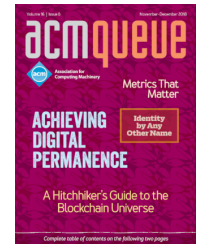


The November/December issue of acmqueue is out now

[Subscribers and ACM Professional members login here](#)



Graphics

May 13, 2010

Volume 8, issue 5



A Tour through the Visualization Zoo

A survey of powerful visualization techniques, from the obvious to the obscure

Jeffrey Heer, Michael Bostock, and Vadim Ogievetsky, Stanford University

Thanks to advances in sensing, networking, and data management, our society is producing digital information at an astonishing rate. According to one estimate, in 2010 alone we will generate 1,200 exabytes—60 million times the content of the Library of Congress. Within this deluge of data lies a wealth of valuable information on how we conduct our businesses, governments, and personal lives. To put the information to good use, we must find ways to explore, relate, and communicate the data meaningfully.

The goal of visualization is to aid our understanding of data by leveraging the human visual system's highly tuned ability to see patterns, spot trends, and identify outliers. Well-designed visual representations can replace cognitive calculations with simple perceptual inferences and improve comprehension,

memory, and decision making. By making data more accessible and appealing, visual representations may also help engage more diverse audiences in exploration and analysis. The challenge is to create effective and engaging visualizations that are appropriate to the data.

Creating a visualization requires a number of nuanced judgments. One must determine which questions to ask, identify the appropriate data, and select effective *visual encodings* to map data values to graphical features such as position, size, shape, and color. The challenge is that for any given data set the number of visual encodings—and thus the space of possible visualization designs—is extremely large. To guide this process, computer scientists, psychologists, and statisticians have studied how well different encodings facilitate the comprehension of data types such as numbers, categories, and networks. For example, *graphical perception* experiments find that spatial position (as in a scatter plot or bar chart) leads to the most accurate decoding of numerical data and is generally preferable to visual variables such as angle, one-dimensional length, two-dimensional area, three-dimensional volume, and color saturation. Thus, it should be no surprise that the most common data graphics, including bar charts, line charts, and scatter plots, use position encodings. Our understanding of graphical perception remains incomplete, however, and must appropriately be balanced with interaction design and aesthetics.

This article provides a brief tour through the "visualization zoo," showcasing techniques for visualizing and interacting with diverse data sets. In many situations, simple data graphics will not only suffice, they may also be preferable. Here we focus on a few of the more sophisticated and unusual techniques that deal with complex data sets. After all, you don't go to the zoo to see Chihuahuas and raccoons; you go to admire the majestic polar bear, the graceful zebra, and the terrifying Sumatran tiger. Analogously, we cover some of the more exotic (but practically useful!) forms of visual data representation, starting with one of the most common, time-series data; continuing on to statistical data and maps; and then completing the tour with hierarchies and networks. Along the way, bear in mind that all visualizations share a common "DNA"—a set of mappings between data properties and visual attributes such as position, size, shape, and color—and that customized species of visualization might always be constructed by varying these encodings.

Most of the visualizations shown here are accompanied by interactive examples. The live examples were created using [Protovis](#), an open source language for Web-based data visualization. To learn more about how a visualization was made (or to copy and paste it for your own use), simply "View Source" on the page. All example source code is released into the public domain and has no restrictions on reuse or modification. Note, however, that these examples will work only on a modern, standards-compliant browser supporting SVG (scalable vector graphics). Supported browsers include recent versions of Firefox, Safari, Chrome, and Opera. Unfortunately, Internet Explorer 8 and earlier versions do not support SVG and so cannot be used to view the interactive examples.

Time-Series Data

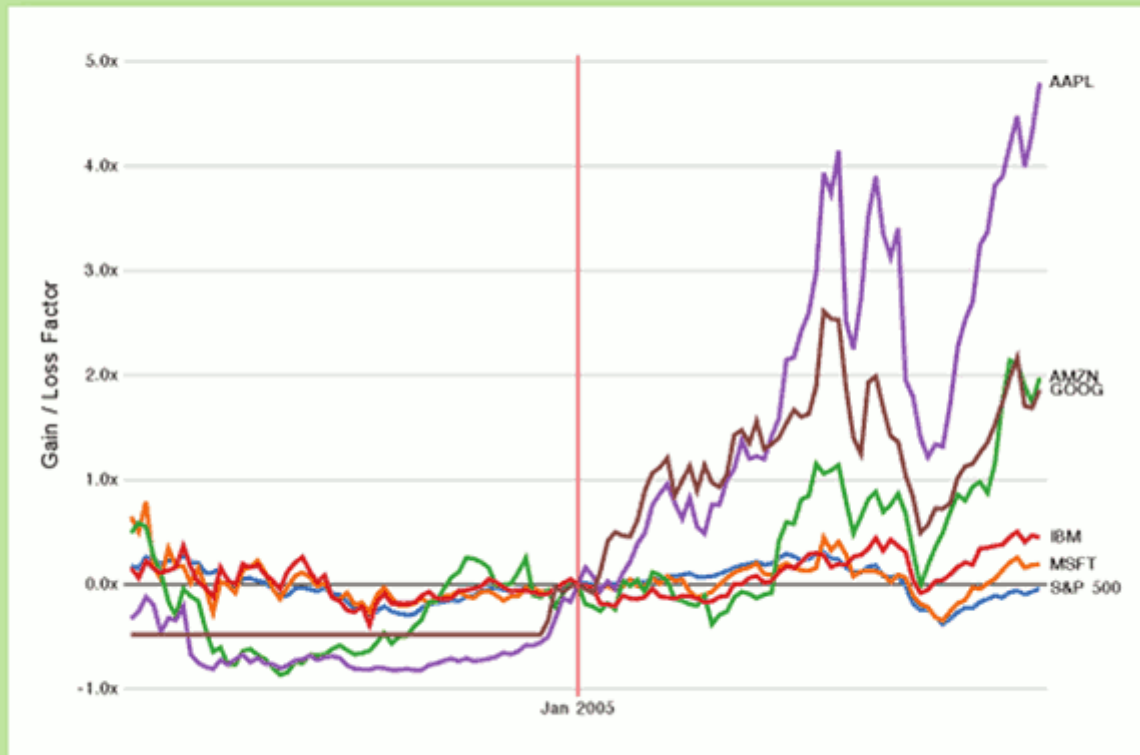
Time-series data—sets of values changing over time—is one of the most common forms of recorded data. Time-varying phenomena are central to many domains such as finance (stock prices, exchange rates), science (temperatures, pollution levels, electric potentials), and public policy (crime rates). One often needs to compare a large number of time series simultaneously and can choose from a number of visualizations to do so.

Index Charts

With some forms of time-series data, raw values are less important than relative changes. Consider investors who are more interested in a stock's growth rate than its specific price. Multiple stocks may have dramatically different baseline prices but may be meaningfully compared when normalized. An *index chart* is an interactive line chart that shows percentage changes for a collection of time-series data based on a selected index point. For example, the image in figure 1A shows the percentage change of selected stock prices if purchased in January 2005: one can see the rocky rise enjoyed by those who invested in Amazon, Apple, or Google at that time.

FIGURE 1A

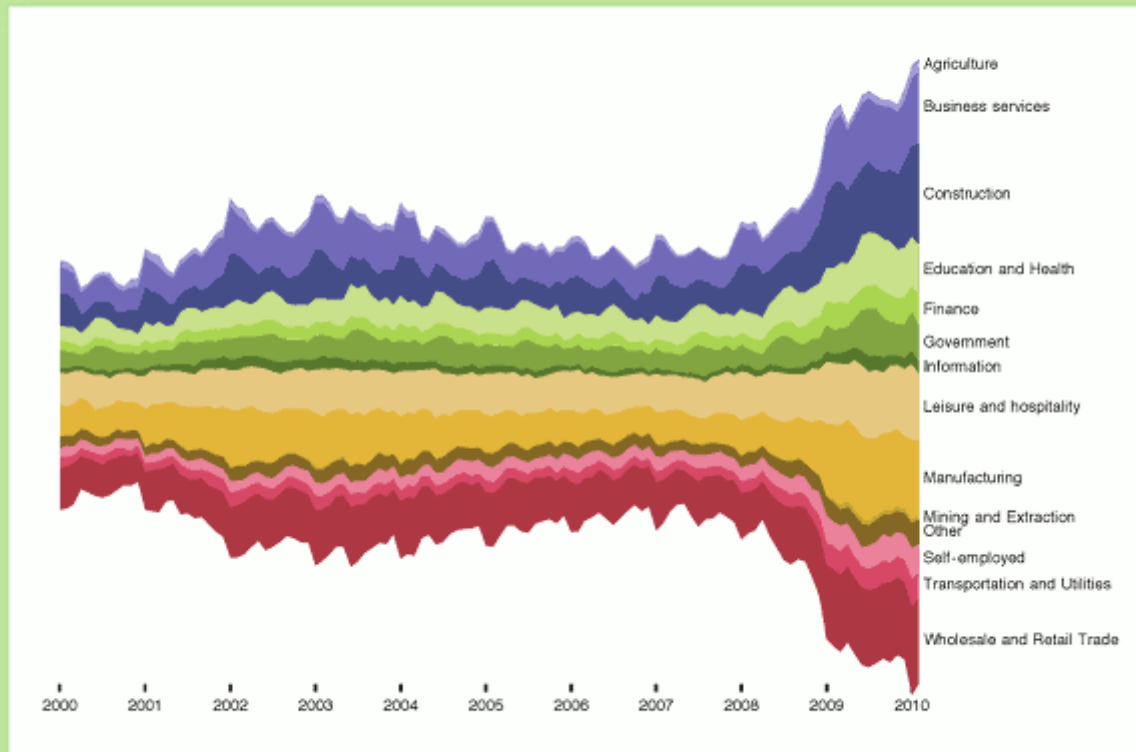
Index Chart of Selected Technology Stocks, 2000-2010

Source: Yahoo! Finance; <http://hci.stanford.edu/jheer/files/zoo/lex/time/index-chart.html>

[Click for full-size version \(requires SVG-compliant browser\)](#)
Source: Yahoo! Finance

Stacked Graphs

Other forms of time-series data may be better seen in aggregate. By stacking area charts on top of each other, we arrive at a visual summation of time-series values—a *stacked graph*. This type of graph (sometimes called a *stream graph*) depicts aggregate patterns and often supports drill-down into a subset of individual series. The chart in figure 1B shows the number of unemployed workers in the United States over the past decade, subdivided by industry. While such charts have proven popular in recent years, they do have some notable limitations. A stacked graph does not support negative numbers and is meaningless for data that should not be summed (temperatures, for example). Moreover, stacking may make it difficult to accurately interpret trends that lie atop other curves. Interactive search and filtering is often used to compensate for this problem.

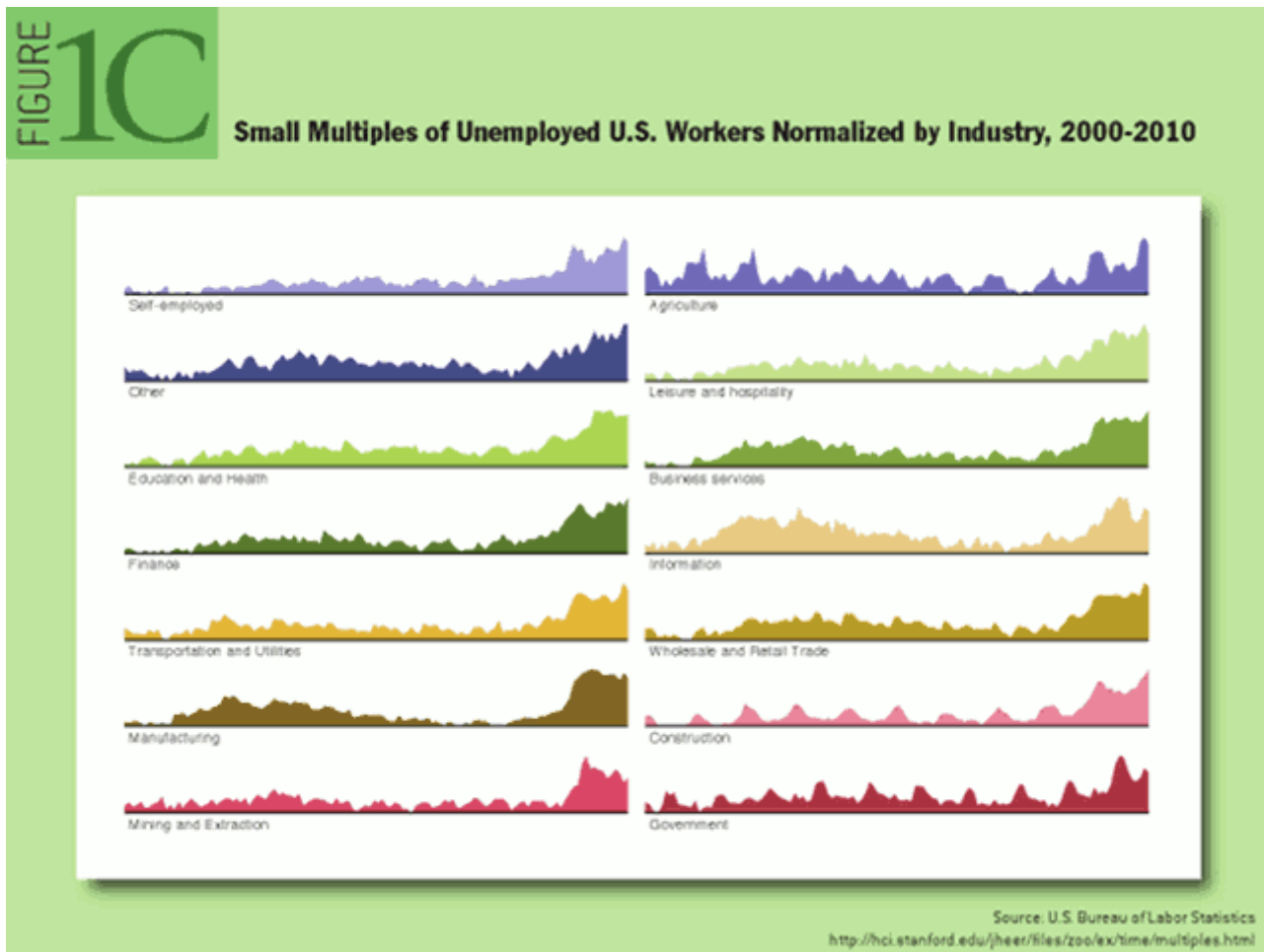
FIGURE 1B**Stacked Graph of Unemployed U.S. Workers by Industry, 2000-2010**

Source: U.S. Bureau of Labor Statistics
<http://hci.stanford.edu/jheer/files/zoo/ex/time/stack.html>

[Click for full-size version \(requires SVG-compliant browser\)](#)
Source: [U.S. Bureau of Labor Statistics](#)

Small Multiples

In lieu of stacking, multiple time series can be plotted within the same axes, as in the index chart. Placing multiple series in the same space may produce overlapping curves that reduce legibility, however. An alternative approach is to use *small multiples*: showing each series in its own chart. In figure 1C we again see the number of unemployed workers, but normalized within each industry category. We can now more accurately see both overall trends and seasonal patterns in each sector. While we are considering time-series data, note that small multiples can be constructed for just about any type of visualization: bar charts, pie charts, maps, etc. This often produces a more effective visualization than trying to coerce all the data into a single plot.

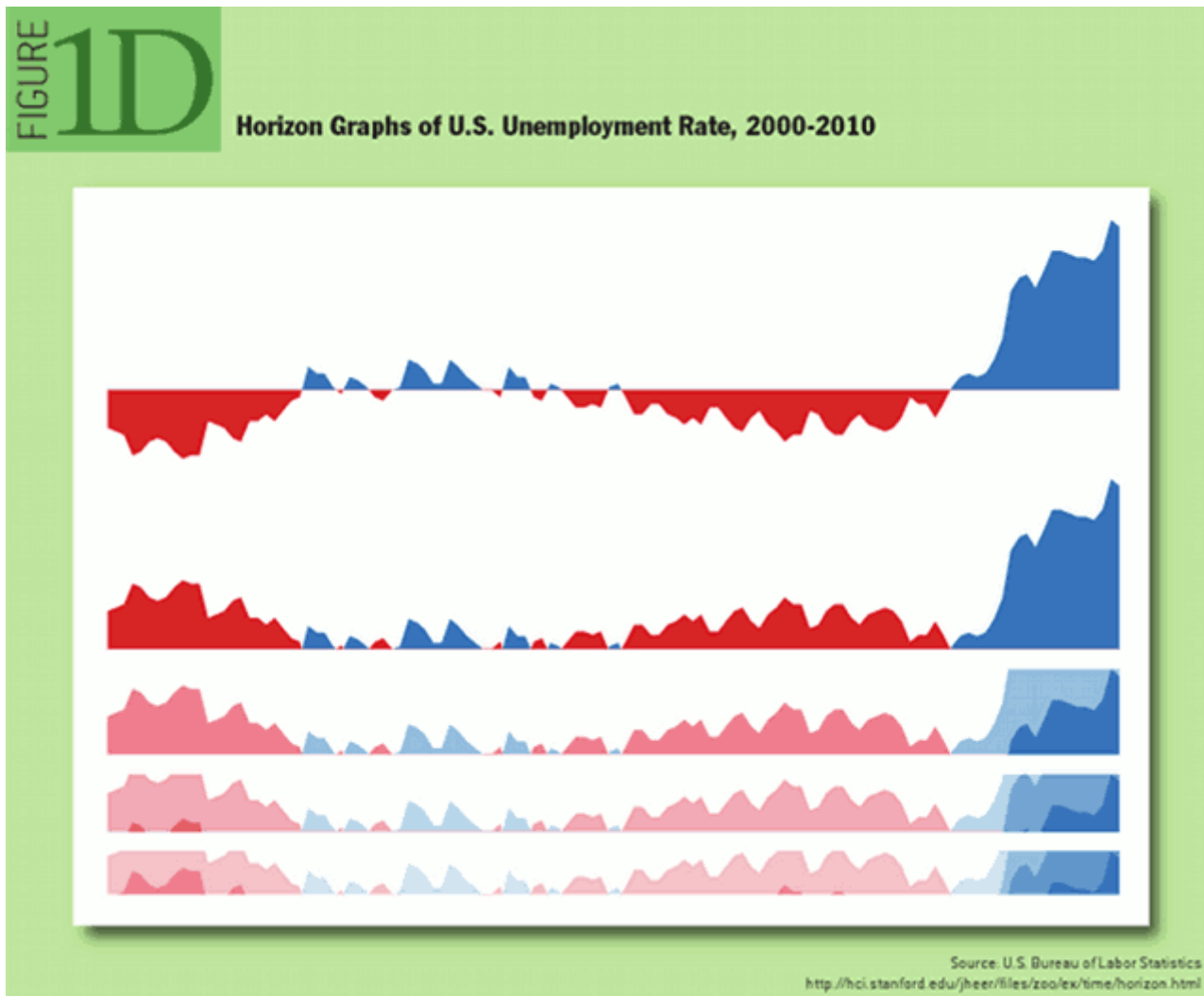


[Click for full-size version \(requires SVG-compliant browser\)](#)

Source: U.S. Bureau of Labor Statistics

Horizon Graphs

What happens when you want to compare even more time series at once? The *horizon graph* is a technique for increasing the *data density* of a time-series view while preserving resolution. Consider the four graphs shown in figure 1D. The first one is a standard area chart, with positive values colored blue and negative values colored red. The second graph "mirrors" negative values into the same region as positive values, doubling the data density of the area chart. The third chart—a horizon graph—doubles the data density yet again by dividing the graph into bands and layering them to create a nested form. The result is a chart that preserves data resolution but uses only a quarter of the space. Although the horizon graph takes some time to learn, it has been found to be more effective than the standard plot when the chart sizes get quite small.



[Click for full-size version \(requires SVG-compliant browser\)](#)

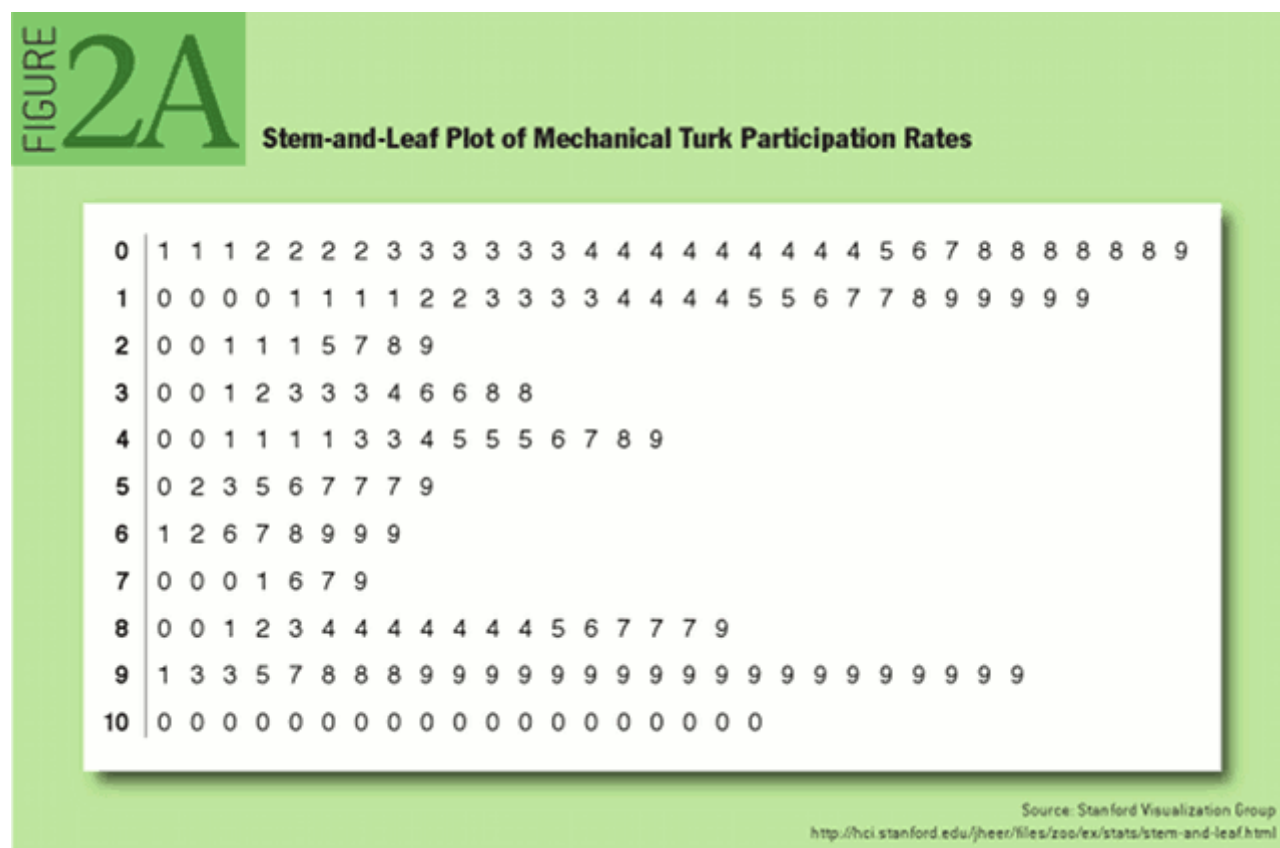
Source: U.S. Bureau of Labor Statistics

Statistical Distributions

Other visualizations have been designed to reveal how a set of numbers is distributed and thus help an analyst better understand the statistical properties of the data. Analysts often want to fit their data to statistical models, either to test hypotheses or predict future values, but an improper choice of model can lead to faulty predictions. Thus, one important use of visualizations is *exploratory data analysis*: gaining insight into how data is distributed to inform data transformation and modeling decisions. Common techniques include the *histogram*, which shows the prevalence of values grouped into bins, and the *box-and-whisker plot*, which can convey statistical features such as the mean, median, quartile boundaries, or extreme outliers. In addition, a number of other techniques exist for assessing a distribution and examining interactions between multiple dimensions.

Stem-and-Leaf Plots

For assessing a collection of numbers, one alternative to the histogram is the *stem-and-leaf plot*. It typically bins numbers according to the first significant digit, and then stacks the values within each bin by the second significant digit. This minimalistic representation uses the data itself to paint a frequency distribution, replacing the "information-empty" bars of a traditional histogram bar chart and allowing one to assess both the overall distribution and the contents of each bin. In figure 2A, the stem-and-leaf plot shows the distribution of completion rates of workers completing crowd-sourced tasks on Amazon's Mechanical Turk. Note the multiple clusters: one group clusters around high levels of completion (99–100 percent); at the other extreme is a cluster of Turkers who complete only a few tasks (~10 percent) in a group.



[Click for full-size version \(requires SVG-compliant browser\)](#)

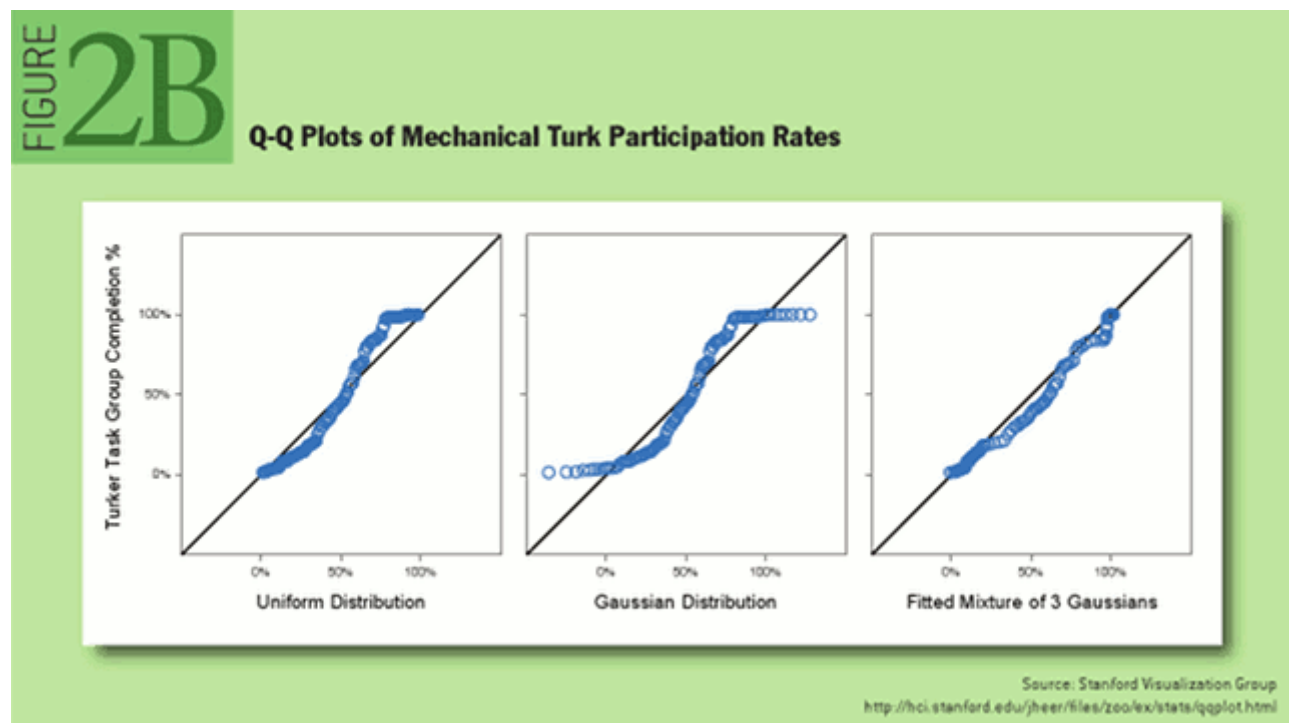
Source: [Stanford Visualization Group](#)

Q-Q Plots

Though the histogram and the stem-and-leaf plot are common tools for assessing a frequency distribution, the *Q-Q (quantile-quantile) plot* is a more powerful tool. The Q-Q plot compares two probability distributions by graphing their [quantiles](#) against each other. If the two are similar, the plotted values will lie

roughly along the central diagonal. If the two are linearly related, values will again lie along a line, though with varying slope and intercept.

Figure 2B shows the same Mechanical Turk participation data compared with three statistical distributions. Note how the data forms three distinct components when compared with uniform and normal (Gaussian) distributions: this suggests that a statistical model with three components might be more appropriate, and indeed we see in the final plot that a fitted mixture of three normal distributions provides a better fit. Though powerful, the Q-Q plot has one obvious limitation in that its effective use requires that viewers possess some statistical knowledge.



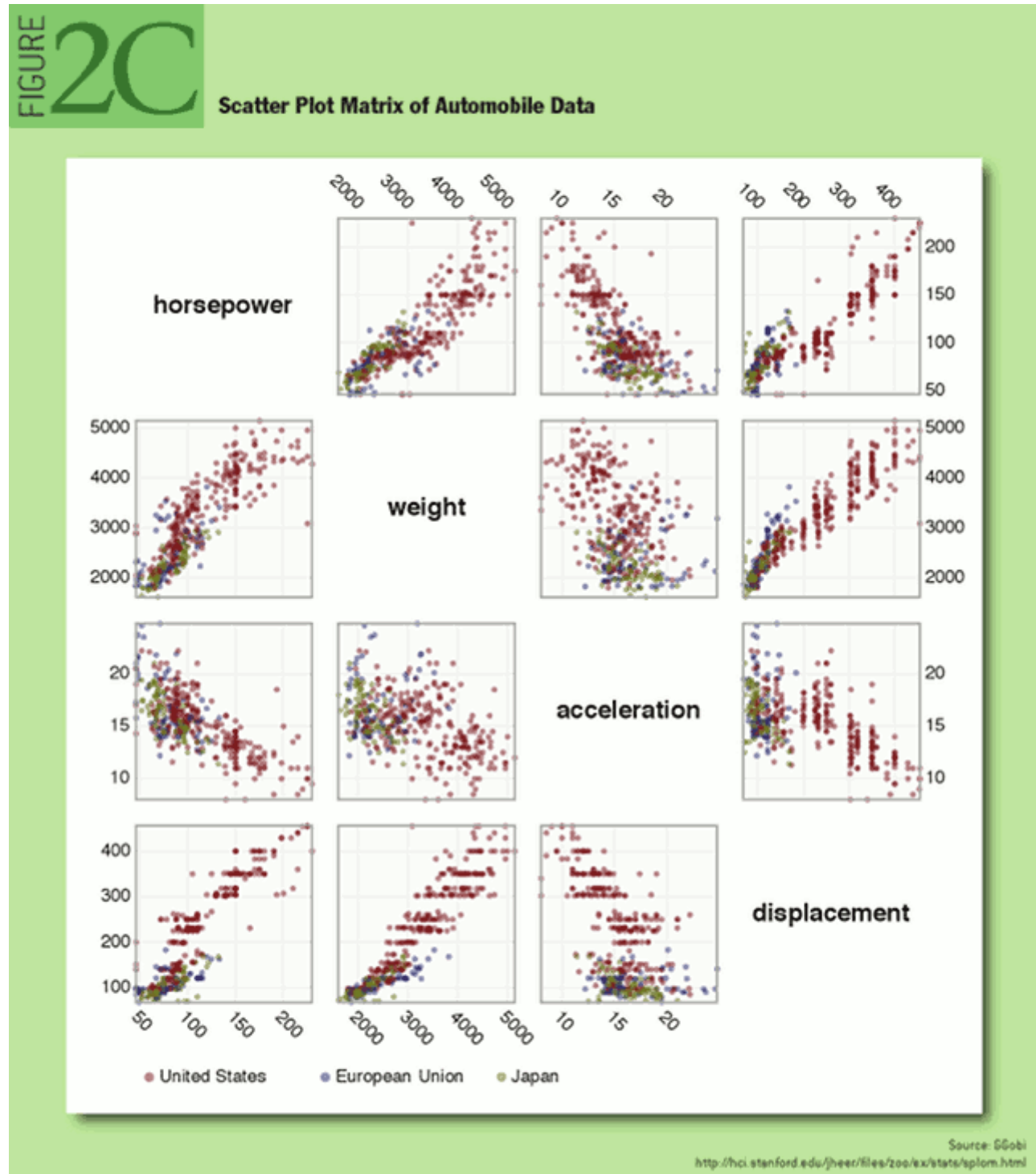
[Click for full-size version \(requires SVG-compliant browser\)](#)

Source: [Stanford Visualization Group](#)

SPLOM (Scatter Plot Matrix)

Other visualization techniques attempt to represent the relationships among multiple variables. Multivariate data occurs frequently and is notoriously hard to represent, in part because of the difficulty of mentally picturing data in more than three dimensions. One technique to overcome this problem is to use small multiples of scatter plots showing a set of pairwise relations among variables, thus creating the *SPLOM* (*scatter plot matrix*). A SPLOM enables visual inspection of correlations between any pair of variables.

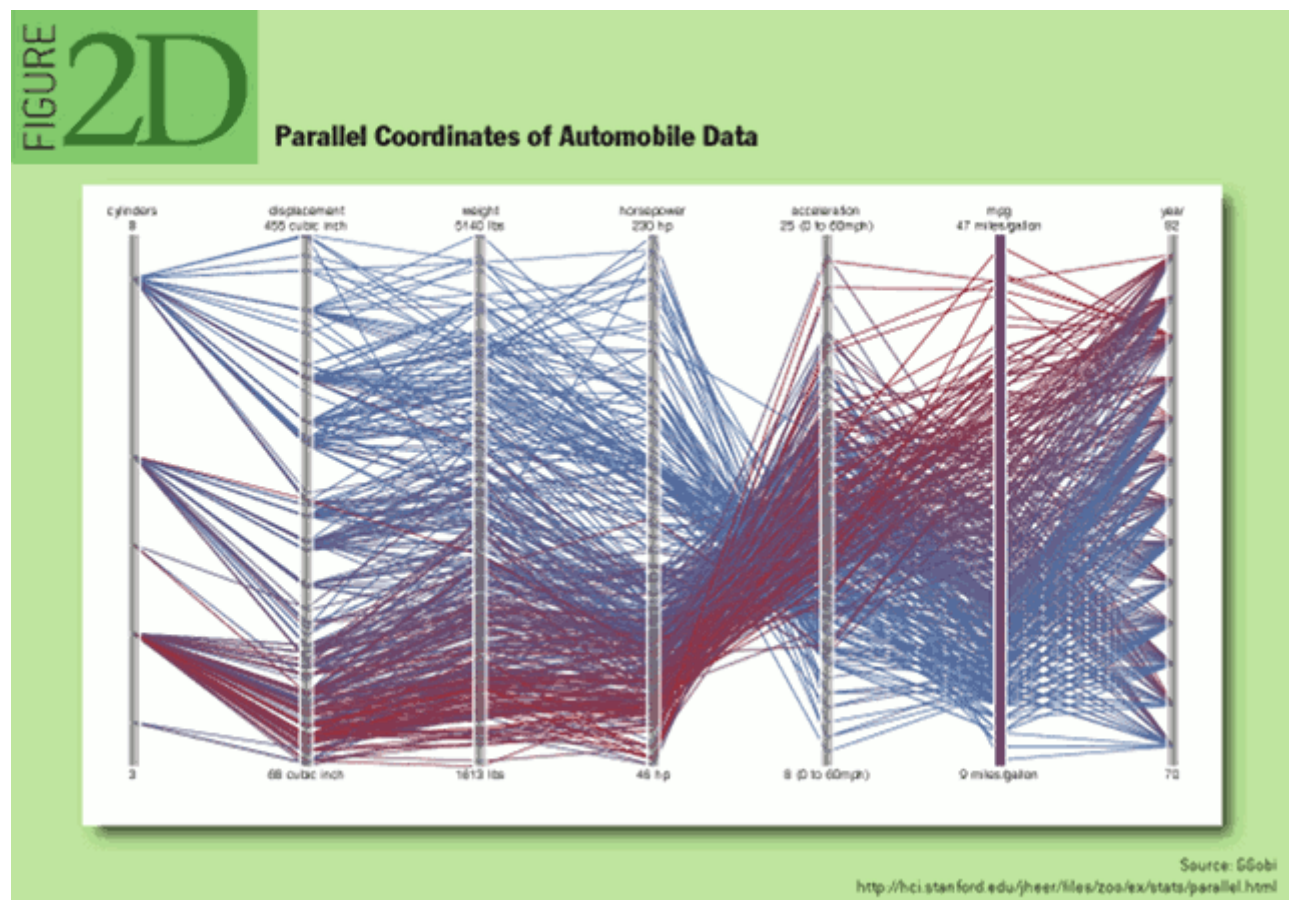
In figure 2C a scatter plot matrix is used to visualize the attributes of a database of automobiles, showing the relationships among horsepower, weight, acceleration, and displacement. Additionally, interaction techniques such as *brushing-and-linking*—in which a selection of points on one graph highlights the same points on all the other graphs—can be used to explore patterns within the data.



Click for full-size version (requires SVG-compliant browser)
 Source: GGobi

Parallel Coordinates

Parallel coordinates (||-coord), shown in figure 2D, take a different approach to visualizing multivariate data. Instead of graphing every pair of variables in two dimensions, we repeatedly plot the data on parallel axes and then connect the corresponding points with lines. Each poly-line represents a single row in the database, and line crossings between dimensions often indicate inverse correlation. Reordering dimensions can aid pattern finding, as can interactive querying to filter along one or more dimensions. Another advantage of parallel coordinates is that they are relatively compact, so many variables can be shown simultaneously.



[Click for full-size version \(requires SVG-compliant browser\)](#)

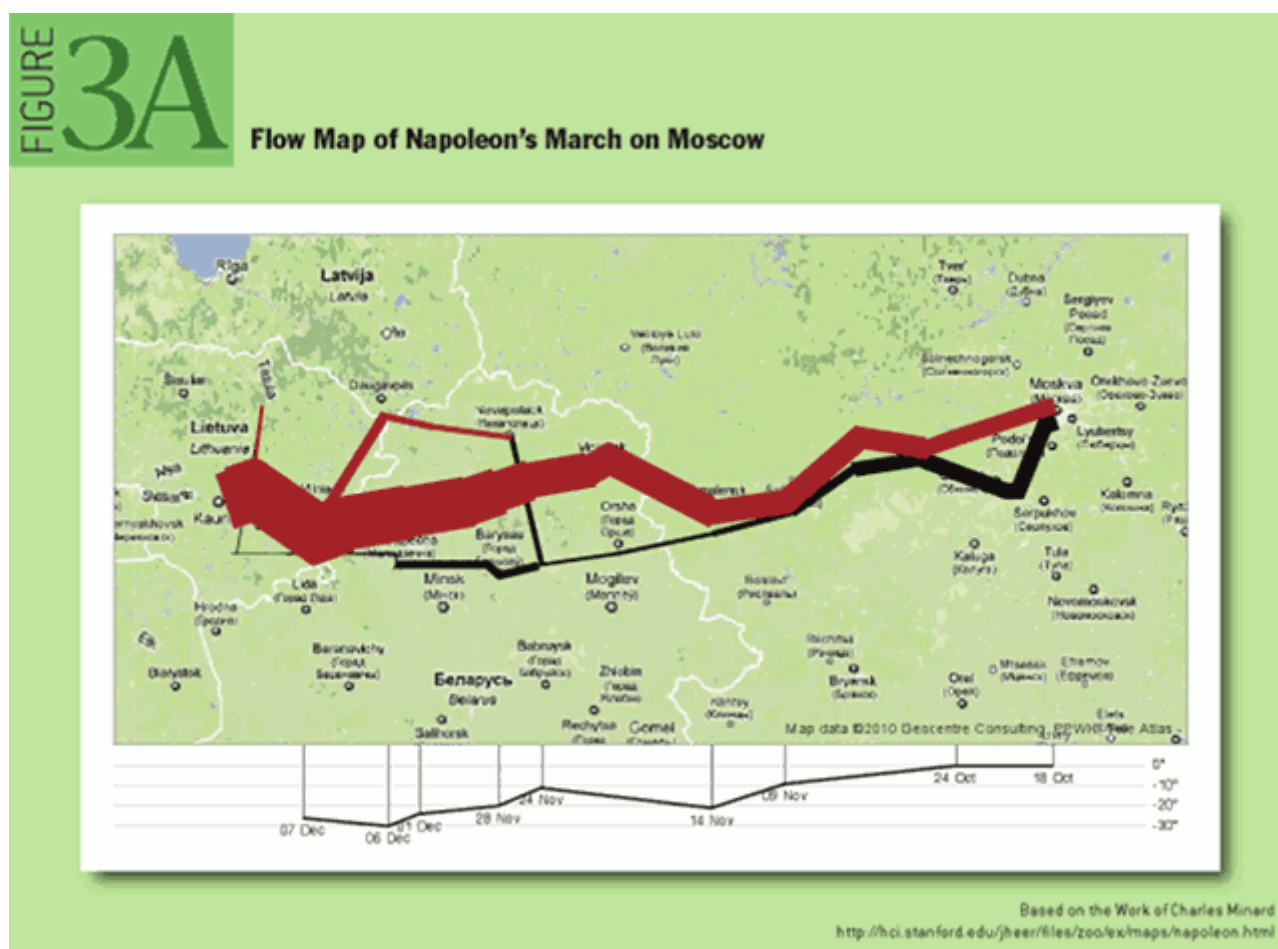
Source: GGobi

Maps

Although a map may seem a natural way to visualize geographical data, it has a long and rich history of design. Many maps are based upon a *cartographic projection*: a mathematical function that maps the three-dimensional geometry of the Earth to a two-dimensional image. Other maps knowingly distort or abstract geographic features to tell a richer story or highlight specific data.

Flow Maps

By placing stroked lines on top of a geographic map, a *flow map* can depict the movement of a quantity in space and (implicitly) in time. Flow lines typically encode a large amount of multivariate information: path points, direction, line thickness, and color can all be used to present dimensions of information to the viewer. Figure 3A is a modern interpretation of Charles Minard's depiction of Napoleon's ill-fated march on Moscow. Many of the greatest flow maps also involve subtle uses of distortion, as geography is modified to accommodate or highlight flows.



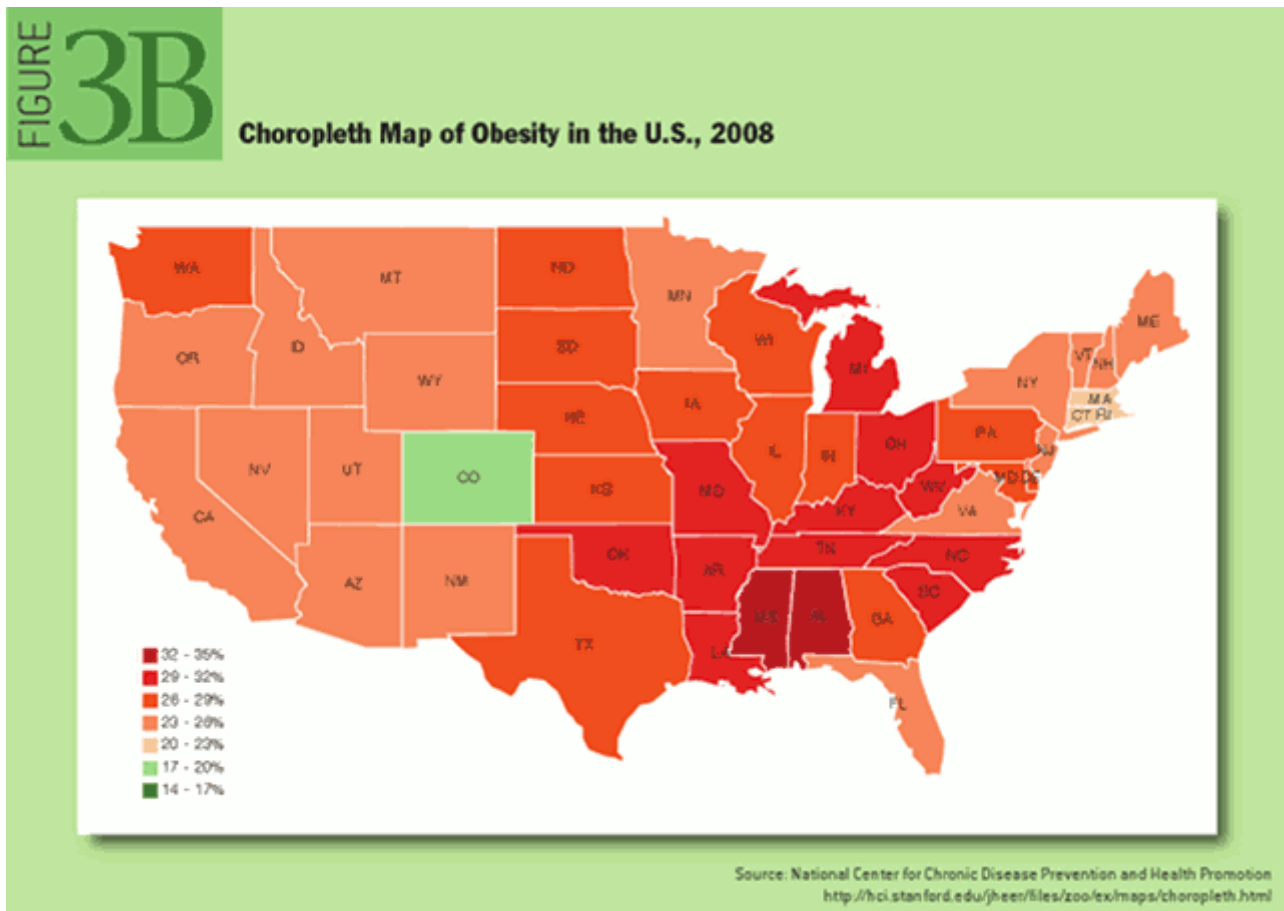
[Click for full-size version \(requires SVG-compliant browser\)](#)

Based on the Work of [Charles Minard](#)

Choropleth Maps

Data is often collected and aggregated by geographical areas such as states. A standard approach to communicating this data is to use a color encoding of the geographical area, resulting in a *choropleth map*. Figure 3B uses a color encoding to communicate the prevalence of obesity in each state in the U.S. Though this is a widely used visualization technique, it requires some care. One common error is

to encode raw data values (such as population) rather than using normalized values to produce a density map. Another issue is that one's perception of the shaded value can also be affected by the underlying area of the geographic region.

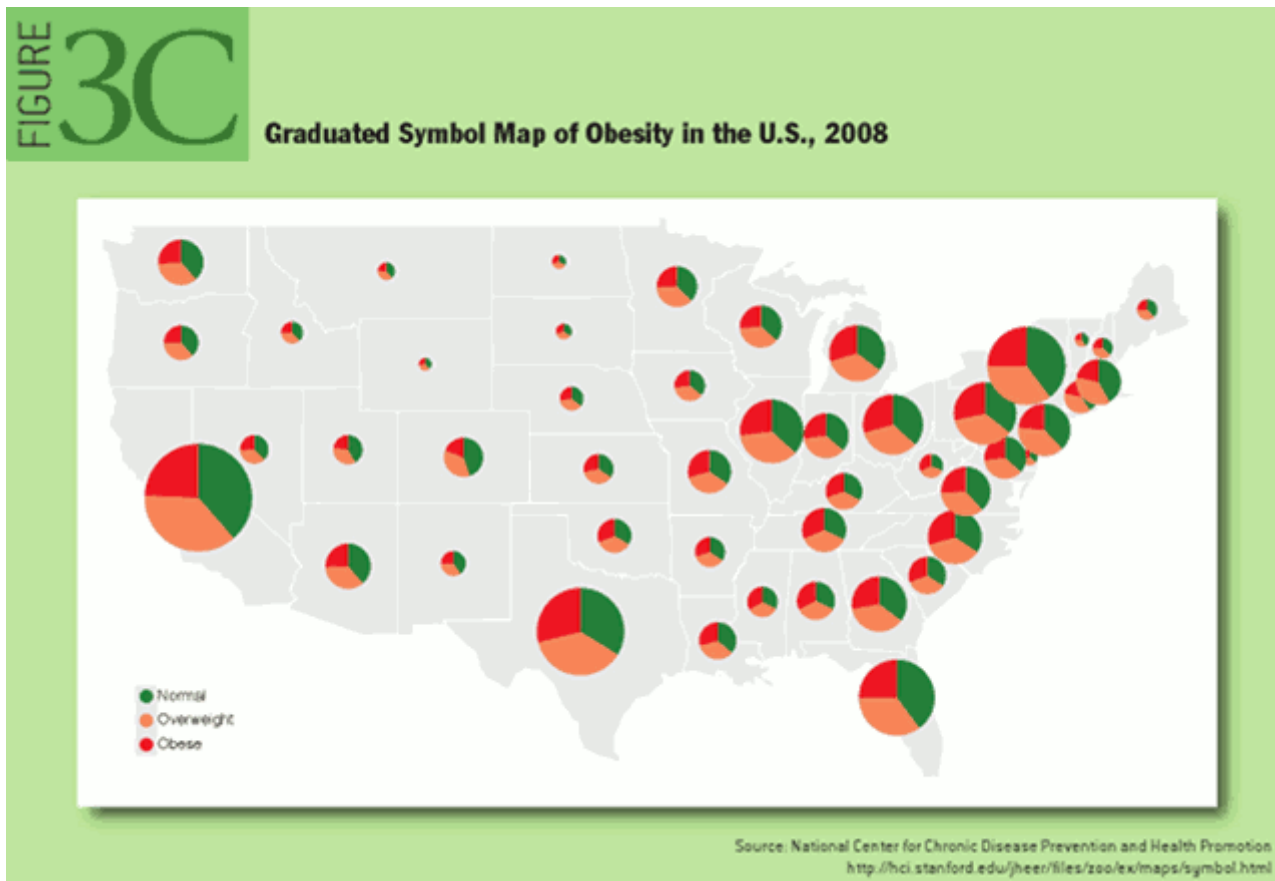


[Click for full-size version \(requires SVG-compliant browser\)](#)

Source: [National Center for Chronic Disease Prevention and Health Promotion](#)

Graduated Symbol Maps

An alternative to the choropleth map is the *graduated symbol map*, which places symbols over an underlying map. This approach avoids confounding geographic area with data values and allows for more dimensions to be visualized (e.g., symbol size, shape, and color). In addition to simple shapes such as circles, graduated symbol maps may use more complicated glyphs such as pie charts. In figure 3C, total circle size represents a state's population, and each slice indicates the proportion of people with a specific BMI rating.



[Click for full-size version \(requires SVG-compliant browser\)](#)

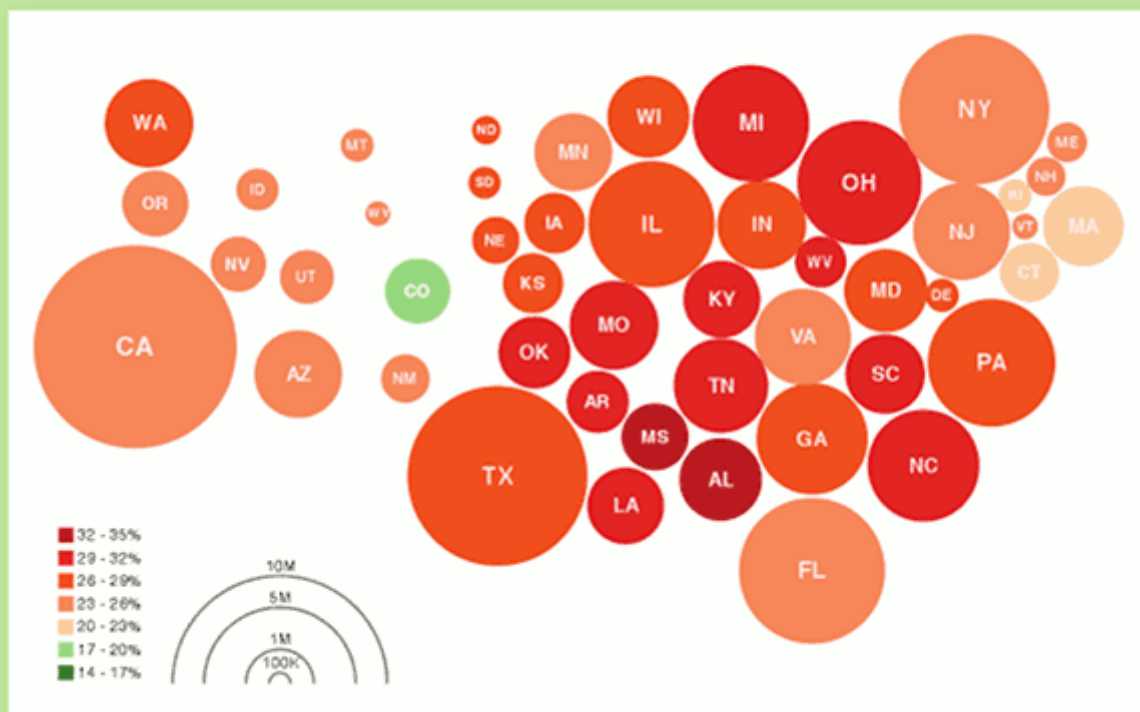
Source: [National Center for Chronic Disease Prevention and Health Promotion](#)

Cartograms

A *cartogram* distorts the shape of geographic regions so that the area directly encodes a data variable. A common example is to redraw every country in the world sizing it proportionally to population or gross domestic product. Many types of cartograms have been created; in figure 3D we use the *Dorling cartogram*, which represents each geographic region with a sized circle, placed so as to resemble the true geographic configuration. In this example, circular area encodes the total number of obese people per state, and color encodes the percentage of the total population that is obese.

FIGURE 3D

Dorling Cartogram of Obesity in the U.S., 2008



Source: National Center for Chronic Disease Prevention and Health Promotion
<http://hci.stanford.edu/jheer/files/zoo/ex/maps/cartogram.html>

[Click for full-size version \(requires SVG-compliant browser\)](#)

Source: [National Center for Chronic Disease Prevention and Health Promotion](#)

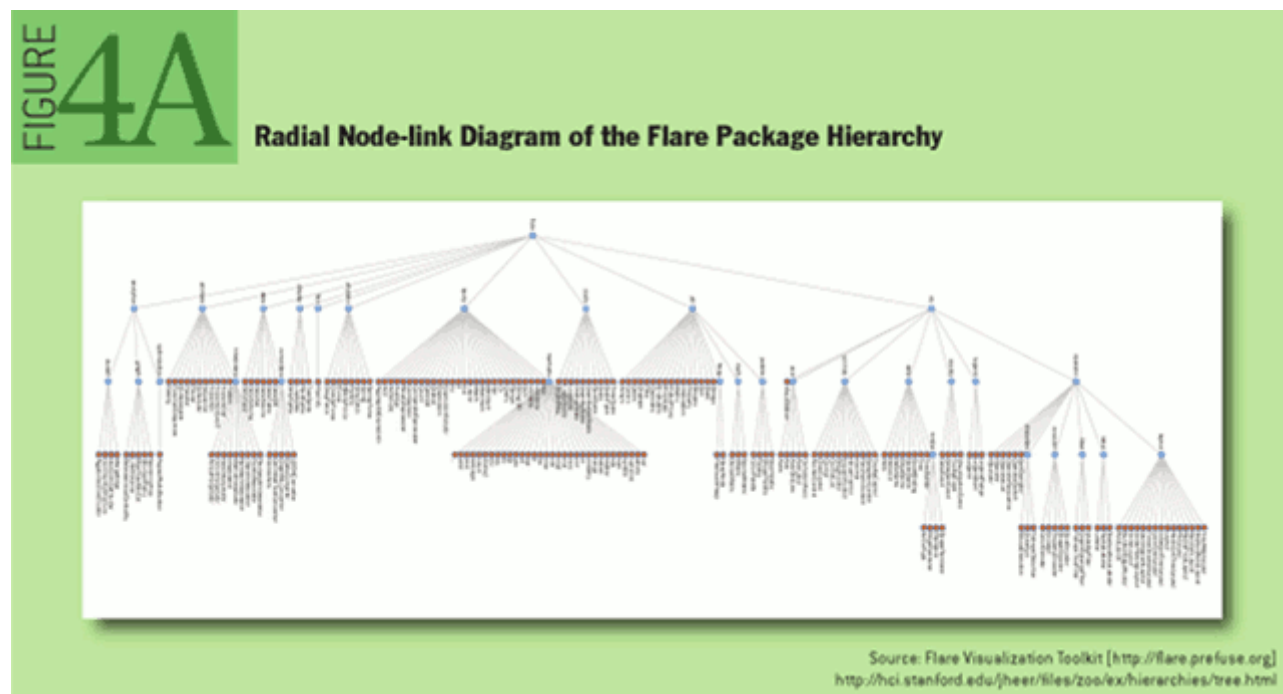
Hierarchies

While some data is simply a flat collection of numbers, most can be organized into natural hierarchies. Consider: spatial entities, such as counties, states, and countries; command structures for businesses and governments; software packages and phylogenetic trees. Even for data with no apparent hierarchy, statistical methods (e.g., *k-means clustering*) may be applied to organize data empirically. Special visualization techniques exist to leverage hierarchical structure, allowing rapid multiscale inferences: micro-observations of individual elements and macro-observations of large groups.

Node-link Diagrams

The word *tree* is used interchangeably with *hierarchy*, as the fractal branches of an oak might mirror the nesting of data. If we take a two-dimensional blueprint of a tree, we have a popular choice for visualizing hierarchies: a *node-link diagram*. Many different tree-layout algorithms have been designed; the

Reingold–Tilford algorithm, used in figure 4A on a package hierarchy of software classes, produces a tidy result with minimal wasted space.



[Click for full-size version \(requires SVG-compliant browser\)](#)

Source: [Flare Visualization Toolkit \(http://flare.prefuse.org\)](http://flare.prefuse.org)

An alternative visualization scheme is the *dendrogram* (or cluster) algorithm, which places leaf nodes of the tree at the same level. Thus, in the diagram in figure 4B, the classes (orange leaf nodes) are on the diameter of the circle, with the packages (blue internal nodes) inside. Using polar rather than Cartesian coordinates has a pleasing aesthetic, while using space more efficiently.

Cartesian Node-link Diagram of the Flare Package Hierarchy

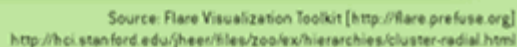


FIGURE 4C

Indented Tree Layout of the Flare Package Hierarchy



Source: Flare Visualization Toolkit (<http://flare.prefuse.org>)
<http://hci.stanford.edu/jheer/files/zoo/ex/hierarchies/indent.html>

Click for full-size version (requires SVG-compliant browser)
 Source: Flare Visualization Toolkit (<http://flare.prefuse.org>)

Adjacency Diagrams

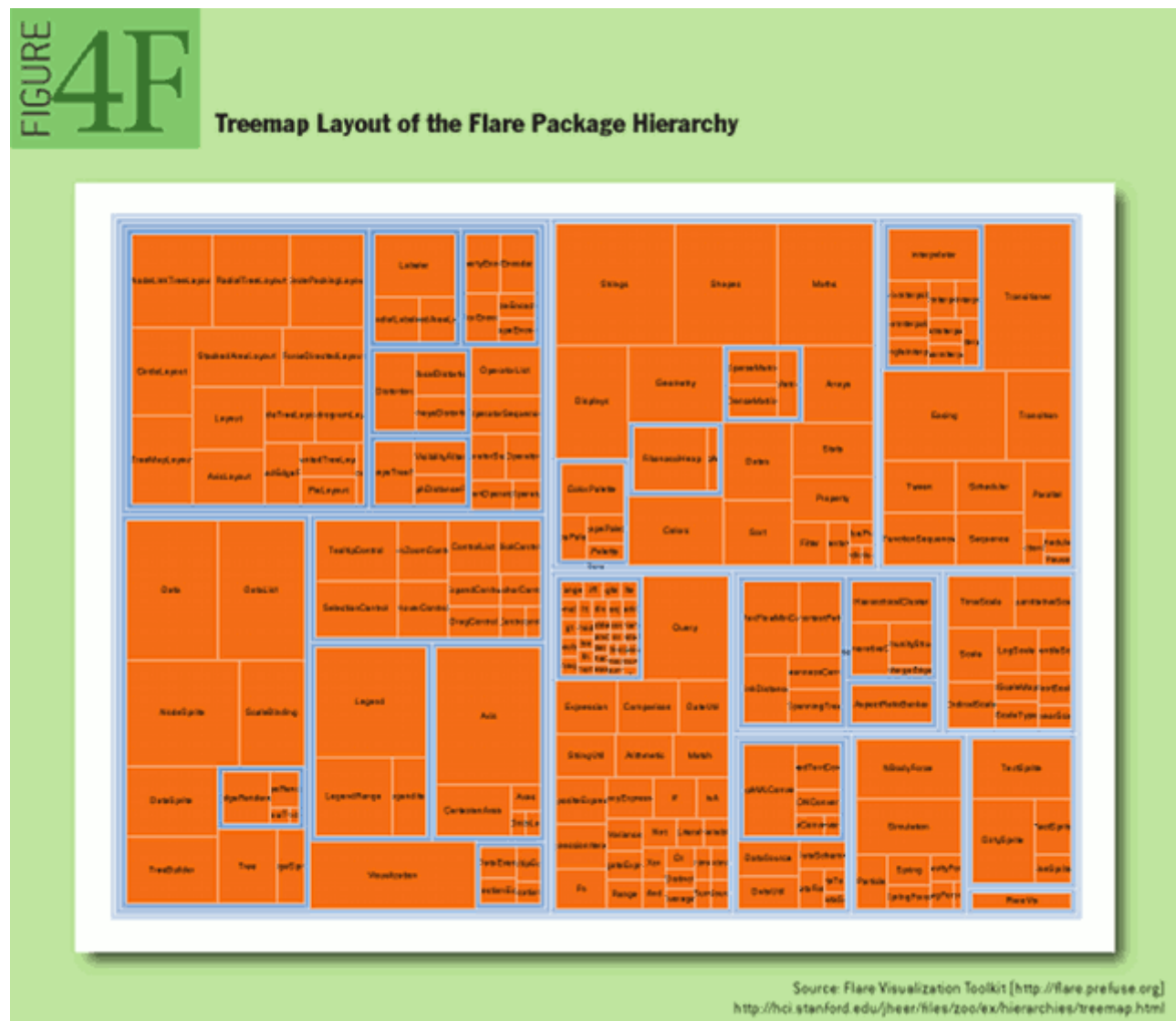
The *adjacency diagram* is a space-filling variant of the node-link diagram; rather than drawing a link between parent and child in the hierarchy, nodes are drawn as solid areas (either arcs or bars), and their placement relative to adjacent nodes reveals their position in the hierarchy. The icicle layout in figure 4D is similar to the first node-link diagram in that the root node appears at the top, with child nodes underneath. Because the nodes are now space-filling, however, we can use a length encoding for the size of software classes and packages. This reveals an additional dimension that would be difficult to show in a node-link diagram.



Click for full-size version (requires SVG-compliant browser)
 Source: Flare Visualization Toolkit (<http://flare.prefuse.org>)

The sunburst layout, shown in figure 4E, is equivalent to the icicle layout, but in polar coordinates. Both are implemented using a partition layout, which can also generate a node-link diagram. Similarly, the previous cluster layout can be used to generate a space-filling adjacency diagram in either Cartesian or polar coordinates.

encoding is sometimes used. *Squarified* treemaps use approximately square rectangles, which offer better readability and size estimation than a naive "slice-and-dice" subdivision. Fancier algorithms such as Voronoi and jigsaw treemaps also exist but are less common.

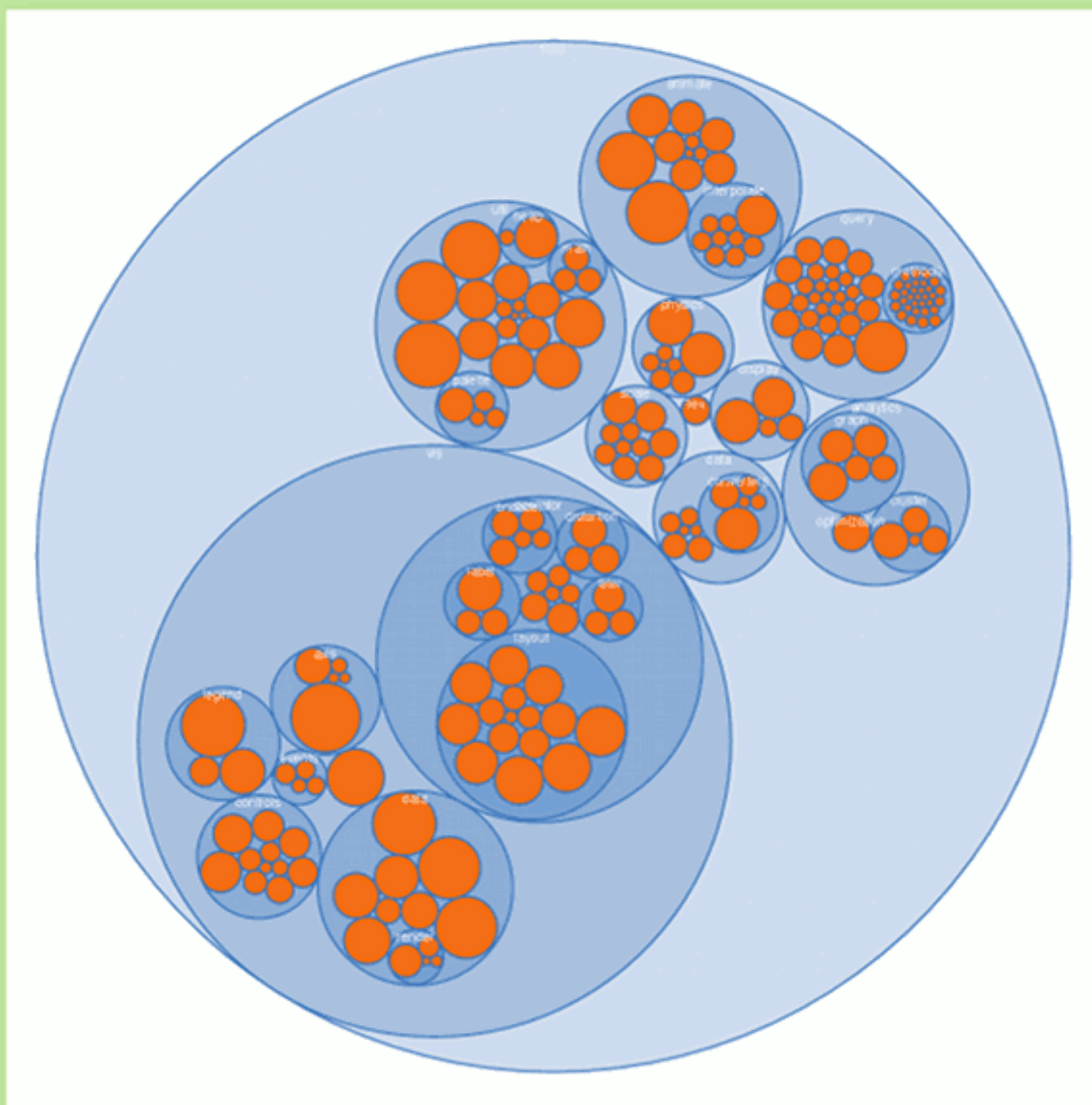


Click for full-size version (requires SVG-compliant browser)
 Source: Flare Visualization Toolkit (<http://flare.prefuse.org>)

By packing circles instead of subdividing rectangles, we can produce a different sort of enclosure diagram that has an almost organic appearance. Although it does not use space as efficiently as a treemap, the "wasted space" of the *circle-packing layout*, shown in figure 4G, effectively reveals the hierarchy. At the same time, node sizes can be rapidly compared using area judgments.

FIGURE 4G

Nested Circles Layout of the Flare Package Hierarchy



Source: Flare Visualization Toolkit [<http://flare.prefuse.org>]
<http://hci.stanford.edu/jheer/files/zoo/ex/hierarchies/pack.html>

[Click for full-size version \(requires SVG-compliant browser\)](#)

Source: Flare Visualization Toolkit (<http://flare.prefuse.org>)

Networks

In addition to organization, one aspect of data that we may wish to explore through visualization is relationship. For example, given a social network, who is friends with whom? Who are the central players? What cliques exist? Who, if anyone, serves as a bridge between disparate groups? Abstractly, a hierarchy is a specialized form of network: each node has exactly one link to its parent, while the root node has no links. Thus node-link diagrams are also used to visualize

networks, but the loss of hierarchy means a different algorithm is required to position nodes.

Mathematicians use the formal term *graph* to describe a network. A central challenge in graph visualization is computing an effective layout. Layout techniques typically seek to position closely related nodes (in terms of *graph distance*, such as the number of links between nodes, or other metrics) close in the drawing; critically, *unrelated* nodes must also be placed far enough apart to differentiate relationships. Some techniques may seek to optimize other visual features—for example, by minimizing the number of edge crossings.

Force-directed Layouts

A common and intuitive approach to network layout is to model the graph as a physical system: nodes are charged particles that repel each other, and links are dampened springs that pull related nodes together. A physical simulation of these forces then determines the node positions; approximation techniques that avoid computing all pairwise forces enable the layout of large numbers of nodes. In addition, interactivity allows the user to direct the layout and jiggle nodes to disambiguate links. Such a *force-directed layout* is a good starting point for understanding the structure of a general undirected graph. In figure 5A we use a force-directed layout to view the network of character co-occurrence in the chapters of Victor Hugo's classic novel, *Les Misérables*. Node colors depict cluster memberships computed by a community-detection algorithm.

FIGURE 5A**Force-directed Layout of *Les Misérables* Character Co-occurrences**

Source: Knuth, D. E. 1993. *The Stanford GraphBase: A Platform for Combinatorial Computing*, Addison-Wesley.
<http://hci.stanford.edu/jheer/files/toolbox/networks/force.html>

[Click for full-size version \(requires SVG-compliant browser\)](#)

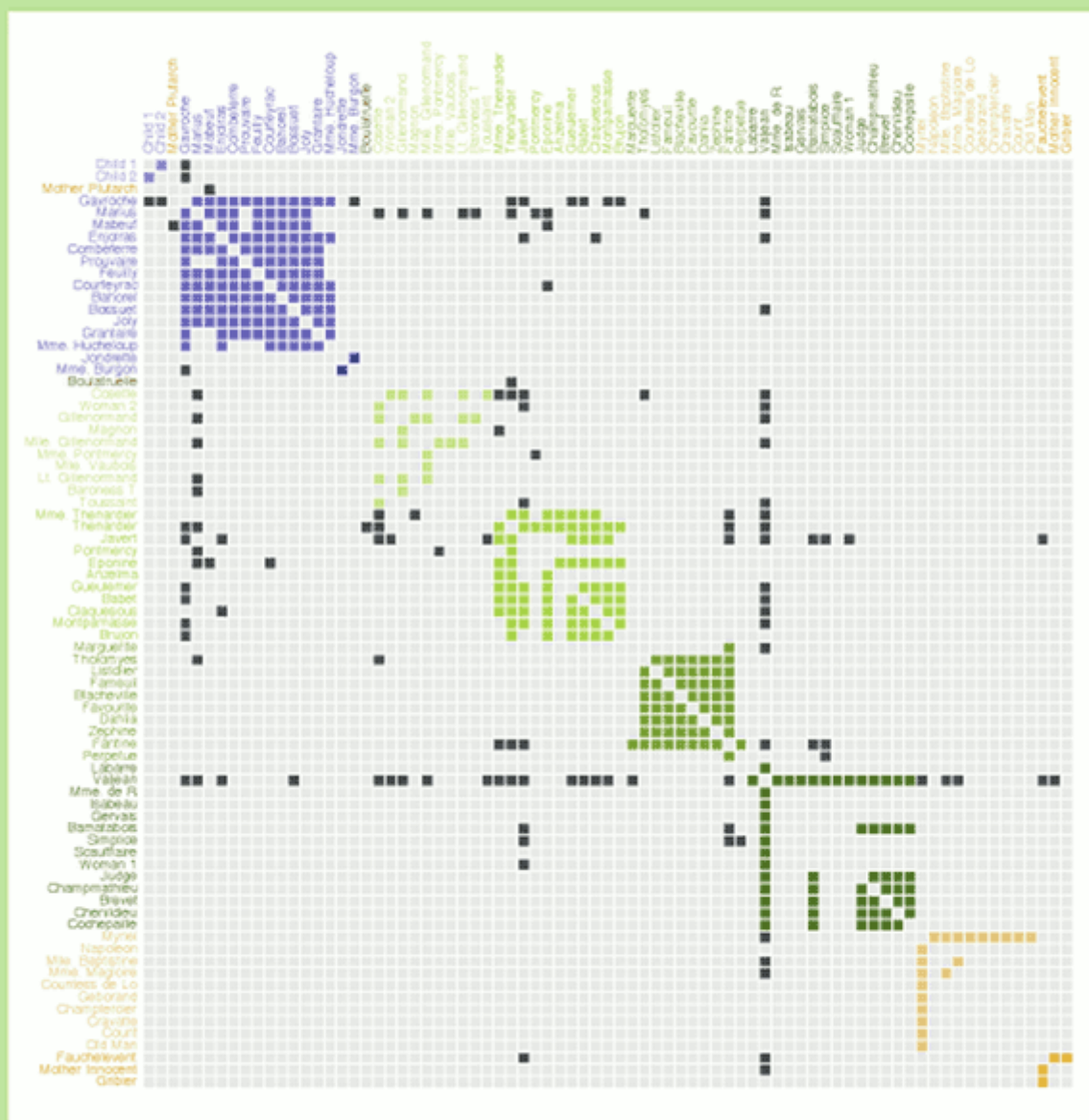
Source: Knuth, D. E. 1993. *The Stanford GraphBase: A Platform for Combinatorial Computing*, Addison-Wesley.

Arc Diagrams

An *arc diagram*, shown in figure 5B, uses a one-dimensional layout of nodes, with circular arcs to represent links. Though an arc diagram may not convey the overall structure of the graph as effectively as a two-dimensional layout, with a good ordering of nodes it is easy to identify cliques and bridges. Further, as with the indented-tree layout, multivariate data can easily be displayed alongside nodes. The problem of sorting the nodes in a manner that reveals underlying cluster structure is formally called *seriation* and has diverse applications in visualization, statistics, and even archaeology.

FIGURE 5C

Matrix View of Les Misérables Character Co-occurrences



Source: Knuth, D. E. 1993. *The Stanford GraphBase: A Platform for Combinatorial Computing*, Addison-Wesley.
<http://hpi.stanford.edu/jheer/files/xoo/ex/networks/matrix.html>

[Click for full-size version \(requires SVG-compliant browser\)](#)

Source: Knuth, D. E. 1993. *The Stanford GraphBase: A Platform for Combinatorial Computing*, Addison-Wesley.

Conclusion

We have arrived at the end of our tour and hope that the reader has found examples both intriguing and practical. Though we have visited a number of visual encoding and interaction techniques, many more species of visualization exist in the wild, and others await discovery. Emerging domains such as bioinformatics and text visualization are driving researchers and designers to

continually formulate new and creative representations or find more powerful ways to apply the classics. In either case, the DNA underlying all visualizations remains the same: the principled mapping of data variables to visual features such as position, size, shape, and color. As you leave the zoo and head back into the wild, try deconstructing the various visualizations crossing your path. Perhaps you can design a more effective display?

Q

Additional Resources

Few, S. 2009. *Now I See It: Simple Visualization Techniques for Quantitative Analysis*. Analytics Press.

Tufte, E. 1983. *The Visual Display of Quantitative Information*. Graphics Press.

Tufte, E. 1990. *Envisioning Information*. Graphics Press.

Ware, C. 2008. *Visual Thinking for Design*. Morgan Kaufmann.

Wilkinson, L. 1999. *The Grammar of Graphics*. Springer.

Visualization Development Tools

Prefuse (<http://prefuse.org/>): Java API for information visualization.

Prefuse Flare (<http://flare.prefuse.org/>): ActionScript 3 library for data visualization in the Adobe Flash Player.

Processing (<http://processing.org/>): Popular language and IDE for graphics and interaction.

Protovis (<http://vis.stanford.edu/protovis/>): JavaScript tool for Web-based visualization.

The Visualization Toolkit (<http://vtk.org/>): Library for 3D and scientific visualization.

LOVE IT, HATE IT? LET US KNOW

feedback@queue.acm.org

Jeffrey Heer is an assistant professor of computer science at Stanford University, where he works on human-computer interaction, visualization, and social

computing. His research investigates the perceptual, cognitive, and social factors involved in making sense of large data collections, resulting in new interactive systems for visual analysis and communication. He has also led the design of the Prefuse, Flare, and Protovis visualization toolkits, in use by researchers, corporations, and thousands of data enthusiasts. Heer is the recipient of the 2009 ACM CHI Best Paper Award and Faculty Awards from IBM and Intel. In 2009 he was named to MIT Technology Review's TR35. He holds B.S., M.S., and Ph.D. degrees in Computer Science from the University of California, Berkeley.

Michael Bostock received the BSE degree in computer science in 2000 from Princeton University. He is currently a Ph.D. student in the Department of Computer Science at Stanford University. His research interests include information visualization and software design. Before joining Stanford, he was a staff engineer at Google, where he developed search quality evaluation methodologies, experimental search user interfaces, and reusable software components such as the Google Collections Library. He is currently working on the Protovis visualization toolkit.

Vadim Ogievetsky is a Masters student at Stanford University specializing in Human-Computer Interaction. He is a core contributor to Protovis, an open-source web-based visualization toolkit. Ogievetsky received a First Class BA degree in Mathematics and Computer Science from the University of Oxford where he specialized in linear algebra and programming languages. In addition to visualization, his interests include massively parallel computing and computer controlled manufacturing processes.

© 2010 ACM 1542-7730/10/0500 \$10.00



Originally published in Queue vol. 8, no. 5—
see this item in the [ACM Digital Library](https://queue.acm.org/detail.cfm?id=1805128)

Tweet

Related:

David Crandall, Noah Snavely – [Modeling People and Places with Internet Photo Collections](#)

Understanding the world from the sea of online photos

Jeffrey Heer, Ben Shneiderman – [Interactive Dynamics for Visual Analysis](#)

A taxonomy of tools that support the fluent and flexible use of visualizations

Robert DeLine, Gina Venolia, Kael Rowan – [Software Development with Code Maps](#)

Could those ubiquitous hand-drawn code diagrams become a thing of the past?

Brendan Gregg – [Visualizing System Latency](#)

Heat maps are a unique and powerful way to visualize latency data. Explaining the results, however, is an ongoing challenge.

Comments

(newest first)

Displaying 10 most recent comments. [Read the full list here](#)

Becka Mays | Fri, 03 Nov 2017 23:11:48 UTC

Cool.....

frankg | Fri, 25 Jul 2014 15:27:30 UTC

I would avoid using red and green colors together since it could be an issue for color blind viewers.

Fernando | Wed, 19 Mar 2014 00:31:26 UTC

Very nice article, really help me in defining a dashboard for social media performance.

puneeth d poojary | Mon, 05 Aug 2013 13:38:07 UTC

Nice article, great work.....

Barrett Elise Harris | Fri, 13 Jul 2012 13:24:22 UTC

This is great. It's like a manifestation of perceptions. There are SO many. So many variables! WHEW! Good stuff. <3

mrugank | Sat, 16 Jul 2011 21:51:51 UTC

hi Jeffrey Heer, is tableau provided it has roots in stanford using any of your chart codes? I am a consultant from India currently working on developing a competing product to tableau. I would like to know whether i can make use of your services? We have developed a platform for data sharing and collection and currently i am looking for chart library APIs for data visualization.

Greg Hooper | Sat, 02 Oct 2010 19:04:06 UTC

What about tools like R – is that a different category of software? Sorry if that's a basic question, I'm just trying to learn what's what.

Roop | Fri, 01 Oct 2010 08:08:35 UTC

Fantastic roundup Protovis looks like a very promising library for our (hopefully) standards-based future.

Larry Tesler | Wed, 23 Jun 2010 22:34:00 UTC

Bryce, you mentioned Wordle. The authors mentioned it, too, in the interview accompanying the web version of the survey.

Cathryn McCormack | Tue, 22 Jun 2010 05:10:12 UTC

While mostly very interesting, I am surprised to see the graduated symbol map 3C and the cartogram 3D included. The graduated symbol map includes pie charts – something abandoned by Stephen Few in Show Me the Numbers because they communicate information poorly. Both pie charts and the round circles in the cartogram 3D would be rejected by him as effective based on the fact it is difficult for human visual perception to accurately assign quantitative values to 2D areas. But I'm glad to see you included my favourite chart of all time – Minard's Napoleonic invasion of Russia!

Displaying 10 most recent comments. [Read the full list here](#)

Post a Comment:

Comment: (Required – 4,000 character limit – HTML syntax is **not** allowed and **will be removed**)



© 2018 ACM, Inc. All Rights Reserved.