniques that represent individual time-dependent variables as layers (↔ p. 289) and also stack these on top of each other like ThemeRiver (↔ p. 293), 3D ThemeRiver (↔ p. 317), and generally stacked graphs (↔ p. 286) inevitably suffer from sine illusion. While said visualization techniques map data values by varying the height of the layers, human perception tends to interpret not the height of layers with respect to the vertical y-axis only, but the “width” of layers with respect to their main direction of flow, more exactly perpendicular to the perceived main direction. This perceptual “illusion” can have adverse consequences for the extraction of individual data values from the visualization. Therefore, optimization techniques have been developed to reduce these adverse effects by appropriately arranging the layers in the visualization. This can be achieved by adjusting the baseline of the visualization and the stacking order of layers. Byron and Wattenberg (2008), Bartolomeo and Hu (2016), and Bu et al. (2021) proposed several optimization strategies with different trade-offs and suitability for different types of data. Their studies show that optimized stream graphs improve the perception of data and make it easier to extract data values from the visualization.

Relevant references: Byron and Wattenberg (2008) • Bartolomeo and Hu (2016) • Bu et al. (2021)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

RankExplorer

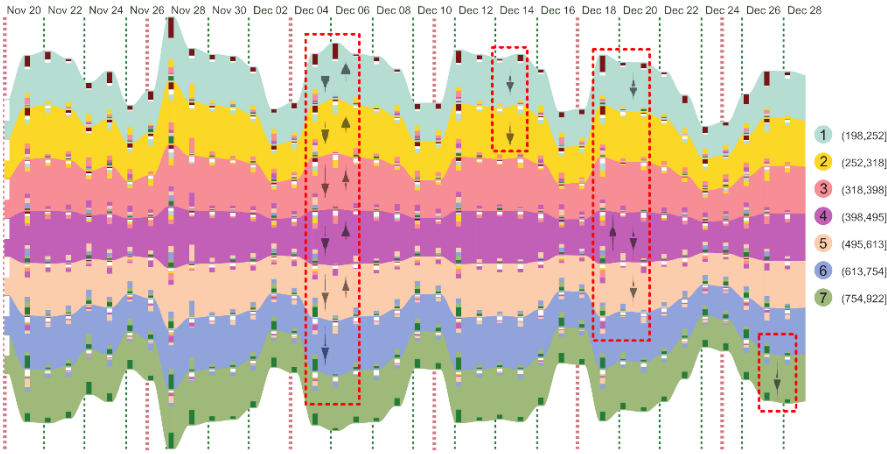


Fig. A.81: RankExplorer visualization of a subset of the top 2000 search keywords on Bing. Each colored layer summarizes multiple keywords. Small bars and glyphs indicate how keywords switch layers from one time step to the next. © ⓘ Courtesy of Conglei Shi.

When it comes to understanding the ranking of many time series, the RankExplorer by Shi et al. (2012) offers a useful solution. The ranking per time point depends on which time series have the highest, second highest, third highest value, and so on. In other words, the sorted data values of a time point define the ranking order for that time point. It is now interesting to analyze how the ranking order changes over time. As this would be too complex on the level of individual time series, the RankExplorer first groups the time series into a smaller number (seven in the figure) of ranking segments or layers. These ranking layers are represented as differently colored, layered bands of varying width along a horizontal time axis, which is analog to the ThemeRiver (↔ p. 293) approach. Changes in the ranking occur when time series leave one layer and enter another one. These changes are represented by small colored bars within the bands. The bars indicate (by the proportion of colors per bar) how many time series enter a layer from the previous time point and how many leave a layer toward the next time point. Additionally, small glyphs (see arrow-like shapes in dashed frames) visualize the content changes between ranking layers. For detailed inspection, users can zoom in to be provided with more fine-grained ranking layers, bars, and glyphs.

Relevant references: Shi et al. (2012)

Sankey Diagram, Alluvial Diagram

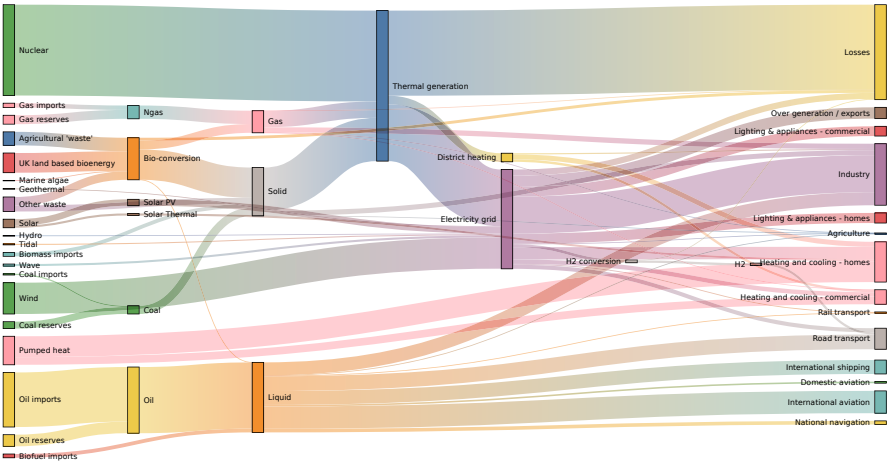


Fig. A.82: Sankey diagram showing how energy in the UK is converted or transmitted before being consumed or lost. Supplies are on the left and demands are on the right. The connecting lines visualize energy flows between stages and line thickness is proportional to the flow quantity. © The authors. Generated with the [Sankey Diagram](#) notebook by Mike Bostock.

Sankey diagrams are used to represent flows of data such as money, people, energy, or material in a system by means of their input and output flow distributions. The width of the lines or arrows connecting different nodes or stages is proportional to the flow quantity. In most cases, Sankey diagrams do not reflect time via absolute position but as a sequence of stepwise changes over time, i.e., they use an ordinal time scale. Originally developed for depicting energy flows, its basic principle is very versatile and can be applied to all kinds of flow data in different application domains. A historical example of the application of this approach is Minard’s graphical representation of Napoleon’s Russian campaign that was created in 1869 (see Chapter *Historical Background*). In addition to changes in line thickness, Minard also used color hue as an additional visual variable for representing the direction of movement. Interestingly, this form of graphic representation is named after Irish Captain Matthew Henry Phineas Riall Sankey who only used this visualization method once in 1898 to represent energy flows in steam engines in Kennedy and Sankey (1898). *Alluvial diagrams* by Rosvall and Bergstrom (2010) are a special form of Sankey diagrams intended to represent structural changes in large complex networks over time, that is, they can be used to show changes in group composition over time. Brinton (1939), the representation concept is called *Cosmograph* and focuses on the comparison of source and destination flow distributions.

Relevant references: Kennedy and Sankey (1898) • Rosvall and Bergstrom (2010) • Brinton (1939)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

TACO



Fig. A.83: Visualization of changes in tables over time using TACO. Multiple levels of detail are employed along an overview+detail concept: (a) selection of types of changes to be displayed, (b) aggregated changes between two consecutive time points, (c) more detailed comparison of two selected time points showing changes and their distribution for rows and columns, (d) a detailed view of raw data table versions (bottom left and bottom right) as well as a diff heatmap (bottom center). The example shows differences between Summer Olympic Games medal tables over time between 1896 and 2012. © The authors. Generated with the TACO demo by Niederer et al. (2018).

Tables may change over time in both structure and content, resulting in multiple versions. A challenging task is to understand what exactly has changed between versions (additions/deletions, reorder, merge/split, and content changes). TACO addresses this by visualizing the differences between multiple tables at various levels of detail. For this, it allows users to gradually add more focused and detailed views by interacting with the interface along three levels of detail. The interface contains a change overview timeline for aggregated differences between multiple table versions over time (level 1), allows for aggregated pairwise comparisons with a more detailed view of the distribution of changes along rows and columns (level 2), and shows a detailed pairwise comparison as well as the raw data (level 3). Across all views, color hue is used to differentiate the four different types of changes. On level 1, stacked bar graphs (↔ p. 234) on a timeline provide an overview of when changes to a table occurred along with the amount and types of changes between consecutive points in time. For the second level, a 2D ratio chart and additional histograms for rows and columns provide more details about changes between two selected time points. On the third level, a detailed view shows heatmap representations of the two selected raw data table versions (left and right) as well as a diff heatmap in the center. Reordering changes between rows are shown with rank charts.

Relevant references: Niederer et al. (2018)

WireVis

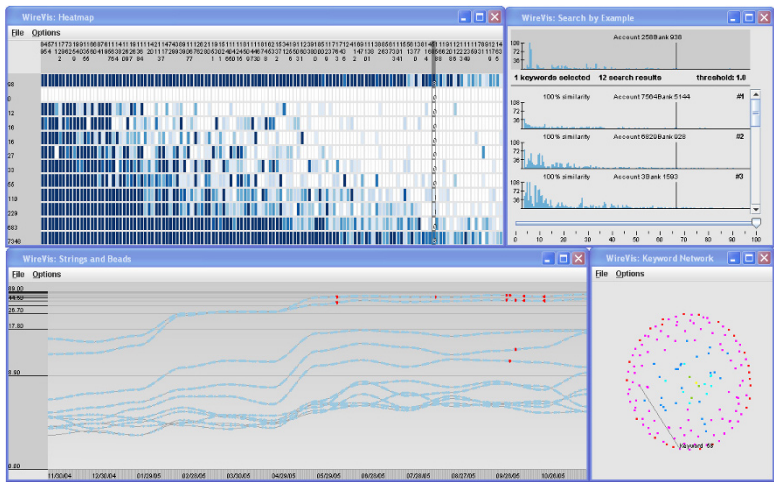


Fig. A.84: WireVis’s multiple coordinated view interface capturing four views to explore complex numerical and categorical time-oriented transaction data: heatmap, search-by-example, strings-and-beads view, and keyword network view. © 2007 IEEE. Reprinted, with permission, from Chang et al. (2007).

Financial transaction data capture complex numerical and categorical time-oriented data, which need to be investigated for suspicious behavior (fraudulent activities, which are changing over time). WireVis by Chang et al. (2007) is a visual analytics approach that utilizes four tightly coordinated views of transaction activity. A heatmap view shows relationships between accounts and keywords. A search-by-example approach supports analysts to discover accounts with similar activities. A strings-and-beads view depicts the transactions over time. And finally, the keyword network view is used to represent the relationships between keywords. All four views rely on high interactivity along with the ability to see global trends and capabilities to drill down into specific transaction records. The usefulness of WireVis was demonstrated in case studies using transaction data of the Bank of America.

Relevant references: Chang et al. (2007)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

MOSAN

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D, 3D

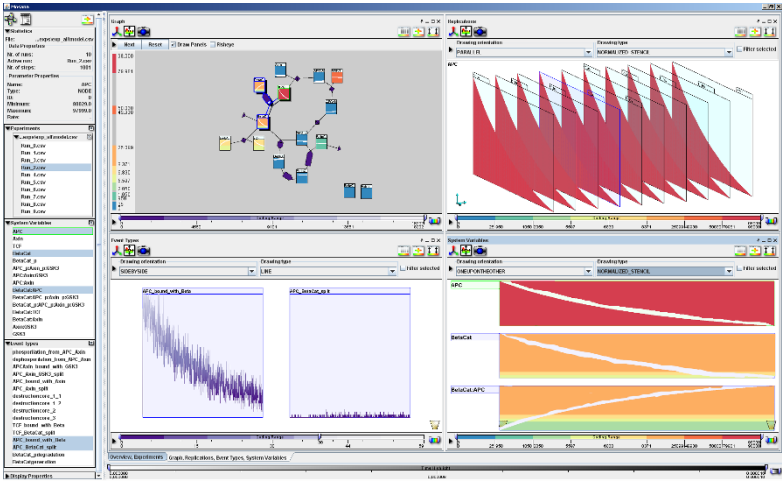


Fig. A.85: The top-left view shows a simulated reaction network. An overview of the time-dependent simulation data is given by the small line plots in the boxes of the network nodes. Three coordinated linked views are provided for the comparison of simulation runs and variables. © *Courtesy of Andrea Unger.*

MOSAN is a tool for visualizing multivariate time-oriented data that result from simulation of reaction networks. Due to the stochastic multi-run simulation, each variable comprises multiple time series. In order to facilitate the understanding of the complex dependencies in the data, it is necessary to jointly visualize structural information and stochastic simulation data together. To this end, Unger and Schumann (2009) combine different views within a single interactive interface. In an overview, time-oriented data are shown along with the structural relations among the variables in the reaction network. The structural relations are shown by a graph layout, where boxes correspond to variables, and simulation data are visualized by small line plots within the boxes (top-left view). The small line plots provide a highly aggregated view of the stochastic simulation data, thus focusing on the communication of the general temporal trends. Furthermore, advanced color-coding is applied to the plots to support the comparison of heterogeneous value ranges among variables. In addition to the overview, coordinated linked views support the inspection of individual time series of the same variable (top right) as well as the detailed inspection and comparison of temporal developments of different variables selected from the overview (bottom). Filter sliders further support the drill down into the data.

Relevant references: Unger and Schumann (2009)

3D ThemeRiver

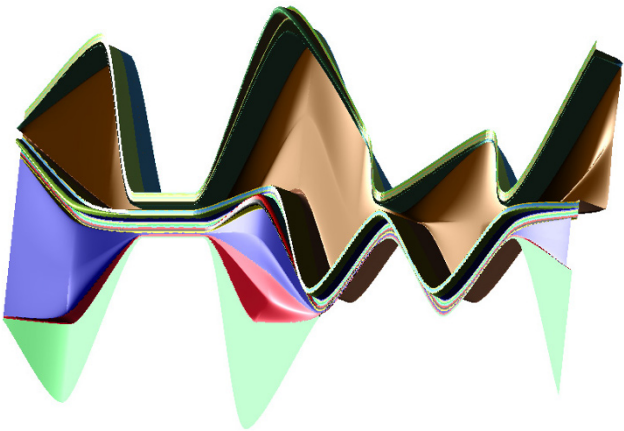


Fig. A.86: Distinctly colored currents form the overall shape of the 3D ThemeRiver. The width, and additionally the height of currents is varied to visualize time-oriented data. In this figure, width encodes the overall distribution of 17 clusters of aerosol data and height indicates the incidence of zinc. © 2003 IEEE. Reprinted, with permission, from Imrich et al. (2003).

Imrich et al. (2003) propose a 3D variant of the ThemeRiver technique (↔ p. 293). The 3D approach inherits the basic visual design from its 2D counterpart: multiple time-oriented variables are encoded to the widths of individually colored currents that form a river flowing through time along a horizontal time axis. In the 2D variant, only one data variable can be visualized per current, namely by varying the current’s width. Imrich et al.’s extension addresses this limitation. By extending the design to the third dimension it is possible to use an additional visual encoding: the height (in 3D) of a current can be varied to encode further information. This design is particularly suited to visualizing ternary covariate trends in the data. Imrich et al. conducted user tests to evaluate the usefulness of the 3D encoding, and indeed got positive results that indicate that the 3D variant has advantages over the 2D variant. Specifically, the availability of appropriate interactive 3D navigation tools is highlighted as an important factor contributing to the success of the 3D ThemeRiver.

Relevant references: Imrich et al. (2003)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

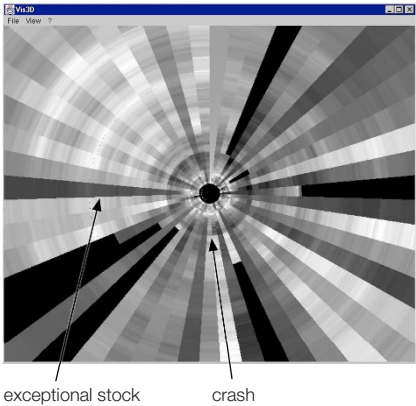
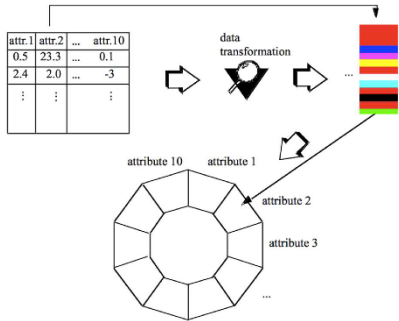
vis

mapping of time: static
dimensionality: 3D

time

Data Tube Technique

primitives: points
arrangement: linear



data

number of variables: multiple
frame of reference: abstract

Fig. A.87: Data are mapped onto the inside of a 3D tube using a tabular layout. Each slice of the tube represents a time point and each cell represents a data parameter by color. Left: visual mapping schema; right: exploring 50 different stocks. © Courtesy of Mihael Ankerst.

In the data tube technique by Ankerst (2001) multiple time-oriented variables are mapped to bands that follow the inside of a 3D tube. The basic mapping procedure is depicted in the left part of the figure. Each slice of the tube represents a time point and each cell represents a data value by color. The tube is viewed from above and time is flowing to or from the center of the tube. The user is able to explore the data by interactively moving through the 3D tube. Because of the 3D perspective distortion, cells that are further away appear to be smaller in size, much like in a focus+context display. As a result of this, the number of displayed variables and the number of displayable time points can be quite large. Later, Ankerst et al. (2008) also developed a comprehensive temporal data mining architecture called DataJewel that is closely integrated with pixel-oriented visualization techniques. Further visual and interactive extensions have been developed by Sureau et al. (2009) and Bouali et al. (2016) in DataTube2. One such extension detailed by Devaux et al. (2014) is to include 3D link arcs into the tube to support the visualization of time-oriented log data.

vis

mapping of time: static
dimensionality: 3D

Relevant references: Ankerst (2001) • Ankerst et al. (2008) • Sureau et al. (2009) • Devaux et al. (2014) • Bouali et al. (2016)

Kiviat Tube

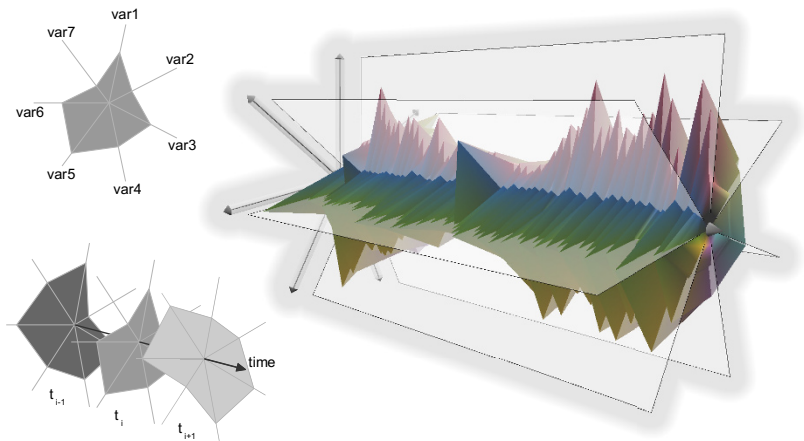


Fig. A.88: Construction of a three-dimensional Kiviat tube representing seven time-dependent variables. Peaks and valleys indicate ups and downs in the evolution of the data over time. Wings assist in associating features of the Kiviat tube to particular variables in the data. © The authors. Generated the VisAxes3D software by Clemens Holzhüter.

The Kiviat tube described by Tominski et al. (2005a) visualizes multiple time-dependent variables. The construction of a Kiviat tube is as simple as stacking multiple Kiviat graphs (see Kolence and Kiviat, 1973) along a shared time axis. Each Kiviat graph represents the data for multiple variables for a specific point in time. But instead of drawing individual Kiviat graphs, a three-dimensional surface is constructed. This way, multiple, otherwise separated time points are combined to form a single 3D body that represents the dataset as a whole. The spatial characteristics of a Kiviat tube can be recognized easily, as it allows users to identify peaks or valleys in the data over time. Additional semi-transparent wings assist in relating identified patterns to particular variables. Similar wings were also employed by Akase and Okada (2015). Common interaction methods can be used for zooming and rotation around arbitrary axes. Rotation specifically around the time axis enables users to quickly access variables on all sides of the Kiviat tube. Interactive axes allow users to navigate back and forth in time to visit different portions of a possibly large time series.

Relevant references: Tominski et al. (2005a) • Kolence and Kiviat (1973) • Akase and Okada (2015)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 3D

time

Temporal Star

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 3D

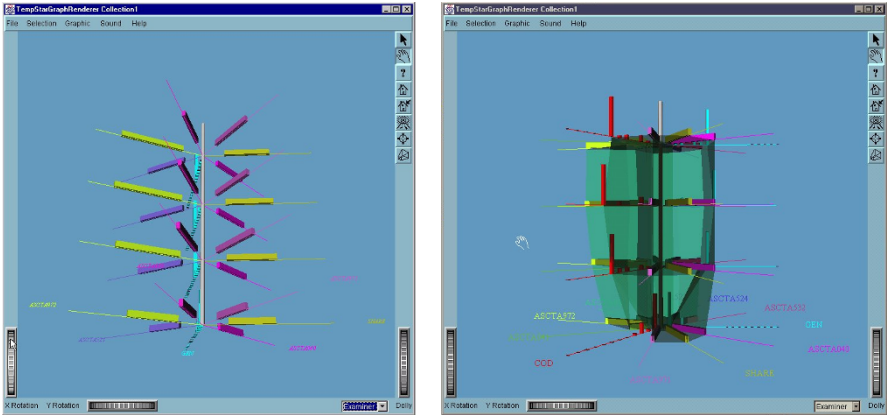


Fig. A.89: 3D representation of circular column graphs that are arranged in a row to represent each time step. A transparent veil can be displayed to enhance the perception of the dataset’s evolution.
© Courtesy of Monique Noirhomme.

The temporal star technique by Noirhomme-Fraiture (2002) visualizes multivariate data structures in 3D. For each point in time, a circular column graph is drawn that represents each variable’s value as a bar in a circular arrangement. These graphs are aligned in a row to represent the development of the dataset over time (left figure). A unique color is assigned to each variable to aid the recognition of variables across time. Moreover, a transparent veil can be displayed to enhance the perception of the dataset’s evolution as a whole (green parts in the right figure). The concept used is similar to that of the Kiviat tube (↔ p. 319), which uses Kiviat graphs instead of circular column graphs. In the temporal star technique, difference plots are also integrated, showing the relative differences between variables rather than absolute values. The rendering parameters, the shown time intervals, and the configuration of the axes can be adjusted interactively. Furthermore, the temporal star technique is integrated with a data warehousing application that provides rich data manipulation features.

Relevant references: Noirhomme-Fraiture (2002)

Time-tunnel

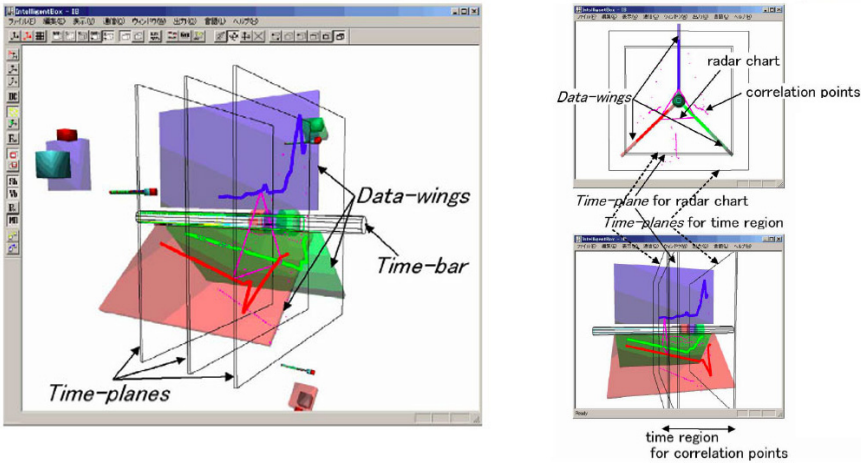


Fig. A.90: Time-tunnel visualization with three time-dependent variables. Left: individual plots are put onto semi-transparent planes (data wings) that are positioned around a central time bar; top right: selected time points can be viewed as radar charts; bottom right: multivariate perspective on the data. © 2004 IEEE. Reprinted, with permission, from Akaishi and Okada (2004).

Akaishi and Okada (2004) developed time-tunnel as a data analysis technique for visualizing a number of time-series plots in a 3D virtual space. The individual plots are put onto semi-transparent planes (data wings) that are positioned around a central time bar in a fan-like manner (left figure). In the example figure, line plots are used for visualizing the time-dependent data but any other linear time visualization might also be used. Multiple planes can be overlapped and compared thanks to their transparency. Furthermore, single time slices or selected time intervals can be viewed as radar charts (top right) and provide a multivariate point of view (bottom right). Additionally, not only time series, but any kind of information can be put onto a plane, making the time-tunnel a multimedia presentation tool.

Relevant references: Akaishi and Okada (2004) • Akase and Okada (2015)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

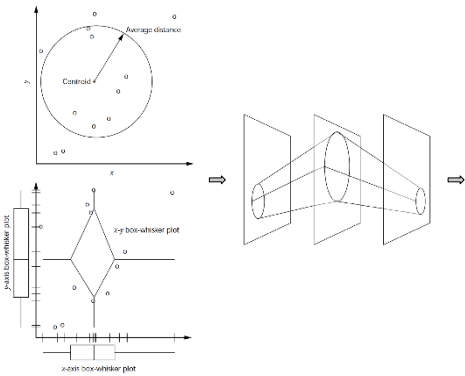
vis

mapping of time: static
dimensionality: 3D

time

primitives: points
arrangement: linear

Worm Plots



data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 3D

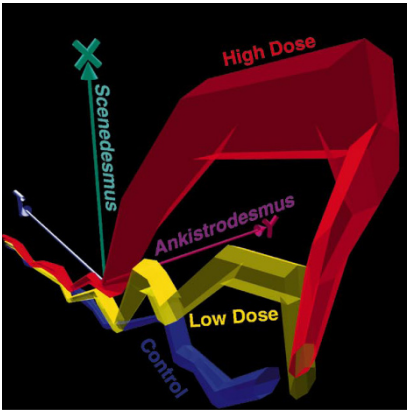


Fig. A.91: Worm plots are generated by creating visual abstractions of point groups at multiple time steps and by assembling a 3D surface that resembles a worm. The worm plot on the right shows three groups (control, high dose, low dose) of data points of toxicology experiments plotted against the two variables Scenedesmus and Ankistrodesmus. © 1997 IEEE. Reprinted, with permission, from Matthews and Roze (1997).

Worm plots have been developed to help scientists gain qualitative insights into the temporal development of groups of points in scatter plots. The initial step necessary to construct a worm plot is to generate a visual abstraction of multiple points. One way to do this is to compute the centroid of a group of points and the average distance of points to the centroid. The visual abstraction is then a circle that is located at the centroid and has a radius equal to the average distance. Alternatively, a 2D generalization of box-whisker plots can be used to form a diamond-shaped visual abstraction. Such abstractions are computed for each time step (i.e., each scatter plot). Subsequently, a three-dimensional surface (worm) is assembled from the visual abstractions of each group. This procedure is illustrated in the left part of the figure. Presented in an interactively manipulatable virtual world, worm plots allow users not only to see where in the variable space point groups are located, but also to discern the compactness of point groups, and to understand the development of these characteristics over time.

Relevant references: Matthews and Roze (1997)

Software Evolution Analysis

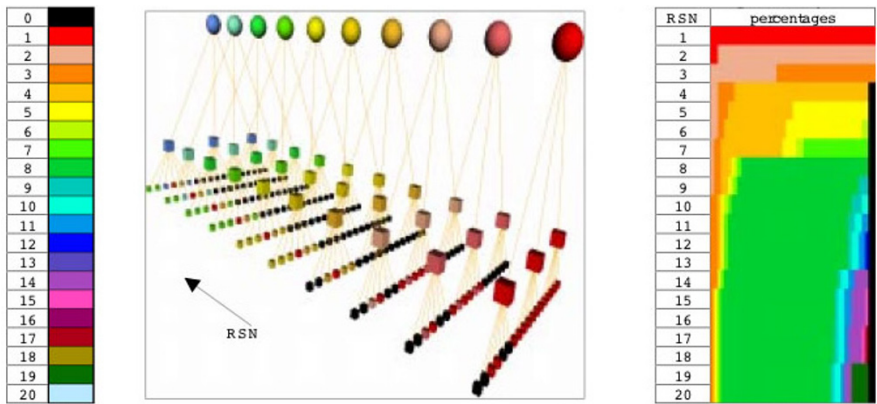


Fig. A.92: 3D visualization to analyze software systems and product families. Left: color legend for different versions; center: hierarchical decomposition by modules, packages, and files (color represents current version); right: individual file where each row represents a single version colored by percentage of code originating from a particular (previous) version. © 1999 IEEE. Reprinted, with permission, from Gall et al. (1999).

The software evolution analysis technique by Gall et al. (1999) uses 3D representations to depict software systems or product families respectively. The information is decomposed hierarchically into modules, packages, and files or similar concepts. This hierarchy is depicted as a three-dimensional tree structure in which the leaf nodes represent individual files. Multiple such trees are aligned in layers in the 3D space, with one layer for each revision of the software. Color is used to distinguish different versions and to show changes over time. Furthermore, individual files might be inspected in more detail to explore the evolution of changes over time using proportionally colored version lines (right in the figure). This way, patterns are formed that can be used to identify, for example, stable parts of a system, frequently changed parts, similarities, and more.

Relevant references: Gall et al. (1999)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 3D

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 3D

3D TimeWheel

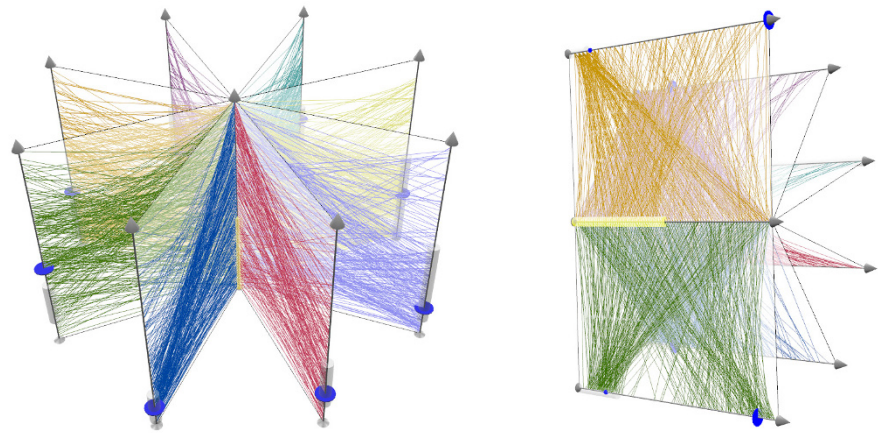


Fig. A.93: The 3D TimeWheel uses a central axis to represent time. Several axes are arranged around the time axis to visualize time-dependent variables. Left: radial layout; right: page-flip layout. © The authors. Generated with the VisAxes3D software by Clemens Holzhüter.

The 3D TimeWheel by Tominski et al. (2005a) is a three-dimensional variant of the TimeWheel technique (\hookrightarrow p. 298). Analogous to the original TimeWheel, the 3D TimeWheel arranges multiple axes of time-dependent variables around a central axis representing the dimension of time. Lines between axes connect time points to data values. Different arrangements of the axes are possible. The central time axis can be surrounded by evenly distributed time-dependent axes (left). This is similar to the classic Parallel Coordinates technique, with the only difference being that for Parallel Coordinates lines are drawn between adjacent axes, while for the 3D TimeWheel lines are always linked to the central time axis. For easier comparison of two selected time-dependent variables, it is also possible to create a distribution of axes that grants a full view on the axes of interest (right). Using a page-flip metaphor, the user can flip the axes panes like the pages of a book. Handles on the axes further support filtering and navigation in time. A study by Hassan et al. (2019) found that 3D representations can be beneficial for the visual exploration of time-oriented data.

Relevant references: Tominski et al. (2005a) • Hassan et al. (2019)

Vanishing-Point Plot

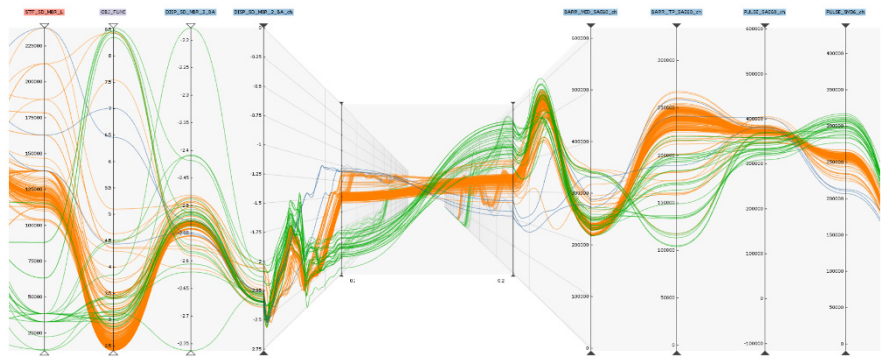


Fig. A.94: Vanishing-point plots add the time dimension to Parallel Coordinates plots by embedding additional 3D perspective time plots for selected data attributes. The example shows a crash simulation dataset containing design parameters and the simulated responses for each simulation run as axes. © 2016 Eurographics Association and John Wiley & Sons. Reprinted, with permission, from Gruendl et al. (2016).

Parallel Coordinates by Inselberg and Dimsdale (1990) are one of the most well-known visualization techniques for exploring multivariate datasets. However, a remaining challenge is the integration of the time dimension in addition to showing multivariate relationships. One option is to add time as an additional axis to the Parallel Coordinates plot. But this leads to clutter as well as a loss of the temporal order for all axes not adjacent to the time axis. A second option is to use animation to dynamically visualize temporal developments. Yet, this makes comparisons between points in time difficult. As a third option, the vanishing-point plot adds time on demand between any two neighboring data variables. This is in contrast to the related TimeWheel (\hookrightarrow p. 298) and 3D TimeWheel (\hookrightarrow p. 324) techniques, where different axes layouts are used to allow for having all variable axes in direct relation to the time axis. Integration of time via the vanishing-point plot is achieved by adding a pseudo-perspective view of two line plots over time that are integrated between two adjacent axes of a Parallel Coordinates plot (see figure). To connect two variables, a translucent parallel coordinates panel is shown that depicts the relationship of the two variables according to the selected time points. Time extends from the foreground into the background toward a vanishing point. Users can move forward and backward in time by moving the panel back and forth. This allows for the analysis of trends over time as well as the exploration of changes in the relationship between two variables over time.

Relevant references: Gruendl et al. (2016) • Inselberg and Dimsdale (1990)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 3D

time

primitives: points
arrangement: linear

InfoBUG

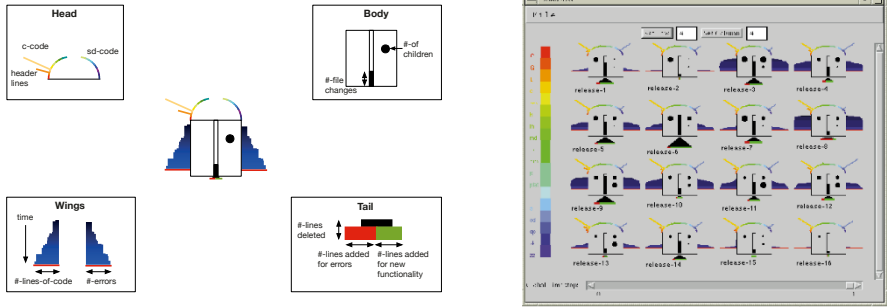


Fig. A.95: The InfoBUG encodes data about software to the wings, head, tail, and body of a glyph (left). The wings show lines of code and number of errors over time, whereas body, head, and tail show further data for a selected time point. A small multiples view can be used to compare different software releases over time (right). © 1998 IEEE. Reprinted, with permission, from Chuah and Eick (1998).

data

number of variables: multiple
frame of reference: abstract

Chuah and Eick (1998) developed InfoBUG for visualizing changes in software projects. The InfoBUG is an information-rich graphic that combines a multitude of different heterogeneous data values. The glyph resembles an insect with wings, head, tail, and body. The different parts of the glyph are used to represent four different classes of information about software projects as shown to the left in the figure: code lines and errors (wings), types of code (head), added and deleted lines of code (tail), and number of file changes and children (body). The wings represent the lines of code (left wing) and the number of errors (right wing) over time as vertical silhouette graphs. While the wings show data over time, the other parts of the glyph show only the data of a user-selected time point, which is indicated as a red line at the wings. Antennas on the InfoBUG's head represent different types of code, where color indicates the type of code, and the relative sizes of different types are encoded by antenna length. The bug's tail represents the number of deleted and added lines. Finally, the InfoBUG's body visualizes information about the number of altered files via a bar in the middle of the body and the number of child objects via filled circles. Small multiples (↔ p. 359) can be used to compare the different releases of a software product over time as shown in the figure to the right. Furthermore, the representation can be animated to follow the course of time.

vis

mapping of time: static, dynamic
dimensionality: 2D

Relevant references: Chuah and Eick (1998)

Gravi++

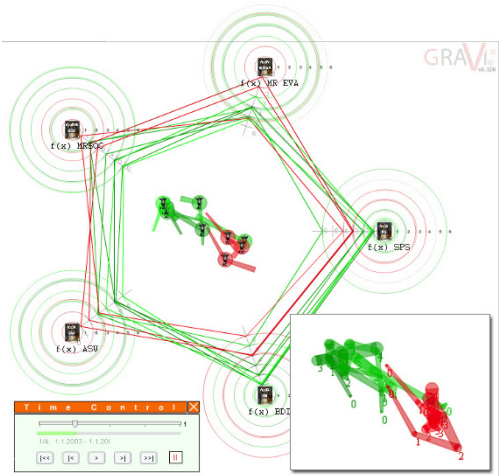


Fig. A.96: Patient icons in the center of the display are positioned relative to the surrounding parameters (in this case items of a questionnaire) following a spring-based model. Individual answers to questionnaire items are shown as concentric rings, and star glyphs show sets of answers as polygonal lines. The user can step through time manually or can use animation, which can be steered via the control panel on the lower left. Furthermore, traces might be displayed to convey information about the evolution of values over time as shown in detail on the lower right. © *Courtesy of Klaus Hinum.*

Gravi++ has been developed as a tool for finding predictors for the treatment planning of anorexic girls. It represents patients and data gathered from questionnaires during treatment over the course of several weeks or months. Patients are represented by icons that are laid out according to a spring-based model relative to the surrounding icons, which represent items of a questionnaire. This leads to the formation of clusters of persons who gave similar answers. Animations are used to visualize the change in values over time. The position of each person’s icon changes over time, making it possible to trace, compare, and analyze the changing values. Alternatively, the change over time can be represented by traces. The size and path of the person’s icon are shown corresponding to all time steps or only to a restricted subset like the previous and the next time step. To visualize the exact values of each question, rings around the question’s icon can be drawn and star glyphs might be shown.

Relevant references: Hinum et al. (2005)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

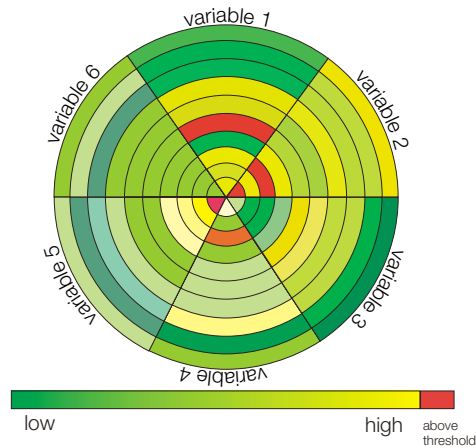
vis

mapping of time: static, dynamic
dimensionality: 2D

time

CircleView

primitives: points
arrangement: linear



data

number of variables: multiple
frame of reference: abstract

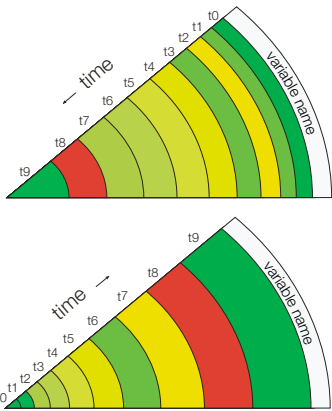


Fig. A.97: Multiple variables are shown as segments of a circle. Each segment is further subdivided into time slots that represent data values using color. Left: six variables over ten time steps; right: different time axes arrangements (outside-in vs. inside-out) and space assignment that emphasizes more recent values. © The authors. Adapted from Keim et al. (2004).

vis

mapping of time: static, dynamic
dimensionality: 2D

Keim et al. (2004) describe the CircleView as a technique for visualizing multi-variate streaming data as well as static historical data. The basic idea is to divide a circle into a number of segments, each representing one variable. The segments are further divided into slots covering periods of time, and color shows the (aggregated) data value for the corresponding interval. Thus, time is mapped linearly along the segments. The user can interactively adjust the number of time slots, the time span per slot, different layouts for the time axis, and might emphasize more recent slots by assigning increasingly more space to them (bottom-right figure). Since the order of segments is important for the visual appearance and the comparison of the data, segments can be reordered interactively by the user or automatically based on similarity measures. For streaming data, the segments of the circle are shifted automatically from the center to the edge (or vice versa). Keim and Schneidewind (2005) also presented a multi-resolution approach on top of CircleView, where time slots for coarser granularities are shown besides detail values.

Relevant references: Keim et al. (2004) • Keim and Schneidewind (2005)

CloudLines

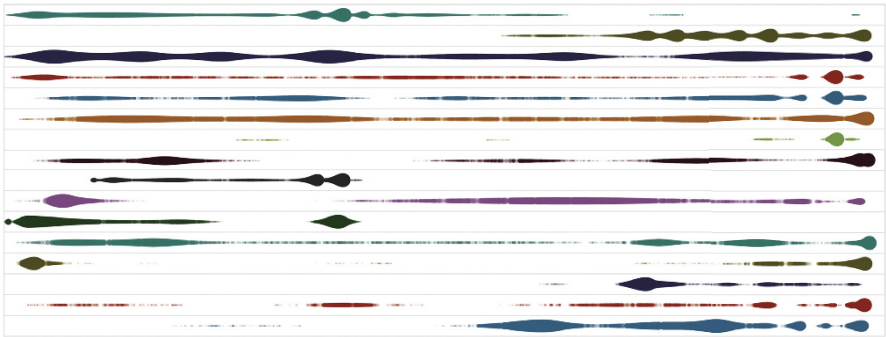


Fig. A.98: CloudLines provide a compact overview of event episodes from multiple time series. Individual events are shown as semi-transparent circles, whereas an importance function is mapped to both opacity and size for improved visibility. The figure shows a dataset of key politicians’ appearances in the news on February 2011, whereas each politician is shown in an individual row. © 2011 IEEE. Reprinted, with permission, from Krstajic et al. (2011).

Application areas that deal large amounts of data, like news publishing, network security, or financial services, are challenging in terms of data analysis as they require scalable solutions. CloudLines is an incremental time series visualization technique that uses interactive distortion to handle time-based representations of large datasets in limited space. It focuses on recent events while providing context on the past, and makes relevant patterns salient at any scale. In order to better focus on recent events, a logarithmic time scale can be used that allows for direct interaction with recent events. The method uses an importance function, which allows the visualization to adjust the opacity and the size of the events according to their age and data density based on a decay function and kernel density estimates. Each row represents a variable, e.g., weekly amount of tweets of a certain person, and qualitative color-coding is used to make row distinction easier. Access to details is provided through a magnifying lens, which takes the distortions in size and opacity into account. This increases readability in selected areas of interest. In order to be applicable for streaming data such as online news streams, incremental kernel density estimate algorithms and smoothly animated transitions for display updates can be used. In the work of Krstajic et al. (2011), CloudLines have been applied for visualizing news appearances of politicians and news event monitoring. They showed that CloudLines help to detect important event episodes in the time series, show the detailed structure of event episodes, and interact with time series on atomic level.

Relevant references: Krstajic et al. (2011)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static, dynamic
dimensionality: 2D

Animated Scatter Plot

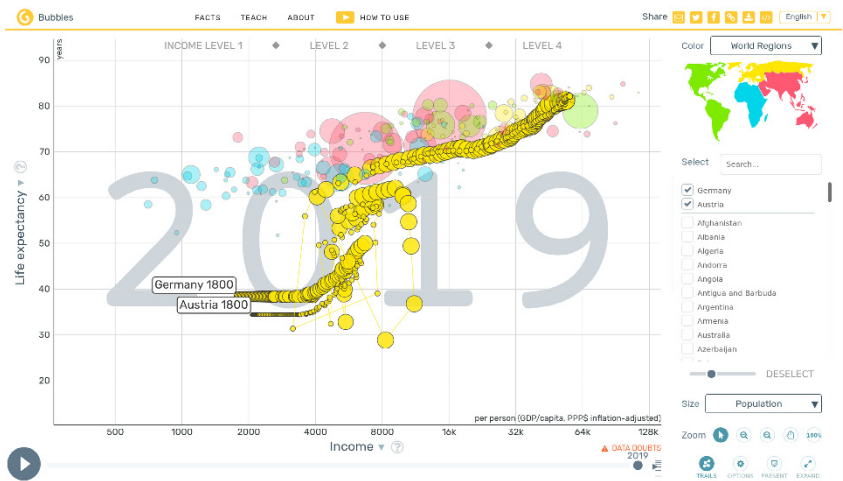


Fig. A.99: Two data variables (life expectancy and income) are mapped to the horizontal and vertical axes, symbol size represents a third variable (population), and animation is used to step through time. Additionally, trails are activated for the selected countries, Austria and Germany, which help viewers track countries over time. © The authors. Generated with *Bubbles* by the Gapminder Foundation.

Animated scatter plots show the data in a Cartesian coordinate system similar to point plots (→ p. 232). For point plots, one axis of the coordinate system represents time, the other axis encodes one time-dependent data variable. For animated scatter plots, the coordinate axes represent two time-dependent variables, while the dimension of time is visualized by means of animation. Animated scatter plots have been popularized by Hans Rosling, who used them extensively to explain facts about the world. The Bubbles tool (formerly known as Trendalyzer) by the Gapminder Foundation (2021) makes animated scatter plots available to a wide audience. The figure shows an example where the two data variables *life expectancy* and *income* are mapped onto the axes of the coordinate system. The dot size represents a third variable, namely *population*, and different dot colors mark groups of data items. An animation visualizes how the dots (and the underlying data) change over time. The animation can be controlled via a time slider, a play/pause button, and a slider for adjusting animation speed. Furthermore, trails can be displayed, which means that dots stay visible and are connected over time. This helps users to better see the path of a variable through time. Robertson et al. (2008) evaluated the use of animation in conveying trends over time and compared animated scatter plots with a modified version of trails, and small multiples (→ p. 359). The results show that animation is both slower and less accurate than the other representations but is well suited as a presentation aid.

Relevant references: Gapminder Foundation (2021) • Robertson et al. (2008)

Process Visualization

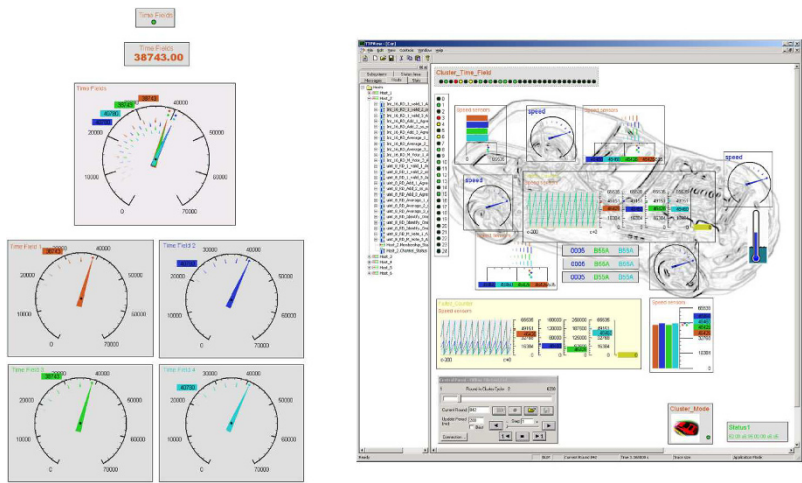


Fig. A.100: Process visualization can be supported by providing virtual instruments at different levels of detail (left). This helps users stay focused on important variables of an automotive process, while less relevant information is presented at a higher level of abstraction only (right). © 2002 IEEE. Reprinted, with permission, from Matković et al. (2002).

Process visualization, for instance in automotive environments, has to deal with a multitude of time-varying input variables to be monitored. Matković et al. (2002) suggest a focus+context approach to help users keep track of the important changes in a process. The key idea is to provide virtual instruments that represent monitored variables at different levels of detail. Instruments representing focused variables provide more detailed information, for example, a brief view on a variable’s history, which is not possible with classic gauges. On the other hand, less relevant variables are visualized using heavily abstracted virtual instruments that might show just the numeric value or even only a colored dot. Multiple such instruments are arranged in a virtual environment that is used as visual reference for the monitoring scenario. Focus and context within the environment can change dynamically during monitoring, either upon detection of certain events in the data or via user interaction. The approach of Matković et al. (2002) is an example of a visualization for dynamic temporal data where only the current state of the data is available without any history.

Relevant references: Matković et al. (2002)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: dynamic
dimensionality: 2D

TimeRider

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: dynamic
dimensionality: 2D

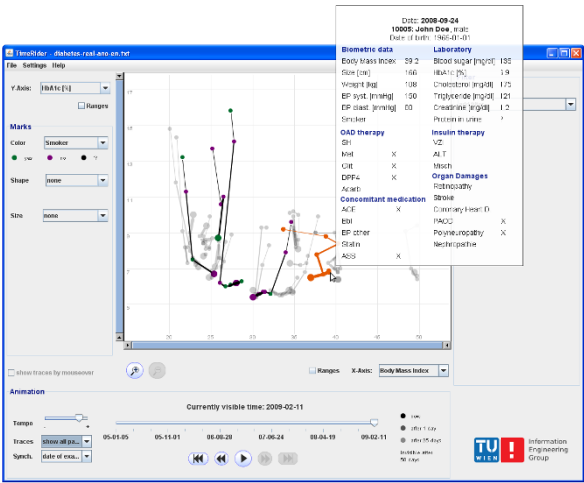


Fig. A.101: Patients are represented as animated marks in a scatter plot. Body mass index is mapped to the horizontal axis, HbA1c to the vertical axis, and mark color shows whether a patient smoked. Time controls for animation and synchronization settings are visible at the bottom. Additionally, traces are displayed that connect values over time. A detail-on-demand window showing further patient data is displayed when hovering over a patient’s mark. © The authors. Generated with the TimeRider software.

TimeRider by Rind et al. (2011a) is an enhanced animated scatter plot (↔ p. 330) for exploring multivariate trends in cohorts of diabetes patients. The enhancements tackle three challenges of medical data: irregular sampling, data wear (i.e., decreasing validity over time), and patient records covering different portions of time. Animation of irregularly sampled data is achieved via interpolation of individual values along a linear trajectory. To account for data wear and to maintain temporal context, transparency and traces are used to enrich the visual encoding of time. For comparing patient histories that cover different portions of time, TimeRider provides four synchronization modes: by calendar date, patient age, start of treatment, and end of treatment. To take better advantage of animation, TimeRider is highly interactive; apart from common interactions to select, pan, zoom, filter, and show details on demand, the user can change the visual mapping of axes, color, shape, and size. Other task-specific features are value ranges that can be highlighted in the background of the scatter plot and dynamic queries on data variables.

Relevant references: Rind et al. (2011a)

Flocking Boids

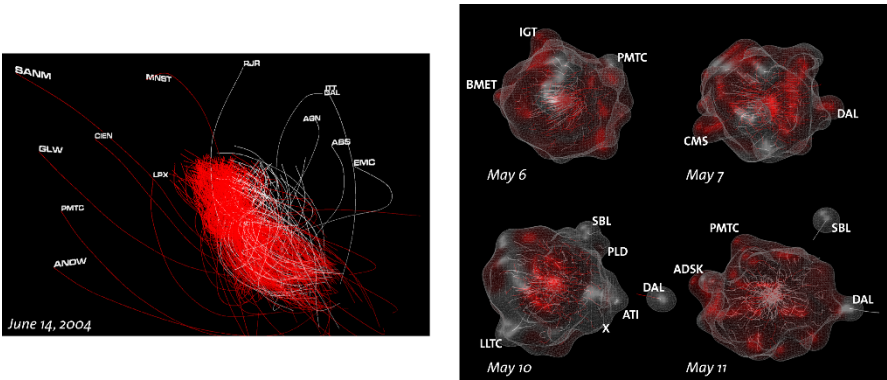


Fig. A.102: Stock market data are represented as flocking boids that move in a three-dimensional presentation space. Left: boids leaving the flock indicate that the corresponding stock price behaves differently than the majority of prices; right: implicit surfaces surrounding boids help users to recognize the spatial structure of the flock. © 2004 IEEE. Reprinted, with permission, from Vande Moere (2004).

Stock market data change dynamically during the day as prices are constantly updated. Vande Moere (2004) proposes to visualize such data by means of information flocking boids. The term boids is borrowed from the simulation of birds (bird objects = boids) in flocks. In order to visualize stock market prices, each stock is considered to be a boid with an initially random position in a 3D presentation space. Upon arrival of new data, boid positions are updated dynamically according to several rules. These rules attempt to avoid collisions of boids, to move boids at the same speed as their neighbors in the flock, to move boids toward the flock’s center, to keep similar boids close to each other, and to let boids stay away from boids that are dissimilar. The visual representation is inherently dynamic and aims at the users’ capability to perceive emergence of patterns as the visualization updates. To this end, boids and corresponding traces are visualized as animated curves, as shown in the left figure. This 3D visual representation is enhanced by enclosing boids within implicit surfaces, which helps users recognize the spatial structure of the flock (right). The flocking boids visualization can be useful for detecting various patterns in the data such as the emergence of clusters, the separation of boids from the main flock, or a general chaotic behavior of boids.

Relevant references: Vande Moere (2004) • Vande Moere and Lau (2007)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

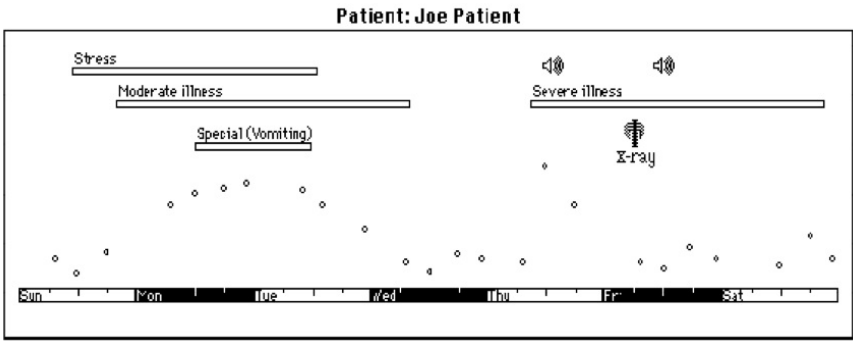
vis

mapping of time: dynamic
dimensionality: 3D

time

Time Line Browser

primitives: points, intervals
arrangement: linear



data

Fig. A.103: Heterogeneous patient information is visualized along a common horizontal time axis. Intervals are displayed as labeled bars and events are displayed as icons. The small circles form a point plot that shows the patient's blood glucose over time. © 1991 Elsevier. Reprinted, with permission, from Cousins and Kahn (1991).

number of variables: multiple
frame of reference: abstract

Cousins and Kahn (1991) developed the time line browser for visualizing heterogeneous time-oriented data. The time line browser integrates qualitative and quantitative data as well as point and interval data into a single coherent view. The time line browser distinguish simple events, complex events, and intervals. Simple events are represented as small circles, whereas complex events are shown as icons. Bars are used to indicate the location and duration of intervals. These depictions are aligned with respect to a common horizontal time axis (\leftrightarrow p. 258), where textual labels might be used to display further details. In addition to the visualization, a formal system for timeline elements and timeline operations has been developed. It defines five basic operations (i.e., slice, filter, overlay, add, new) for manipulating timelines and also supports composite operations. These operations are useful for addressing the issues of different temporal granularities and the calendar mapping problem.

vis

mapping of time: static
dimensionality: 2D

Relevant references: Cousins and Kahn (1991)

PatternFinder

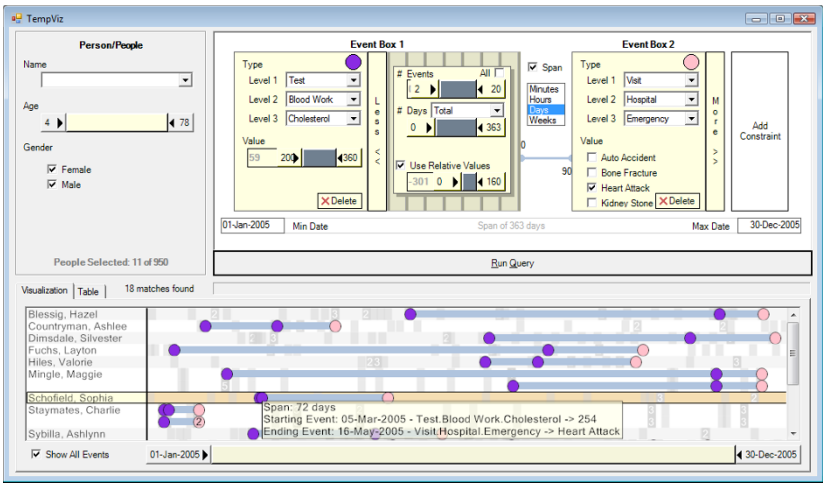


Fig. A.104: The query formulated in the visual interface (top) relates a cholesterol test to a subsequent emergency visit to a hospital. The resulting visualization (bottom) shows several patient records matching the query as vertically stacked ball-and-chain representations. © *Courtesy of Jerry Alan Fails.*

PatternFinder by Fails et al. (2006) can be used for constructing queries to find temporal patterns in medical record databases. The temporal patterns consist of events that are associated with data, and time spans between events. Users formulate queries by imposing constraints on events and time spans. Events can be selected from a hierarchically structured vocabulary, and constraints for associated variables can be specified in a visual interface along with temporal constraints. This way, users can build queries for the existence of events (e.g., persons with heart attack), temporally ordered events (e.g., heart attack followed by stroke), temporally ordered value changes (e.g., BMI of 25 or higher followed by BMI of 20 or lower), and trends over time (e.g., BMI decreasing). Event sequences might not only be specified in terms of temporal order, but also in terms of temporal distance (e.g., time span of 28 days or less between heart attack and stroke). Moreover, all of the mentioned query types can also be combined. For visualizing query results, a so-called ball-and-chain representation is used: results are shown as vertically stacked timelines, where colored circles represent matched events and bars stand for matched time spans.

Relevant references: Fails et al. (2006)

time

primitives: points, intervals
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

time

FacetZoom

primitives: points, intervals
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

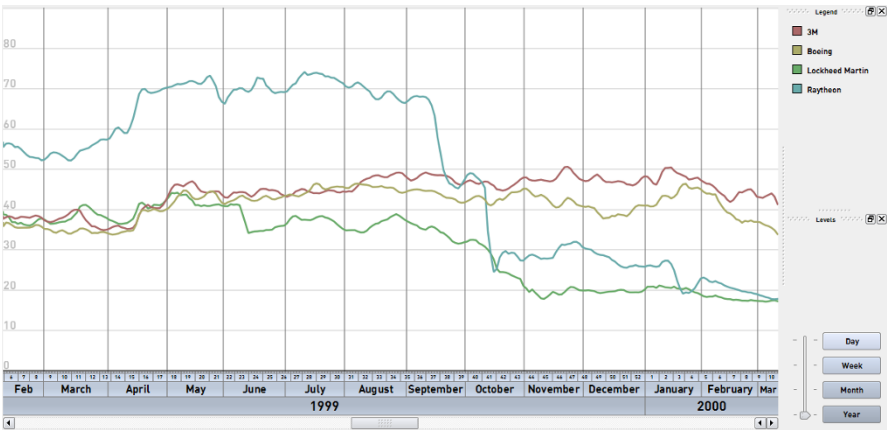


Fig. A.105: The hierarchical structure of time is shown as an interactive horizontal time axis widget that has a data view attached to it, in this case, a visualization of stock market data. © *Courtesy of Raimund Dachzelt.*

FacetZoom is a technique that enables users to navigate hierarchically structured information spaces (see Dachzelt et al., 2008). The hierarchical structure of time is a natural match for this technique. What Dachzelt and Weiland (2006) originally called TimeZoom is a visual navigation aid for time-oriented data. The basic idea is to display a horizontal time axis that represents different levels of temporal granularity as stacked bars (e.g., decades, years, months, weeks, days). The time axis is an interactive widget that can be used to access data from different parts of the time domain at different levels of abstraction. In addition to continuous zooming and panning via mouse, it is also possible to simply select discrete intervals from the time axis. Depending on the user’s selection, the time axis display is altered to accommodate the selected part of the time axis with more display space. Accordingly, the data view, which is attached to the time axis, can use the extra space to represent more data items in greater detail. While the actual mapping of time is static, the navigation steps of the user, including the visual adjustment of the time axis, are smoothly animated.

Relevant references: Dachzelt et al. (2008) • Dachzelt and Weiland (2006)

KNAVE II

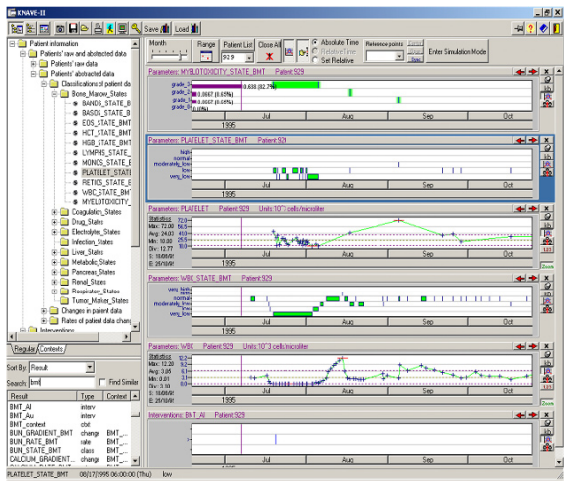


Fig. A.106: KNAVE II supports the visualization and knowledge-based navigation of patient data and abstractions thereof. Left: tree view of clinical domain ontology for navigation; right: visualization panels for raw data and abstractions; bottom left: search panel. © 2006 Elsevier. Reprinted, with permission, from Shahar et al. (2006).

KNAVE II enables visual browsing and exploring of patient’s data (raw measured values and external interventions such as medications). The system focuses mainly on the visual display of temporal abstractions of the data and shows domain-specific concepts and patterns. In order to abstract the raw data, a predefined knowledge base is used that defines three types of interpretations: classification of data (e.g., low–normal–high), change of data (e.g., increasing–decreasing), and rate of change (e.g., slow–fast). Colored timelines (↔ p. 258) depict abstracted intervals as bars where a bar’s vertical position within a panel encodes its qualitative value. For example, the second panel (blue frame) in the figure shows the platelet state abstracted to the qualitative values: very low, low, moderately low, normal, and high (from bottom to top). KNAVE II also allows users to view the raw data as line plots (↔ p. 233). Moreover, statistics for parameters and corresponding abstractions can be superimposed as bar graphs (↔ p. 234) as shown in the topmost panel, or can be given as text labels, as shown in the third and fifth panels. A granularity-based zoom (bottom of each panel) allows users to quickly navigate in time by simply clicking individual granules.

Relevant references: Shahar et al. (2006)

time

primitives: points, intervals
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

Continuum

time

primitives: points, intervals
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

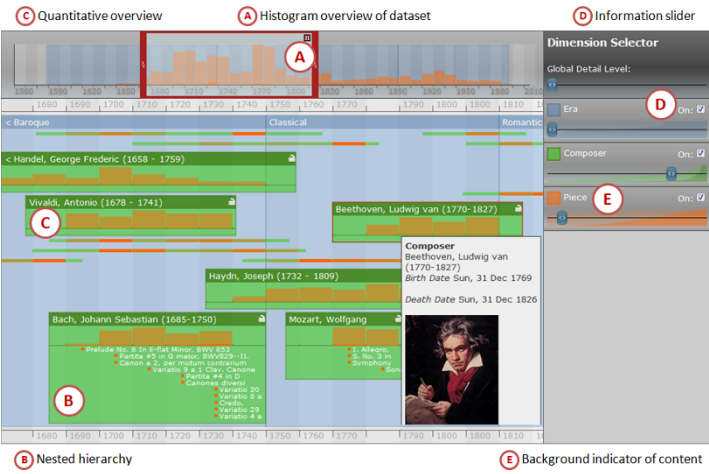


Fig. A.107: Continuum showing a music dataset with the three variables era, composer, and piece. Scalable histograms provide a complete representation of the data and quantify the focal data item. Top left: timeline overview; bottom left: timeline detail view; right: dimension selector panel. © *Courtesy of Paul André.*

Collections of small events often constitute larger, more complex events, like for example talks at conferences or legs of a race. Moreover, events might also be related to other events at other points in time (e.g., a paper written at some point in time and referenced later). Continuum by André et al. (2007) is a timeline visualization tool to represent large amounts of hierarchically structured temporal data and their relationships. It addresses the three problems of scale, hierarchy, and relationships by using scalable histogram overviews, flattening high-dimensional data into dynamically adjustable hierarchies, and arching connection lines for representing non-hierarchical relationships. The interface consists of three main panels that show overview, detail, and dimension configuration. The timeline overview always represents the complete timespan of the dataset using scalable histograms where the vertical axis quantifies the user-selected focal data item. The timeline detail view shows hierarchical relationships as nested elements and applies semantic zooming depending on the amount of information to be displayed. With the dimension selector, users can interactively control the hierarchical buildup and the level of detail to be shown.

Relevant references: André et al. (2007)

VisuExplore

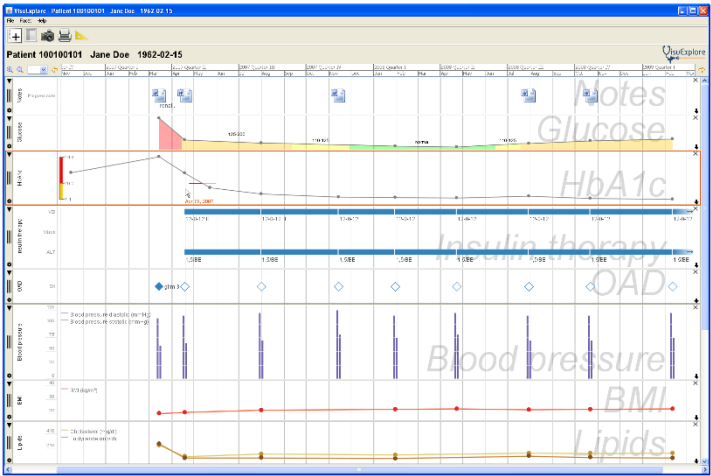


Fig. A.108: Visualization of heterogeneous medical parameters of a diabetes patient. Eight visualization panels show progress notes in a document browser, glucose and HbA1c as line plots with semantic zoom, insulin therapy as timelines, OAD as event chart, blood pressure as bar graphs, and BMI as well as lipids as line plots (top to bottom). © The authors. Generated with the VisuExplore software.

VisuExplore by Rind et al. (2011b) is an interactive visualization system for exploring a heterogeneous set of medical parameters over time. It uses multiple views along a common horizontal time axis to convey the different medical parameters involved. VisuExplore provides an extensible environment of pluggable visualization techniques and its primary visualization techniques are deliberately kept simple to make them easily usable in medical practice: line plots (↔ p. 233), timeline charts (↔ p. 258), bar graphs (↔ p. 234), event charts, line plots with semantic zoom, and document browsers (notes panel on the top). Furthermore, data can also be presented as textual tables to augment the visual representations. VisuExplore’s interactive features allow physicians to get an overview of multiple medical parameters and focus on parts of the data. Users may add, remove, resize, and rearrange visualization views. A measurement tool can be employed to determine the duration of time spans between user-selected points of interest. This works not only within one but also across different views.

Relevant references: Rind et al. (2011b)

time

primitives: points, intervals
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

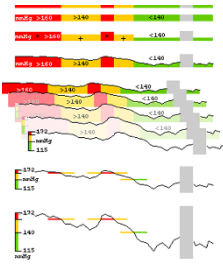
vis

mapping of time: static
dimensionality: 2D

Midgaard

time

primitives: points, intervals
arrangement: linear



data

number of variables: multiple
frame of reference: abstract

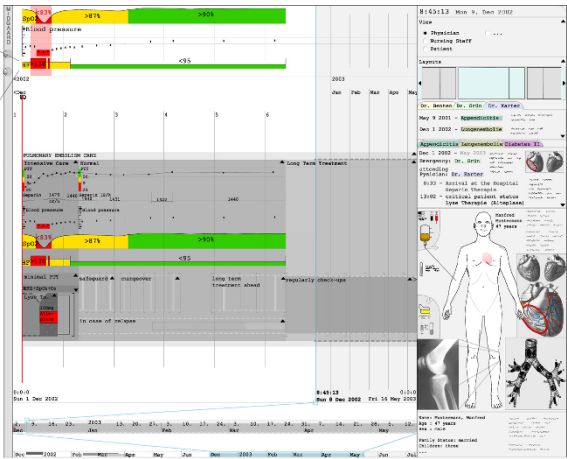


Fig. A.109: Midgaard employs different steps of semantic zooming of a time series from a broad overview (top left) to a detailed view with details of fine structures (bottom left). The main user interface shows different measurements (e.g., blood gas measurements, blood pressure), treatment plans, and additional patient information. © The authors.

Several tightly integrated visualization techniques have been developed in the Midgaard project by Bade et al. (2004) to enhance the understanding of heterogeneous patient data. To support the user in exploring the data and to capture as much qualitative and quantitative information as possible on a limited display space, Midgaard supports different levels of abstractions for time-oriented data (left). Switching between these levels is achieved via a smoothly integrated semantic zoom functionality. These methods were designed to allow users to interact with data and time. Navigation in time is done using three linked time axes (bottom right). The first one (bottom) provides a fixed overview of the underlying time interval covering its full range. Selecting a subrange in that time axis defines the temporal bounds for the main display area and the second time axis (middle). Selecting a further subrange in the middle time axis defines detail and surrounding context areas in time. By interactively adjusting the subranges, users can easily zoom and pan in time.

Relevant references: Bade et al. (2004)

vis

mapping of time: static
dimensionality: 2D

LifeLines

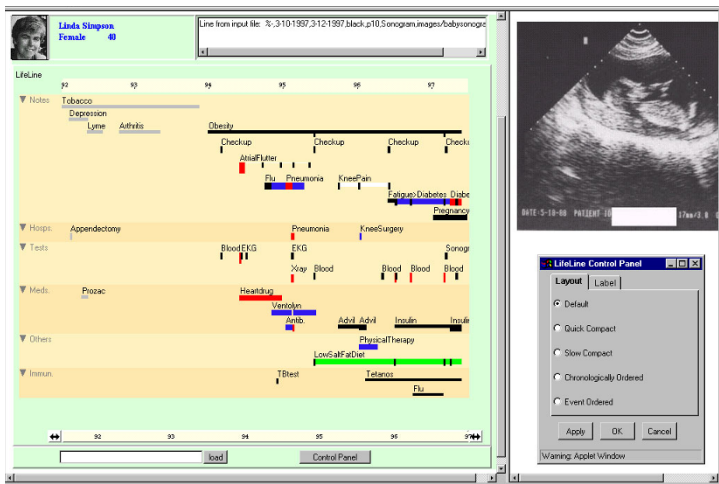


Fig. A.110: Horizontal bars are used to show the temporal location and duration of health-related incidents. The example shows several facets of patient information and an additionally linked sonogram on the right. © Courtesy of Catherine Plaisant and University of Maryland Human-Computer Interaction Lab.

A simple and intuitive way of depicting incidents is by drawing a horizontal line on a time scale for the time span the incident took. This form of visualization is called timeline (\hookrightarrow p. 258). Plaisant et al. (1996) apply and extend this concept for visualizing health-related incidents in personal histories and patient records. Consequently, they call their approach LifeLines. Horizontal bars are used to show the temporal location and duration of incidents, treatments, or rehabilitation. Additional information can be encoded via the height as well as the color of individual bars. In order to structure the displayed information in groups, the so-called facets are introduced. Multiple such facets are stacked vertically. Depending on the information sought by the user, facets can be expanded and collapsed. When collapsed, only a very small, geometrically and semantically reduced visual representation without textual labels is displayed. When expanded, a facet shows full detail. External information related to certain incidents are provided on demand in a linked view, for example, x-ray images or sonograms.

Relevant references: Plaisant et al. (1996) • Plaisant et al. (1998)

time

primitives: points, intervals
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

EventRiver

time

primitives: points, intervals
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

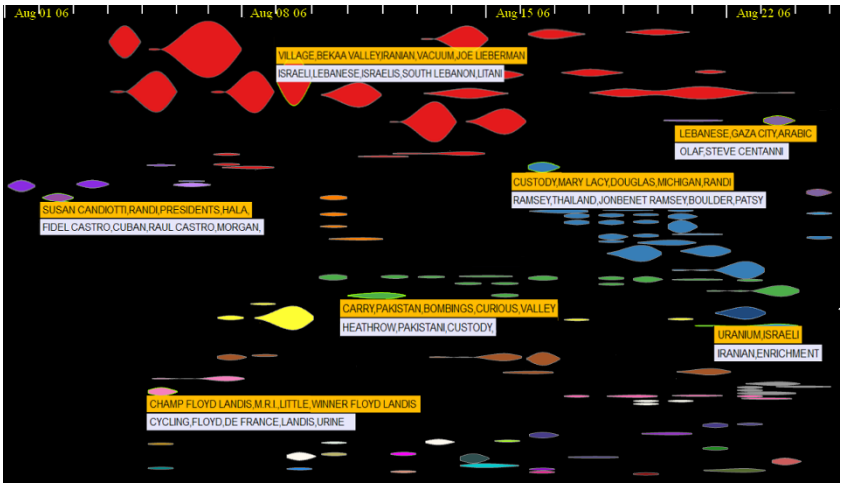


Fig. A.111: EventRiver visualization of CNN news data from August 2006. Event bubbles flow in a horizontal river of time, where important events are highlighted in red in the top part of the river. © 2011 IEEE. Reprinted, with permission, from Luo et al. (2012).

Text collections such as news corpora or email archives often contain temporal references, which embed the text’s information into a temporal context. Luo et al. (2012) describe a technique, called EventRiver, for exploring such text collections interactively in terms of important events and the stories that these events constitute. In the first step, events are extracted from the data using a number of analytical methods, including keyword identification and temporal locality clustering. This analysis yields a set of events that are characterized by their position in time, by their duration, and by several other measures (e.g., temporal influence, strength, and co-strength). The visual design of EventRiver is based on the so-called event bubbles that flow in a horizontal river of time, that is, along a horizontal time axis. The bubbles are placed horizontally where events are located in time. A bubble’s shape illustrates how an event has emerged and disappeared over time. Colors and the bubbles’ vertical positions in the river are chosen so as to highlight important interconnected events that constitute long-term stories in the text documents. While tooltip labels show the important keywords of events, document details are provided on demand in separate views. Analysts can adjust the EventRiver by using various interaction techniques including dynamic filtering, semantic and temporal zooming, and manual relocation of event bubbles.

Relevant references: Luo et al. (2012)

Story Curves



Fig. A.112: Story curves representation of the movie “Pulp Fiction”. The narrative order is mapped to the horizontal axis, whereas the chronological order of events is mapped to the vertical axis. The curve visualizes how the movie’s story unfolds and which events are told at what time. © The authors. Generated with the *Story Curves* software by Nam Wook Kim.

The temporal order in which stories are told and the actual chronological order of the events being told are two distinct things. Movies often make use of this fact. They tell a story deliberately disrupting the chronological order by flashbacks, reversing time, and switching between past, present, and future. Story curves by Kim et al. (2018) is a technique for visualizing such nonlinear narratives. The core idea is to map the narrative order to the horizontal axis and the chronological order of story events to the vertical axis. A curve then visualizes how the story unfolds and which events are told at what point during the movie. The curve can be enhanced with color-coded bars representing additional information, such as the characters appearing in a scene or the location in which a scene takes place. The figure shows the story curve of the movie “Pulp Fiction”. We can see that the initial scene (leftmost cyan bar) is actually almost in the middle of the chronological order of the story. Moreover, we see that the actual first events of the story are only told in the middle of the movie (topmost yellow bar). From the paired bars that run in parallel, we can further learn when key characters of the movie co-occur. With such insights, story curves offer new and interesting perspectives on movies (or other narratives) in terms of how they twist the dimension of time to tell a compelling and fascinating story.

Relevant references: Kim et al. (2018)

time

primitives: points, intervals
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

TextFlow

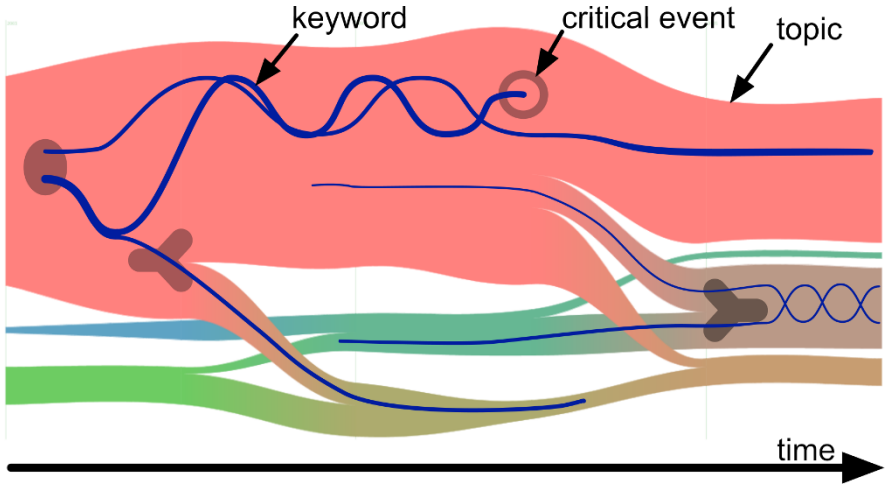


Fig. A.113: The TextFlow technique uses bands of varying widths to represent the evolution of topics over time. Critical events in the evolution are marked with glyphs. So-called threads (blue lines) show details about selected keywords. © Courtesy of Weiwei Cui.

The TextFlow technique by Cui et al. (2011) has been designed to support the analysis of text over time. The general idea is similar to the ThemeRiver (↔ p. 293) approach. Text topics are represented as colored bands whose height varies over time to represent topic importance. Yet, TextFlow also allows bands to split and merge, and to change their vertical positions to represent important changes in the topic evolution. As such, TextFlow bears resemblance to storyline visualizations (↔ p. 264). Split and merge events as well as source (emergence of a topic) and sink (disappearance of a topic) events are additionally marked in the visualization by small glyphs. For a more detailed analysis of topic evolution, so-called threads are used to visualize the co-occurrence of user-selected keywords. This allows users to understand how individual keywords contribute to the formation of topics and their overall relevance in comparison to other topics. The TextFlow technique provides interactive filtering to focus the analysis on selected topics and also provides recommendations on what keywords are worth investigating in detail based on correlation scores.

Relevant references: Cui et al. (2011)

Event-Flow Visualization



Fig. A.114: Event-flow visualization via Outflow. Outflow processes multiple temporal event data and visualizes aggregate event-flow pathways together with associated statistics. This screenshot shows a visualization of Manchester United’s 2010-2011 soccer season. Green and red show pathways with good and bad outcomes (i.e., wins and losses), respectively. © 2012 IEEE. Reprinted, with permission, from Wongsuphasawat and Gotz (2012).

Event-flow visualization is concerned with the question of how linked events literally flow through time. This question is relevant in various domains, ranging from electronic medical records to sports events, where events can be linked to time points, time intervals, or combinations thereof. The general design of event-flow visualizations is similar to Sankey diagrams (↔ p. 313). To overcome visual clutter, methods to reduce edge crossing and straighten unnecessarily curvy edges while preventing overlaps are usually applied. A concrete example of an event-flow visualization is Outflow by Wongsuphasawat and Gotz (2012). It supports the visual analysis of multiple event progression pathways and their associated properties (timing, cardinality, and outcomes). The EventFlow technique by Monroe et al. (2013b) tackles the complexity of point- and interval-based event sequences by providing user-driven data simplification and search techniques to support the users to discover temporal patterns in electronic health records. The DecisionFlow approach by Gotz and Stavropoulos (2014) directly supports the visual analysis of high-dimensional temporal event sequence data with orders-of-magnitude more event types than previously achieved. Various interaction techniques are usually provided, including navigation, simplification, and correlated factor analysis.

Relevant references: Wongsuphasawat and Gotz (2012) • Monroe et al. (2013b) • Gotz and Stavropoulos (2014)

time

primitives: points,intervals
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

time

Co-Bridges

primitives: intervals
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

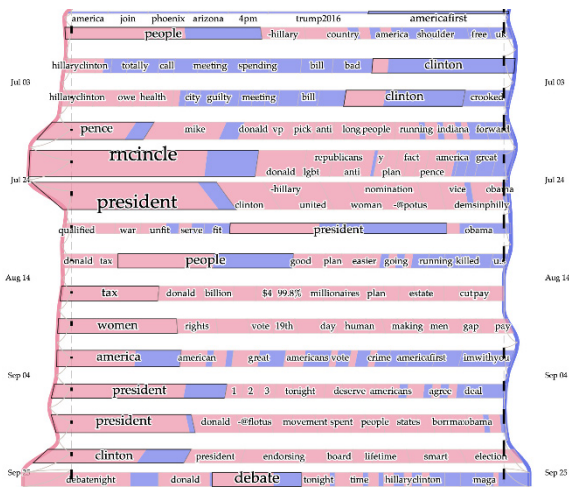


Fig. A.115: The Co-Bridges visualization is based on the metaphor of a river flowing top to bottom, whose river banks represent two social media streams by Hillary Clinton (left, pink) and Donald Trump (right, blue). Bridges cross the river and their segments represent the frequency of keywords in the two data streams. © *Courtesy of Siming Chen.*

The Co-Bridges technique supports the visual comparison of two time-varying data streams with multiple items. Chen et al. (2021) designed Co-Bridges with two metaphors in mind: the river metaphor and the bridge metaphor. The river part of the technique depicts a quantitative characteristic (e.g., stream volume) about the two data streams being compared via two distinctly colored vertical line plots (left and right in the figure), which correspond metaphorically to the river banks. Several bridges cross the river and connect corresponding parts of the two data streams. The bridges' width represents the accumulated frequency of data items. Thin curves at each bridge head mark the time interval covered by a bridge. The bridges themselves are paved with labeled item segments. There is one segment per item, and per segment, the proportion of colors indicates the frequencies of the associated item for the two data streams. The segments are sorted in such a way that they are closer to the river bank whose stream has the higher count of the associated item. Dynamic querying and semantic zooming allow users to refine their analysis and compare data streams in detail. The figure shows an example of Co-Bridges being applied to social media data. One can see, for example, that the item "president" is more frequent in the left stream, which represents posts by former presidential candidate Hillary Clinton.

Relevant references: Chen et al. (2021)

LiveGantt

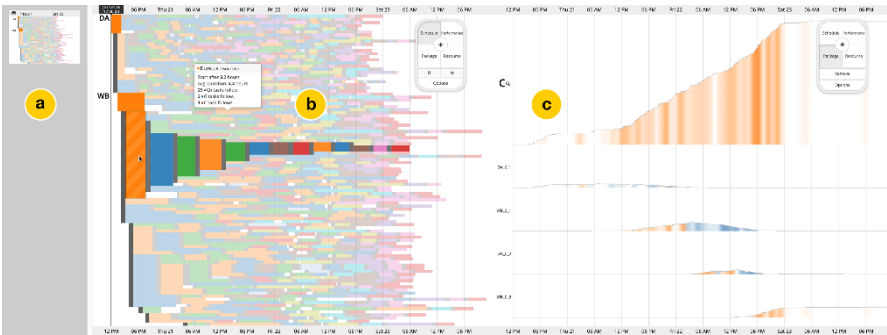


Fig. A.116: Interactive visualization of large manufacturing schedules. The interface shows (a) an exploration history, (b) the schedule view that shows tasks with their durations and orders, as well as (c), a package view that shows changes in packages’ production. The example depicts a schedule containing thousands of tasks on hundreds of machines. © The authors. Generated with the *LiveGantt* demo by Jaemin Jo.

LiveGantt aims for supporting the visual exploration of large manufacturing schedules from different perspectives. It extends traditional Gantt charts (↔ p. 253) which are limited in terms of scalability and interaction support. To achieve a scalability of the method to thousands of tasks and hundreds of machines, time-based task aggregation and resource reordering are applied. The LiveGantt approach consists of five different interactive views. First, the main view is the scheduling view in the center (b) which is based on timeline bars as used in Gantt charts. To tackle scalability, an interactive aggregation algorithm is used. It is controlled interactively by the user selection of a focus time that is shown by a thick black vertical line. Second, the performance view is used to monitor the performance of a schedule over time. It consists of a line plot (↔ p. 233) that shows resource utilization over time which is superimposed over a bar graph (↔ p. 234) that represents concurrent changeovers over time. Third, the resource view contains information about utilization and total changeover time for each resource in bar graphs. In addition, operation start and finish times are displayed in a Gantt-like chart. Fourth, the package view on the right (c) shows the number of completed packages and work-in-progress (WIP). It allows for stepwise drill down by expanding the contents of each package and showing individual charts. For each package, a line plot shows the development of WIPs over time. To emphasize sudden changes, a dual encoding is used by coloring the area below the line depending on the slope of the line plot. Color hue is representing the direction of the slope (orange for upwards and blue for downwards) and saturation represents the steepness of the slope (more saturated colors for steeper slopes). Fifth, the exploration history view on the left (a) visualizes the exploration sequence with thumbnails and allows users to interact with it.

Relevant references: Jo et al. (2014)

time

primitives: intervals
arrangement: linear

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

time

primitives: points
arrangement: linear, cyclic

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

PeopleGarden



Fig. A.117: This PeopleGarden shows about 1200 posts entered into a discussion board during a period of two months. Users are represented by flowers whose petals represent individual messages posted by a user. © 1999 ACM. Reprinted, with permission, from Xiong and Donath (1999).

PeopleGarden is a graphical representation of users’ interaction histories in discussion groups. PeopleGarden visualizes data about users and messages posted to an online interaction environment. It integrates information on the time of posting, amount of response, and whether a post starts a new conversation. For intuitive understanding, PeopleGarden uses the metaphor of a garden of flowers. The garden represents the whole environment and flowers represent individual users within the environment. The petals of a flower stand for the messages posted by a user, the time of posting is mapped to the ordering and saturation of the petals, the amount of response is represented by circles that are stacked on top of petals, and the color is used to depict whether a post starts a new conversation. Furthermore, the height of the flower gives information about how long a specific user has been a member of the discussion group. Using these visual representations, one can easily spot dominant voices, long-time participants, or very active groups.

Relevant references: Xiong and Donath (1999)

PostHistory

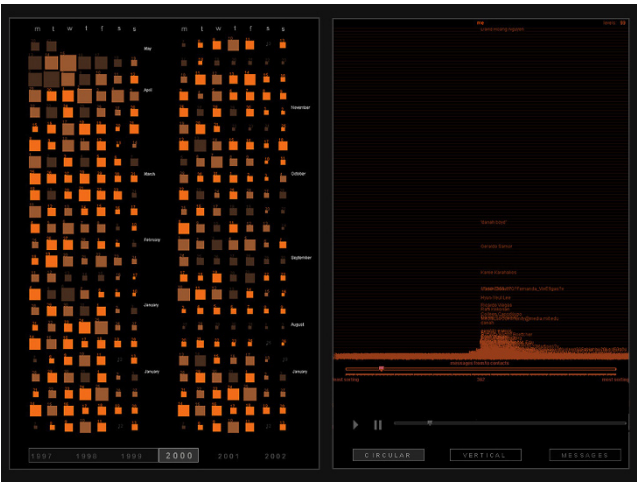


Fig. A.118: The calendar panel on the left shows e-mail activity on a daily basis, where the number of emails and their average directedness are mapped to box size and color, respectively. The contacts panel on the right displays the names of people who sent messages to the user. © *Courtesy of Fernanda B. Viégas.*

PostHistory is a visualization technique for visually uncovering different patterns of e-mail activity (e.g., social networks, e-mail exchange rhythms) and the role of time in these patterns. PostHistory is user-centric and focuses on a single user’s direct interactions with other people through e-mail. The social patterns are derived from analyzing e-mail header information. So, not the content of messages, but the tracked traffic is used as the basis for the analysis of people’s e-mail conversations over time. Basically, the user interface visualizes a full year of e-mail activity and is divided into two main panels: a calendar panel on the left and a contacts panel on the right. The calendar panel shows the intensity of e-mail activity on a daily basis: a square represents a single day and each row of squares represents a week. The size of a square is determined by the quantity of e-mail received on that day and its color represents the average directedness of messages, i.e., whether a mail was received via TO, CC, or BCC. The brighter the color of a square, the more directed the messages are on that day. The contacts panel is used for displaying the names of people who sent messages to the user.

Relevant references: Viégas et al. (2004a)

time

primitives: points
arrangement: linear, cyclic

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

time

Pixel-Oriented Network Visualization

primitives: points
arrangement: linear, cyclic

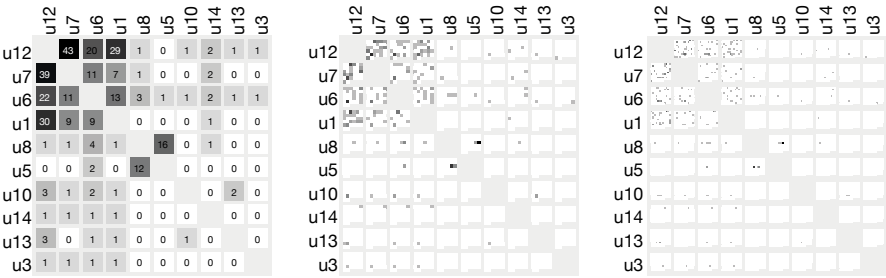


Fig. A.119: Wiki collaboration patterns. Left: adjacency matrix showing users as rows and columns and collaboration intensity by color brightness of cells that connect two users (darker means more collaboration); center: pixel-oriented view where collaboration dynamics are shown in a 6x6 pixel array laid out row by row and each pixel represents a four week period; right: more fine-grained configuration that shows weekly steps. © *Courtesy of Klaus Stein.*

data

number of variables: multiple
frame of reference: abstract

Social networks consist of actors and relationships between them. Unlike most static node-link representations of graph-like structures would suggest, these networks are dynamically changing over time. The two most common forms of visualizing time-varying networks are applying animation to node-link diagrams or applying the concept of small multiples (\leftrightarrow p. 359) by showing snapshots of different points in time. An alternative display is suggested by Stein et al. (2010). They developed a pixel-oriented visualization of networks (PONV) that reveals interaction patterns between actors by integrating pixel-based representations (\leftrightarrow p. 282) within the cells of an adjacency matrix. An adjacency matrix can be represented visually as a matrix table whose rows and columns represent the nodes of the network. The table cell at the intersection of a particular column and row visualizes information about the relationship between the corresponding nodes. The left figure shows an example of an adjacency matrix where the darkness of a cell represents the collaboration intensity of two individuals (the value 0 and a white cell background indicate that there is no relationship between two individuals). The figure in the center uses a different representation to show the temporal evolution of the relationships. Now each cell contains a 6x6 pixel glyph, where each pixel represents the aggregated collaboration intensity of a four-week period. The user can interactively control various parameters, including the color scale, the pixel pattern arrangement, and the time period to be covered by each pixel (e.g., daily values as in the right figure).

vis

mapping of time: static
dimensionality: 2D

Relevant references: Stein et al. (2010)

Parallel Glyphs

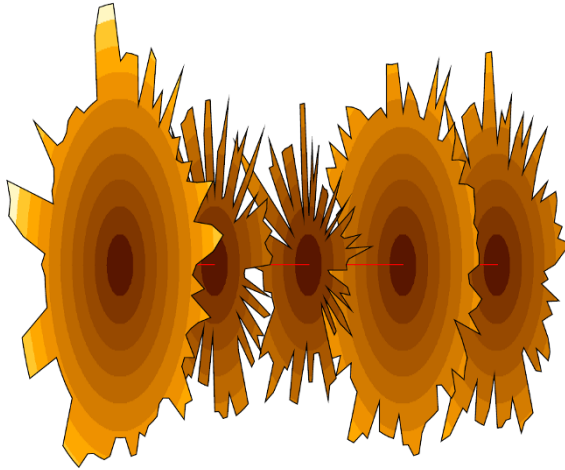


Fig. A.120: Parallel glyphs are used to visualize five different variables over time. Each radial glyph shows a single variable over time where each point on the outside of a glyph corresponds to a data value measured at a specific point in time. Differently colored rings assist in comparing data values. © 2005 IEEE. Reprinted, with permission, from Fanea et al. (2005).

Multivariate time series can be visualized as parallel glyphs. Fanea et al. (2005) synthesized this technique as a combination of parallel coordinates and star glyphs. The visualization uses multiple star glyphs, each of which consists of as many radially arranged spikes as there are time points in the data. The length of a spike corresponds to the data value measured at the spike's associated time point. The tips of subsequent spikes are connected via a polyline, effectively creating a polygonal shape that visualizes the data of one variable in a radial fashion. As an alternative representation, the shape can be filled with differently colored rings (as in the figure) to make the data values easier to compare. Multiple such star glyphs are generated, one for each variable of the dataset. These glyphs are then arranged in the three-dimensional space along a shared axis in a parallel fashion. To assist users in identifying correlations among variables, polylines can be used to connect the same time step along the glyphs. In order to avoid clutter, it is possible to restrict this feature to a user-selected number of time steps. The technique offers various ways of manipulating the display, including switching the role of variables and data records, rotation and zooming in the 3D presentation space, and adjustment of colors.

Relevant references: Fanea et al. (2005)

time

primitives: points
arrangement: linear, cyclic

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 3D

time

Circos

primitives: points, intervals
arrangement: linear, cyclic

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

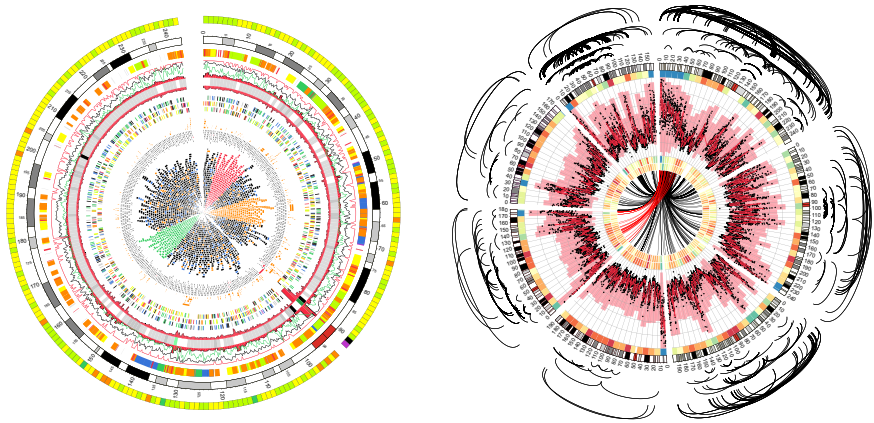


Fig. A.121: Display of multivariate data using data tracks in a radial layout. Images show human chromosome data using point plots, line plots, tiles, histograms, heatmaps, and connectors that link points on the circle. © Courtesy of Martin Krzywinski.

The Circos technique by Krzywinski et al. (2009) uses a circular design to generate multivariate displays. It uses concentric bands (data tracks) as display areas and is capable of displaying data as point plots (↔ p. 232), line plots (↔ p. 233), histograms, heat maps, tiles, connectors, and text. Time is mapped circularly to the circumference of data tracks. The configuration of a visual representation is based on rules that filter and format data elements based on position, value, or previous formatting. Circos was initially developed for genomics and bioinformatics data to visualize alignments, conservation, and intra- and inter-chromosomal relationships. Relationships between pairs of positions are represented by the use of ribbons that connect elements. In the same way, relational data encoded in tabular formats can be shown. Thanks to its flexible approach, Circos has also been applied to numerous other application areas, such as urban planning, and has been used for infographics in newspapers and ads to display complex relationships.

Relevant references: Krzywinski et al. (2009)

Kaleidomaps

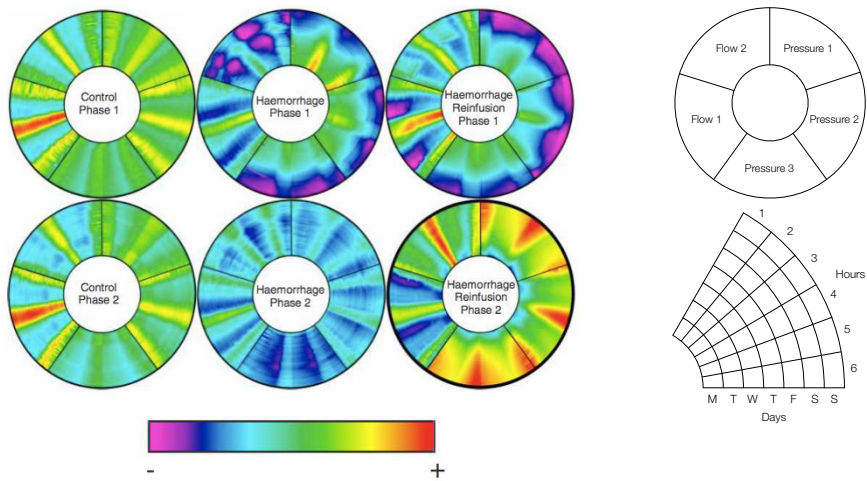


Fig. A.122: Left: six kaleidomaps show the morphology of blood pressure and flow waves over two experimental phases; top right: illustration of the layout of variables within a kaleidomap; bottom right: layout of time within a segment. © 2006 IEEE. Reprinted, with permission, from Bale et al. (2006).

Kaleidomaps visualize multivariate time-series data and the results of wave decomposition techniques using the curvature of a line to alter the detection of possible periodic patterns. The overall idea of kaleidomaps is similar to the rendered output of a kaleidoscope for children, from whence the name comes. A base circle is broken into segments of equal angles for different variables. Each circle segment has two axes representing time, one along the radius and one along the arc of the segment. The data values and categories are represented using color. Due to the circular nature of kaleidomaps, the number of variables in one circle is limited to a maximum of six to eight. Interaction techniques within the kaleidomaps allow an analyst to drill down both in time and frequency domains in order to uncover potential relationships between time, space, and waveform morphologies. Kaleidomaps were developed in the domain of critical care medicine, but case studies have shown their usefulness in other domains as well, like in environment analysis.

Relevant references: Bale et al. (2006) • Bale et al. (2007)

time

primitives: points
arrangement: cyclic

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

KAVAGait



Fig. A.123: KAVAGait’s user interface, which combines three main components: (1) the explicit knowledge store (EKS) capturing an overview of the stored gait patterns, (2) the patient explorer combining patient’s information and different visualizations of the ground reaction force, and (3) the parameter explore visualizing the 16 calculated spatio-temporal parameters of the patient to be explored in connection with various statistical measures. © Wagner et al. (2019).

Understanding the patient’s gait performance is critical to provide appropriate diagnoses and treatment planning. KAVAGait by Wagner et al. (2019), which was designed in close cooperation with domain experts, provides a visual exploration environment incorporating principles of knowledge-assisted visual analytics to support clinicians inspecting complex data derived during clinical gait analysis. KAVAGait combines well-known visualization techniques data- and tasks-specifically, like interactive twin box plots to analyze the current patient with various statistical measures of the corresponding patient cohorts. Various interaction concepts as well as the modeling of explicit knowledge about various gait patterns ease the overall exploration process. The effectiveness of KAVAGait was validated by moderated expert reviews, user studies, and a case study with a national expert.

Relevant references: Wagner et al. (2019)

SentiCompass

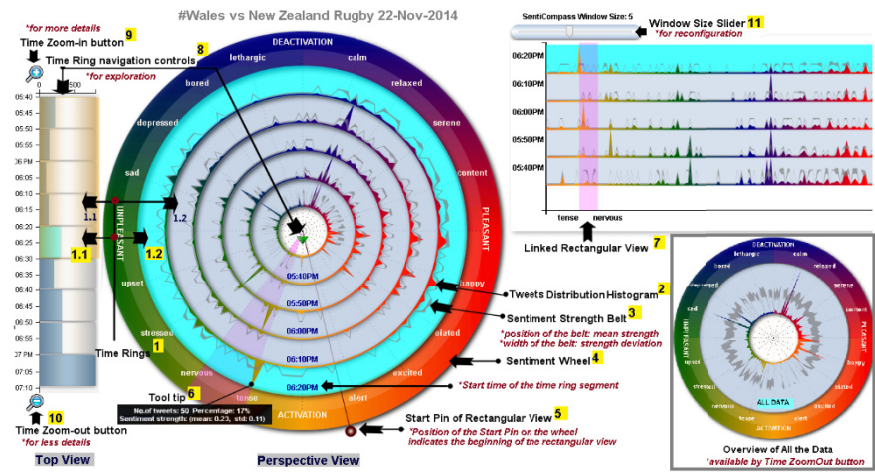


Fig. A.124: SentiCompass is used for exploring and comparing sentiments of Twitter data over time. The emotional spectrum of the sentiments is displayed on the outermost, colored ring. © 2015 IEEE. Reprinted, with permission, from Wang et al. (2015).

SentiCompass collects tweets of specific events, such as sports events or elections, and performs text mining for sentiment analysis in combination with supervised classification along an affective dictionary. Thus, the approach does not limit sentiment to a one-dimensional variable but expands it to a 2D sentiment spectrum. The positive ends of the sentiment spectrum are pleasantness and activation, while the negative ends are deactivation and unpleasantness. SentiCompass contains three different views: a top view, a perspective view, and a linked rectangular view. For the main visual metaphor, the perspective view (center), time is shown as a cylindrical tunnel cylindrical tunnel (↪ p. 284), with different time periods represented by time rings with time advancing from the inside to the outside. The emotional spectrum of the sentiments is displayed on the outermost, colored ring. The distribution of the number of tweets is shown as circular histograms along the different sentiments within each ring. In addition, each ring contains a sentiment strengths belt, i.e., the mean and standard deviation of sentiment strengths as band graph (↪ p. 233) (line plot with varying line widths). As an alternative to the circular perspective view, a linked rectangular view can be shown (top right) which shows the same data in a linear, rectangular fashion. The top view (left) is used as an overview and navigation element for the perspective view. It splits a time interval into slices of same lengths that are displayed below each other. Each slice is proportionally filled to represent the total number of tweets collected in the respective time interval.

Relevant references: Wang et al. (2015)

time

primitives: points
arrangement: cyclic

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

TreeRose

time

primitives: points
arrangement: cyclic

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

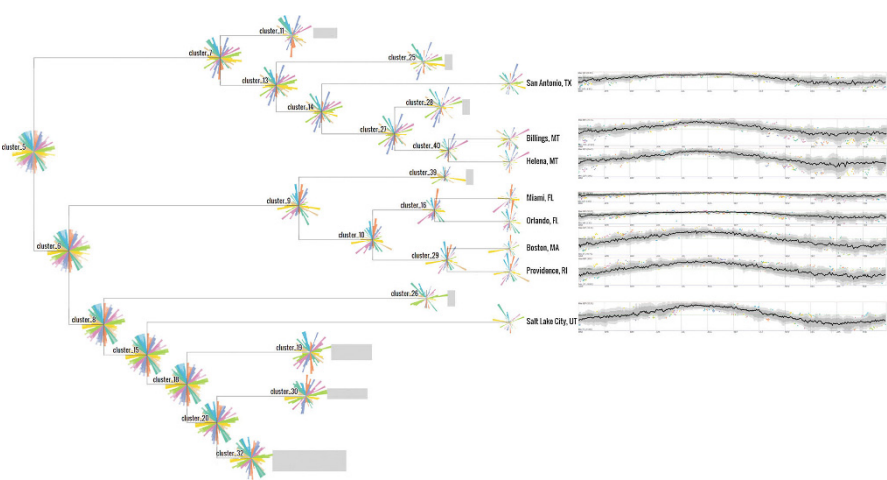


Fig. A.125: TreeRoses representing 15-year temperature data of major US cities. On the left-hand side, collapsible dendrogram clusters group cities with similar anomalous temperatures. The wind roses summarize the anomaly distributions and the gray rectangles beside the wind roses represent the depth and width of collapsed branches. On the right-hand side, the band curves visualize detailed temporal patterns and outliers. © 2019 Springer. Reprinted, with permission, from Tang et al. (2019a).

Outlier detection in cyclic time series is an important, but challenging task. TreeRose by Tang et al. (2019a) is a visual analytics framework, which transforms a large number of multivariate time series into manageable visual representations that enable analysts to identify and monitor anomalies and trends. The time series are expected to be periodical with a known period (e.g., yearly, daily). Wind roses are used to represent the anomalies, which are glyph-based summaries of outliers for each time series. Hierarchical clustering is applied to group time series with similar anomalies (dendrogram clusters). The applicability and usability of TreeRose are illustrated using weather and building datasets as use cases.

Relevant references: Tang et al. (2019a)

Intrusion Detection

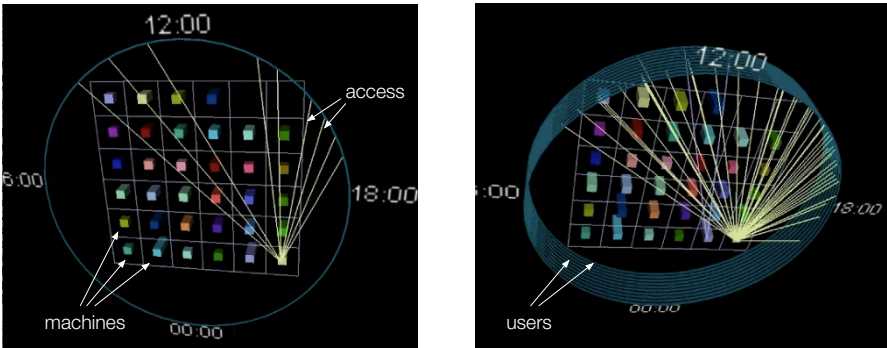


Fig. A.126: Machines in a network are represented by a matrix of 3D cubes in the center of the display. Time is mapped to the circumference of a circle enclosing the matrix of machines. When a particular machine is accessed, a line is drawn that links the particular machine with a point in time on the circular time axis. © Courtesy of Kovalan Muniandy.

A 3D visualization technique by Muniandy (2001) helps administrators analyze user access to computers in a network over time for intrusion detection. The different parameters time, users, machines, and access are mapped onto a 3D cylinder. In the figure, time is mapped onto the circumference of a circle showing the 24 hours of a day. The units along the circle can be configured to represent either hours, months, or years. Different users are represented by individual cylinder slices that are stacked upon each other and machines are represented as cubes that are arranged in a matrix. Access to a machine by a user is visualized by a line connecting the user slice at the corresponding access time with the accessed machine. This way, certain patterns of network access can easily be spotted visually and suspicious behavior can be revealed. Details are displayed when hovering with the mouse over an element of the visualization. To mitigate occlusion, the representation can be zoomed and rotated freely by the user. Moreover, filtering can be applied to remove clutter.

Relevant references: Muniandy (2001)

time

primitives: points
arrangement: cyclic

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 3D

time

KronoMiner

primitives: points, intervals
arrangement: cyclic

data

number of variables: multiple
frame of reference: abstract

vis

mapping of time: static
dimensionality: 2D

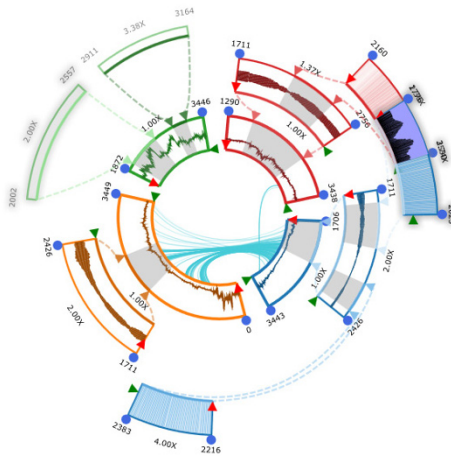


Fig. A.127: KronoMiner shows the original data, four variables colored in red, blue, orange, and green, as radial line plots in the central ring and subintervals of interest on the outer rings for closer inspection. © Courtesy of Jian Zhao.

KronoMiner by Zhao et al. (2011a) is a multipurpose time-series exploration tool providing rich navigation capabilities and analytical support. It aims to make the analysis of selected regions of interest easier. To this end, the central view of KronoMiner is based on a hierarchical radial layout of line plots showing the raw data in the center and focused intervals in the periphery, allowing users to drill down into different parts of the data at different scales. The central ring contains all data, four time-dependent variables colored in red, blue, orange, and green in the figure. From there, users can select focus intervals, which are then extracted and shown in the outer rings. The focused intervals can be rotated, dragged, stretched or shrunk, supporting various kinds of time-series analysis and exploration tasks. KronoMiner also introduces two analytical techniques. One is a MagicAnalytics Lens, which shows the correlations between two focus intervals when overlapped. The second is the Best Match mode in which an arch shape is displayed indicating the matching parts of two intervals under a specific similarity measure.

Relevant references: Zhao et al. (2011a)

A.3 Techniques Supporting a Spatial Frame of Reference

Small Multiples

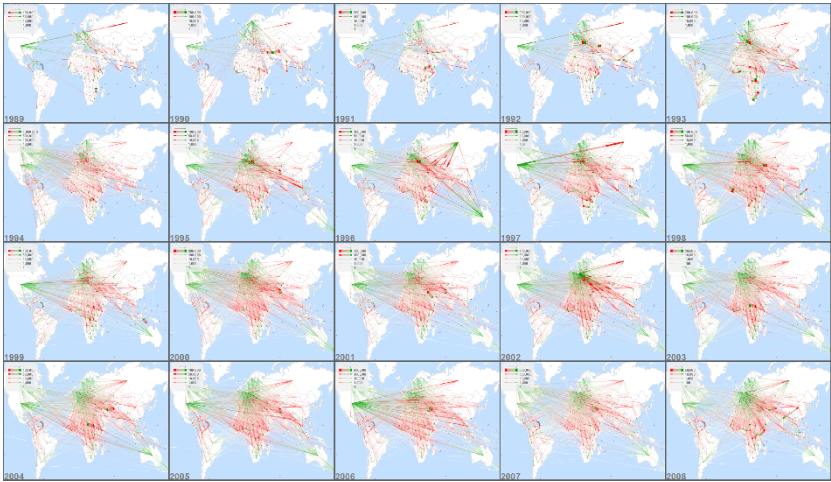


Fig. A.128: Small multiples showing migration data. Each miniature map visualizes the migration of people as links between countries. Links start in red to mark the origin of the migration, they end in green at the destination, and their line width indicates the volume of the migration. © The authors. Generated with the *JFlowMap* software by Ilya Boyandin.

Small multiples are more of a general concept than a specific technique. They are described as sets of miniature visual representations (see Tufte, 1983; Tufte, 1990). For time-oriented data, each miniature visualizes a selected time point. The concrete depiction may show a single variable or multiple variables in an abstract or spatial context using a 2D or 3D presentation space. Particularly relevant is the arrangement of the small multiples as it dictates how the time axis is perceived. Linear or circular arrangements can be used, or specific arrangement patterns can be applied to account for different granularities of the time axes. Small multiples provide an overview of the data and allow users to visually compare the data at different time points. Another advantage of small multiples is that the concept can be applied to virtually any existing visualization technique. The only thing to do is to create a thumbnail from an existing visual representation for each time step. Depending on the amount of screen space occupied by each thumbnail, however, the number of representable time steps could be rather small. Or, if the images are shrunk to fit more time steps, fewer details are visible.

Relevant references: Tufte (1983) • Tufte (1990)

time

primitives: points
arrangement: linear, cyclic

data

number of variables: single, multiple
frame of reference: abstract, spatial

vis

mapping of time: static
dimensionality: 2D, 3D

time

EventViewer

primitives: points, intervals
arrangement: linear, cyclic



Fig. A.129: Visual exploration of event data via the EventViewer. Spatial, temporal, and thematic dimensions of events can be assigned to configurations of bands, stacks, and panels. Left: low pressure and high wind events are shown along three locations and two years; right: display configuration to reveal temporal patterns along hours of a day. © The authors. Adapted from Beard et al. (2008).

data

number of variables: single, multiple
frame of reference: abstract, spatial

The EventViewer by Beard et al. (2008) is a framework that has been developed to visualize and explore spatial, temporal, and thematic dimensions of sensor data. The system supports queries on events that have been extracted from such data and are stored in an events database. The spatial, temporal, and thematic categories of selected events can flexibly be assigned to three kinds of nested display elements called bands, stacks, and panels. Bands are the primary graphic object and act as display containers for a set of events. The horizontal dimension of a band represents time and bars within a band represent instances of events. The length of a bar corresponds to the event’s duration and color can be used to encode other data values. Furthermore, missing data are shown by using gray bars to make a clear visual distinction to areas without events (shown as empty areas). Stacks consist of event bands that are placed on top of each other and panels are collections of stacks. Each of the three data dimensions space, time, and theme can be modeled along hierarchies or lattices. For time, calendric systems consisting of time granularities like hours, days, weeks, and years are used. The configuration of display elements is broken down along these hierarchical and lattice structures and form small multiples (↔ p. 359). The assignments can be changed interactively by the user via direct manipulation, thus revealing different kind of patterns, for example, periodic patterns, spatial and temporal trends, or event-event relationships.

vis

mapping of time: static
dimensionality: 2D

Relevant references: Beard et al. (2008)

Ring Maps

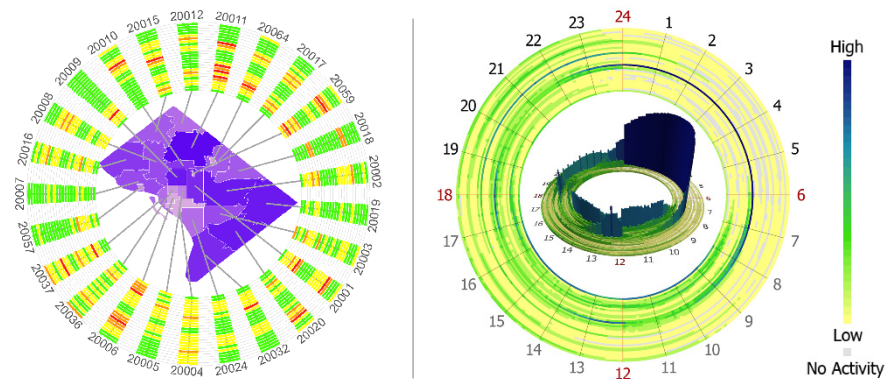


Fig. A.130: Ring maps representing health-related alert levels (green, yellow, orange, red) for various zip code regions during a period of 24 weeks (left) and degree of activity during the course of a day for 96 human activities, one shown per ring (right). *Left: © Courtesy of Guilan Huang. Right: © Courtesy of Jinfeng Zhao.*

The basic idea of ring maps is to create multiple differently sized rings, each of which is subdivided into an equal number of ring segments (see Zhao et al., 2008b; Huang et al., 2008). The rings and their segments as well as the center area of the overall visual representation can be used in various ways. One can utilize ring maps to visualize spatio-temporal data. To this end, a map is shown in the center and the ring segments of a particular angle are associated with a specific area of the map. This is depicted in the left figure, where different angles show the data for different zip code regions. A time series for each region can then be represented by the rings, for instance, by assigning the first series entry to the innermost ring and the last one to the outermost ring. The actual data visualization is done by color-coding. There are other ways of mapping information to rings and segments. The right figure shows an application of ring maps where the hours of the day are mapped to the ring segments and the rings represent different activities a person can be busy with during the course of a day. The degree of activity is encoded by color. This time the center of the display is used to show a complementary 3D representation of the same data to assist users in spotting highly active regions.

Relevant references: Zhao et al. (2008b) • Huang et al. (2008)

time

primitives: points, intervals
arrangement: linear, cyclic

data

number of variables: single, multiple
frame of reference: abstract, spatial

vis

mapping of time: static
dimensionality: 2D, 3D

ThermalPlot

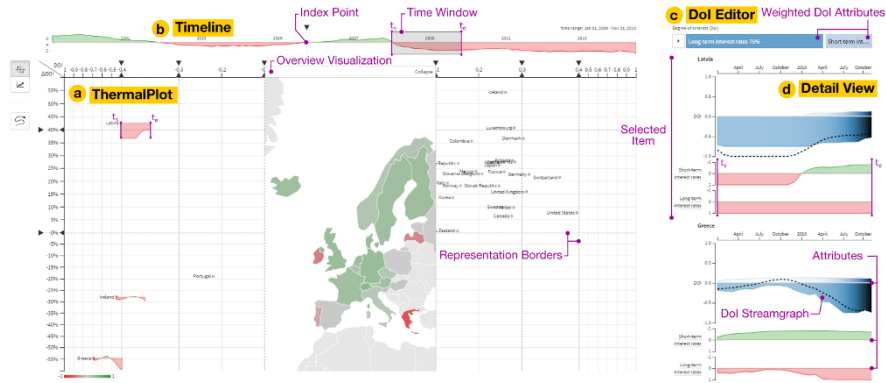


Fig. A.131: ThermalPlot summarizes combinations of multiple attributes over time using user-defined degree-of-interest (DoI) functions. The example shows the development of short-term and long-term interest rates for OECD countries between January 2000 and July 2015. © The authors. Generated with the ThermalPlot demo by Holger Stitz and Samuel Gratzl.

Visual exploration of large and multivariate time-series datasets to identify interesting items and patterns is an important analysis task in many application domains. However, this poses significant challenges to users as particularly the identification of important attributes and comparison of temporal developments among large numbers of items are complex tasks. Stitz et al. (2016) tackle this challenge in their ThermalPlot technique by mapping a user-specified degree-of-interest (DoI) value on the x-axis and the change of the DoI value (ΔDoI) on the y-axis. That is, the DoI and its first derivative are plotted against each other. Moreover, the DoI value is composed of a weighted combination of one or multiple attributes over time. Users can interactively compose the resulting DoI value and adjust individual weights to reflect the relevance of each component. The interface consists of three main parts: the ThermalPlot view (a) depicting the development of DoI vs. ΔDoI relative to a selected index point including a multi-level display, the overview timeline (b) which shows the overall DoI over time using a silhouette graph (\hookrightarrow p. 281), and the DoI editor (c) that uses silhouette graphs and streamgraphs (\hookrightarrow p. 286) to visualize the contribution of each component to the overall DoI. For high absolute or relative DoI ranges in the ThermalPlot that contain potentially more interesting items, a more detailed representation based on silhouette graphs is used. Toward the center of the representation, less details are shown (label + point or point only) and optionally, an overview representation that summarizes the less interesting items according to the DoI specification might be displayed (e.g., a choropleth map for spatial data or a treemap for abstract data).

Relevant references: Stitz et al. (2016)

Circular

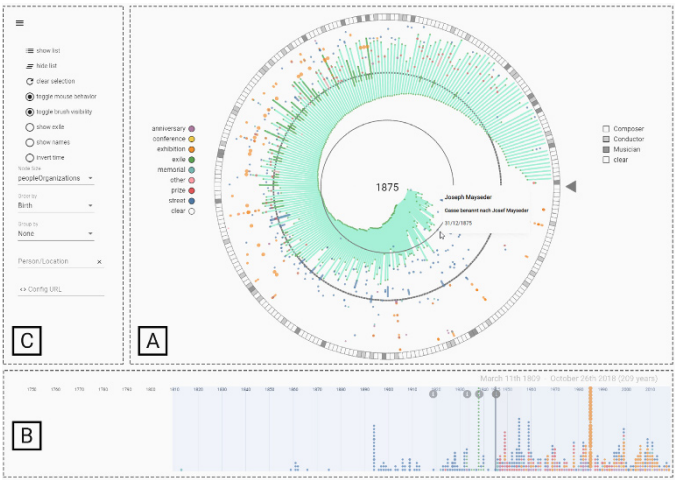


Fig. A.132: An overview of Circular. The main view (A) represents people, locations, events, and thematic changes by means of radially arranged rays. The rays are ordered according to user-selected criteria, in this case, by date of birth. People of interest in this dataset have many events occurring after their death. A timeline (B) shows historic events in a temporally ordered view. Several controls (C) allow users to modify the visualization. © [Filipov et al. \(2021\)](#).

Circular by Filipov et al. (2021) is a technique to visualize a large amount of linked historical events and entities that are embedded in spatial and temporal frames of reference. Circular depicts entities as rays that are arranged in a radial fashion and sorted depending on a user-selected data attribute. Along the rays, one can see the events related to the temporal development of an entity represented as colored dots. The colors represent different types of event themes. On the circle’s exterior, a selected categorical attribute can be visualized. While the data are based on time points, they can also be grouped according to a particular context, like artists’ lifelines and related events. Circular provides a straightforward and engaging interface to present and get an overview of the data. Various interaction techniques support data exploration. The data can be sliced in various ways based on analysis tasks and user interests. The grouping and ordering can be changed to allow users to extract different insights from the visualization. Domain experts from the digital humanities tested Circular with data containing historical events and entities related to public music festivities in Vienna. Through the visualization they could better understand their data and answer questions like “Can you name exiled musicians that do not have a street in the city named after them?”, “Which location was used most often for political stagings during the Second Republic?”, or “Which exiled musicians never returned home?”.

Relevant references: Filipov et al. (2021)

time

primitives: points, intervals
arrangement: linear

data

number of variables: multiple
frame of reference: abstract, spatial

vis

mapping of time: static
dimensionality: 2D

ViDX

time

primitives: points, intervals
arrangement: linear, cyclic

data

number of variables: multiple
frame of reference: abstract, spatial

vis

mapping of time: static, dynamic
dimensionality: 2D, 3D

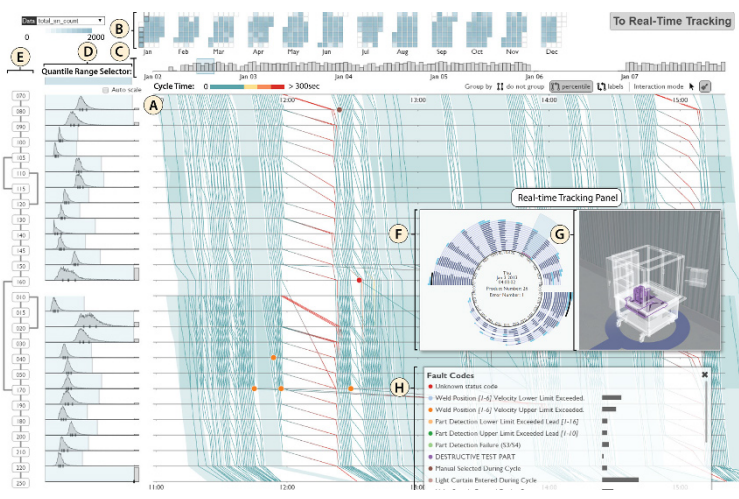


Fig. A.133: The ViDX system combines several views (e.g., Marey’s graph, calendar view, bar chart, histograms, and radial graph) to facilitate the visual analysis of assembly lines. © 2017 IEEE. Reprinted, with permission, from Xu et al. (2017).

The ViDX system by Xu et al. (2017) is a comprehensive solution for visually analyzing and monitoring assembly line performance in smart factories. As such, the system is related to the Industry 4.0 trend, which is concerned with increasing interconnectivity and smart automation. ViDX is based on the concept of coordinated multiple views (CMV) according to which the visual representation of complex data is divided into several dedicated views that are tightly linked via interaction mechanisms. The figure shows several such linked views, including a calendar view (↔ p. 269) and a bar graph (↔ p. 234) timeline at the top, a Marey’s graph (center) and histograms and the assembly line schema (right). These views show historical data about the assembly line, its efficiency, and failure rates. For example, the Marey’s graph shows the assembly line’s stations along the vertical axis and time along the horizontal axis. Vertical lines in the Marey’s graph represent products being processed simultaneously on the assembly line. A smoothly operating assembly line can be recognized by the lines forming parallel patterns without any gaps. Failures, interruptions, and delays are visible as line crossings and gaps in Marey’s graph. Further information about such problems is available in a separate on-demand view. In addition to supporting the investigation of historical data, ViDX also allows users to monitor the current state of the assembly line. For this purpose, ViDX offers a radial graph and a 3D representation of the assembly line (center). While the radial graph represents the current status of all the products being processed at the stations of the assembly line, the 3D view provides the spatial context necessary to understand the overall operation of the assembly line.

Relevant references: Xu et al. (2017)

Value Flow Map

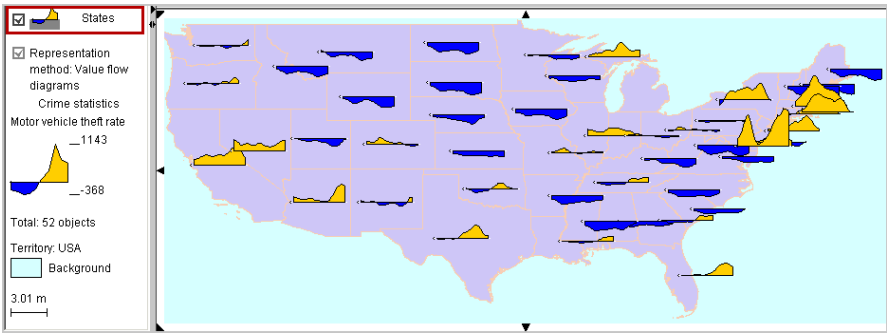


Fig. A.134: Univariate spatio-temporal data are represented by embedding multiple miniature silhouette graphs into regions of a map. The graphs use a specific encoding where the yellow color corresponds to a positive deviation of the variable from the data’s mean and the blue color indicates a negative deviation. © *Courtesy of Gennady Andrienko.*

What Andrienko and Andrienko (2004) call value flow map is a technique to visualize variation in spatio-temporal data. A value flow map is an instance of the icons on maps approach (\hookrightarrow p. 379), where a value flow map shows miniature silhouette graphs (\hookrightarrow p. 281) for each map area to represent the temporal behavior of one data variable per area. Typically, temporal smoothing is carried out by replacing the values of a point-based time scale with the mean values of an interval-based time scale. In this way, small fluctuations are disregarded and major trends become visible. Moreover, a number of data transformations can be applied to define the mapping of the graphs. An example of such a transformation is to show the variable’s deviation from the data’s mean, rather than the raw data, that is, data values are replaced by their differences from the mean in order to represent positive and negative variations. This way, the silhouette graphs visualize quite well how the data values flow in time and space (hence the value flow map). This is a necessary requirement to enable analysts to detect patterns, and thus to support exploring spatial distributions, comparing data evolution at different locations, as well as finding similarities and outliers.

Relevant references: Andrienko and Andrienko (2004)

time

primitives: points
arrangement: linear

data

number of variables: single
frame of reference: spatial

vis

mapping of time: static
dimensionality: 2D

time

primitives: points
arrangement: linear

data

number of variables: single
frame of reference: spatial

vis

mapping of time: static
dimensionality: 2D

Flowstrates

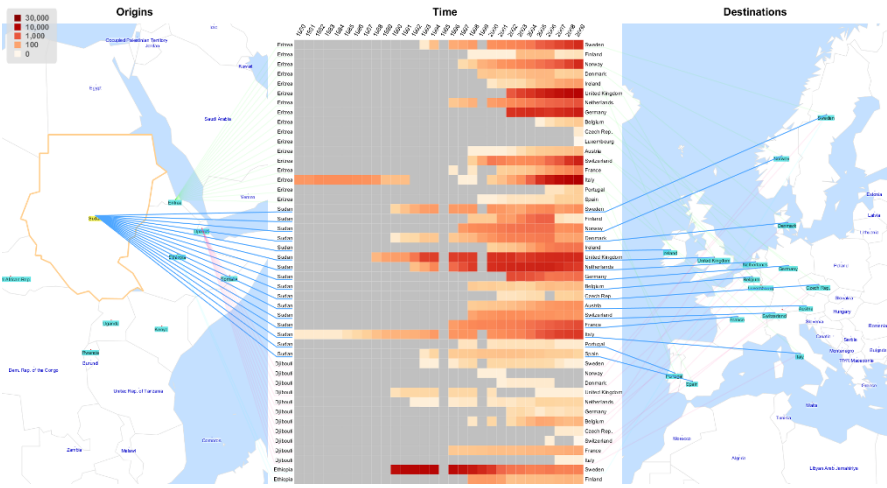


Fig. A.135: Flowstrates visualization showing migration of refugees from eastern Africa to European countries. Currently highlighted is the flow of refugees from Sudan. © *Courtesy of Ilya Boyandin.*

Flowstrates by Boyandin et al. (2011) extend the idea of flow maps (↔ p. 371) to the temporal dimension and allow the user to analyze changes in the flow magnitudes over time. In Flowstrates, the origins and the destinations of the flows are displayed in two separate maps, and the temporal changes of the flow magnitudes are displayed between the two maps in a heatmap in which the columns represent time periods. As in most flow maps that focus on representing the flow magnitudes, the exact routes of the flows are not accurately represented in Flowstrates. Instead, the flow lines are rerouted so that they connect the flow origins and destinations with the corresponding rows of the heatmap as if the flows were going through it. The flow lines help viewers see in the geographic maps the origin and the destination corresponding to each of the heatmap rows. To allow users to explore the whole data in every bit of detail, Flowstrates provide interactive support for performing spatial visual queries, focusing on different regions of interest for the origins and destinations, zooming and panning, sorting and aggregating the heatmap rows.

Relevant references: Boyandin et al. (2011)

Traffigram

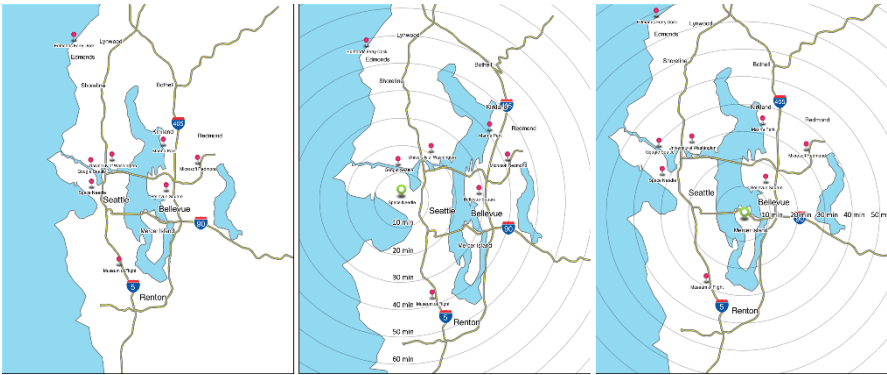


Fig. A.136: Regular maps (left) show spatial distances. The Traffigram distorts a map so that distances correspond to travel times from a selected origin. The distorted map in the center shows the Seattle Space Needle as the origin, the example to the right has the origin set to Mercer Island. © Courtesy of Sungsoo (Ray) Hong.

Traffigram is a technique for visualizing time, more concretely the time it takes to travel from a selected origin in space to other locations on the map. This technique is also called distance cartogram. Hong et al. (2014) designed their Traffigram as a distortion-based approach where a map is distorted so that distances between one origin to the rest of the points on the map correspond to travel time. The first step is to pinpoint a user-selected location with respect to which the Traffigram is to be generated. Secondly, isochronal contours are computed. These contours connect points in the map that are reachable from the focus location in the same amount of time. In other words, traveling from the focus point to any point on an isochronal contour takes constant time. The third step in generating a Traffigram is to distort the map by warping. Hong et al. (2014) use the thin-plate spline (TSP) algorithm for this purpose. The effect of this warping is that the isochronal contours become a circular shape so that readers can visually compare travel time between multiple locations at a glance. Two such distorted maps (center) and (right) and the original undistorted map (left) are shown in the figure. It can easily be seen from the figures that much of the distortion takes place in west/east directions away from the major highways, which go in north/south directions. So traveling in areas further away from highways takes more time. More recent work related to distance cartograms includes the design of an advanced visualization technique for creating the outcome without introducing excessive distortion that can hamper readers' understanding of geographical relationships (see Hong et al., 2017) and understanding the user-side effect of using distance cartograms in a more realistic situation through a two-weeks deployment study (see Hong et al., 2018).

Relevant references: Hong et al. (2014) • Hong et al. (2017) • Hong et al. (2018)

time
primitives: points
arrangement: linear
data
number of variables: single
frame of reference: spatial
vis
mapping of time: static
dimensionality: 2D

time

primitives: points
arrangement: linear

Time-Varying Hierarchies on Maps

data

number of variables: single
frame of reference: spatial

vis

mapping of time: static
dimensionality: 3D

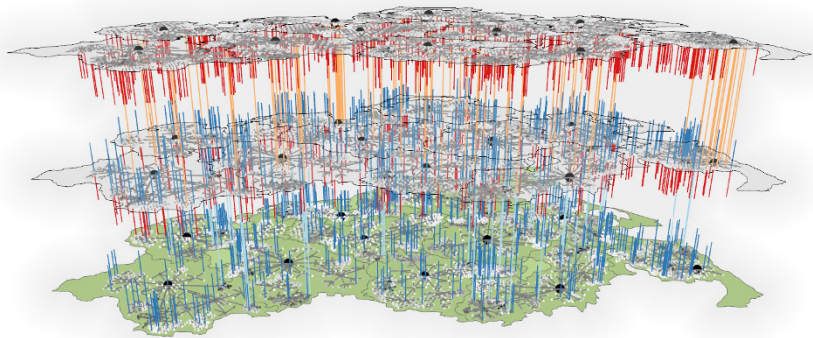


Fig. A.137: Hierarchy layouts are embedded into areas of a map, where each map layer corresponds to one time step. Colored links and spikes between layers indicate significant changes from one time step to the other. © ⓘ The authors. Generated with the LandVis system by Christian Tominski.

Hierarchical structures can be found in many application areas. A technique for visualizing hierarchies that change over time in a geo-spatial context is described by Hadlak et al. (2010). This technique follows the idea of using the third dimension of the presentation space to represent the dimension of time, which is analog to the space-time cube approach (↔ p. 377). For a series of time steps, individual map layers are constructed, where each map region shows an embedded hierarchy layout and each node's color visualizes a data value. To facilitate the identification of changes between two layers, visual cues are added. Differently colored links between subsequent layers are used to indicate nodes that have moved or whose attribute values have changed significantly. Significance is determined by a user-selectable threshold. Positive attribute changes are shown as red links and negative changes are shown in blue. Links representing node movements are colored with a shade of gray. Addition or deletion of nodes and edges is indicated by spikes. Spikes that represent deletion leave a layer in the direction of the time axis and are shown in blue. Those that mark addition enter a layer and are shown in red. The layering approach in combination with the described visual cues allows users to compare successive time steps more closely.

Relevant references: Hadlak et al. (2010)

Great Wall of Space-Time

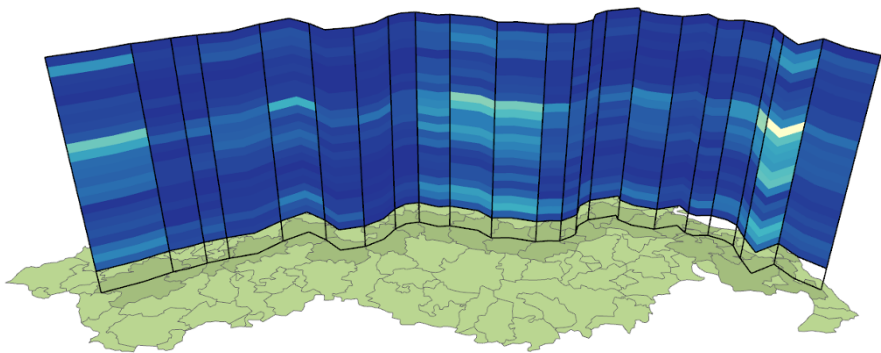


Fig. A.138: A Great Wall of Space-Time showing human health data. A path through space is extruded along the third dimension to create a slice through space-time, the wall, onto which the actual data representation is projected. © The authors. Generated with the LandVis system by Christian Tominski.

Tominski and Schulz (2012) introduce a visualization technique for spatio-temporal data that refers to 2D geographical space and 1D linear time. The idea is to construct a non-planar slice – called the Great Wall of Space-Time – through the 3D (2D+1D) space-time continuum. As such, the technique is based on the concept of the space-time cube (\hookrightarrow p. 377). The construction of the wall is based on topological and geometrical aspects of the geographical space. First, a topological path is established automatically or interactively based on a neighborhood graph of the map regions. The topological path is transformed into a geometrical path that respects the geographic properties of the areas of the map. The geometrical path is then extruded to a 3D wall, whose 3rd dimension can be used to map the time domain. Different visual representations can be projected onto the wall in order to display the data. Examples illustrate data visualizations based on color-coding (as shown in the figure) and parallel coordinates. The wall has the advantage that it shows a closed path through space with no gaps between the information-bearing pixels on the screen.

Relevant references: Tominski and Schulz (2012)

time

primitives: points
arrangement: linear

data

number of variables: single
frame of reference: spatial

vis

mapping of time: static
dimensionality: 3D

time

primitives: points
arrangement: linear

MoSculp



Fig. A.139: MoSculp-rendered sculpture of an Olympic runner. The sculpture captures the 3D characteristics of the motion of the runner’s limbs and torso. © ⓘ The authors. Generated with the *MoSculp* software by Xiuming Zhang.

data

number of variables: single
frame of reference: spatial

Zhang et al. (2018) describe their MoSculp approach as a novel way of visualizing motion and time. MoSculp is similar to stroboscopic photography and shape-time photography in that it makes intermediate states of motion visible. In fact, MoSculp creates a three-dimensional sculpture of the motion of complex objects, such as human bodies or animals. While existing approaches can represent only one or a few discrete states of an object’s motion at a time, a MoSculp sculpture creates a contiguous visual and tangible representation that captures all the motion’s states. The figure shows MoSculp applied to the movement of an Olympic runner. More specifically, the sculpture (in blue) captures the movement of several parts of the runner’s body, including the lower legs, the lower arms, and the torso. The creation of a MoSculp sculpture takes several steps. First, key points are extracted from the frames of a video clip. These 2D key points are then mapped to a sequence of 3D body models, which are then used to estimate the 3D motion for the entire video. The skeletons of the 3D body models form the basis for the creation of a MoSculp sculpture. The final visualization shows either the sequence of 3D models or a rendering of the sculpture, where different rendering styles can be selected by the user. It is even possible to create a physically 3D-printed version of the sculpture. In this sense, the MoSculp approach not only makes the key characteristics of motions visible, but also physically explorable.

vis

mapping of time: static
dimensionality: 3D

Relevant references: Zhang et al. (2018)

Flow Map

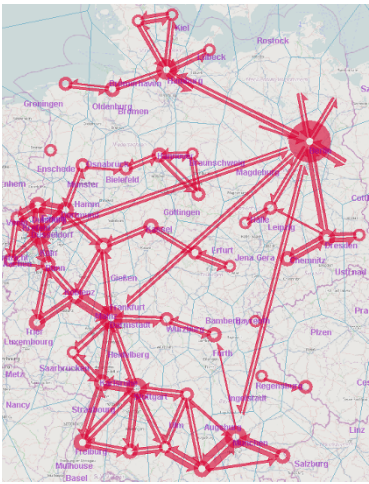


Fig. A.140: The flow map shows characteristic movements of photographers over time extracted from the metadata of more than 590,000 geo-referenced and time-stamped photographs. © *Courtesy of Gennady Andrienko.*

Flow maps show movements of objects over time, that is, they show a change of positions over time, rather than a change of data values. Usually, such movements form directed (optionally segmented) trajectories connecting the starting point of a movement and its end point. Such trajectories can be represented visually as more or less complex arrows or curves, where width, color, and other attributes can be used to encode additional information (see Kraak and Ormeling, 2020). A famous example is Minard’s flow map of Napoleon’s Russian campaign. A high number of flows, however, leads to overlapping trajectories and thus to visually cluttered flow maps. In order to represent massive data, flow maps can show abstractions of movements, rather than individual movements (see Andrienko and Andrienko, 2011). To faithfully communicate the underlying data, characteristic movements need to be extracted. First, time points are aggregated to larger time intervals and individual places are substituted with larger regions so as to arrive at abstracted trajectories that show mean trends. Secondly, the trajectories are grouped based on a similarity search, e.g., by applying cluster analysis or self-organizing maps (SOM). In this way, places with similar dynamics are merged, and individual trajectories are replaced by trajectories associated with the groups.

Relevant references: Kraak and Ormeling (2020) • Andrienko and Andrienko (2011)

time

primitives: points, intervals
arrangement: linear

data

number of variables: single
frame of reference: spatial

vis

mapping of time: static
dimensionality: 2D

Visits

time

primitives: points, intervals
arrangement: linear

data

number of variables: single
frame of reference: spatial

vis

mapping of time: static
dimensionality: 2D

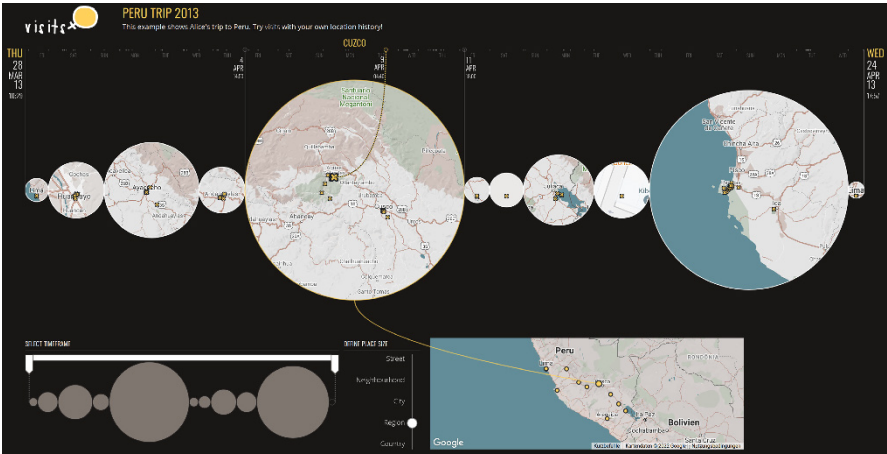


Fig. A.141: Location history of a trip to Peru visualized using the Visits approach. Map insets (center) with location markers are temporally arranged along a time axis (top). The insets have been generated automatically to match a region-level spatial resolution. Sliders (bottom left and center) can be used to adjust the spatial resolution and filter the time axis. An overview map (bottom right) provides spatial context. © The authors. Generated with the Visits software by Alice Thudt.

Location histories contain information about a person’s spatial and temporal whereabouts. That is, they describe at what time (when) a person was at certain locations (where). The Visits approach by Thudt et al. (2013) makes location histories visually accessible with a combined spatial and temporal visualization. The dimension of time is represented as a horizontal time axis (top in the figure). Each visited location is marked as a small dot at the time axis. More details about the locations’ spatial context are represented in circular map insets (center) that are aligned along with the time axis. A map inset represents a temporally contiguous group of locations, which are marked by small crosses. The specific time point when a location was visited can be interactively highlighted via an arc drawn between the location and the time point at the time axis. The diameter of a map inset corresponds to the time interval covered in the data. The number of map insets depends on the chosen spatial resolution. Different resolutions (e.g., street, neighborhood, city, region) can be selected via a vertical slider (bottom center). Choosing street-level resolution will lead to more and smaller map insets, and region-level resolution leads to fewer and larger map insets. To keep users oriented, an overview map (bottom right) indicates where the map insets are located on a global scale. Moreover, a dual-handle range slider can be used to filter the time axis.

Relevant references: Thudt et al. (2013)

Time-Oriented Polygons on Maps

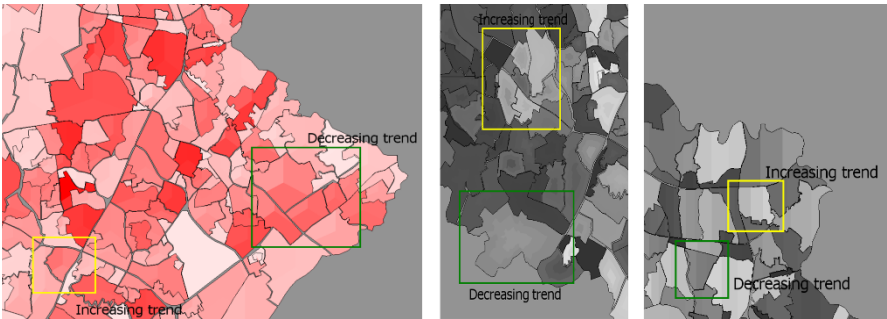


Fig. A.142: Time-oriented polygons are sub-divided according to time using three different schemes: wedges (left), rings (center), and time slices (right). Data values are encoded by color. The maps show the development of the high school population over the course of three years. © 2005 IEEE. Reprinted, with permission, from Shanbhag et al. (2005).

Shanbhag et al. (2005) present three time-oriented visualization methods to analyze and support the effective allocation of resources in a spatio-temporal context. Wedges, rings, and time slices are the three basic layouts used to display changes in data values over time on a map. For all three variants, data values and categories are represented using color components (hue, saturation, and brightness). In the layout of the wedges, the area of a polygon is partitioned into radial sectors in a clock-like manner (left figure). The ring layout is inspired by the concentric rings of a tree trunk where the innermost ring corresponds to the earliest time point and the outermost ring corresponds to the latest time point (center figure). The time slices layout divides a polygon into vertical slices that are ordered from left to right according to the progress in time (right figure). The wedges, rings, and time slice layouts are applied to polygonal areas of a map. Time-oriented polygons on maps were used, for example, to plan the future allocation of resources for schools based on time-dependent variables such as student population by grade, number of students requiring free meals, and test scores needed.

Relevant references: Shanbhag et al. (2005)

time

primitives: points
arrangement: linear, cyclic

data

number of variables: single
frame of reference: spatial

vis

mapping of time: static
dimensionality: 2D

time

primitives: points
arrangement: linear, cyclic

data

number of variables: single
frame of reference: spatial

vis

mapping of time: static
dimensionality: 2D

Growth Ring Maps

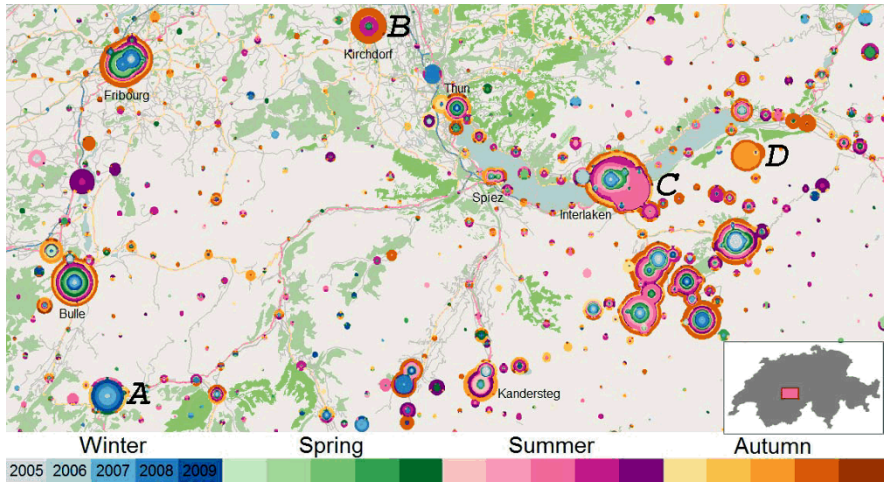


Fig. A.143: Growth Ring Maps visualize spatio-temporal events as colored pixels accumulating in space around locations where the events occurred. This map shows information from a photo database, more concretely, it shows where and when photos were taken. © Courtesy of Peter Bak.

Growth Ring Maps by Bak et al. (2009) are a technique for visualizing the spatio-temporal distribution of events. Every spatio-temporal event is represented by one pixel, which makes the technique highly scalable with the number of events. Each location (for example the centroid of spatial clusters of events) is taken as the center point for the computation of growth rings. The pixels (i.e., events) are placed around this center point in an orbital manner resulting in the so-called Growth Ring representations. The pixels are sorted by the date and time the event occurred: the earlier an event happened, the closer the pixel is to the central point. Additional color-coding is used to visualize the association of a pixel with a time primitive. When two or more neighboring growth rings are about to overlap, the layout algorithm displaces the pixels in such a way that none of them is covered by another pixel. Hence, when big clusters of events are close in space, the corresponding growth rings will not have perfectly circular shapes but will be distorted. The resulting Growth Ring Maps are overlaid over a cartographic map to capture their spatial context. The figure illustrates Growth Ring Maps where events correspond to photos taken by tourists.

Relevant references: Bak et al. (2009) • Andrienko et al. (2011)

Trajectory Wall

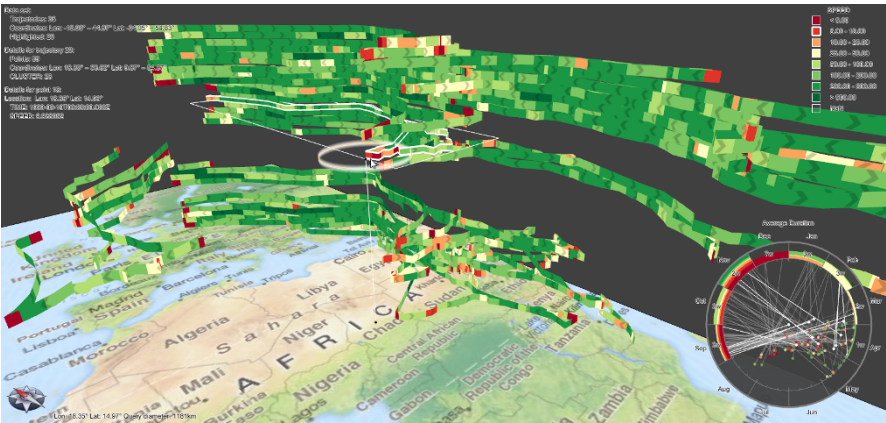


Fig. A.144: Visualization of trajectories of migrating storks. Colored trajectory bands represent the speed of migration in km/d. A radial time lens (bottom right) represents the distribution of speeds for the months of a year. © The authors. Generated with the TrajectoryVis software by Christian Tominski.

The Trajectory Wall by Tominski et al. (2012b) shows temporal, spatial, and attribute information of movement data as a hybrid 2D/3D visual representation. Starting with an appropriate grouping of movement trajectories, the trajectories are mapped to 3D bands that are stacked above a map display to erect a wall of trajectories. A user-selected data attribute associated with the movement (e.g., speed, acceleration, sinuosity) is color-coded along the individual bands. The 3D representation is suitable for being watched from an oblique perspective. When looking at the map from a bird’s-eye view, colored 2D lines appear on the map and provide an overall visualization of all movement trajectories. An interactive time lens (bottom right) enables the user to access temporarily aggregated information about a selected spatial region of the data. While the trajectory bands communicate the spatial information and linear temporal character of the data, the time lens emphasizes the cyclic temporal components. In combination, the interactive visualizations enable users to explore trajectory attributes with regard to their spatial and temporal dependencies. Movement patterns such as general commuting behavior, unexpected deviations, or trends in the development of trajectory attributes can be discerned.

Relevant references: Tominski et al. (2012b) • Andrienko et al. (2014)

time

primitives: points
arrangement: linear, cyclic

data

number of variables: single
frame of reference: spatial

vis

mapping of time: static
dimensionality: 2D, 3D

time

Spatio-Temporal Event Visualization

primitives: points
arrangement: linear

data

number of variables: single, multiple
frame of reference: spatial

vis

mapping of time: static
dimensionality: 3D

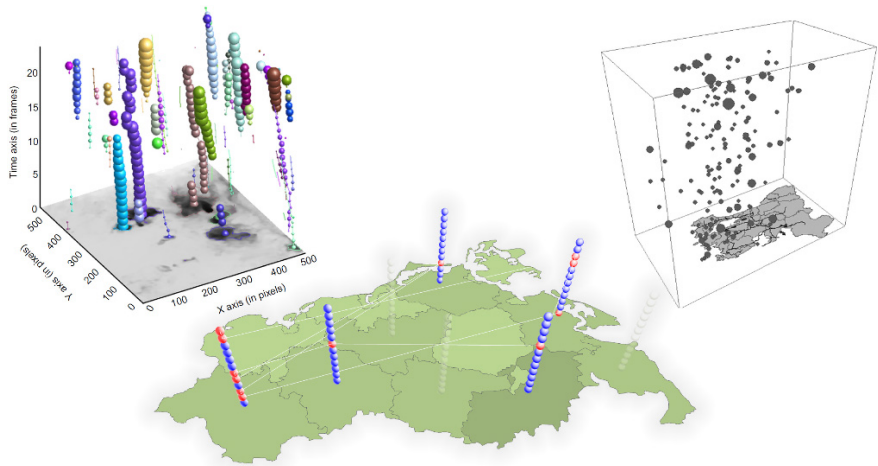


Fig. A.145: Events in space and time are visualized by embedding graphical objects of varying size and color into space-time cubes. From left to right, the cubes show events related to convective clouds, human health data, and earthquakes. Left: © 2007 Elsevier. Reprinted, with permission, from Turdukulov et al. (2007).; Center: © The authors. Generated with the LandVis system by Christian Tominski; Right: © 2004 IEEE. Reprinted, with permission, from Gatalsky et al. (2004).

Events usually describe happenings of interest. In order to analyze events in their spatial and temporal context, one can make use of the space-time cube concept (\hookrightarrow p. 377), where space is mapped to the x-y plane and time is mapped to the z-axis of a 3D presentation space. The actual events are visualized by placing graphical objects in the space-time cube at those positions where events are located in time and space. Attributes associated with events can be encoded, for example, by varying the size, color, shape, or texture of the graphical objects. Marking events in a space-time cube is a general concept with a wide range of applications: Turdukulov et al. (2007) explore events related to the development of convective clouds, Tominski et al. (2005b) consider maxima in human health data as events of interest, and Gatalsky et al. (2004) visualize earthquake events.

Relevant references: Turdukulov et al. (2007) • Gatalsky et al. (2004) • Tominski et al. (2005b)

Space-Time Cube

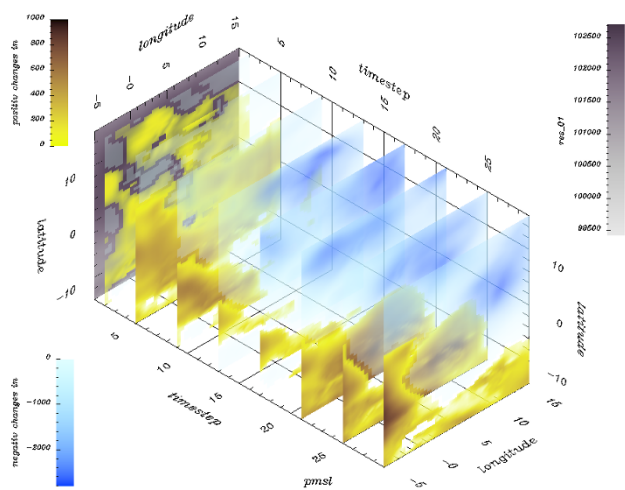


Fig. A.146: This space-time cube represents two spatial dimensions (latitude and longitude) along the y-axis and the z-axis, and time along the x-axis. Multiple color-coded layers are embedded into the cube to visualize spatio-temporal climate data. © Courtesy of Thomas Nocke.

A classic concept combining the visualization of space and time is the space-time cube, which is attributed to the pioneering work of Hägerstrand (1970). The basic idea is to map two spatial dimensions to two axes of a virtual three-dimensional cube and to use the third axis for the mapping of time. The spatial context is often represented as a map that constitutes one face of the space-time cube. The three-dimensional space inside the cube is used to represent spatio-temporal data, where manifold visual encodings are possible. One can place graphical objects in the cube in order to mark points of interest, or one can construct trajectories that illustrate paths of objects (↔ p. 378). Associated data can be encoded to the properties of graphical objects and trajectories, where color and size are common candidates. Another technique is to place multiple layers along the time axis, each of which encodes the data for a specific time point. Space-time cubes usually rely on appropriate interaction to allow users to view the data from different perspectives. A contemporary review of the concept can be found in the work by Kraak (2003). User studies found that the 3D nature of space-time cubes matches quite well with the conditions in immersive environments (see Filho et al., 2020).

Relevant references: Hägerstrand (1970) • Kraak (2003) • Filho et al. (2020)

time

primitives: points, intervals
arrangement: linear

data

number of variables: single, multiple
frame of reference: spatial

vis

mapping of time: static
dimensionality: 3D

time

Space-Time Path

primitives: points, intervals
arrangement: linear

data

number of variables: single, multiple
frame of reference: spatial

vis

mapping of time: static
dimensionality: 3D

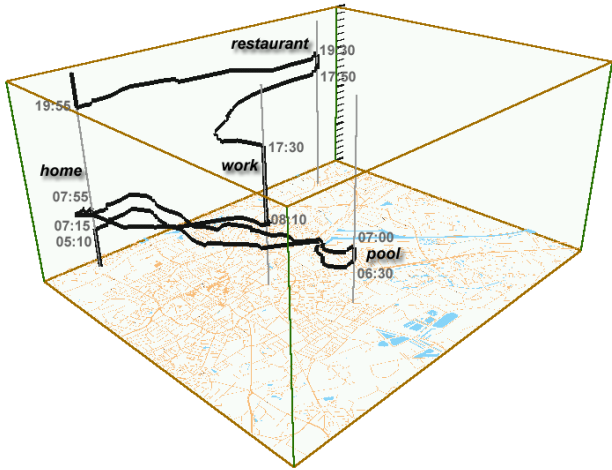


Fig. A.147: A space-time path is embedded into a space-time cube to show a person’s movement through space and time. For better orientation, important places are marked by vertical lines and annotations. © 2003 International Cartographic Association. Reprinted, with permission, from Kraak (2003).

The space-time path is a specific representation of data in a space-time cube (\hookrightarrow p. 377). The roots of the concept of space-time paths can be found in the work by Lenntorp (1976). Kwan (2009) describes contemporary visual representations that are based on the classic concept. A space-time path is constructed by considering the location of an object as a three-dimensional point in space and time. Multiple such points ordered by time describe the path that an object has taken. The path can be rendered as a polyline that connects successive points. In order to encode data along a space-time path, one can vary the line’s color, use differently dashed line segments, or employ other visual attributes. Alternatively, a space-time path can be represented as a three-dimensional tube, where the tube’s radius can be varied to encode additional data values. Today’s implementations usually offer interaction to allow for virtual movements through space and time, or for rotation and zoom.

Relevant references: Kraak (2003) • Lenntorp (1976) • Kwan (2009)

Icons on Maps

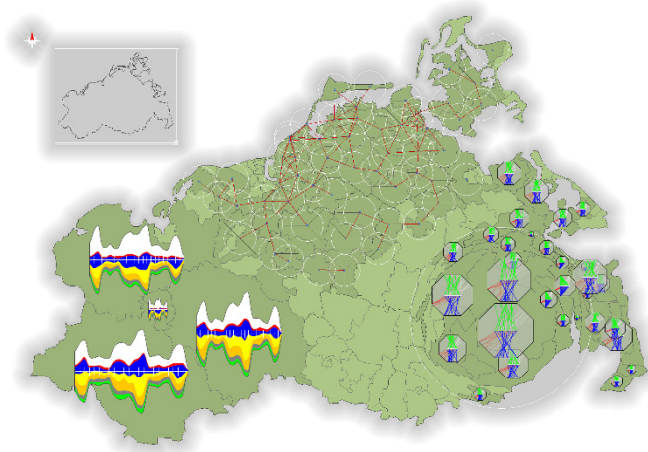


Fig. A.148: The figure illustrates the embedding of time-representing icons into a map in order to visualize spatio-temporal data. This illustration shows ThemeRiver icons and TimeWheel icons in the left and the right part of the map, respectively. The northern part of the map illustrates a conflict graph as used for local optimization of icon positions. © The authors. Generated with the LandVis system by Christian Tominski.

When time-oriented data contain additional spatial dependencies, it is necessary to visualize both the temporal and the spatial aspects of the data. A sensible approach to achieving this is to adapt and combine existing solutions. Maps are commonly applied to represent the spatial context of the data. In order to combine maps with existing visualization techniques for representing the temporal context, they must be made compatible with the map display. First and foremost, this implies a reduction in size, which effectively means creating icons from otherwise full-size visual representations. In a second step, it is then possible to place icons on the map, exactly where the data are anchored in space. Tominski et al. (2003) and Fuchs and Schumann (2004b) demonstrate the integration of the ThemeRiver (\leftrightarrow p. 293) and the TimeWheel (\leftrightarrow p. 298) into a map display as shown in the figure. If there are too many icons on a map, they might occlude each other. Therefore, an additional computational step can be used to determine suitable overlap-free icon positions, a problem that is very much related to the cartographic map labeling problem. This problem can be tackled by global or local optimization of icon positions (see Fuchs and Schumann, 2004a; McNabb and Laramee, 2019).

Relevant references: Tominski et al. (2003) • Fuchs and Schumann (2004b) • Fuchs and Schumann (2004a) • McNabb and Laramee (2019)

time

primitives: points, intervals
arrangement: linear, cyclic

data

number of variables: single, multiple
frame of reference: spatial

vis

mapping of time: static
dimensionality: 2D, 3D

VIS-STAMP

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: spatial

vis

mapping of time: static
dimensionality: 2D

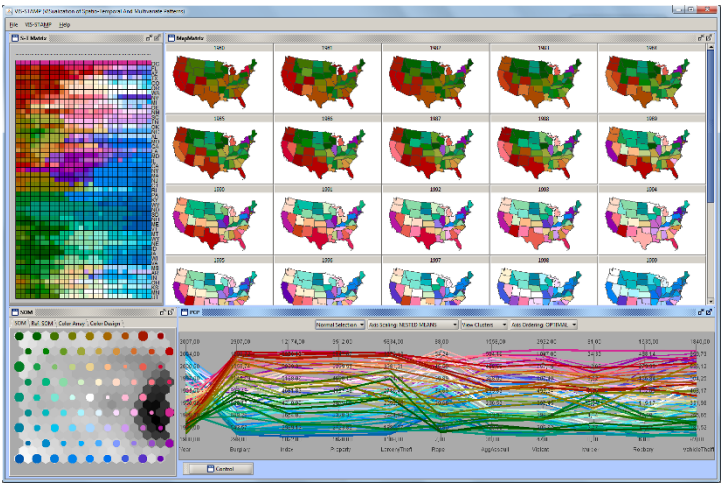


Fig. A.149: Multiple views show multivariate spatio-temporal crime data that have been clustered by a self-organizing map (SOM). An enhanced color-coding schema is used consistently across all views to visualize cluster affiliation. © The authors. Generated with the VIS-STAMP system by Diansheng Guo.

Spatio-temporal data can be complex and multi-faceted. Guo et al. (2006) developed a system called VIS-STAMP that integrates computational, visual, and cartographic methods for visual analysis and exploration of such data. At the heart of the system is a self-organizing map (SOM) that is used for multivariate clustering, sorting, and coloring. The visual ensemble comprises a matrix view (top left), a map view (top right), a parallel coordinates view (bottom right), and a SOM view (bottom left). The matrix view's columns represent time points and its rows stand for geographic regions. Cluster affiliation of the matrix cells is visualized by means of an enhanced color-coding schema. The color-coding is consistent across all views. The map view follows the small multiples approach (↔ p. 359) and shows color-coded map thumbnails, one for each time point. The parallel coordinates view addresses the multivariate character of the data. Finally, the SOM view offers a detailed view and control interface of the underlying self-organizing map. A number of automatic and interactive manipulation techniques (e.g., reordering and sorting) facilitate the data analysis.

Relevant references: Guo et al. (2006)

Time-Ray Maps

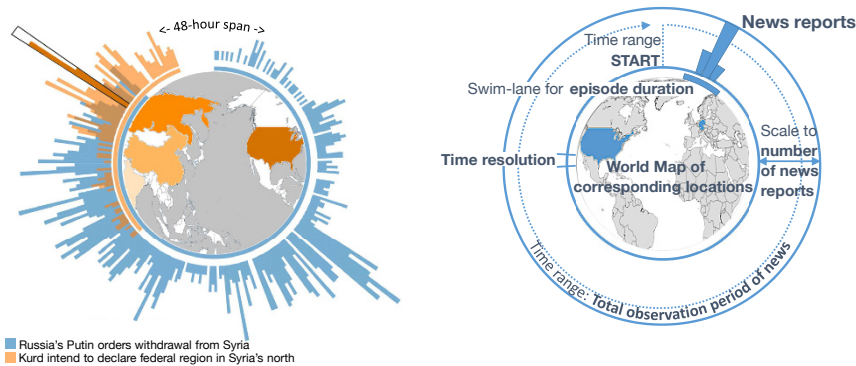


Fig. A.150: Time-Ray Maps allow users to assess the temporal evolution along with the geographic origins of news items at a glance. A circular time axis around the circumference of the world map is used to depict the observation period starting from the top. Bar height represents the amount of news items for a particular topic while color hue is used to distinguish between different news report types. Upon selection of a time interval in the bar graph, corresponding spatial information is highlighted in the world map. © ⓘ Courtesy of Julia Sheidin.

Time-Ray Maps represent the temporal and spatial evolution of news episodes. Moreover, news stories can be compared to each other, as well as a quick analysis of a specific story in terms of influence is made possible. The circular bar graphs (↔ p. 234) along the outer circle represent the amount of news reports over time, starting at the top and continuing clockwise. Inside the inner circle, highlighted regions are displayed on a circular world map. For representing multiple news episodes, different color hues are overlaid semi-transparently to allow for direct comparison of temporal and spatial patterns. Interactive brushing by selecting individual bars or time intervals along the circular time axis highlights corresponding spatial regions in the center of the representation. The highlighted regions and bars are linked by using the same color hues. Inside the world map, the user can pan/rotate the visible area and zoom to regions of interest.

Relevant references: Sheidin et al. (2017)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: spatial

vis

mapping of time: static
dimensionality: 2D

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: spatial

vis

mapping of time: static
dimensionality: 3D

Wakame

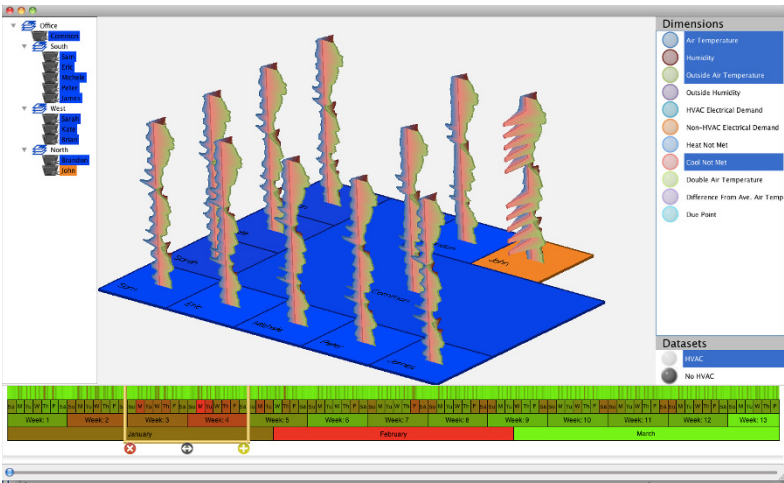


Fig. A.151: Three-dimensional visualization objects, so-called wakame, show multiple variables along time. Placing multiple wakame on a map allows analysts to make sense of spatio-temporal data. A time scale widget on the bottom can be used to select intervals of interest. © *Courtesy of Clifton Forlines.*

Forlines and Wittenburg (2010) describe an interactive system for visualizing multivariate spatio-temporal data. The temporal aspects are encoded to multivariate glyphs, so-called wakame. A single wakame basically corresponds to a radar chart that has been extruded along the third dimension. In a radar chart, different variables are represented on radially arranged axes that are connected to form a polyline. Wakame are constructed as solid three-dimensional objects whose shapes indicate temporal trends and relations among time-dependent variables, similar to the Kiviat tube (↔ p. 319). Embedding multiple wakame into a map display facilitates the understanding of spatial aspects. Noteworthy about the wakame approach are its interaction and animation facilities. An intelligent camera positioning mechanism supports users in finding perspectives on the wakame that most likely bear interesting information. A hierarchical time axis widget (↔ p. 336) denotes by color how different each time primitive is from its neighbors. This allows users to pick interesting time primitives easily. Upon interaction, an animation is used to smoothly interpolate between views. Additional animation schemes are offered to switch between the three-dimensional wakame view and traditional two-dimensional radar charts and line plots (↔ p. 233), which might be better suited for certain analysis tasks. Given their 3D nature, wakame can also be useful in the context of immersive analytics as demonstrated by Reski et al. (2020), who refer to their technique as 3D radar charts.

Relevant references: Forlines and Wittenburg (2010) • Reski et al. (2020)

GeoTime

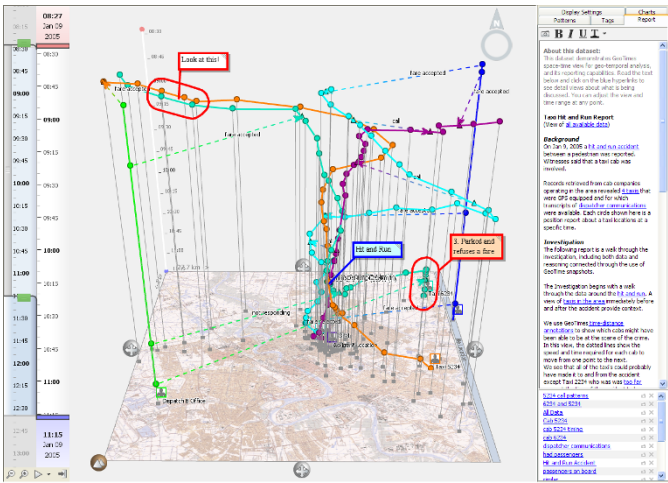


Fig. A.152: The visualization shows taxis involved in a hit and run accident as colored points and paths. Patterns of interest are annotated in the scene, and a narrative can be authored on the right. © Courtesy of William Wright.

Kapler and Wright (2005) describe GeoTime[®], which is a registered trademark of Oculus Info Inc., as a system to visualize data items (e.g., objects, events, transactions, flows) in their spatial and temporal context. It provides a dynamic, interactive version of the space-time cube concept (↔ p. 377), where a map plane illustrates the spatial context and time is mapped vertically along the third display dimension. Items and tracks are placed in the space-time cube at their spatial and temporal coordinates. GeoTime provides a variety of visual and interactive capabilities. Time intervals of interest can be selected by the user and events are smoothly animated along the time axis. Alternative projections of the display allow users to focus more on either temporal or spatial aspects. Notable about GeoTime are its annotation, storytelling, and pattern recognition features (see Eccles et al., 2008). They enable automatic as well as user annotation of the representation with findings, as well as the creation of stories about the data for analytic exploration and communication. Additional functionality allows the analysis of events and transactions in time above a network diagram (see Kapler et al., 2008).

Relevant references: Kapler and Wright (2005) • Eccles et al. (2008) • Kapler et al. (2008)

time

primitives: points
arrangement: linear

data

number of variables: multiple
frame of reference: spatial

vis

mapping of time: static, dynamic
dimensionality: 3D

time

primitives: points, intervals
arrangement: linear

Multiple Temporal Axes Model

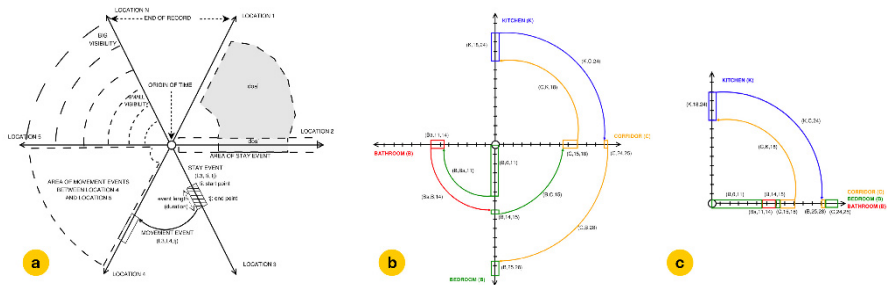


Fig. A.153: Multiple Temporal Axes Model. (a) Using the model to represent movement and activity events across different locations in the home. (b) A person’s movements between the locations kitchen, corridor, bedroom, and bathroom in her home. (c) Axes representing different locations might also be collapsed on top of each other. © 2013 IEEE. Reprinted, with permission, from Juarez et al. (2013).

data

number of variables: multiple
frame of reference: spatial

Tracking a person’s movements between a set of known locations over time is an important task in a number of domains. In their work in the context of ambient assisted living (AAL), Juarez et al. (2013) designed a multiple temporal axes (MTA) model to detect accidents at home, which are still one of the major risk factors for the elderly. The data are collected via sensors and automated preprocessing is applied to detect stay as well as movement events. Using a specialized visualization model helps users to identify common risk scenarios via visual mining. In the visualization model, axes represent different locations and are arranged in a star-like layout. Time extends from the center to the outside on each axis and distinct colors are used for each location. Thus, each axis represents both, a spatial and a temporal perspective. When a person stays in one location (stay event), a colored rectangle is shown along the time axis. A movement event is represented by an arc that connects the origin and destination location. As time evolves from the inside to the outside of the axes, arcs are getting longer the later the activity events occur. For summarizing movements in the house, a number of axes can be collapsed on top of each other (see (c) in figure). This can also be used to alleviate problems with arcs crossing multiple axes in case they cannot be arranged next to each other. The MTA model supports users in gaining an overview of the stay and activity sequences in terms of intervals and interval relations and identifying densities in the activities in order to monitor and detect accidents.

vis

mapping of time: static
dimensionality: 2D

Relevant references: Juarez et al. (2013)

Data Vases

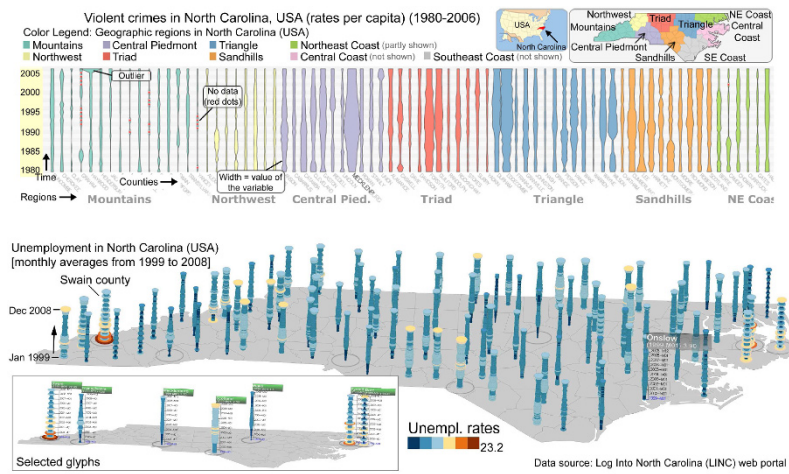


Fig. A.154: Shape and color of a data vase encode time-varying data values. The spatial component of the data can be communicated by a vertical alignment of 2D data vases (top), or by embedding 3D data vases into a space-time cube (bottom). © 2010 IEEE. Reprinted, with permission, from Thakur and Hanson (2010).

The data vases technique has been designed to visualize multiple time-varying variables. Thakur and Rhyne (2009) describe two alternative designs: a 2D and a 3D variant. A 2D data vase is basically a graph constructed by mirroring a line plot (↔ p. 233) against the time axis, effectively creating a symmetric shape that can be filled (segment-wise) with a data-specific color. Such data vases can then be arranged on the screen to create a meaningful visualization. For spatio-temporal data, one can use multiple vertically aligned data vases, each of which represents an individual geographic region. Thakur and Hanson (2010) further elaborate on the idea of extending data vases to the third dimension. In 3D, data vases are constructed by stacking discs along a vertical time axis, where each disc maps the data for a particular time primitive to disc size and color. Such 3D data vases can then be embedded into a space-time cube (↔ p. 377), i.e., a virtual 3D world where two dimensions are used to show a geographic map and the third dimension encodes time.

Relevant references: Thakur and Rhyne (2009) • Thakur and Hanson (2010)

time

primitives: points, intervals
arrangement: linear

data

number of variables: multiple
frame of reference: spatial

vis

mapping of time: static
dimensionality: 2D, 3D

time

Pencil Icons

primitives: points, intervals
arrangement: linear

data

number of variables: multiple
frame of reference: spatial

vis

mapping of time: static
dimensionality: 3D

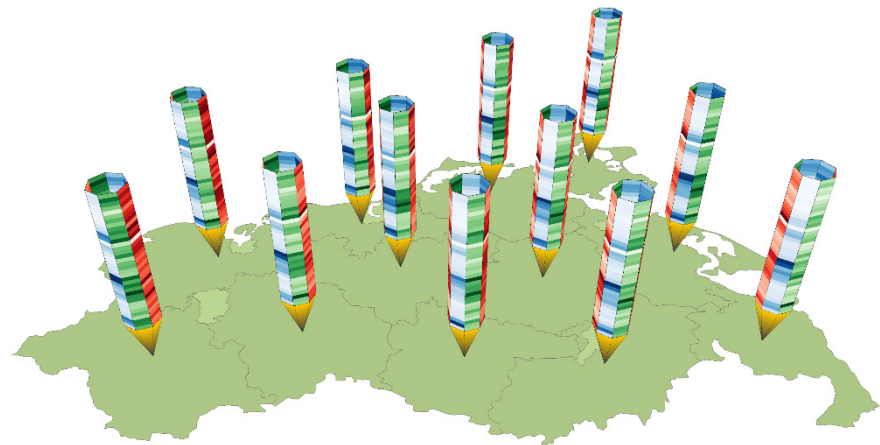


Fig. A.155: Multiple time-dependent variables are mapped onto the faces of pencil icons to visualize temporal dependencies in the data. By placing the icons on a map, the spatial dependencies are communicated. © ⓘ The authors. Generated with the LandVis system by Christian Tominski.

Pencil icons visualize multivariate spatio-temporal data. The technique is based on the space-time cube concept (↔ p. 377), where the spatial frame of reference is represented as a map in the x-y plane of a virtual three-dimensional cube. The dimension of time is mapped along the cube’s z-axis. Within the cube, pencil icons are positioned where data are available. This way, the spatial context is communicated. Each pencil icon represents the temporal context and multiple time-dependent variables by mapping time along the pencil, starting at the tip, and associating each face of the pencil with an individual time-dependent variable. Color-coding is applied to visualize the data. Color lightness is varied according to data values, and different hues are used to help users identify particular variables. The linear shape of the pencil is suited to represent the linear characteristics of the underlying time axis. Heterogeneous data can be depicted by using appropriate color scales. In order to deal with occlusion and information displayed at the pencils’ back faces, several interaction techniques are provided, including navigation in the virtual world as well as the individual and linked rotation of pencils.

Relevant references: Tominski et al. (2005b)

Temporal Focus+Context

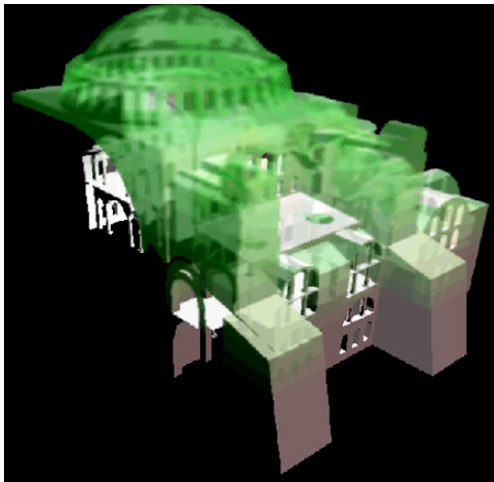


Fig. A.156: Using temporal focus+context, phases of a building’s history are visualized using different graphical properties (e.g., rendering style, transparency) to emphasize and attenuate different time periods. © *Courtesy of Alexandre Carvalho.*

Carvalho et al. (2008) introduce a temporal focus+context visualization model to meaningfully display several time points simultaneously. In this model, focus+context is applied to time rather than, more typically, to attributes or space. Underlying the proposed technique is the calculation of a temporal degree of interest (TDoI), which is driven by the valid time attribute, by specific analysis, exploration, or presentation goals as well as by user-defined visualization requirements. The TDoI is used to convey the temporal aspects of the data via adjusting graphical properties, such as transparency, color, sketchiness, or other non-photorealistic enhancements. This makes it possible to meaningfully compress information about distinct temporal states of the data into the same visualization display. The concept of temporal focus+context is generally applicable to different types of visualization of time-oriented data.

Relevant references: Carvalho et al. (2008)

time

primitives: intervals
arrangement: linear

data

number of variables: multiple
frame of reference: spatial

vis

mapping of time: static
dimensionality: 3D

time

Chro-Ring

primitives: points, intervals
arrangement: cyclic

data

number of variables: multiple
frame of reference: spatial

vis

mapping of time: static
dimensionality: 2D

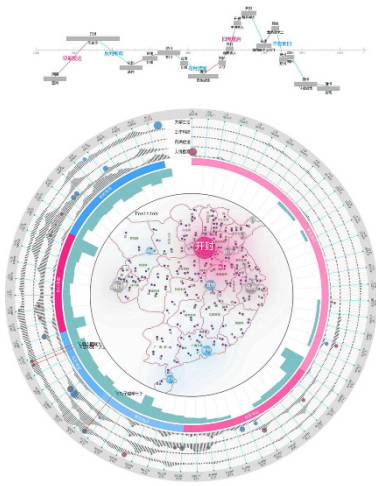


Fig. A.157: The time ring of Chro-Ring illustrates the life of the writer Su Shi. The upper part shows the life bar, illustrating the various events. The lower part shows the composition histogram, summarizing his life in different rings. Red arcs are successful periods and blue arcs indicate setbacks. In the middle of the ring, the map illustrates the various locations. © 2016 Springer Nature. Reprinted, with permission, from Zhu et al. (2016).

Visualizing and exploring the personal histories of writers according to temporal and geo-spatial dimensions is a challenging task. Chro-Ring by Zhu et al. (2016) is a visual exploration environment that captures the individual histories of Chinese writers according to their life stories, geographic information, and relationships. The chronology graph is represented as ring, and various facets, views, and interaction techniques are provided to visually explore the writers' history. The most important views are (1) time ring, (2) life river, (3) histograms, (4) cloud map, (5) line chart, (6) tag cloud, and (7) radial spikes. Color-coding (red and blue) annotates successful and unsuccessful life periods. A usage scenario and user study (with 24 participants) illustrate the usability of Chro-Ring.

Relevant references: Zhu et al. (2016)

Helix Icons

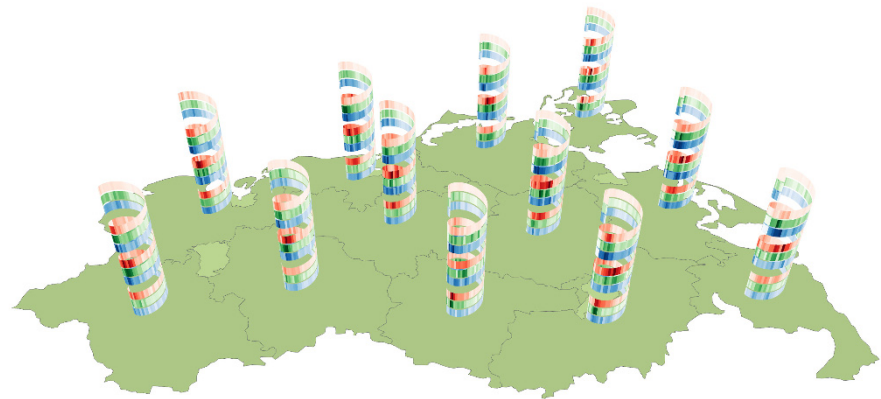


Fig. A.158: Helix icons use color-coding to visualize multivariate spatio-temporal data along helix ribbons, which emphasize the data’s cyclic temporal character. The spatial aspect of the data is illustrated by embedding helix icons in a space-time cube. © The authors. Generated with the LandVis system by Christian Tominski.

Helix icons by Tominski et al. (2005b) are useful for emphasizing the cyclic character of spatio-temporal data. The underlying model of this technique is the space-time cube (\hookrightarrow p. 377), which maps the spatial context to the x-axis and the y-axis, and the dimension of time to the z-axis of a virtual three-dimensional cube. The actual data visualization is embedded into the cube. Helix icons use a helix ribbon to roll up the time domain along the z-axis. Each segment of the helix ribbon visualizes a specific time point (or interval) in time by means of color-coding. Multiple time-dependent variables can be visualized by subdividing the helix ribbon into narrower sub-ribbons, each of which represents a different variable. Using unique hues for each sub-ribbon helps the user distinguish variables. As for spiral graphs (\hookrightarrow p. 284), interaction techniques help users in finding an appropriate number of segments per cycle so that periodic patterns in the data are revealed. The inherent 3D representation problems (i.e., information displayed on helix back faces or inter-icon occlusion) are dealt with by offering 3D navigation through the space-time cube and rotation of helix icons. It is worth noting that helical representations are useful for time-oriented data with and without spatial context (see Gautier et al., 2016; Weber et al., 2001).

Relevant references: Tominski et al. (2005b) • Gautier et al. (2016) • Weber et al. (2001)

time

primitives: points, intervals
arrangement: cyclic

data

number of variables: multiple
frame of reference: abstract, spatial

vis

mapping of time: static
dimensionality: 3D