

## Get the Best out of SAS/GRAPH® and ODS

LeRoy Bessler, Assurant Health, Milwaukee, Wisconsin, USA

### Abstract

This tutorial on visual communication presents graphic (and web) design principles, examples, and solutions for making best use of those precious resources: the time and attention of your information audience. The design principles are software-independent, but the solutions are derived from real-world application development experience with SAS/GRAPH and (for web-enabled solutions) ODS.

### Introduction

As the most powerful graphic application development software in the world, SAS/GRAPH has long been rich, and becomes ever richer, in features. With these features comes the need to apply the technology in a communication-effective manner. This tutorial cuts through what can be the problem of “Options Over-Choice” to empower you to make elegant and effective graphs. Communication-effective use of SAS/GRAPH gives you The Power to Show: to inform and influence, to reveal and persuade.

ODS (Output Delivery System), even when its use is limited to web publishing, brings an array of opportunities and challenges of its own. What you see as web page creator may not agree with what your web browser viewer sees. This tutorial addresses this and other matters with tips, techniques, and best web practices. (However, lest I seem to over-promise, I must mention that most of the coding details in this paper are for creating graphs, not for web enablement. Except for creating custom styles with PROC TEMPLATE, web enablement for graphs or tables requires rather simple code, and not much additional code.)

You will find no axis lines on my graphs, no tick marks, and axis values only when really needed. Better than axis information to enable the viewer to estimate the length of a bar, or the y-value of a plot point, is the precise value for it somewhere on the graphic image. You will rarely see axis labels. Most of the time they would be redundant with information that is already in the graph title or subtitle (or that should be there). Dates are usually unmistakable as such, and the need to explain them as “Date of . . .” can be addressed in a title or subtitle.

Simplicity is elegant. A graph or web page that tells the viewer only what needs to be said, but everything that needs to be said, is a thing of beauty.

This paper is part of a trio. Reference 2 covers: fully commented coding details to customize three popular types of presentation graphs, to implement design principles presented here; ODS destinations; graphic device drivers; preparing graphs for, and using them in, Microsoft Word or PowerPoint; and “Accessibility”. Reference 3 covers communication-effective pie charts.

Figures 1-16 were built at 960 pixels by 900 pixels and were resized to 3.2 inches by 3 inches after insertion in Microsoft Word.

### The Best Advice I Have Ever Found on Communication (from Joseph Pulitzer)

Put it before them briefly so they will read it.  
Put it before them clearly so they will understand it.  
Put it before them picturesquely so they will remember it.  
And, above all, put it before them correctly so they will be guided by its light.

### My General Recommendations on Visual Communication

Construct charts or web pages that deliver a quickly assimilated picture, for rapid decision-making, *and* as much precise detail as possible, for reliable decision-making.

Keep your graphic image and/or web page simple and focused. Simplicity provides focus. Focus usually requires simplicity. Simple and focused messages are easily and quickly interpreted.

Do not let the medium get in the way of the message. Let the message in your data reveal itself. Let the data talk.

Do not accept traditional, and usually superfluous, graphic paraphernalia that is present by default. With good design, none of it is usually needed. Artifacts of axis lines, tick marks, and grid lines are just holdovers from the days of grid paper, pen, and ink.

Show them what is important. When ranking is available, it is a usually effective, always easy, way to show what is important.

Let part stand for the whole. Give your audience a way to find out about the rest, if they are interested. Presenting the minimum for the intended message provides focus on what is important.

Beware of 3D to display two variables and misuse of color, which are frequently encountered obstacles to visual communication.

## The Most Important Design Consideration for the Web

Dilbert (Scott Adams' cartoon character) said, "I calculated the total time that humans have waited for web pages to load. It cancels out all the productivity gains of The Information Age."

Simple web pages load faster. The number one concern of web users is download time.

## Specific Graphic Design Principles for Visual Communication with Simplicity and Focus

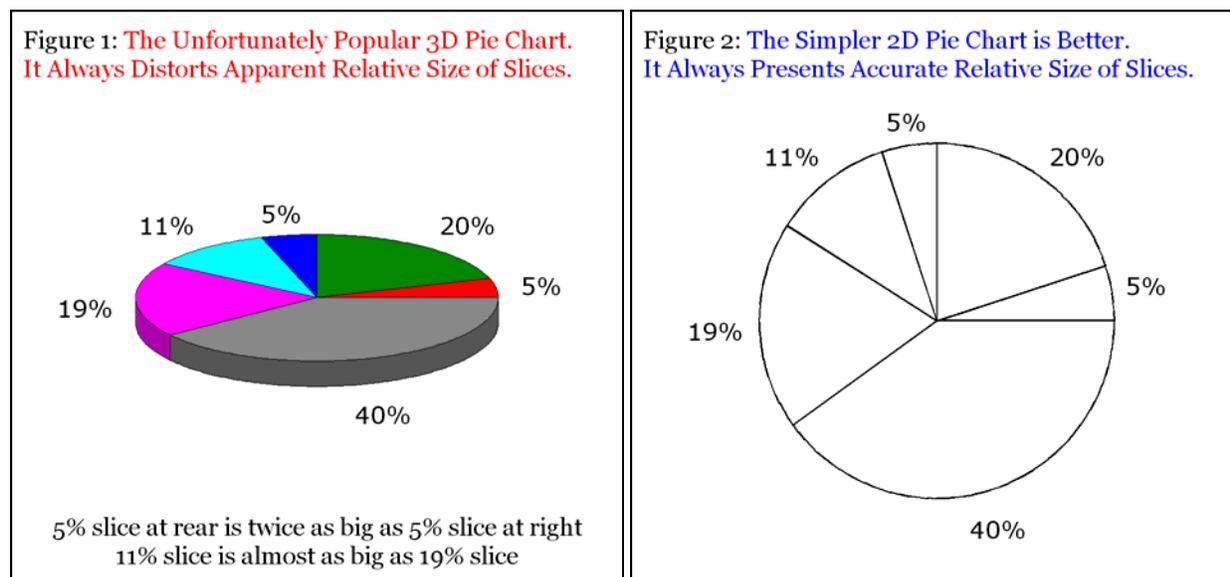
Focus on the Data by Removing Axis Clutter. Turn off axis lines. Turn off tick marks, but not tick mark labels. If not turning off axis labels, supply your own, but not to state the self-evident—e.g., that dates are dates. What variables are being graphed can be adequately described in the graph title or subtitle. Axis paraphernalia are historic holdovers from the bygone era of graph paper, pen, and ink. They are not an aid to communication. Communication is more reliable (and efficient) by annotating plot points and bar ends than by requiring visual comparisons with axis tick marks and/or reference lines.

Show Them What Is Important. Rank data in bar charts and pie charts. On maps, supply rank as an annotation.

Use Sparse Annotation. Whenever sufficient, annotate y-values only for the critical points of a plot line, and label only corresponding (invisible) x-axis tick marks. For a crowded or crossing multi-line plot, put critical values in the legend. *Critical points* are start, end, maximum, minimum, and points where the rate of growth or decline persistently changes.

Special Effects are for Movies. 3D is for 3 Variables. Good design and interesting data can stand on their own. Omit drop-shadows, clip art, etc. 3D pie charts *always distort* the apparent relative size of shares of the whole, defeating the visual communication purpose of pie charts. 3D bar charts needlessly complicate a simple image, usually making it more difficult to interpret. 3D maps are either communication-problem-prone (PROC GMAP's PRISM and BLOCK maps have the solids for high response areas hiding those for low response areas) or impractical (the PROC GMAP SURFACE map).

Pie charts are covered in detail in Reference 3, but let me show you this comparison of 3D and 2D with no code:



Use Area Fill Wisely, if at all. Always omit area fill on line charts. When fill is really needed on a graph, use solid colors or grays, not cross-hatching. On bar charts, use light gray or some other light hue. Dark or intense area fill can create a distracting, vision-dominating mass. On a chart with many bars, use of empty bars creates visual confusion between the bars and the spaces.

Usually Start the Response Axis At Zero for Trend Charts (whether line plots or vertical bar charts). Minimize meaningless fluctuations, and avoid magnifying what may be insignificant "growth" or "decline". Prevent needless questions and undeserved elation or alarm. Note the desirable near-flatness of the upper chart in Figure 9—variation over twenty-two years was less than ten percent. Conversely, when fine structure of a chart IS important, devote all of the available space to the actual data range.

**Communication with Color** (For a more comprehensive discussion, please see Reference 1.)

The commonest color blindness cannot distinguish red and green. (See Reference 1 for communication-effective color-coding alternatives.) Consider using red and blue, or orange and green, instead.

Maximize color contrast between text and background. There are problems using yellow (or light green) on white, or using black (or other dark) text on dark or intense background colors. Also, adequate contrast for an online display does not guarantee the same for hardcopy, which is not brightly backlit. Use high contrast (best exemplified by Black with White or Yellow).

NOTE: Background color for a graph is controlled with GOPTIONS CBACK=, which is defaulted to WHITE. For web-delivered graphs, I usually recommend use of GOPTIONS TRANSPARENCY, which makes the web page background, and its color, show through as the background for the image and the text of the graph.

Make colored text, lines, and symbols sufficiently thick. For the hue of any colored graphic element to be distinguishable, it must have sufficient mass, regardless of its contrast with the background. Text is sometimes colored for emphasis or needless decoration. For text emphasis, instead consider the alternatives of **bold** or *italic* or ALL CAPS (use caps sparingly).

When using a single hue for color-coding, limit it to no more than five shades (degrees of lightness/darkness). The human eye cannot reliably distinguish more than five shades of a single hue. If one or both are distinct from the background, you can add white and/or black to the color-coding palette. White (Black) is the lightest (darkest) shade of every color hue.

Use RGB color codes, not SAS® “Predefined Color Names”, and not color names from the SAS HTML Color Registry. There are 292 “SAS Predefined Color Names”, listed in Table 7.2 in the Version 8 SAS/GRAPH manual. They have names such as “PINK”, or “LIPK” for “Light Pink”. However, many of the names are misleading. If you display or print PINK and LIPK, you will see that Light Pink is darker than Pink. There are other contradictions like this. Also, many of the colors are too dark to be useful. To make wise color choices, create samples. Below is a simple way to do so. Add more notes for more colors.

```
proc gslide;
note j=C font='Georgia' h=1 c=BLACK 'C=LIPK' font='Monotype Sorts' h=5 c=LIPK '6E'X;
note j=C font='Georgia' h=1 c=BLACK 'C=PINK' font='Monotype Sorts' h=5 c=PINK '6E'X;
run; quit;
```

If you cannot use this Windows TrueType font, use f=CENTX and replace '03'X with '6E'X.

There are also about 150 newer color names in the SAS Color Registry. You can find them, and their RGB codes, using this click sequence in your SAS session: Solutions -> Accessories -> Registry Editor -> Colornames -> HTML. With these, too, assume nothing. Make yourself a sample chart.

In Version 9 of SAS/GRAPH, CNS color names have replaced the SAS Predefined Color Names.

Use color to communicate, not to decorate. The use of color-coding with a legend is an example of using color to communicate, not decorate. If you have no need to distinguish response levels/categories, use black and white. If you have a few levels or categories, gray shades may suffice. If you have many levels or categories, color is necessary.

The Benefits of “Boring” Black-and-White.

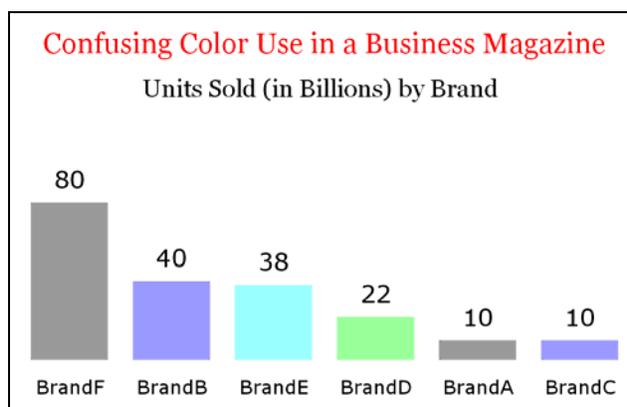
Technology to print black and shades of gray is faster, cheaper, and more reliable.

Black, white, and shades of gray are easier to use. Not only is the equipment simpler, but also their use requires no agonizing over color selection.

Finally, such output is reproducible. Regardless of the proliferation of cheap color printers, the copiers that you find in abundance in the workplace are still almost always black-and-white. Why does that matter? Good graphs or web pages—after they have been printed—will get copied when people want to share them.

Avoid confusing with color. Viewers try to draw inferences from your use of color, even if you intend none. When there is no intended pattern to color use, viewers will assume that there is one, or will be confused, or at least slowed down, by attempting to find it.

The figure below is a simulation, with different data, of a real bar chart that I saw in a business magazine. It had me perplexed as to the relationship between brands A and F and the relationship between brands B and C. There was no relationship. Finally, based on seeing colors used in a different chart in the article, I made a conjecture that the chart preparer was limited, for what reason I do not know, to a palette of four colors.



Use color consistently. Consistency in use of your preferred color palette from graph to graph will avoid confusing your viewers. As noted above, even when there is no intended pattern to color use, viewers will assume that there is one, or will be confused, or at least slowed down, by attempting to find it. After you have settled on a particular color palette, you should use it across all of your work. Not only is there communication benefit, but also you need not agonize about color choices from project to project.

If possible, test color graphic output on the ultimate target medium/media. If developing it with CRT monitor, be aware that an LED monitor will change the colors. (E.g., very light colors are washed out to white or near white on an LED screen.) Printed color does not look exactly like the screen-displayed color, which shines at your eyes. Not only are discrepancies introduced by the printer, but also viewing printed color relies on reflected light. Test any color graphic slides on the LED (or CRT) projector that you will use. The projected colors will not match the colors on your laptop LED screen.

For the web, use “browser-safe” (a.k.a. “web-safe”) colors. Unless you have a captive web audience, and can personally and perpetually control the characteristics and settings of all of their PCs, you cannot assume that they see the same colors in their web browser that you see. Unless you use browser-safe colors, color substitution can occur—not merely a change of shade, but an actual change of hue. If you want confidence that they do see what you see (within the limits set by differences in human visual perception and differences in hardware), use of the universal browser-safe color palette is the solution. These are RGB colors, and there are “only” 256 of them. Recalling from above that it is impossible to distinguish more than 5 shades of any single hue, and realizing that a SAS/GRAPH does not support more than 255 colors on an image, this limit of 256 colors is a benefit, not a handicap. Would you really want to select colors from the more than 4 billion choices available with what is called “32-bit color”, especially when you can not be sure that your web viewer has 32-bit color available? Now, having said all this, I must admit that color substitution is not highly likely, because most PCs will be able to support a sufficiently large color palette (i.e., more than 8-bit color). However, if you want to be safe, for details about browser-safe colors, for color charts (the complete palette, and some “coordinated” subsets), and for a demo of screen captures of color substitution observed due to use of unsafe colors, please see Reference 1.

The simplest guidance is: Use color with a purpose.

### Communicate with Text

Make the graph title your headline. You are using the graphic image to reveal and/or persuade. Don’t be reluctant to tell the viewer what you know or think the graph means. Bland description of graph content is often self-evident information. The bland description, if required, can be in the graph subtitle (e.g., “Monthly Sales of Widgets from April 2006 through April 2007”).

Use sparse text to make the graph talk. Be sure every letter or number *must* be there. Superfluity detracts from emphasis.

Please keep text horizontal. That’s the way we read. Do not permit software to stack the letters of a report column label. Do not apply a rotated label to the vertical axis of a graph just because it will fit nicely. Instead, use a sufficiently informative title or subtitle to make graph axis labels superfluous.

For text emphasis, consider the alternatives of **bold** or *italic* or ALL CAPS, rather than color. Use emphasis sparingly. Otherwise it starts to lose emphasis. Especially use ALL CAPS sparingly. ALL CAPS is more difficult to read in large quantities. To see that, prepare an entire paragraph in ALL CAPS.

Beware of the SIMULATE font sometimes substituted by SAS/GRAPH. Check your SAS log for a note that the undesirable SIMULATE font has been substituted for your requested font. Unfortunately, you are not always notified. Therefore, know how to recognize it when you see it. For a sample of this font (or any other SAS/GRAPH software font, specified with name=), use:

```
proc gfont name=SIMULATE nobuild NOKEYMAP H=1; run; quit;
```

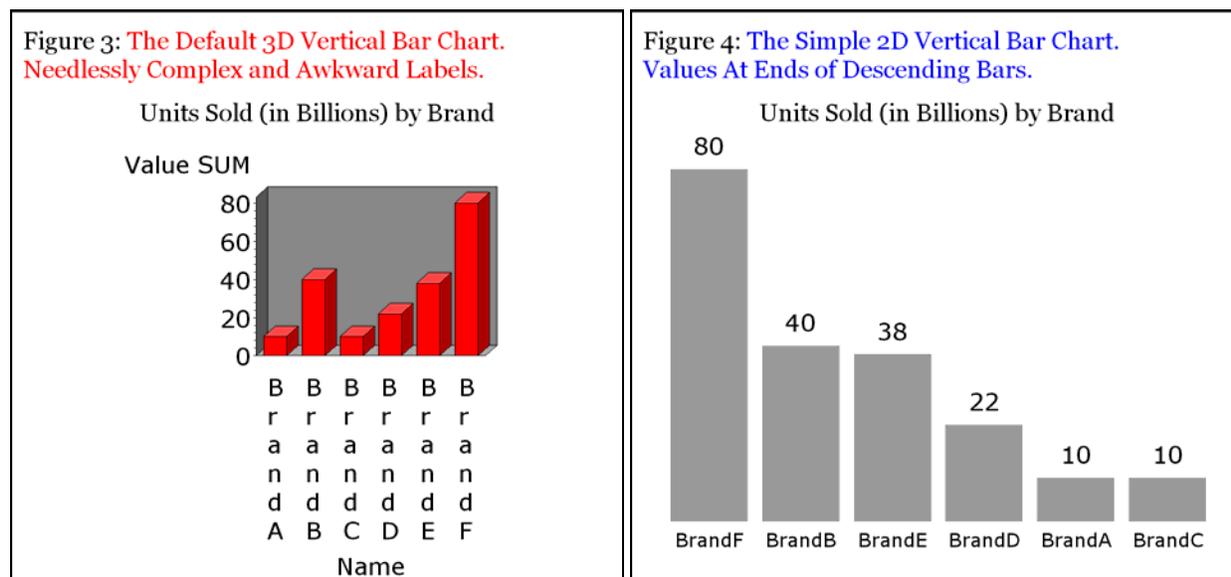
A common error is a misspelled font name, in which case the SAS log seems to always notify you of substitution. If you use font=NONE, or fail to specify font= where applicable, in your SAS/GRAPH program, the device driver uses its default font. If, when getting a default font, you specify a height other than one cell (i.e., not height=1), then the SIMULATE font will be used, but you will not be notified. SAS/GRAPH software, as shipped, comes with GOPTIONS SIMFONT=SIMULATE. You can override SIMFONT= to use any other SAS/GRAPH software font (not, e.g., a Windows font). I cannot guarantee that you will like the outcome with a different font, but I can tell you that I think the SIMULATE font is never acceptable.

Set your preferred SAS/GRAPH defaults with FTEXT, HTEXT, and CTEXT. The text for certain features of some SAS/GRAPH charts are not controllable with font=, height=, color=. For them, these GOPTIONS parameters are your only recourse.

Use Windows TrueType fonts. Good fonts include Matthew Carter's creations designed for readability on the screen and the web, Verdana (sans serif—useful for small letters and numbers on your graph) and Georgia (serif—useful for titles and important footnotes). See how these fonts are used in the Figures.

## Bar Charts

In Figure 3, the third dimension adds no communication value. I recommend the obviously better 2D bar chart in Figure 4.



Here are the strengths of Figure 4. The bars are ranked from highest to lowest, to “Show them what is important.” There is nothing in the image that is dispensable. The bars are lightly shaded. A color would add nothing. Using BLACK as the area fill would cause the image to be unnecessarily dominated by the area fill. Using an EMPTY area fill (hollow rectangles) can cause visual confusion when there are numerous bars, between what are the bars and what are the spaces.

Tip: In a horizontal bar chart with numerous entries (e.g., some measurement for each of the fifty United States of America), rather than ordering bars by size, it may be more useful to support quick look-up of a specific entry with alphabetical order.

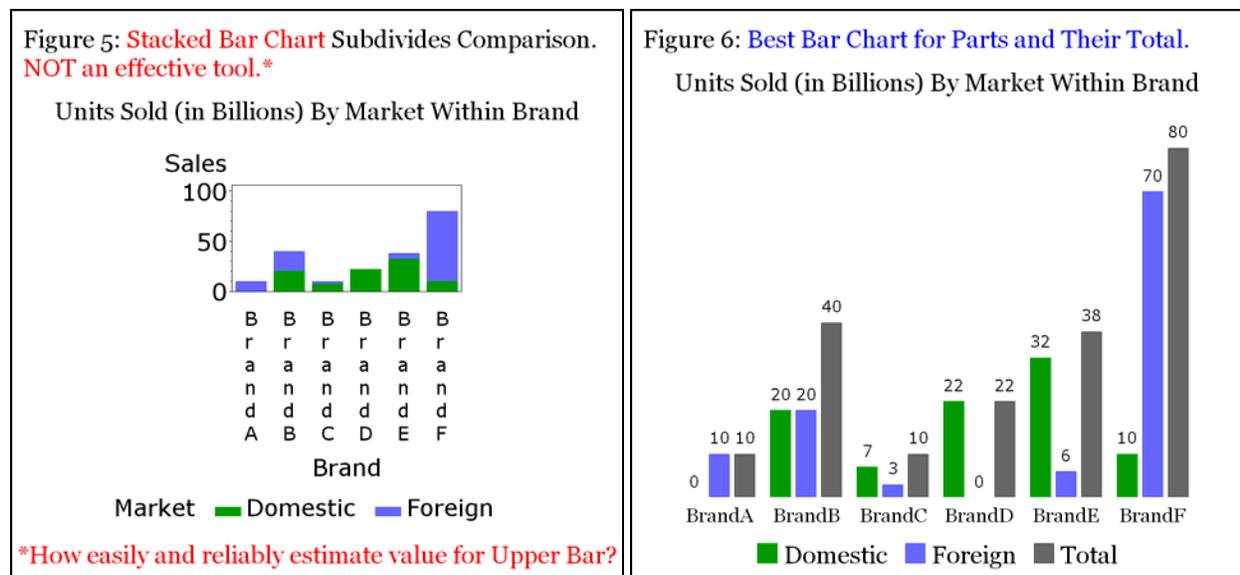
Here is the code used to create Figure 4:

```

/* Common Preliminary Code presented in the Appendix would be inserted here. */
title2 height=5 PCT font='Georgia'
  justify=LEFT ' Figure 4: ' color=CX0000FF 'The Simple 2D Vertical Bar Chart.'
  justify=LEFT ' Values At Ends of Descending Bars.';
title3 height=2.5 PCT ' ';
title4 height=5 PCT font='Georgia' justify=CENTER 'Units Sold (in Billions) by Brand';
footnote1 height=2.5 PCT ' '; /* white space at bottom */
pattern1 v=solid color=CX999999; /* browser-safe light gray */
axis1 label=none major=none minor=none style=0 value=(height=4 PCT);
axis2 label=none major=none minor=none style=0 value=none;
proc gchart data=DataForSimpleCharts;
vbar Name / sumvar=Value sum /* The SUM option causes the display of the SUMVAR at the top of each bar. */
  maxis=axis1 raxis=axis2 descending width=15 space=3;
run; quit;

```

A Stacked Bar Chart (which can also be rendered horizontally) is another regrettably popular tool for ineffective communication.



For the bar chart in Figure 6, SAS/GRAPH does not provide a legend. So, a custom footnote is used to create a legend. This example also provides a solution for the general case of a side-by-side bar chart, even when pre-processing has not been needed to provide a set of Total bars.

Here is the code used to create Figure 6:

```
data UnitsSoldByBrandAndMarket;
infile cards;
input @1 Brand $6. @8 Market $8. @17 Value 2.;
cards;
BrandF Domestic 10
BrandF Foreign 70
BrandA Domestic 0
BrandA Foreign 10
BrandB Domestic 20
BrandB Foreign 20
BrandC Domestic 7
BrandC Foreign 3
BrandD Domestic 22
BrandD Foreign 0
BrandE Domestic 32
BrandE Foreign 6
; run;

proc sort data=UnitsSoldByBrandAndMarket; by Brand Market; run;

data UnitsSoldWithTotals;
length Market $ 8;
retain Total 0;
set UnitsSoldByBrandAndMarket;
by Brand;
Total=Total+Value; output;
if last.Brand;
Market='Total'; Value=Total; output; Total=0;
run;

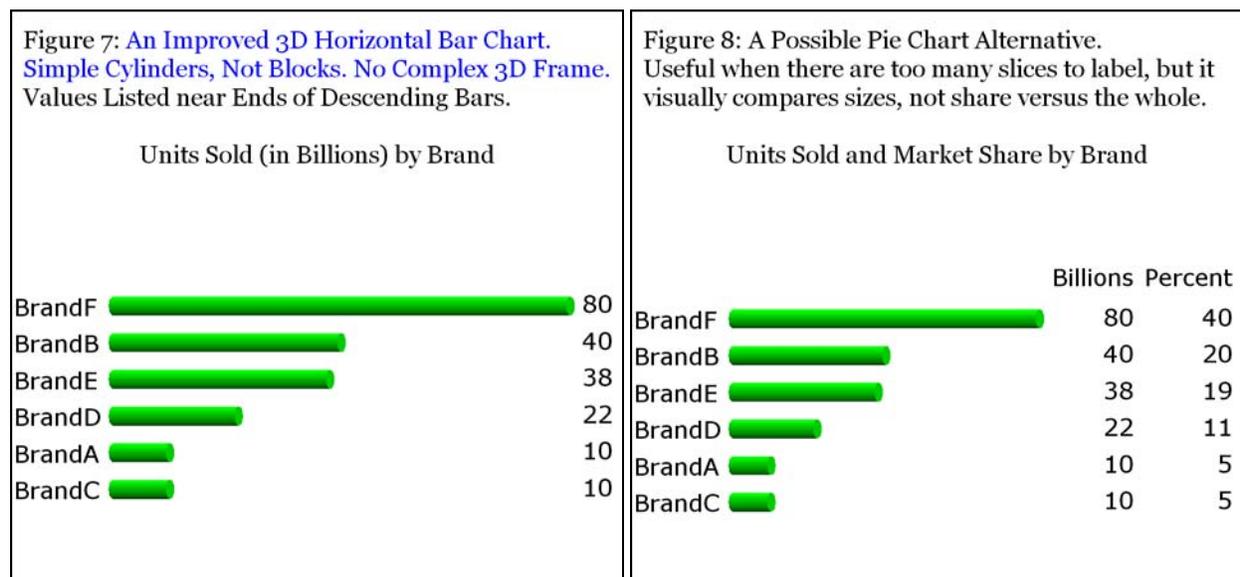
/* Common Preliminary Code presented in the Appendix would be inserted here. */
title2 height=5 PCT font='Georgia' justify=LEFT
' Figure 6: ' color=CX0000FF 'Best Bar Chart for Parts and Their Total.';
title3 height=2.5 PCT ' ';
title4 height=5 PCT font='Georgia' 'Units Sold (in Billions) By Market Within Brand';
title5 height=5 PCT ' ';
```

```

footnote1 height=5 PCT justify=CENTER
  font='Monotype Sorts' color=CX009900 '6E'X
  font='Georgia' color=CX000000 ' Domestic '
  font='Monotype Sorts' color=CX6666FF '6E'X
  font='Georgia' color=CX000000 ' Foreign '
  font='Monotype Sorts' color=CX666666 '6E'X
  font='Georgia' color=CX000000 ' Total';
footnote2 height=2.5 PCT ' ';
pattern1 v=solid color=CX009900;
pattern2 v=solid color=CX6666FF;
pattern3 v=solid color=CX666666;
axis1 label=none major=none minor=none style=0 value=none;
axis2 label=none major=none minor=none style=0 value=none offset=(0.5,0);
axis3 label=none value=(font='Georgia' height=4 PCT);
goptions htext=3.5 PCT; /* size of bar-end value labels */
proc gchart data=UnitsSoldWithTotals;
vbar Market / sumvar=Value sum group=Brand
  maxis=axis1 raxis=axis2 gaxis=axis3 patternid=midpoint;
run; quit;

```

From the few examples in Figures 3-6, it is obvious that vertical bar charts have limited ability to display bar labels, unless you resort to (what I regard as an undesirable option of) tilted labels. Here is a pair of horizontal bar charts that are always communication-effective, never require tilted labels for readability, and use 3D with no detrimental effects.



With a minor enhancement to Figure 7 we can achieve the considerable benefit provided by Figure 8. Namely, by specifying five parameters in the HBAR statement, we get what can be used in lieu of a pie chart in situations where no pie chart is feasible. The alternative of combining multiple small slices into OTHER—is anti-communicative, and just invites two questions, “What is in OTHER? How big are the pieces?” To communicate, your graph should answer questions, not create them.

Here is the code to create Figure 7:

```

/* Common Preliminary Code presented in the Appendix would be inserted here. */
title2 height=5 PCT font='Georgia'
  justify=LEFT ' Figure 7: ' color=CX0000FF 'An Improved 3D Horizontal Bar Chart.'
  justify=LEFT ' Simple Cylinders, Not Blocks. No Complex 3D Frame. '
  justify=LEFT color=CX000000 ' Values Listed near Ends of Descending Bars.';
title3 height=2.5 PCT ' ';
title4 height=5 PCT font='Georgia' justify=CENTER 'Units Sold (in Billions) by Brand';
pattern1 v=solid color=CX00FF00;
axis1 label=none major=none minor=none style=0 noplane;
axis2 label=none major=none minor=none style=0 noplane value=none;
proc gchart data=DataForSimpleCharts;
hbar3d Name / sumvar=Value sum sumlabel=' '
  descending maxis=axis1 raxis=axis2 width=1.6 space=1.6 noframe shape=cylinder;
run; quit;

```

The keys to creating Figure 8 are these statements:

```
axis1 label=none major=none minor=none style=0 noplane offset=(0,7 PCT);
/* Without the OFFSET as a problem circumvention, the labels would overlap the first row of values */
hbar3d Name / freq=Value freq freqlabel='Billions' percent percentlabel='Percent'
descending maxis=axis1 raxis=axis2 width=1.6 space=1.6 noframe shape=cylinder;
```

The use of FREQ= rather than SUMVAR= above is counterintuitive, but is necessary to get the result shown. An annotated alternative to Figure 8 that is useful for a chart with numerous bars (especially with many short bars) is provided in Reference 3.

## Trend Charts

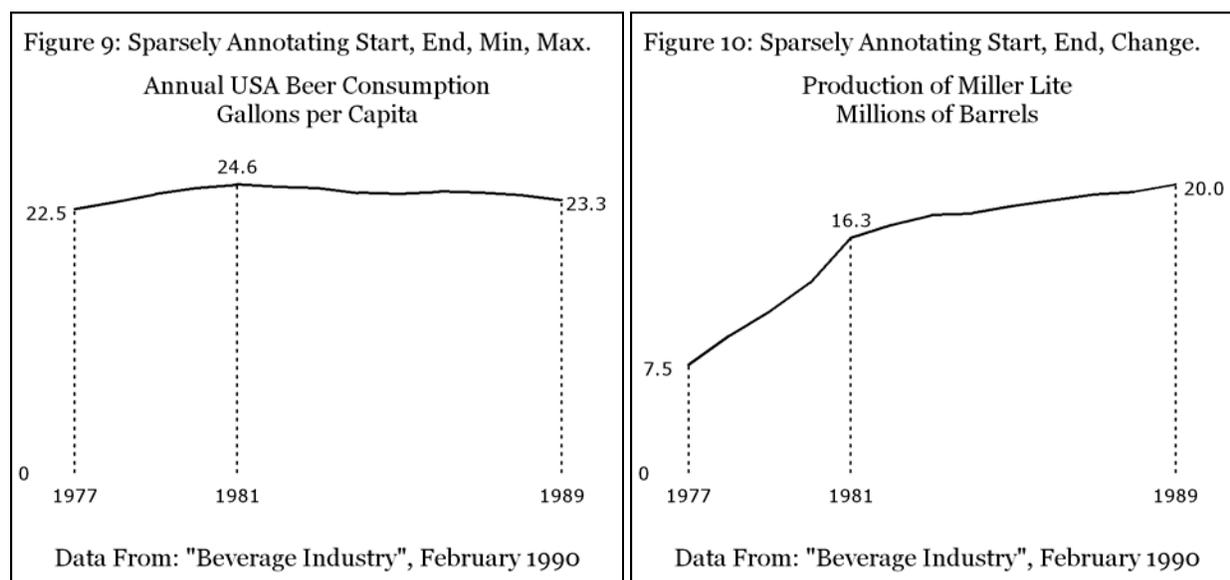
For bar charts, it is easy to deliver both image (for quick and easy inference) and precise detailed numbers (for reliable inference), and there is no real need for the traditional axis paraphernalia. For trend charts (or scatter plots), unless you deliver them via the web with pop-up text at the data points, it is usually more difficult to deliver both image and unambiguous detail. I do not recommend estimating y values by comparing the position of a data point with vertical axis tick mark values. For even the start and end points, it is always only an estimate, not the precise number.

Earlier I have emphasized the value of sparse annotation. For typical management reporting or an enterprise performance management system, the graph user is rarely interested in precise information about every data point displayed.

There are four common questions. How are we doing right now? What is the trend? What have been the best and worst numbers? When did the trend change?

Below is a pair of graphs for real business data. I prepared them a very long time ago when I worked in the beer industry. For both graphs I deliberately start the vertical axis at zero, rather than artificially magnify any trends. As point of information, back in 1990 the US beer industry recent historical trend was described as “flat”, and my axis-range handling conveys such an image, reinforcing that characterization.

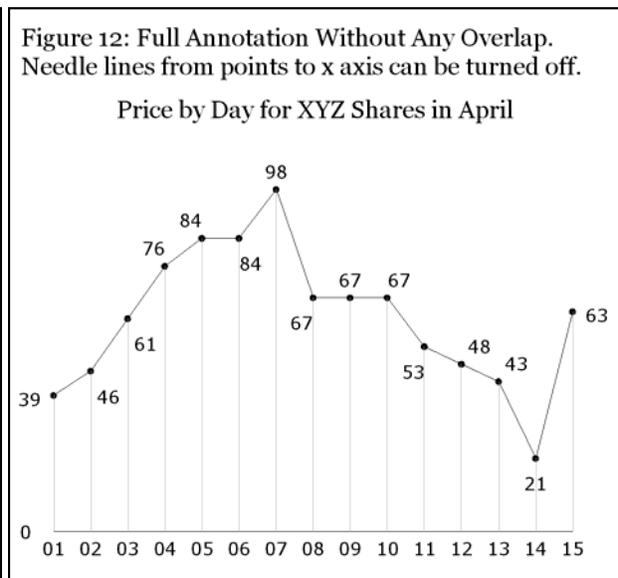
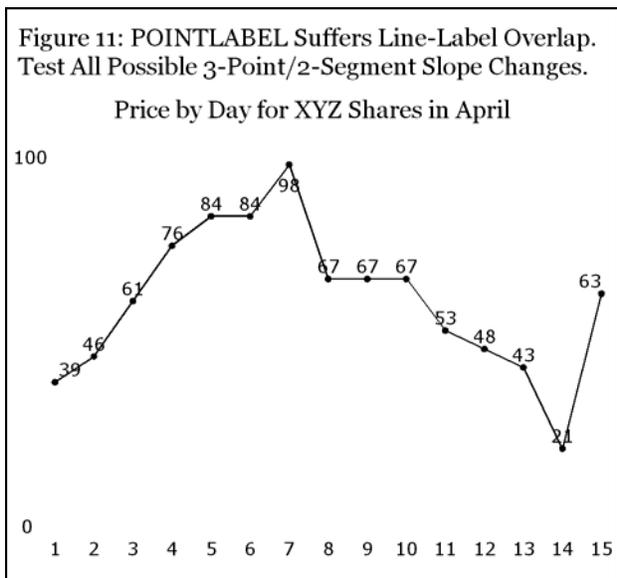
In principle, it would have been nice to put both trends in the same graph. SAS/GRAPH permits you to use one scale on the left axis for one y-variable and a different scale on the right axis for a second y-variable. In the general case, annotated multi-line graphs inevitably suffer annotation label overlap. You can sometimes remove the overlap through iterative trial-and-error, but I am always looking for robust solutions that get things right the first time every time, and can be employed for production graphic applications, which must run “hands-off”. Though these graphs were prepared in an ad hoc situation, I decided to keep things simple. Combining graphs would have just had significance in this special situation of a conjunction of significant critical points.



Here is the code to create Figure 9 (from this, the code for Figure 10 should be obvious):

```
data AllUSAbbeer;
input yy measure;
datalines;
77 22.5
78 23.1
79 23.8
```





I prefer to use a custom macro that yields the results shown in Figures 12 and 13. The needle lines in Figure 12 are a macro option, and can be suppressed. When they are used, there is some text-line overlap, but its effect is insignificant. In Figure 13, only the x-values for the critical points are displayed on the axis, and there is no ambiguity about which annotated y-value belongs to which x-value.

The result in Figure 12 makes it clear that there is no inherent reason why the SAS/GRAPH POINTLABEL algorithm cannot be improved. Figures 12 and 13 are created with a custom Trend macro that is available via email upon request to the author. An old edition of the Trend macro was provided in Reference 4. The macro is too complex and lengthy to present here. Frankly, I became convinced that Figure 14 is a better solution, and its generalization to Figure 16 addresses the multi-line plot situation, where collision-free annotation is almost always impossible.

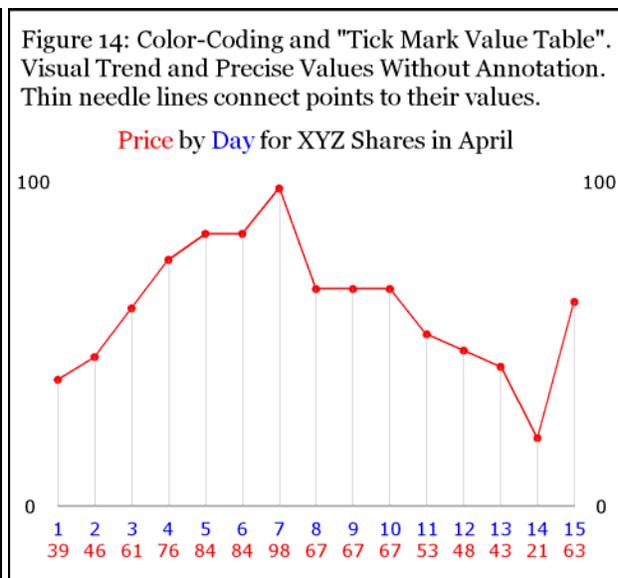
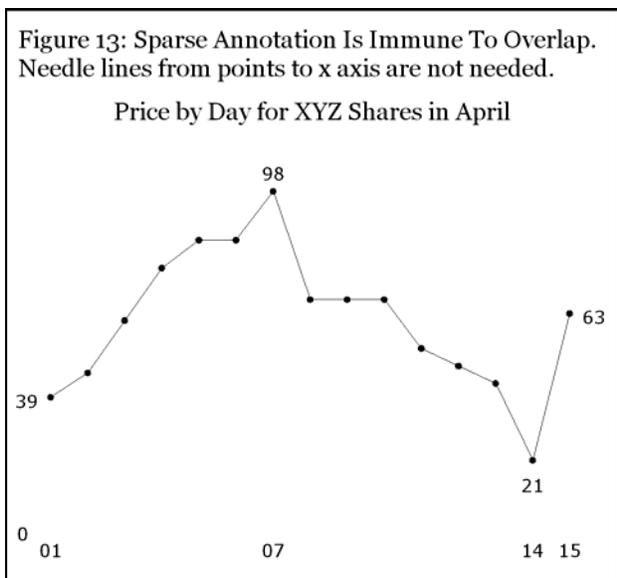


Figure 14 provides a trend line for quick assessment, plus the precision of an on-graph table built into tick mark labels. A graph is a visual aid to accelerate decision-making and inferences. A table of precise values (or annotation of them on the graph) is a necessity for reliable decision-making and inferences. Since all y-values are in the table, there is no need for a y-axis.

Here is the code used to create Figure 14:

```

%macro MakeTicks;
%do i = 1 %to &ValCount;
tick=&i "&&x_value&i" j=C color=CXFF0000 "&&y_value&i"
%end;
%mend MakeTicks;
    
```

```

data _null_;
set allways end=lastobs;
call symput('y_value' || trim(left(_N_)), trim(left(yvalue)));
call symput('x_value' || trim(left(_N_)), trim(left(xvalue)));
if lastobs;
call symput('ValCount', _N_);
run;

/* Common Preliminary Code presented in the Appendix would be inserted here. */
title2 height=5 PCT font='Georgia'
  justify=LEFT ' Figure 14: Color-Coding and "Tick Mark Value Table".'
  justify=LEFT ' Visual Trend and Precise Values Without Annotation.'
  justify=LEFT ' Thin needle lines connect points to their values.';
title3 height=2.5 PCT ' ';
title4 font='Georgia' height=5 PCT
  color=CXFF0000 'Price ' color=CX000000 'by ' color=CX0000FF 'Day '
  color=CX000000 'for XYZ Shares in April';
footnotel height=2 PCT ' ';
axis1 label=none major=none minor=none style=0
  value=(font='Verdana' color=CX0000FF height=3.75 PCT %MakeTicks);
axis2 label=none major=(number=2 color=CXFFFFFF) minor=none style=0
  value=(font='Verdana' height=3.75 PCT color=CX000000
    tick=1 justify=LEFT tick=2 justify=RIGHT);
axis3 label=none major=(number=2 color=CXFFFFFF) minor=none style=0
  value=(font='Verdana' height=3.75 PCT color=CX000000
    tick=1 justify=LEFT tick=2 justify=RIGHT);
symbol1 color=CXFF0000 height=2.5 PCT width=2 v=dot interpol=join;
symbol2 v=none interpol=needle color=CX999999;
proc gplot data=allways;
plot yvalue*xvalue / haxis=axis1 vaxis=axis2 vzero;
plot2 yvalue*xvalue=2 / haxis=axis1 vaxis=axis3 vzero;
format xvalue 2. yvalue 3.;
run; quit;

```

The disadvantage of Figure 14 is that it will not communicate effectively if printed in black and white, or if a color print of it is subsequently copied in black and white. A solution to this problem is provided in the example shown in Figure 16, which is a generalization of the tick mark value table to a multi-line plot.

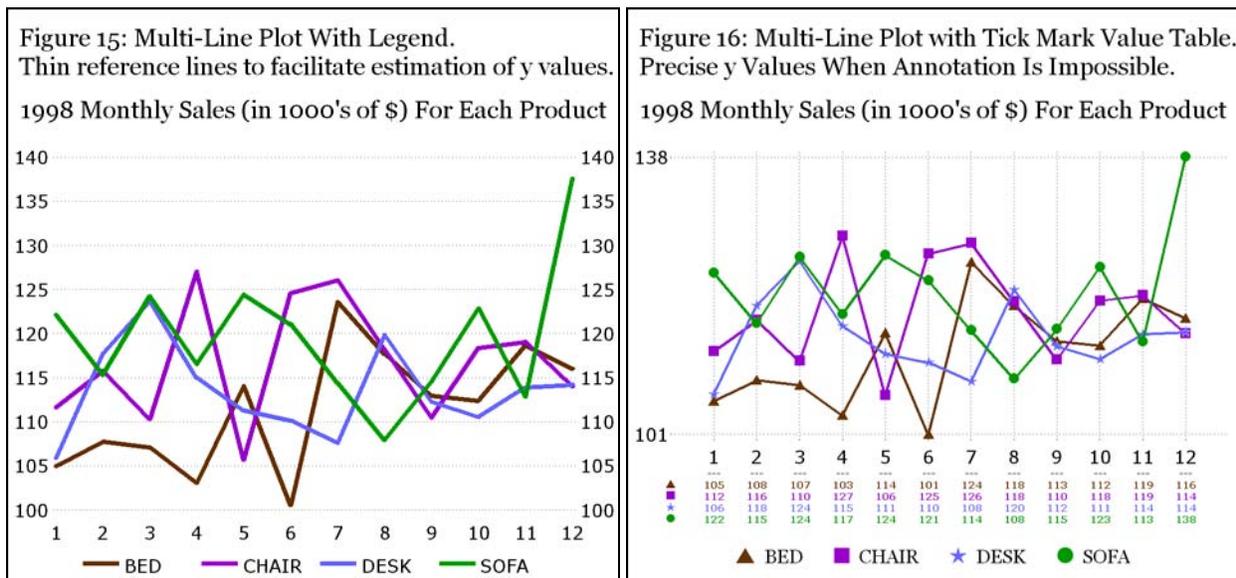


Figure 15 facilitates estimation of y values on a multi-line plot, but delivers neither certainty nor precision.

Here is the code used to create Figure 15:

```

data DataForFourLineTrendPlot(drop=Date Year);
set sashelp.prdsal3(where=(Year=1998) keep=Year Product Actual Date);

```

```

Month = month(Date);
run;

proc summary nway data=DataForFourLineTrendPlot;
class Product Month;
var Actual;
output out=DataForFourLineTrendPlot(drop=_type_ _freq_) sum=Actual;
run;

data ToPlot;
drop Actual;
set DataForFourLineTrendPlot;
Sales = Actual / 1000;
run;

/* Common Preliminary Code presented in the Appendix would be inserted here. */
title2 height=5 PCT font='Georgia'
  justify=LEFT ' Figure 15: Multi-Line Plot With Legend.'
  justify=LEFT ' Thin reference lines to facilitate estimation of y values.';
title3 height=2.5 PCT ' ';
title4 height=5 PCT font='Georgia'
  '1998 Monthly Sales (in 1000's of $) For Each Product';
axis1 label=none major=none minor=none style=0 value=(height=3.75 PCT);
axis2 label=none major=(color=CXFFFFFF) minor=none style=0 value=(height=3.75 PCT)
  order=100 to 140 by 5;
axis3 label=none major=(color=CXFFFFFF) minor=none style=0 value=(height=3.75 PCT j=L)
  order=100 to 140 by 5;
symbol1 v=none i=join w=6 color=CX663300;
symbol2 v=none i=join w=6 color=CX9900CC;
symbol3 v=none i=join w=6 color=CX6666FF;
symbol4 v=none i=join w=6 color=CX009900;
symbol5 v=none i=none r=4;
legend1 label=none value=(height=3.75 PCT) shape=line(7);
proc gplot data=ToPlot;
plot Sales*Month=Product / haxis=axis1 vaxis=axis2 legend=legend1 autovref lvref=33;
plot2 Sales*Month=Product / vaxis=axis3 nolegend;
run; quit;

```

Figure 16 delivers the perfect combination of an easily and quickly interpreted visual with precise data for the plot points. For the very complex code required to create it, please see the Appendix.

## Design and Construction for the Web

To put your graph in a web page, use the GIF driver, and wrap your graph code as follows:

```

goptions reset=all; goptions device=GIF; ods listing close;
ods noresults;
ods html path='c:\YourFolderName' (url=none) /* where web pages and graph image files are stored */
  gtitle gfootnote /* to force graph titles and footnotes into the graph image file, not the web page background */
  body= 'YourWebPageFileName.html'
  (title='text appears in upper left corner of web page in the web browser title bar')
style=styles.YourCustomStyle; /* or, say, styles.Minimal, but NOT styles.Default */
/* YOUR GRAPH CODE GOES HERE */
ods html close; ods listing;

```

When creating web graphs, you omit the `gsfname` `GOPTION`, and the `FILENAME` statement to connect the `gsfname` with a specific location and filename for the GIF file on disk. To control the name of the GIF file that is stored in the `PATH=` folder, put `NAME='XXXXXXXX'` after the `/` in the `PIE`, `HBAR`, `VBAR`, or `PLOT` statement (or `CHORO` statement, in the case of a map), where `XXXXXXXX` is a unique 8-character name.

Using the `PATH='...' (url=none)` assignment above guarantees that the folder of web pages and graphs is relocatable. You can copy the folder to anywhere, and the web pages will display the graphs. You can even zip the folder, and email the zip file to anyone, who can then unzip the folder to any location of her/his choosing, and the web pages will display the graphs. (For when to use `GPATH=`, please see the SAS/GRAPH Online Doc.)

Here are various uses of the text in TITLE=:

1. captured by search engines
2. title bar for the browser window
3. web page browse History list entry
4. default text for Internet Explorer Favorite Bookmark

The web page title text helps people find your web pages.

If the web page uses frames (e.g., an ODS Table-of-Contents-based presentation of information), then it is the TITLE= assignment for the frame file that appears in the browser window title bar, and presumably serves the other uses listed above.

For web page TITLE text, I usually use the graph title. For the filename, I usually use a compressed version of the web page title.

## Communicating with Text on the Web

No Blinking. It is annoying to some viewers. And it frustrates web-page-reading software for the visually impaired.

Beware of Marquee Text. This is like the traveling text at the bottom of a TV screen during a new broadcast. It, too, may frustrate web-page-reading software for the visually impaired.

Make your web page title your headline. If you are using the web page to persuade and/or reveal, don't be reluctant to tell the viewer what you know (or you think) it implies and/or shows. Presuming that the only thing you display in the web page is your graph, then its title presumably should be the web page title.

See the discussion of TITLE= in the section above.

Choose your background. The ODS default web page background of gray is boring, and does not enhance readability of foreground text. My recommendation: use one solid color. Textures and backgrounds vary the contrast with foreground text, impairing readability, besides being unnecessary and sometimes actually annoying. The presence of fancy backgrounds on web pages is too often a case of confusing the possible with the necessary.

There is only one way to preserve text appearance on the web. Fonts used inside graphs on a web page are embedded in the graph file that is part of the web page. So, the appearance of graph text is the same for all web viewers, and the same as that for the graph creator, except for screen resolution differences.

However, text outside graphs on a web page is affected by ODS, by the web browser, by what fonts are available in the font library of the viewing PC, and, potentially, by the web browser user, who can command the browser to enlarge or shrink ALL web page text outside of the graph. (In Internet Explorer, the user can assert control by clicking View and then Text Size.)

When putting SAS graphs in your web page, always use ODS HTML options GTITLE and GFOOTNOTE. This forces the TITLE and FOOTNOTE statements to be built inside your graph. Even if you use an "exotic" font, your choice will be seen on every web browser user's computer because its rendering will be inside your graph.

Provide a Web Page Magnification Option (a.k.a. "Zoom") for Viewers If Needed. At a certain level of complexity, it may become necessary to use small text on a graph. If web-deployed, it is possible to provide a magnification capability. The code required is very simple. See, e.g., <http://support.sas.com/faq/032/FAQ03230.html>.

## How Big Is Their Web Window to the World (and to Your Web Pages)

Designing for the web must take into account the pixel counts of the width and height of your graphic image and the likely pixel counts of the width and height of the web viewers' PC and browser.

The most recent report I have seen said that the commonest resolution (58%) on PCs is 1024 X 768. Fewer than 20% of the PCs were known to be higher resolution. None were found at the 640 X 480 resolution. A non-trivial fraction of the PC display is used by the web browser (and a little bit is used by Windows itself). The remainder is called "live space". Use it wisely. The most conservative design would be for a resolution of 800 X 600.

Some viewers may shorten their live space by use of an optional extra toolbar, such as that from Yahoo or Google.

To find out the characteristics of typical web users, try: [http://www.w3schools.com/browsers/browsers\\_stats.asp](http://www.w3schools.com/browsers/browsers_stats.asp). These statistics evolve over time, and I cannot guarantee currency of the information at this web site.

### Your Web Page Viewers Don't Really Want to Scroll

Design and build your web pages for FULL view on the smallest probable screen that will display it. The live space varies for PC vs. Mac, FireFox vs. Internet Explorer. For more information about live space, please see also References 2 and 5.

Vertical scrolling is tolerable, barely, and definitely not preferred. If your web page absolutely requires scrolling, put the most important information at the top of the page, and the least important at the bottom.

Horizontal scrolling not only is disliked by most web viewers, but also can frustrate effective viewing. Requirement for scrolling in both directions on the same page is unacceptable. However, there are exceptions. A large complex map cannot be displayed on the screen of any desktop or laptop PC, no matter how big the screen. A very large dense plot would be another exception. For a good example of the latter, see "Visualizing Patterns with Scrollable Web Graphics" by Eric C. Brinsfield and Caroline C. Bahler, in *Proceedings of the Twenty-Seventh SAS Users Group International Conference*, 2002.

If your web page content is difficult to fit on the screen without scrolling, you might consider making the text and image smaller, and deploying it with the web page zoom feature mentioned above. The viewer can magnify the image if it is a bit hard to read. This is not an ideal solution.

### Focus Your Web Page

Normally deliver only one graph per web page, if possible. If you can fit multiples on one screen, this suggestion does not apply. However, a scrollable multi-image, multi-element page can confuse the viewer. Also, if you click on a hyperlink whose target is really just part of a long page, and then print what you think is a small package of information about that topic, it is annoying to find 10 or 20 wasteful pages of output on your printer, with most of it being irrelevant to your interest.

A focused web page is less likely to require scrolling.

### Navigation Alternatives and the CrossLink Method

Please see References 5 and 6 for a discussion of web page navigation alternatives, including how to create a better ODS Table of Contents.

### Assure That Your Web Pages Work As You Intend: Avoid browser-specific or version-specific web design

The World Wide Web Consortium provides a facility to validate your web pages (and to evaluate conformance with their standards) at [validator.w3.org](http://validator.w3.org). Do not assume your web pages, which may work as desired with Internet Explorer, will work the same way with other browsers, or that they will work the same with all versions of IE. Recent (November 2006) statistics: 50% of browsers are Internet Explorer 6 and 30% of browsers are FireFox.

### Benefits of ALT Text

Those boxes of text that may pop up when you rest your mouse on an image, or on the place on the web page where an image will appear when it completes its download, are called ALT text. "ALT" is short for "Alternative", as in "Alternative to the Image". It is also called mouseover text, flyover text, floatover text, hover text, or a tool tip.

It is a recommended practice for web page developers to provide ALT text for all images. It has numerous benefits.

Vision-impaired users can use web-page-reading software that converts the ALT text to audible speech, but there are numerous benefits for unimpaired users as well.

With ALT text, you do not need to sacrifice live space on the web page to provide a "hard label" for an image.

You can supply labels-available-upon-mouse-request for situations where annotation is infeasible: (a) a plot that is very dense with points; or (b) a map with tiny geographic unit areas. If your web page viewers are not expected to need a printable record, ALT text can avoid forcing them to download a separate companion look-up table (which might be too large to fit on the same web page as the graph).

Web users can find out something about the image file that they are waiting to download until the picture is fully painted. Some may skip the wait and move on to some linked web page, or may focus on a different aspect of the current web page until the image arrives/completes.

You can be concise or verbose with ALT text. You can even format it as an information-rich multi-line, single-column table (as in, e.g., Figure 17).

In HTML, ALT= is a parameter that can be used with the IMG tag. In ODS, ALT= is a parameter that can be used with the PREHTML= and POSTHTML= options of the STYLE statement. With SAS/GRAPH, it is possible to provide ALT text for various parts of a graphic image, regardless of whether or not they are hyperlinked to other web pages. It can be assigned with the HTML parameter available for commonly used graphic PROCs. It can also be assigned with the HTML variable available for many Annotate functions. Here is an example of how you assign ALT text, after you have determined where to use the HTML (and/or HTML\_LEGEND, if your graph has a legend) parameter in your SAS/GRAPH PROC:

```
html= (or html_legend=) TextVar;
```

where TextVar is typically a data-dependent variable defined, e.g., with statements like:

```
length TextVar $ 200; /* or whatever is big enough to accommodate your longest possible value */
TextVar = ' alt="describe this area/point" href="OtherPageName.html" ';
```

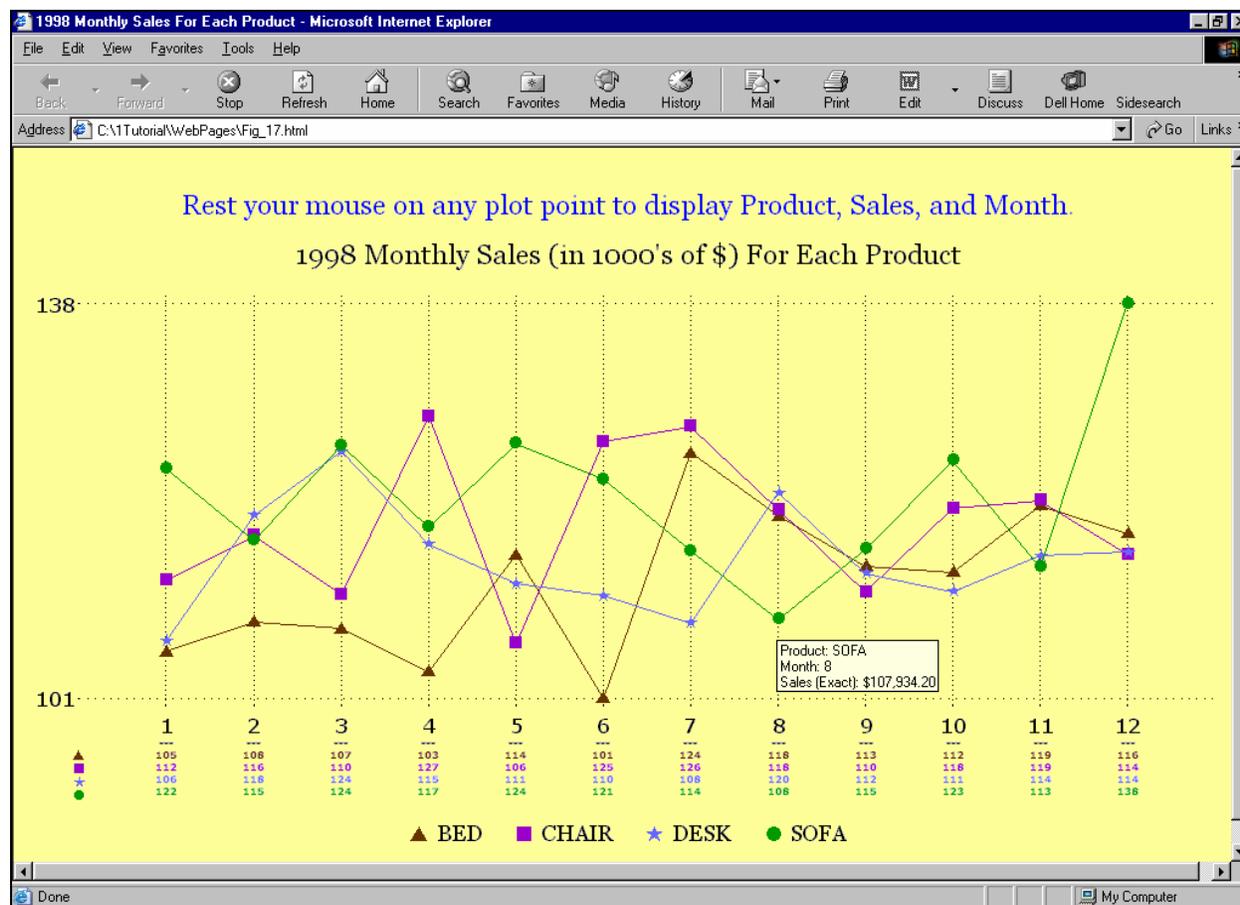
You need not use href= (i.e., to assign a hyperlink) to use **alt=**. In the Appendix, please see the ForWebGraph DATA step for an example of creating data-dependent ALT text.

### “Graph Overview” ALT Text with Minimal Effort in SAS 9.1.3

If you use **DES=** (in full, **DESCRIPTION=**) to describe a graph, that text will pop up automatically when the mouse hovers over the graphic image. If you omit **DES=**, then SAS will provide a default generic description, such as **plot of y\*x**, where y and x are replaced by your actual variable names. You can suppress the ALT text by specifying **DES= ' '**. The descriptive text can be up to 256 characters, and can include use of #BYLINE, #BYVAL, and #BYVAR substitution variables supported for BY processing. You can use **DES=** in the PIE, HBAR, VBAR, PLOT, and CHORO statements, among others.

**Figure 17. Web-Enabled (with ALT text) Version of Figure 16.**

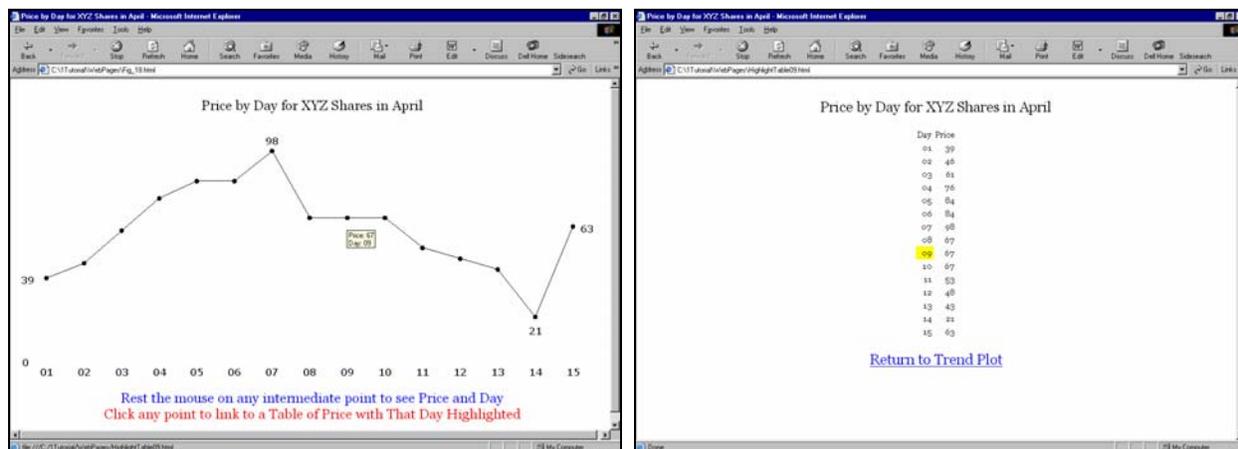
Mouse was resting on the plot point for Product Sofa in Month 8. (My screen “print” did not show the little hand of the mouse.)



The notice to the viewer about availability of ALT text could be displayed instead as a footnote. Since there is a chance that the viewer's web browser might not display the entire image without scrolling, and since I am willing to risk telling too much rather than too little, I prefer to assure that the viewer will know what is available. Please see the Appendix for the code used here.

**Figures 18. Web-Enabled (with ALT text and drill-down) Version of Figure 13.**

Mouse was rested on the plot point for Day 09, and then clicked to link to the Table with Day 09 Highlighted.



If you prefer the tick mark value table solution in Figures 14, 16, and 17, then the solution in Figures 18 may be unnecessary.

## Conclusion

If you can't do it with SAS/GRAPH, you may not really need to do it. So far, I have always been able to find a way.

## References (Related Work by the Author)

1. Communication-Effective Use of Color for Web Pages, Graphs, Tables, Maps, Text, and Print, *Proceedings of the Twenty-Ninth Annual SAS Users Group International Conference*, 2004.
2. How to Make the "Best Choice" from the Many Ways to Create and Deliver SAS Graphs, *Proceedings of the SAS Global Forum 2007*.
3. Communication-Effective Pie Charts, *Proceedings of the SAS Global Forum 2007*.
4. With Francesca Pierri, %TREND: A Macro to Produce Maximally Informative Trend Charts with SAS/GRAPH, SAS, and ODS for the Web or Hardcopy, *Proceedings of the Twenty-Seventh Annual SAS Users Group International Conference*, 2002.
5. Web Communication Effectiveness: Design and Methods to Get the Best Out of ODS, SAS, and SAS/GRAPH, *Proceedings of the Twenty-Eighth Annual SAS Users Group International Conference*, 2003.
6. With Francesca Pierri, Show Your Graphs and Tables at Their Best on the Web with ODS, *Proceedings of the Twenty-Seventh Annual SAS Users Group International Conference*, 2002. A much more expansive paper, with complete code, is on the *VIEWS 2001 Conference CD*, VIEWS - the Independent UK SAS Users Group, London, UK, 2001. The latter paper is available via email request to LeRoy Bessler.

## Acknowledgments

I am grateful to my colleague Alix Riley for her helpful review of this paper, and to SAS Global Forum 2007 Tutorial Section Chairs Sue Douglass and Harry Droogendyk for the opportunity to share my ideas with other SAS users.

## Contact Information

Your comments, questions, and suggestions are welcome. I am always interested in design ideas or construction solutions that enhance graphic communication.

LeRoy Bessler PhD

Email: [bessler@execpc.com](mailto:bessler@execpc.com)

Phone: 1 414 351 6748 (evenings and weekends—time is six hours earlier than Greenwich Mean Time)

SAS/GRAPH, SAS, and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.

## Appendix.

### Common Preliminary Code for All of the Non-Web-Enabled Graphs

```

goptions reset=all; /* It is best to do this reset before EVERY graph. */
goptions device=GIF;
goptions cback=CXFFFFFF; /* Background color is RGB White, which is actually the default. */
goptions border; /* Put the graph in a box, to separate it from text when being published in, e.g., Microsoft Word. You can instead apply
a border to the inserted graph using Microsoft Word itself. For this paper, both borders were applied. */
goptions htext=5 PCT ftext='Verdana'; /* height and font used for parts of graph for which you do not make an explicit
assignment, or for which no direct controls are available in SAS/GRAPH */
goptions ypixels=900 xpixels=960; /* Default for GIF is 800 X 600, which is a good ratio if later inserting to PowerPoint.
Here the target was really this Word document. */
title1 height=2.5 PCT ' '; /* Insert empty space between border and TITLE2.
Also, erase any titles left from prior graphs / reports in same session. */
footnotel; /* Erase any footnotes left from prior graphs / reports in same session. */
goptions gsfmode=replace gsfname=anyname; /* Output graph to disk with this fileref used in FILENAME statement below. */
filename anyname C:\YourFolderName\YourFileName.GIF";

```

### Input Data

Below is the code used to create the input for some of the graphs above. The values for any observations not listed can be read off the graphs created with them.

```

data DataForSimpleCharts;
infile datalines;
input @1 Name $6. @8 Value 2.;
datalines;
BrandA 10
etc.
; run;

data AllWays;
infile datalines;
input @1 xvalue 2. @5 yvalue 2.;
cards;
1 39
...
15 63
; run;

```

### Code Used To Create Figures 16 and 17.

```

%macro TickMarkValueTable(MarkerType=WindowsFont);
tick=1 color=CXFFFFFF height=3.75 PCT font='Verdana' "XXX"
justify=CENTER height=2 PCT font='Verdana/Bold' "XXX"
%do j = 1 %to &Zcount;
justify=CENTER color=&&color&j
%if %upcase(&MarkerType) eq WINDOWSFONT
%then %do;
font='Monotype Sorts' "&&winMarker&j"X
%end;
%else %do;
font=Marker "&&sasMarker&j"
%end;
%end;
%do i = 2 %to %eval(&Xcount + 1);
tick=&I color=CX000000 height=3.75 PCT font='Verdana' "&&Xvalue&i"
justify=CENTER height=2 PCT font='Verdana/Bold' '---'
%do j = 1 %to &Zcount;
justify=CENTER color=&&color&j "&&Y_value_X&i._Z&j"
%end;
%end;
%mend TickMarkValueTable;

%macro SymbolStatements
(NumberOfThemNeeded=,SymbolSize=2 PCT,LineWidth=2,MarkerType=WindowsFont);
%do i = 1 %to &NumberOfThemNeeded;
symbol&i interpol=join width=&LineWidth color=&&color&i height=&SymbolSize
%if %upcase(&MarkerType) eq WINDOWSFONT
%then %do;
font='Monotype Sorts' value="&&winMarker&i"X;
%end;

```

```

    %else %do;
        font=Marker          value="&&sasMarker&i";
    %end;
%end;
%global TotalLineCount;
%let TotalLineCount = %eval(&NumberOfThemNeeded + 1);
symbol&TotalLineCount interpol=none color=CXFFFFFFF;
%mend SymbolStatements;

%macro Legend(NumberOfYvalueLines=,
              SymbolSize=2.50 PCT,
              TextSize=3.75 PCT,
              MarkerType=WindowsFont);
%do i = 1 %to &NumberOfYvalueLines;
    height=&SymbolSize color=&&color&i
    %if %upcase(&MarkerType) eq WINDOWSFONT
    %then %do;
        font='Monotype Sorts' "&&winMarker&i"X
    %end;
    %else %do;
        font=Marker          "&&sasMarker&i"
    %end;
        height=&TextSize    color=CX000000
        font='Georgia'      " &&ZvalueDescription&i"
    %if &i ne &NumberOfYvalueLines %then %do;
        " "
    %end;
%end;
%end;
%mend Legend;

/* SAS markers for plots using MarkerType=SAS: C for triangle, U for square, V for star, W for circle */
%let winMarker1 = 73;
%let winMarker2 = 6E;
%let winMarker3 = 48;
%let winMarker4 = 6C;

%let color1 = CX663300;
%let color2 = CX9900CC;
%let color3 = CX6666FF;
%let color4 = CX009900;

data ForPlot; drop Month Product;
set DataForFourLineTrendPlot;
xvalue=Month; yvalue=Actual / 1000; zvalue=Product;
run;

proc means data=ForPlot min max range noprint;
var yvalue;
output out=MinMaxRangeY min=miny max=maxy range=rangey;
run;

data _null_;
set MinMaxRangeY;
call symput('minY',miny);
call symput('maxY',maxy);
call symput('rangeY',rangey);
call symput('DisplayMinY',trim(left(put(miny,3))));
call symput('DisplayMaxY',trim(left(put(maxy,3))));
run;

proc sort data=ForPlot out=DistinctX nodupkey;
by xvalue; run;

data _null_;
set DistinctX end=LastOne;
call symput('Xvalue' || trim(left(_N_ + 1)),trim(left(xvalue)));
if LastOne;
call symput('Xcount',trim(left(_N_)));
run;

proc sort data=ForPlot out=DistinctZ nodupkey;
by zvalue; run;

data _null_;
set DistinctZ end=LastOne;

```

```

call symput('ZvalueDescription' || trim(left(_N_)), trim(left(Zvalue)));
if LastOne;
call symput('Zcount', trim(left(_N_)));
run;

proc sort data=ForPlot;
by xvalue zvalue; run;

data VisiblePlotLines(keep=yvalue Actual xvalue XforPlot zvalue)
  AnyYvalueWouldSuffice(keep=yvalue);
retain WhichZ 0 WhichX 1;
set ForPlot end=LastOne;
by xvalue zvalue;
if first.xvalue then WhichZ = 1;
call symput('Y_value_X' || trim(left(put(WhichX+1,2.))) || '_Z' || trim(left(put(WhichZ,2.))),
  trim(left(put(yvalue,3.))));
XforPlot = WhichX;
output VisiblePlotLines;
WhichZ = WhichZ + 1;
if last.xvalue;
WhichX = WhichX + 1;
if LastOne;
output AnyYvalueWouldSuffice;
run;

data HiddenPlotLine(keep=XforPlot yvalue zvalue);
retain zvalue 'Z';
set DistinctX(keep=yvalue);
if _N_ eq 1 then set AnyYvalueWouldSuffice;
if _N_ eq 1 then do;
  XforPlot = 0;
  output;
end;
XforPlot = _N_;
output;
run;

data VisibleAndHidden;
set VisiblePlotLines HiddenPlotLine;
run;

%macro HrefValues;
%do i = 1 %to %eval(&Xcount);
&i
%end;
%mend HrefValues;

/* Common Preliminary Code presented earlier in this Appendix would be inserted here. */
title2 height=5 PCT font='Georgia'
  justify=LEFT ' Figure 16: Multi-Line Plot with Tick Mark Value Table.'
  justify=LEFT ' Precise y Values When Annotation Is Impossible.';
title3 height=2.5 PCT ' ';
title4 height=5 PCT font='Georgia' '1998 Monthly Sales (in 1000's of $) For Each Product';
footnote1 justify=CENTER height=2.5 PCT ' ' /* this will insert white space BELOW the footnote */
  justify=CENTER
  %Legend(NumberOfYvalueLines=&Zcount, SymbolSize=3.75 PCT, MarkerType=WindowsFont);
footnote2 angle=+90 height=1 PCT ' ' /* prevent VREF lines extending to edge of web page */
axis1 label=none major=none minor=none style=0 order=(0 to 12 by 1) offset=(0,7 PCT)
  value=(%TickMarkValueTable(MarkerType=WindowsFont));
axis2 label=none major=none minor=none style=0 order=&miny to &maxy by &rangeY
  value=(height=3.75 PCT font='Verdana' "&DisplayMinY" "&DisplayMaxY");
%SymbolStatements(NumberOfThemNeeded=&Zcount, LineWidth=3,
  SymbolSize=2.5 PCT, MarkerType=WindowsFont);
proc gplot data=VisibleAndHidden;
plot yvalue*XforPlot=zvalue / haxis=axis1 vaxis=axis2
  lhref=33 href=(%HrefValues) lvref=33 vref=(&miny &maxy)
  nolegend; /* suppress the automatic default legend */
run; quit;

proc template;
edit styles.minimal as styles.Minimal_YellowBackground;
  style body / background = CXFFFF99; /* Light Yellow web page background:
  all background not otherwise specified */
end; run;

```

```

data ForWebGraph;
length AltText $ 100;
set VisibleAndHidden;
if zvalue ne 'Z' /* if it is a visible plot point */
then AltText =
' alt=" ' || 'Product: ' || trim(left(zvalue))
|| '0D'X ||
'Month: ' || trim(left(xvalue))
|| '0D'X ||
'Sales (Exact): ' || trim(left(put(Actual,dollar11.2))) || ' " ' ;
run;

/* Do not use the Common Preliminary Code when doing web graphs. */
goptions reset=all;
goptions device=GIF;
goptions transparency;
goptions noborder; /* web graph border would be hidden, but would take up space */
goptions htext=5 PCT ftext='Verdana';
goptions xpixels=994 ypixels=555; /* defaults: 800 X 600 */
ods listing close; ods noresults;
ods html path='c:\YourFolder\WebPages' (url=none) /* where web pages and graph image files are stored */
gtitle gfootnote /* force titles and footnotes into the image, not into the web page background */
body= 'Fig_17.html' (title='1998 Monthly Sales For Each Product')
style=styles.Minimal_YellowBackground;
%SymbolStatements(NumberOfThemNeeded=&Zcount,LineWidth=1,
SymbolSize=2.5 PCT,MarkerType=WindowsFont);
axis1 label=none major=none minor=none style=0 order=(0 to 12 by 1) offset=(0,7 PCT)
value=(%TickMarkValueTable(MarkerType=WindowsFont));
axis2 label=none major=none minor=none style=0 order=&miny to &maxy by &rangeY
value=(height=3.75 PCT font='Verdana' "&DisplayMinY" "&DisplayMaxY");
footnote1 justify=CENTER height=2.5 PCT ' ' /* insert white space BELOW the footnote */
justify=CENTER
%Legend(NumberOfYvalueLines=&Zcount,SymbolSize=3.00 PCT,MarkerType=WindowsFont);
footnote2 angle=+90 height=1 PCT ' ' /* prevent VREF from extending to edge of web page */
title1 font='Georgia' height=5 PCT color=CX0000FF
'Rest your mouse on any plot point to display Product, Sales, and Month.'
justify=CENTER height=2.5 PCT ' ' justify=CENTER height=5 PCT color=CX000000
'1998 Monthly Sales (in 1000's of $) For Each Product';
proc gplot data=ForWebGraph;
plot yvalue*XforPlot=zvalue / html=AltText name='Fig_17'
haxis=axis1 vaxis=axis2 lhref=33 href=(%HrefValues) lvref=33 vref=(&miny &maxy)
nolegend; /* suppress the automatic default legend */
run; quit;
ods html close; ods listing;

```