Better Information from Better Data Visualization

Nicole Arksey, INETCO Systems Ltd.
Scott Chapman, American Electric Power

As performance analysis and capacity planners, we collect data to share with co-workers and management in order to explain and justify our technical capacity and performance recommendations. To display this data, we often create the easiest graphs and don't consider how it is perceived. This paper discusses ways to visualize data and describes scenarios in which these visualizations should be used to transform data into useful information. We discuss common 'mistakes' and ways to improve your charts to better get your point across and better represent your intended message.

Introduction

As performance analysts and capacity planners, we have lots of data that we need to turn into information. We all use various techniques and tools to help us visualize the data to find trends, relationships, and potential problems. However, have you really thought about the best way to visualize your data? In most cases we simply accept the default charts, graphs, and reports that our tools generate, but all too often these were created without an emphasis on clearly and succinctly transforming the data into information. In many cases these charts and graphs are basically serviceable but could be improved with a few adjustments. In other cases, the default charts may actually obscure important information. The choice for including a particular visualization in a product may have been driven by what the marketing department thought would sell the product, not what would be useful to a skilled practitioner.

Whenever you create a new chart, graph, or report, you need to carefully consider your audience and what story you are trying to tell. The story to be told is often simple ("we need to purchase more capacity..."), but is often backed by data that is complicated to both present and understand ("...because these workloads are suffering unacceptably at these times"). The goal should be to develop a picture of your data that tells your story clearly and succinctly without distracting the reader. A poorly designed visualization can easily lead the reader to the wrong conclusion.

We hope that the examples presented in this paper inspire you to avoid common data visualization problems and inspire you to create better visualizations of your data. If you are a software vendor that provides charting and graphing of user data, we hope that you'll take some of these lessons to heart and improve your products.

Before you start

You have gone to a lot of trouble to collect a lot of data, but now you have to figure out how you should display that data so that it can be understood and valued by others. We have all been presented with a visualization that is overwhelming with colors, unclear scales, and we can't figure out what is the important information. In order to make a useful and powerful visualization there are a few important considerations. These considerations include understanding the purpose of your visualization and the needs of your audience, selecting appropriate colors, selecting the right chart type, and knowing what things to include and not to include in your visualization.

The first thing to do before creating your visualization is to think about what information you are trying to show and who your audience is. Your question may be exploratory in nature ("What workloads might have contributed to slow batch turn-around yesterday?") or very specific ("What was the heap size trend in the hour before the JVM crashed?"). You may want to compare many different data sources, such as different servers, or look at how a single server has been performing over a longer period. If the purpose of your visualization is exploratory you may want to include different views of the same data or add more interactive features so your users can 'drill-in' to get

more details as needed. If the purpose of your visualization is more specific, you want to ensure the information that you would like to get across is clear by highlighting it while minimizing other parts of the visualization that may distract from that. You will not want to include extra charts or data that does not speak to the point you need to prove. Understanding the purpose of your visualization is a way to evaluate whether or not your visualization is successful.

As well as understanding the purpose of your visualization, you also need to understand your audience. Is your audience expert network users who want to see a lot of detail to help solve a tricky performance issue or are you trying to prove something to your executives so your budget will be increased. At the executive level, you will want to give a summary of important values while being able to back up this information with further information if needed. For instance, you may have a summary page that gives your results, but displays simple charts on subsequent pages for further evidence. If you are sharing data with your peer group, they may be more interested in more exploratory approach where they can view all the data and coming up with their own conclusions. Also, the language you use to label your visualizations and data must reflect your user's knowledge base. Using overly technical terms may only confuse and frustrate business owners.

Picking the right colors

There is a lot of research into how we perceive colors, the attributes of color and how it can be used in design. Your choice of colors has a huge impact on how people understand your visualization, or if they even want to look at it. Rather than go through the research here, we are going to pick out a few key rules that will help you when picking colors for your visualization.

Generally, it is best to use a white as your background color for your visualization. This is not a hard and fast rule, but it makes your data stand out (rather than everything surrounding your data standing out), it prints out easier and leads to a minimal and clean aesthetic.

There are two types of data: quantitative and categorical. Quantitative data is numeric and refers to quantities of things. For instance, measuring the CPU utilization of a server is quantitative data. On the other hand, categorical data is represented by distinct items, for instance different types of servers are categorical. When viewing quantitative data, using a scale of a single color with varying intensities is usually best because it lends itself to a feeling like there is a little and a lot of something i.e. if using blue, low values could be represented by a pale blue and high values could be represented by a deep blue. One thing to keep in mind though is that we can only perceive differences for about 5 shades of a particular color so you may need to use multiple colors if you have more than 5 data sources [FEW2006].

When choosing colors for categorical data, use distinct colors and colors with a similar intensity. Stephen Few recommends using colors that can be found in nature, soft grays, browns, oranges, greens, blues and reserving really vivid colors for information you want to highlight [FEW2006].

Another option for colors is to utilize the gray scale. While it might seem 'boring', there can be some benefits to this approach. Gray colors generally print closer to what you see on your screen than colors do. Also if you want to highlight certain values, then using gray colors will make this highlighted values stand out even more.

Figure 1 - Example color pallets

You should also not forget about your color blind readers as 7-10% of the population has some form of red-green color blindness and 6% of the population has yellow-blue color blindness. Our default choices are often red and green to display something that is good or bad, but these colors are probably not our best choices because 10% of our readers may not be able to distinguish between them. If you really need to use red-green, they should be used with other indicators as well, such as a check-mark or an x icon so people with color blindness can see the difference between them. Red and green or blue and yellow are not good choices for line graphs, pie graphs or other graphs that cannot be directly labeled. In a quantitative scale, do not use red-green or blue-yellow as your scale because some people will not be able to view these differences. You can check how your visualizations look for people with color blindness at VisCheck [VIS2012].

Picking a chart type

Now that you know who your users are and what they need to know the next step is selecting what type of chart to use to display your data. You may have found yourself confused by all the choices including pie charts, line charts, heat maps and many more. First decide if you need to display data over time, or whether finding relationships or comparing values is the most important task. The following sections describe some familiar and maybe not-so-familiar visualizations and their benefits and limitations.

Viewing Values Changing Over Time

Many times we want to show how values change over some time period. For example, if you have been collecting the per minute transaction rates of various applications and want to compare the transaction rates of a few key applications to one another. Line charts and area charts are an easy to understand solution for this type of data. Line charts are typically good at showing several data samples for 1-10 data sources, but can get pretty overwhelming and confusing for a user if there are many more data sources than that. If there is a lot of variability in your data, a line chart can start looking like a saw or the teeth from Jaws and be hard to make any sense out of the data. A solution to this problem might be to just plot the dots as you would see in a scatter plot (more information below) and not connect them with a line. By not connecting values though it may be difficult to follow a particular data source as it changes over time so this only works for one or two data sources.

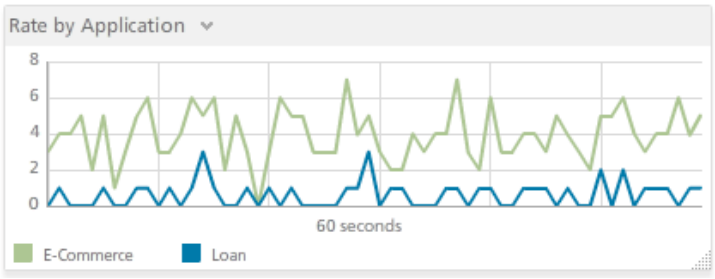

Figure 2 Line plot of per minute data by application

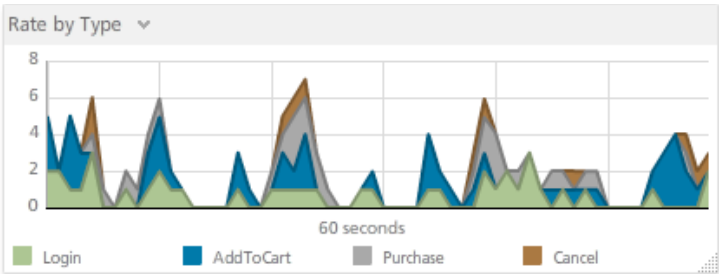

Figure 3 Area chart of per minute data by application

An area chart also displays values over time, but the area underneath the lines are filled in. You can also stack these areas in a stacked area chart if you are trying to understand how the total of these values change over time. However, be aware that it is hard to compare these values once they are stacked. If knowing the exact values for the data sources is important you should stick with a line graph.

If you have more than ten data sources for a single time period and find that your chart is getting crowded and hard to see, sparklines are a great alternative to putting all the values in a single chart. A sparkline is a small word-size graph which displays only the data in a line chart while leaving out the grid lines, labels and other extraneous information [TUF2006,]. The idea is that you can put the sparkline chart within text as you are writing about the data or you can stack many of these charts together so you can compare lots of values in a single screen. A sparkline should only have one data source at a time and minimal graphics other than the actual values. Some nice features to sparklines include highlighting the minimum and maximum values with a colored dot, showing a lighter threshold band so users can quickly see when values were out of these thresholds, and to add small labels with the minimum, maximum and average values beside each chart. Sparklines are particularly effective when you need to compare several data sources and are interested in the slope of the data, but may not be a good solution if it's important for your audience to view specifics at a given time.

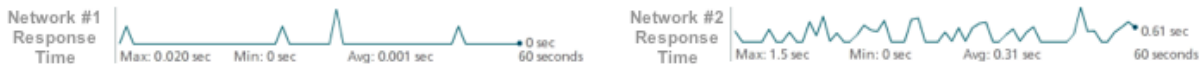

Figure 4 Sparklines showing network response time history

Showing Ratios & Comparing Values

Another point you may want to make with your visualization is to show how parts of the data make up the whole or compare a group of values. For instance, if you have collected the average CPU utilization for a particular server for an hour and want to see how much each of these processes contribute to the whole or you want to compare each of these values to one another. Pie charts, bar graphs and heat maps are visualizations to display this type of data.

A pie chart is a common visualization where each part of a ratio is represented as a pie 'slice'. A pie chart is great at showing how a few different pieces contribute to a whole, but is doesn't show a lot of data besides that and is an often-overused visualization. One of the major problems with pie charts is that it is actually harder for human perception to understand the specific difference in the pieces of a pie chart because people are less accurate at perceiving differences in angles versus differences in length [CLE1985]. This means that if you want your audience to understand the differences in two values, it would be better to show them in a bar chart (which uses length of a bar to display a value). Also, the labeling of data sources in a pie chart can be challenging. If you are using legends to label your data sources, people have to match the colors in the pie charts to the color in the legend. While this seems like an easy task, if you have a lot of different data sources and colors are similar, this matching becomes more difficult. You can add labels to the pie pieces in a 'call-out', but this potentially leads to a lot of visual clutter. If you are using pie charts, ensure you add the actual value for each data source as well since there are no other ways for your audience to get your values. For the reasons above, many visualization experts have strong feelings against the overused pie chart, but there is still a place for pie charts in showing how a few pieces make up the whole.

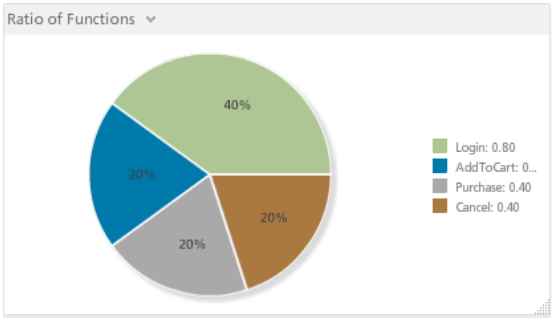

Figure 5 Pie chart

A far less contentious visualization, the bar chart displays rectangular bars where the length of the bar represents the value for each data source. Bar charts are much easier than a pie chart to view biggest to smallest values and this can be especially effective if the bars are ordered intelligently. Bar charts are also better at displaying many more data sources than a pie chart (you can reasonably show 50 different data sources) and still be able to compare values across these data sources. One problem with bar charts is that if your values are close together, it can be difficult to view the small differences between them.

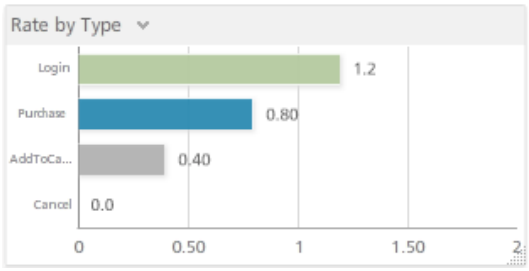

Figure 6 Bar chart

Often times you want to display a single value in the context of some thresholds so your audience can better understand the health of the reported system. Speedometers are often used in this circumstance but suffer from the problem of taking up a lot of screen real-estate without showing a lot of data. Stephen Few introduced bullet graphs as a way to solve the real-estate problem [FEW2006]. Instead of using a circular speedometer, a bullet graph is a variation of a bar graph, which allows you to visualize many more data values than would have been possible if you were using speedometers. The bar graph is encoded with three shades where each shade represents your threshold values and the length of the bar graph is the maximum value of your data source (think of your maximum value in your speedometer view). A thinner black bar is used in the middle of the bar to display the actual value and a small line represents some comparative value you want to highlight on your bar graph. Bullet graphs are a great alternative to the speedometer because you can still encode all the same information as your speedometer while fitting in many more graphs onto your screen.

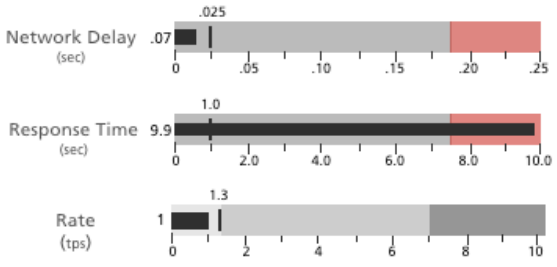

Figure 7 Bullet graphs

A heat map is another visualization for comparing many values, an example of which can be seen in Figure 17. A heat map is a visualization where data sources are mapped across both x and y axis into small squares and the value of the data source is represented by a color. This visualization is great at showing many different data sources at a single time and can be relatively easy to understand. Patterns, maximums and minimums are easy to pick out if color choices are done correctly. One challenge with the heat map is that the exact values are difficult to place on the chart without it looking cluttered and taking the impact away from the color. This problem could be mitigated with tool tips (if you have an interactive visualization) and a well-placed and detailed legend. The heat map is still a good choice for comparing the relative value of many different data sources.

Finding Relationships

Showing how data is related to each other is often an important goal in visualizations. If the primary goal of your visualization is to explore or highlight relationships, you may want to think about using scatter plots, bubble charts or parallel co-ordinates charts.

Scatter plots are a common visualization which use dots to plot values for a data source on an x and y axis. Scatter plots are good at showing if data sources are closely related because they will be closely grouped together. You can also see if values are positively or negatively correlated very quickly as well by looking at the slope of the dots on the chart. Adding a best-fit line helps demonstrate the trend and correlation of the dots. Scatter plots work only if both your variables on the axis are quantitative (i.e. numerical), but you can add a third dimension to plot using different shapes instead of just a circle to represent categorical information.

Bubble charts are similar to a scatter plot except that changing the size and color of the dots i.e. bubbles allows you to plot 2 more variables. Figure 8 shows the current average duration, failures and rate of different parts of a network. A bubble chart could be used if you want to find relationships but you have more than two variables you want to show. Some interactive bubble charts also have a playback feature which lets you trace bubbles as they change throughout time. A bubble chart's benefit is that you can view lots of variables at a time yet it's still easy to understand. Similar to a scatter plot, if you have a lot of values that overlap it can be difficult to view each individual bubble. Also, recognizing small differences in the size of the bubbles is difficult for human perception although adding labels to the bubbles can mitigate this problem. One of the better examples of a full featured bubble chart is GapMinder [GAP2012], an interactive visualization tool which displays global health and finance information in a compelling way.

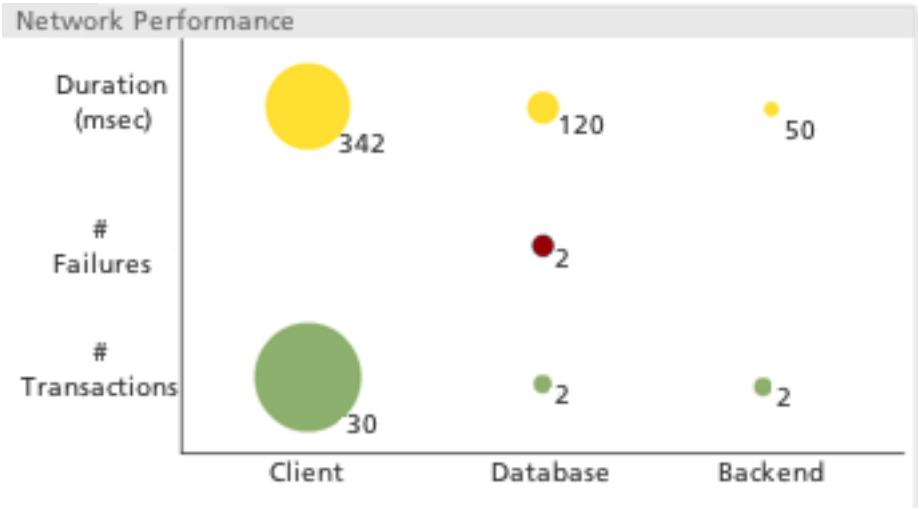

Figure 8 Bubble graph

There may be times when you need to look for relationships among many variables, where some are quantitative and some are qualitative. Parallel co-ordinates is a visualization for viewing such data. A set of parallel lines, each representing a variable, are displayed. Each point in each line represents a value for that variable. Then for each data source, you draw a line across all parallel lines. With this method you can see how data correlates and relates to one another by comparing where each line lands on the parallel lines. While this method may take some time to explain to your audience if they have never seen it before, you can visualize thousands of different data sources for many variables which makes this a really powerful

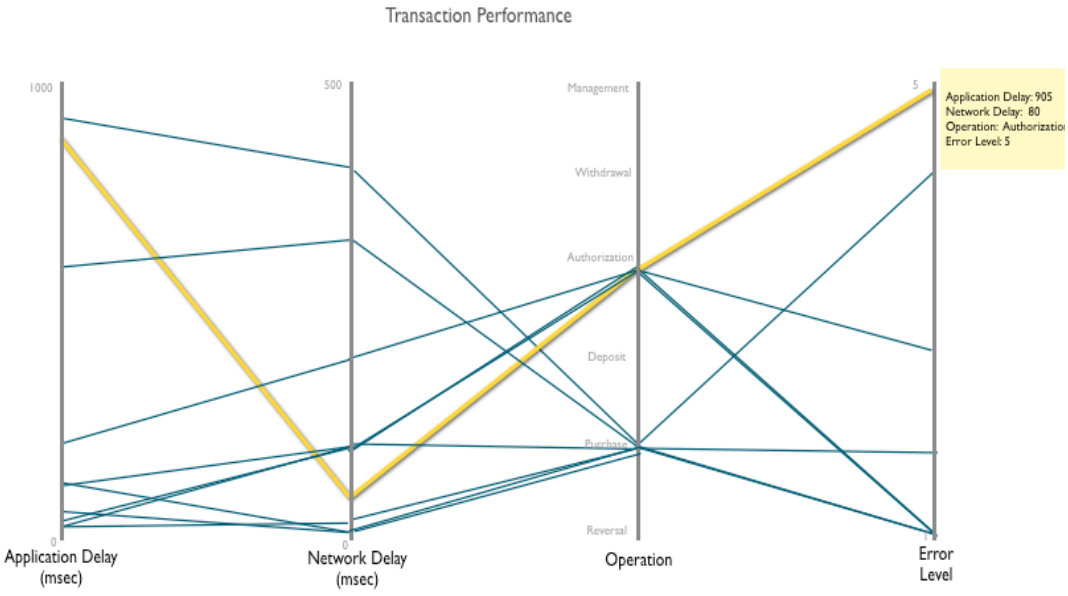

Figure 9 Parallel co-ordinates

visualization. Interaction techniques which can highlight a particular line and give more specific values for the variables can be very useful as well.

What not to include

Now that you have selected a useful and compelling way to display your data, there are some other things you can do to make sure your visualization is as useful as possible. The golden rule is that you want the data to tell your story, so you should only be including elements to your visualization that help make your point. 'Chartjunk' is a phenomenon where unnecessary information is included in visualizations and reduces the impact and visibility of the data [TUF1983]. The term was coined by Edward Tufte and includes anything from dark and heavy grid lines, decorative fonts and backgrounds, unnecessary icons and dimensions of data. Worse than these ornamental elements is actually junk that can skew the understanding of your data, including 3-D elements and inconsistent scales. Even though 3-D charts are very popular, 3-D effects often make it more difficult to see the actual values and may imply there is another dimension to the data. As for scaling, your y-axis generally should start from zero; otherwise small variations on the y-axis will look misleadingly large. If you must start a y-axis at a value other than zero, it should be very clearly labeled. You also should have reasonable maximums. For instance, if you are plotting the ratio of something your maximum value should not be over 100%. While logarithmic scales are sometimes necessary, they may mislead the casual reader and so should generally be avoided.

Examples Gone Wrong (and Improvements)

Poor Axis Choices

This section reviews some actual examples taken from our everyday lives of how visualizations can go wrong. Figure 10 shows the CPU utilization for multiple systems and demonstrates several of the problems discussed above. First of all, the Y-axis scale maximum is 90% although most people would assume that in CPU utilization the maximum would be 100%. By lowering the maximum it looks as though SysJ is consuming essentially the entire machine at its peak, although it really is reaching "only" about 88% of the machine. This is also a prime example of how 3-D can make the values more difficult to view the values of the data. Not only are the bars in 3-D but the Y and X axis are also in 3-D. The values in 'front' can occlude the values of data in the back in the front values are higher. What part of the machine was SysJ using at 6:00? It is somewhat difficult to even tell which bars represent 6:00, but when we do find 6:00, we discover that SysJ's utilization is hidden behind SysB's bar.

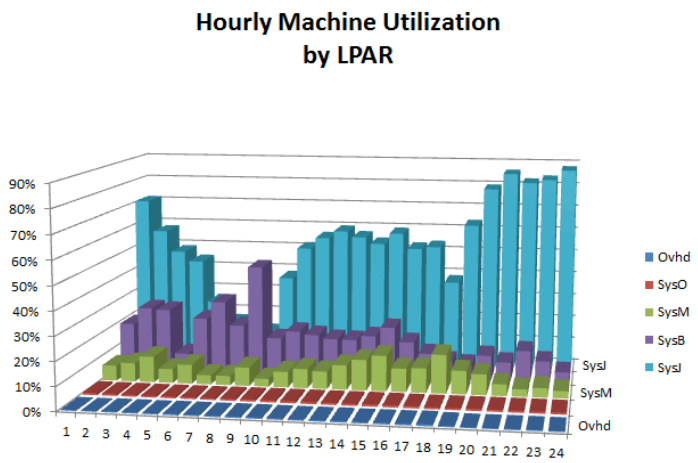

Figure 10 Hourly system utilization as 3D bar chart

Figure 11 contains the same data, but this time presented as a stacked area graph. This is better in that no data is hidden. However, area and line graphs can imply a continuum of data when you actually have discreet values. This can make it hard to interpret the actual values. For example, what is the machine utilization at 7pm? The values at 6pm and 8pm are fairly obvious, but it's unclear exactly where 7pm falls on the line between those two times. Also note in this case the software automatically chose to scale the y-axis to 120%, which is nonsensical for the most common ways of measuring machine capacity. Presented without context, the user is left to wonder what the machine utilization truly represents: is there some way that it could get to 120%? Is 120% the maximum it ever could get to? Or is it just a poorly chosen maximum for the y-axis?

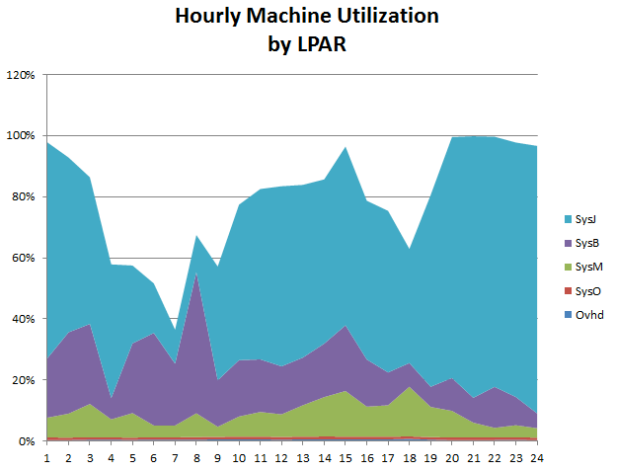

Figure 11 System utilization stacked area chart

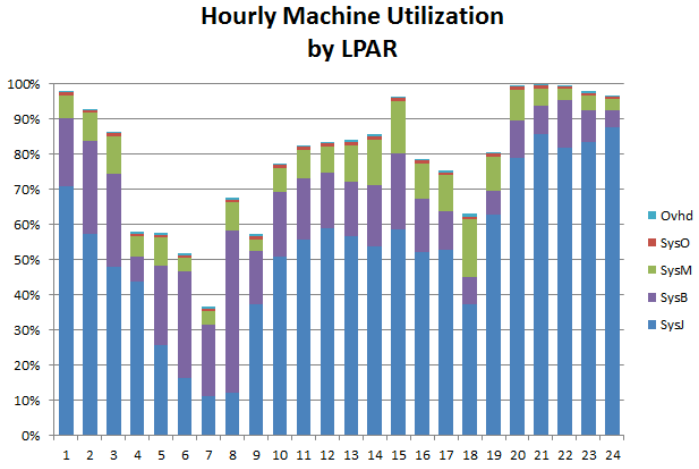

Figure 12 System utilization stacked bar chart

Compare the previous charts to the stacked bar chart in Figure 12. Note how each of the discrete data values is readily visible and understandable. We intentionally chose to put the most important system on the bottom so that its utilization is always directly readable on the y-axis. The y-axis is sized intelligently.

Low Data Density

As mentioned previously, speedometers have very low information density but take up a lot of screen space. In the example in Figure 13, the current values are displayed in a label at the bottom of the needle but there are no other values labeled so users have to estimate what the maximum values could be. The only "value" that the speedometers add to the basic display of the number is a feel for relatively how far into the green or yellow range the values are. However even that is somewhat misleading. Looking at the speedometer labeled MTD (month-to-date), the reader might presume that the monthly SLA (Service Level Agreement) target is something a little under 40 as it seems the current value is more than half-way to the yellow, and about half-way to the red. But we don't know whether the start of the yellow or the start of the red is the SLA value. In the YTD (year-to-date) speedometer, it appears that we're already in the yellow for the year, and not far from the red zone. That's probably not good. If it's currently the middle of March, that's probably even more alarming. Perhaps however, the graphs' yellow and red thresholds vary from day to day based on how far into the year or month we currently are. Or perhaps the thresholds are reset less often. There's no way to tell from the graphs though and no way to tell what the actual SLA is for either timeframe.

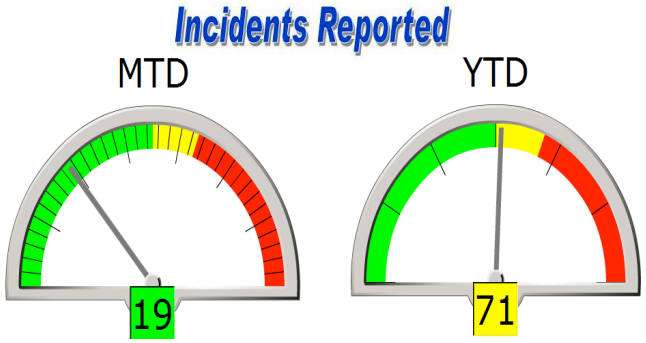

Figure 13 Incident count SLA shown as speedometers

Providing a meaningful visualization for this type of data can is difficult. Simply graphing the number of incidents over time will yield a wedge-shaped graph of increasing values, which isn't very interesting. We know that the number of incidents can only increase over the course of the year; the past can't be undone. Perhaps if we have a large number of incidents per day, graphing the daily values might be interesting, but in this case we're averaging about 1 incident per day, so graphing the daily values is also uninteresting.

While the visualization in Figure 14 may not be as colorful as the speedometers, it provides a better context for the message that it's presenting. Here, the value that we're displaying is the YTD number of incidents above or below the annual SLA value, pro-rated for that particular date. For example, if the annual SLA is 336 incidents, then on February 1st, the pro-rated SLA value is 29. If we have had 28 incidents as of February 1st, then the graph in Figure 14 would show -1 for that date, because we've had one less incident than the annual SLA would allow on a pro-rated basis.

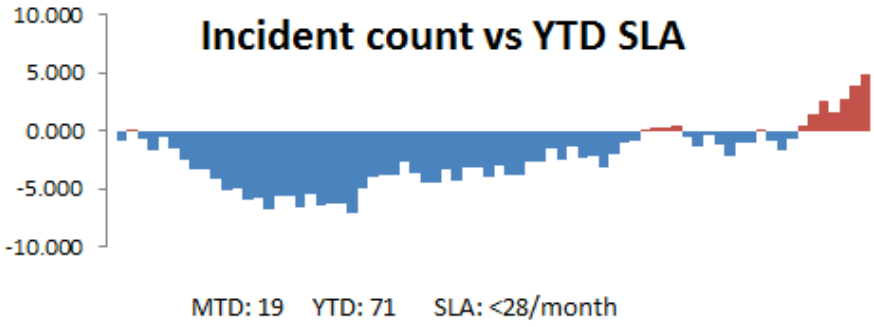

Figure 14 Incident count SLA attainment

While this alternative visualization might not be immediately understood by the reader it is also not as likely to be misinterpreted as the speedometers in Figure 13. The message that we've had a total of about five more incidents than we should at this time of year is clearer in Figure 14 and it does not incorrectly imply that making the yearly SLA is going to be nearly impossible.

Poor color choice

In this example we are comparing response times for a particular application in different regions. Both examples utilize a simple bar chart, but the Figure 15 demonstrates that color choices for data and surrounding areas can be distracting. This example shows very saturated colors for each bar, a background that is using a gradient for no particular reason and a very dark color as the background color to the whole chart. Also, using red as a color may show something bad or unwanted is occurring. In comparison, Figure 16 uses different shades of blue for each region while using a brighter red color to highlight a higher value which needs attention. The viewer is drawn to this data first and is not distracted by the other colors. The background is a soft gray which still makes the labels easy to read and separates the auxiliary information from the actual data. The white background of the chart allows the viewer to see the colors of the bar charts more easily as well.

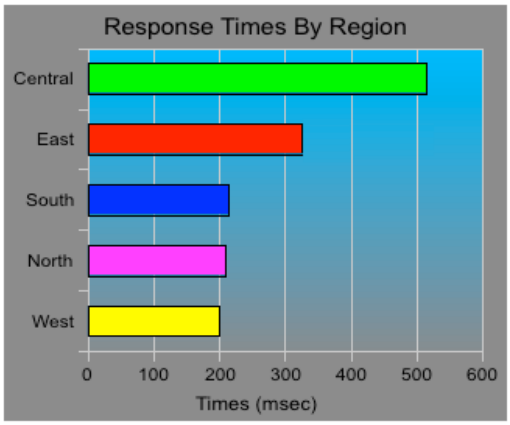

Figure 15 Poor color choices

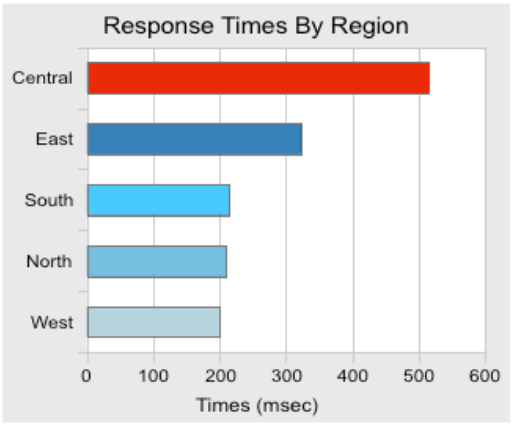

Figure 16 Better color choices

Too Much Data

Sometimes our inclination to show all the data results in data visualizations that are too complicated to be meaningful, at least in some contexts. In a mainframe environment, workloads run at different importance levels and the amount of CPU delay suffered by workloads at each importance level is a useful metric. While some CPU delay is to be expected and indicative of a financially well-managed environment, the intention is to avoid significant delay for important workloads. Of course, it is typical to have multiple systems, and the delay suffered by importance level, by system is going to vary across the course of the day. The first attempt to produce a visualization of this data for management resulted in the confusing heat map chart in Figure 17.

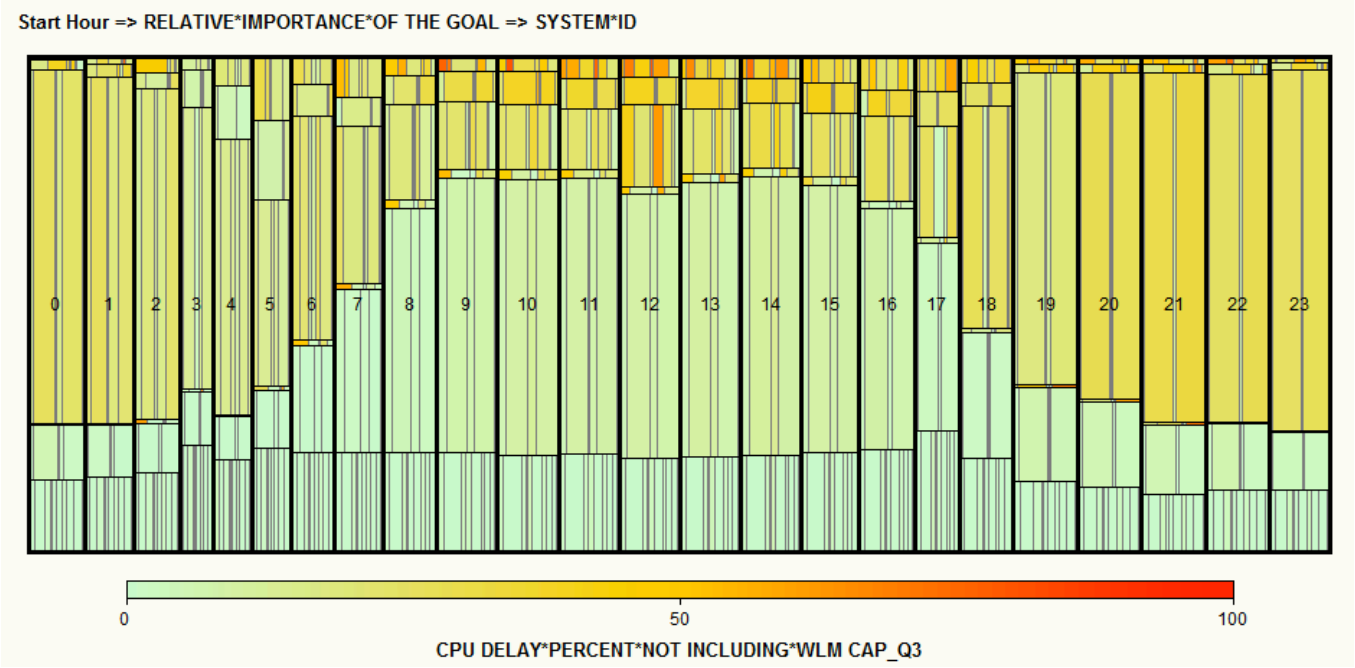

Figure 17 CPU delay heat chart

The horizontal striations in the above chart represent the relative importance levels, and the vertical bars within each horizontal striation represent the individual system. The total height of the blocks indicates the overall amount of work produced in that hour at that importance level, across all systems. The width of each bar represents how much of that total work was executed by each system. The color represents the average CPU delay suffered over the course of the month for that combination of hour, system, and importance level. While this information is interesting, and if the chart is interactive (e.g. if one can hover over or click on a box to get the detailed information), perhaps it might be appropriate for a technician. But it's not appropriate for a management level report. But how can we display this relatively large amount of data in a usable format?

When presented with an unwieldy amount of data, it is often useful to think about the information that needs to be presented. The process of turning data into useful information usually involves some form of statistical transformation that summarizes the data into representative values that can be easily explained.

In the above case, the message that needs to be presented is that some workloads are suffering significantly at times, while the most important workloads are generally not suffering substantially. Since the daytime hours are the time period that is most important, it was decided to just focus on those. Because extreme delays, not modest average delays, cause people to complain, the 90th percentile of the delay suffered by each workload importance level for each system was calculated for the daytime period for the month. Using the 90th percentile instead of an average keeps the values from being overly skewed by days where the workload is relatively light and more accurately reflects users' perceptions of the system. Users' overall perception of the system being slow is not reduced by the days when their response time is faster than necessary, so average values may not accurately reflect user perception.

While the bar charts in Figure 18 eliminate a significant amount of detail data found in the tile chart, this summary data is much easier to visualize in a meaningful way. Even without a detailed explanation, the reader is much more likely to receive the intended message. Clearly there are times on multiple systems where the lower importance (0 is the highest importance value) workload is being significantly delayed for CPU. Because there are more than five categories of data (importance levels), shades of blue and green were chosen for most of the categories. Since importance level zero should very rarely suffer any delay, it is shown in a perhaps more alarming maroon. The background shading attempts to suggest how significant the delay values are, but the graphs would be cleaner on a plain background. While it is preferable to leave the background plain white, in this case the background shading attempts to add value to the overall chart and so may be acceptable.

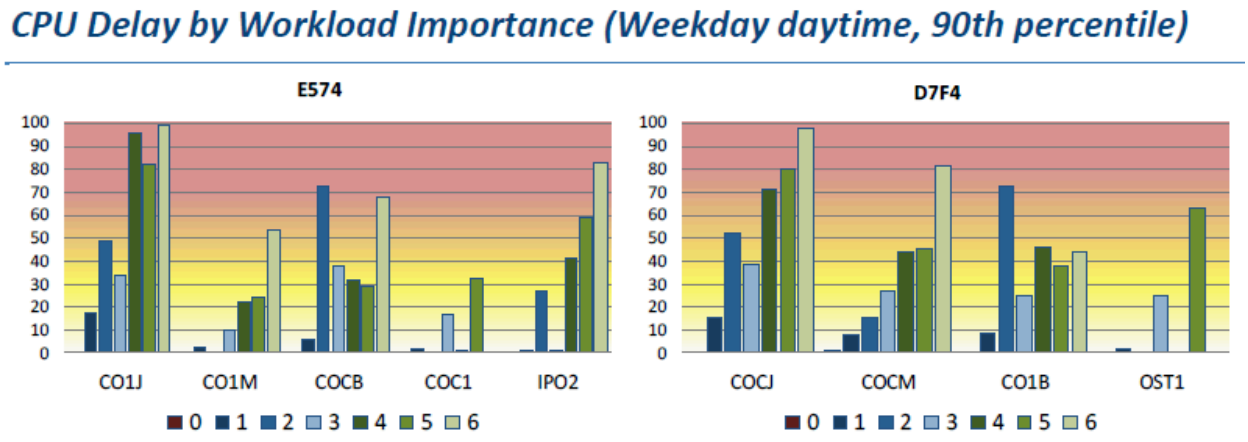

Figure 18 CPU delay bar graphs

Chart junk

As mentioned above, chart junk refers to anything added to a chart or visualization that is not needed and therefor may deter from the actual data that the chart is trying to show. Here are two charts that are looking at the response times of an application by 5 different regions. Figure 19 includes a bunch of extraneous visual components that deter from the data. For instance, the axis labels are difficult to read because they are far too frequent, are in a difficult color to read and in the case of the Y-axis, they include too many decimal places. The title font is difficult to read and includes a shadow for no reason. The shapes for each data point add to the clutter on the graph and instead of making the values easier to read, they end up making the values more difficult to see when the data source's values begin to overlap. The dark gray background does not help to view any of the data, but just draws attention to the background rather than the data. The second example in Figure 20 shows a simplified version of the same data. The number of labels on both axis are reduced and a neutral color is used to make them easier to read. Instead of including the units for the milliseconds in each of the axis labels, the title for the axis includes the units. The background of the graph is white which makes the lines for each data source easier to read, while a light gray color is used to frame the graph's labels and legend.

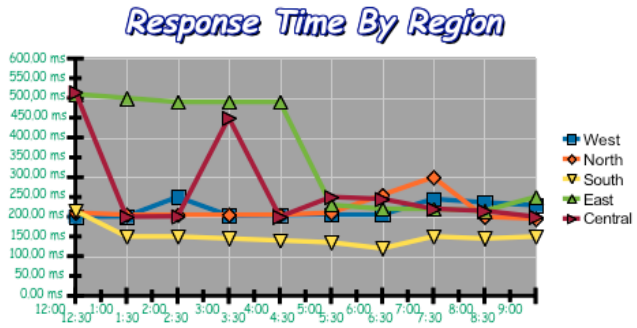

Figure 19 Chart junk line graph

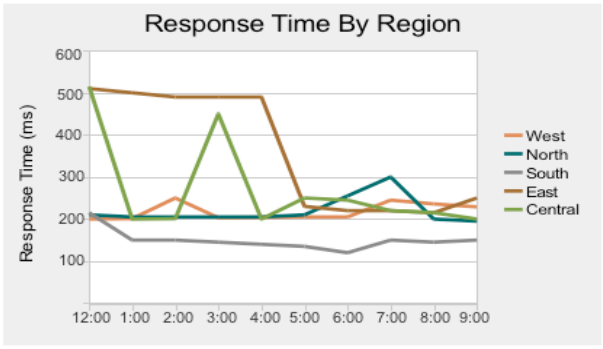

Figure 20 Clean line graph

Too Much Clutter

Below is an example of how to utilize sparklines (individual line graphs) instead of cluttering up a simple line chart. In Figure 21, there are 12 different data sources each showing the rate and duration for 6 different applications running on the network. With this many data sources it is difficult to follow a single source and almost impossible to compare the rates or durations of the applications.

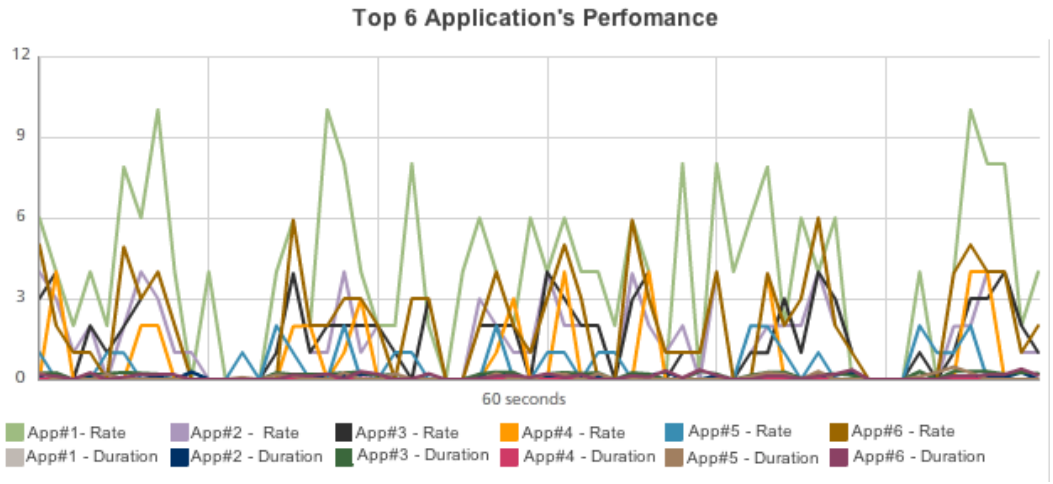

Figure 21 Multiple applications' statistics as a multiple-line graph

The visualization in Figure 22 shows an example of how you could make this cluttered line graph into 12 sparklines without using much more screen area. For each application there is a small, simplified line chart for its rate and duration. Viewing a column allows you to compare the general values for the rate or duration for all applications. These sparklines also include a light threshold line and anything above this threshold is shown in a darker color to bring attention to it. The maximum, minimum and average for each sparkline is also included for easy reference. By separating out each data source, we are able to include specific thresholds and statistics for each data source. While some people complain that sparklines don't include enough details, if you are creating your own sparkline component you can add interactive elements when you move your mouse over a specific chart to get values at any given time.

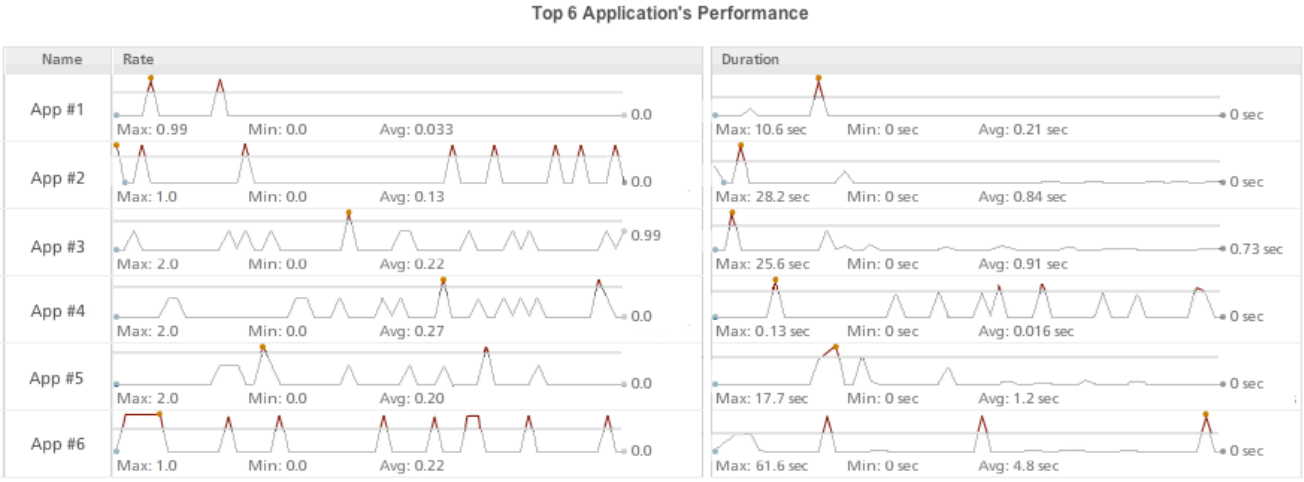

Figure 22 Multiple applications' statistics as individual sparklines

Tools

While one could make beautiful visualizations of data with oil paint on canvas or a set of artist pencils and a sketchbook, most of us will be using some sort of computer program to both process our data and display and/or print it. Major statistical packages such as SAS and R provide a wide variety of capabilities for graphing data. Certainly the ability to visualize the results of your analysis in the tool you are using to analyze the data is convenient. However, customizing the graph output from these packages may be a skill entirely different than the skills needed to use the tools to analyze the data in the first place.

In contrast, Microsoft Excel and similar spreadsheet programs usually make it very easy to alter the look of a graph, often as simple as right-clicking on what you want to change and selecting the new look you want for that element. Adding useful elements (such as shading part of the background of the graph yellow to indicate an area

of concern) can also be easily accomplished. So the best option may be to process and analyze your data in SAS or R and then pass the final processed data into a spreadsheet for graphing. For charts you're going to produce on a regular basis, you may want to have your statistics package create a comma separated values file in a specific place and create a spreadsheet that references that external data source. You can set options such that the spreadsheet will automatically re-read the CSV file every time you open the spreadsheet. Word documents can even be created that will pull in the Excel graphs automatically. Such Word documents could be easily saved as PDF files for easy distribution via email.

To explore different visualizations for your data you can use the website Many Eyes [MAN2012], a website run by IBM where you can upload your data and try out different visualizations including bubble charts, line charts, scatter plots, and bar charts. Most of these visualizations include interactive features as well so you can really get a sense of what different visualizations can offer.

It is very common to need to deliver your data via a web browser. In these cases, it is quite likely that your performance tools and data analysis tools can produce some form of HTML-based output. Customizing this output may again be difficult. It may be more advantageous to employ a strategy similar to the one used above, only instead of reading the data with Excel, you could read an XML file with a simple JavaScript program embedded in a static HTML page. JavaScript is a fairly easy programming language to learn, and every computer professional should have a basic knowledge of HTML. (If you need a quick primer on HTML see [CHA2008].)

Browsers are getting ever more sophisticated in their ability to display rich datasets and there are visualization tool kits and libraries available that help you build more complicated and interactive visualizations. Protovis is a graphic library built in JavaScript and SVG for visualizations to be displayed on a website [PRO2012]. The library was built by a group out of the Stanford Visualization Group and includes all the basic visualizations plus a few more unique ones including bullet charts. D3 is the group's extension of this project and "allows you to bind arbitrary data to a Document Object Model (DOM), and then apply data-driven transformations to the document" [D32012]. The authors have not used this new approach but it looks promising and is built on recognized standards such as CSS3, HTML5 and SVG.

There are several other JavaScript libraries for creating basic charts and graphs. See Appendix A for a list of some example libraries. While fully leveraging any of these libraries will require writing code that is likely at least as complex as the code you would have written in SAS or R, the result can be a very impressive and interactive display of your data delivered directly to your reader's browser.

Conclusion

While we often graph and chart our data relatively casually, carefully crafted data visualizations improve the chance that our message will be received as intended. Keeping a few simple rules in mind can help us create better visualizations. Avoid unnecessary "chart junk" and trendy, but often confusing, chart types such as 3D charts and speedometers. Make sure your axis limits make sense for the data you are presenting. Select your colors with care, avoiding overly saturated color schemes and remember that a significant percentage of your viewers are likely color blind. Consider techniques you may not have tried before such as parallel coordinates and bullet graphs. Think about the information that needs to be conveyed and don't present excessive data when simpler summary data will suffice. Following these principles will help you produce visualizations that better inform your readers and are less likely to be misinterpreted.

References

[CHA2008] Scott Chapman, “<HEAD>, <BODY, Links and Code, An Introduction to Using HTML to Present Your Data”, Computer Measurement Group Proceedings, 2008

[CLE1985] William S. Clevland, *The Elements of Graphing Data*. Pacific Grove, CA: Wadsworth & Advanced Book Program, 1985.

[D32012] http://mbostock.github.com/d3/

[FEW2006] Stephen Few, *Information Dashboard Design*, 2006

[GAP2012] http://www.gapminder.org/

[MAN2012] http://www-958.ibm.com/software/data/cognos/manyeyes/

[PRO2012] http://mbostock.github.com/protovis/

[TUF1997] Edward Tufte, *Visual Explanations, Images and Quantities, Evidence and Narrative*, 1997

[TUF2006] Edward Tufte, *Beatiful Evidence*, 2006

[VIS2012] http://www.vischeck.com/

Copyrights and Trademarks

Appendix A – JavaScript Graphing Libraries

Library	Website	License	Browsers	Comments
Raphael	raphaeljs.com	MIT	FF, S, C, O, IE6+	Support for all sorts of vector graphics needs
JSCharts	jscharts.com	Commercial	FF, S, C, O, IE6+	Supports line, bar, and pie charts
flotr	solutoire.com/flotr	MIT	FF, S, C, O, IE6, 7	Bar and line graphs, supports interactivity
YUI Charts	yuilibrary.com/yui/docs/charts/	BSD	FF, S, C, O, IE6+	Bar, line, pie charts, mouse-over tool-tips
Highcharts	highcharts.com	Commercial	FF. S. C, O, IE6+	Bar, line, area, pie, mixed types; zoom and drill down capabilities
d3	mbostock.github.com/d3/	Free	FF, S, C, O, IE9	Multiple, but not really a library of graph types

Browsers: FF = Firefox, S = Safari, C = Chrome, O = Opera, IE*n* = Internet Explorer version *n*