

Data Visualization Tips and Techniques for Effective Communication

LeRoy Bessler

Bessler Consulting and Research, Mequon, Milwaukee, Wisconsin, USA

ABSTRACT

This tutorial presents my principles of communication-effective data visualization, and shows widely usable ways to implement them with new technology (ODS Graphics and Statistical Graphics Procedures), or, when it is the better choice, old technology (SAS/GRAPH Procedures). Most of the solutions use the new technology, which is imbedded in Version 9.3 of Base SAS software at no additional cost. I identify what the new technology can do best, and what to be wary of when using that technology.

INTRODUCTION

For over thirty years starting with SAS/GRAPH and now also with ODS Graphics and the new SG (Statistical Graphics) Procedures, I have been on a quest in pursuit of data insights and visual communication effectiveness

ODS Graphics and the SG procedures have options and features that need optimization by you to get communication-effective output rather than simply the default results that come with no added thought or effort. Also, the new tools manifest unexpected and unacceptable behaviors, which do not even announce themselves in the SAS log. The paper points these out, and shows how to cope with them. Nevertheless, after twenty-five years of sharing my SAS/GRAPH solutions with SAS users, I have become a convert to the new graphics technology.

This tutorial shows you easy ways to get beyond taking the defaults, to get better results with minimum time and effort. It covers pie, bar, and line charts written to disk to be inserted later into Microsoft PowerPoint, Word, or Excel, or simply printed, but also emphasizes and demonstrates the benefits of web-enabled graphs, and shows how to create them. For pie charts and bar charts it has long been possible to display the associated precise numbers as part of the image, but for graphs with plot points on multiple lines and/or with point-dense single lines, it has not. For time series or line charts, the paper demonstrates three ways to address this: (1) ALT text (a.k.a. "data tips", mouseover text, flyover text, hover text, floatover text, tool tips, or pop-up text—which is my name for it); (2) a companion spreadsheet linked to the plot; and (3) a table imbedded in the plot image. The first two options can be used only with web-enabled plots. The third option is available for static plots that are printed or deployed in slides or electronic documents, and for web-enabled plots, and is easier to do with ODS GRAPHICS than with SAS/GRAPH.

My focus is on graphs used for management reporting in business, government, or other organizational settings, not statistical or heavy-duty analytical graphs. Although the SG (Statistical Graphics) procedures were originally developed for statistical graphics, they can be used for management reporting. By graphs for management reporting I mean graphs or web graphs used to answer several common questions. How are things going? Are they better, worse, or about the same? How do measurements for different entities compare? Questions like these are commonly answered visually with time plots (or time series graphs), bar charts, and pie charts.

For dramatic and unequivocal proof of the importance of graphing your data, see Reference 1.

NOTE: In the PharmaSUG 2013 conference proceedings, there is a companion paper. For some of the coding examples presented in this paper, macro-based solutions are provided there to make it easier to create their results, and for other solutions, use of macro code is mandatory to make them possible. See Reference 2 for the macro tools.

UNEXPECTED THINGS ODS GRAPHICS DOES WITHOUT ANY WARNING MESSAGES

1. It can omit tick mark values that are essential (categorical character values, not values of a continuous numeric variable) or in cases where you have explicitly specified the numeric or date values to be listed. This astonishing (and frustrating) feature is described in great detail in the paper, and circumvention options are presented.
2. It creates line breaks in your titles and footnotes when font size is too large for all of the text to fit on one line.

MY WISH FOR ODS GRAPHICS—the ability to force DESIRED line breaks with coding so that specification of text characteristics (font, size, color, etc.) does not have to be repeated in every TITLE statement.

ALSO MISSING IN ODS GRAPHICS—the ability to specify default characteristics for all text in the graph for which characteristics are not explicitly specified in code. Using GTL to create a custom style is extra and complex work.

My thanks to Laura MacBride for her helpful review of this paper.

PRINCIPLES OF COMMUNICATION-EFFECTIVE GRAPHICS

I have recently been using *Principia Graphika* as a title for concise handouts on the Principles of Communication-Effective Graphics. They are some of the ideas and techniques that I have been implementing, advocating, and writing/speaking about for more than two decades.

So, what are the Principles? Well, there are a lot of them, previously reported by me elsewhere. In this paper, the focus is on these:

- Deliver image plus precise numbers: image for quick easy inference, precise numbers for reliable inference
- Provide ordering—Show them what's important
- Subset the content where appropriate—Show them what's important
- Provide a reliably usable legend—A SAS user should be able to, but cannot, assume that this is built-in
- Suppress and avoid graphic frills—Let your data talk
- Make wise use of web enablement
- IF using color, which is often unnecessary, use it in a communication-effective manner

Communication-Effective Graphic Design Principle 1: Deliver image plus precise numbers

Graphs accelerate and facilitate inference and decision-making, but the actual numbers are required for reliable inference and decisions.

My scope does include static graphs, i.e., ones destined directly for print (less frequent these days) or for imbedding in PowerPoint for presentations, in PDF for Adobe Acrobat Reader, or in RTF for Microsoft Word. However, web-enabled graphs are my preferred choice now, whenever they are an acceptable information delivery method. Web enablement provides quick and easy navigation between graph and graph or between graph and table/spreadsheet. It also supports what is officially called ALT text.

ALT text is an accessibility aid. For visually impaired web users, the HTML source code to support ALT text can be converted into audio by so-called screen reader software. But the pop-up data tips that ALT text provides are helpful for any web user without visual impairment.

Among the options that web-enabled graphs offer is the ability to connect them, forwards and backwards, with SAS-created spreadsheets. Besides allowing detail look-up, the benefit of a spreadsheet rather than just a SAS table is that the user then has the option of post-processing the graph-supporting data in any way desired, using a tool that almost any user already has and knows how to use. That option was previously implemented with SAS/GRAPH (see Reference 4), but it is extended here to ODS GRAPHICS and PROC SGPLOT. However, unlike the case when using SAS/GRAPH PROC GPLOT, **you must add NOGTITLE to the ODS HTML statement in order to get the link to the spreadsheet from the web-deployed graph to work.**

For scatter plots and line charts, if they are very point-dense or multi-line, it can be difficult, if not impossible, to provide annotation for the data points that does not suffer collisions between point labels and/or between lines or points and the point labels. For them, web enablement with ALT text and/or a companion table is essential.

Bar charts and pie charts can be created with all detail numbers in the image. Code here routes the charts to a disk location, from which they can be inserted in a slide or a Microsoft Word document, but they could be web-enabled.

Not demonstrated here is use of the URL parameter for HBAR or SERIES statements to make bars or points drillable. URL= specifies a variable to contain the bar- or point-appropriate hyperlink.

Available as a solution for both static plots and web-enabled plots is a table imbedded in the image, or using ODS to package the image of a plot with a report procedure table concatenated below it. ODS Graphics makes table imbedding easy, as is shown in this paper. However, the possibility of compound graph-plus-table ODS packaging is a well-known and long-ago well-documented capability, and will not be covered here.

Communication-Effective Graphic Design Principle 2: Provide ordering

One of my favorite titles from past papers that I have written is—

Show them what's important: Solutions for a finite work day in an era of information overload

The default ordering provided by SAS graphics software has always been alphabetical ordering of label text: labels of bars in bar charts, labels of pie slices in pie charts, or labels of legend entries.

If you are doing a horizontal bar chart of measures for, say, all 50 states in the USA, or all of the counties in one of the states, or all of the over 200 countries of the world, and you want the chart to be used as an easy look-up tool, rather than primarily as a tool to quickly identify the entity with the most or least impactful measure, then alphabetical order of the labels is the right choice.

However, if largeness (or smallness) of the measure of interest in a bar chart or pie chart is important, then it is best to order the bars or slices from largest to smallest (or smallest to largest). This technique is even more effective when combined with use of Principle 3.

Less obvious kinds of ordering that I have also used (but not here) for the legend of a multi-line plot is to order the entries by:

- the value of the ending y-value for each line
- the average y-value for each line
- the peak y-value for each line

Communication-Effective Graphic Design Principle 3: Subset the content where appropriate

One of my favorite titles from past papers that I have written is—

Show them what's important: Solutions for a finite work day in an era of information overload

The default action performed by SAS graphics software, rightly, is to use all of the input data that you provide it.

If you are doing a horizontal bar chart of measures for, say, all 50 states in the USA, or all of the counties in one of the states, or all of the over 200 countries of the world, and you want the chart to be used as a comprehensive look-up tool, rather than primarily as a tool to quickly identify the entity with the most or least impactful measure, then presenting all of the possible bars is the right choice.

However, if largeness (or smallness) of the measure of interest in a bar chart, then you can choose to present only a subset of the ranked bars from largest to smallest (or smallest to largest). This technique combines use of Principles 2 and 3.

The idea of subsetting the information was concisely and best described many years ago in a recommendation of Jim White in an article on effective communication with print:

Let part stand for the whole.

It is a fact that often an overwhelming share—80%, 90%, or more—of the total measure of interest is often accounted for by a small number of contributors to the total.

Focus attention where attention is merited, rather than overwhelming the graph viewer with all possible information.

When subsetting the information, it can be useful to also make it easy to get to The Big Picture for anyone who really wants to or needs to see it all. With web-enabled graphs, as is shown in Reference 2, you can provide a pair of linked-by-a-click graphs: one for the “Top N” (N is your choice of 10, 25, or whatever) and the other for “All”.

Other ways to subset the data are with a simple cut-off filter, or (a method demonstrated by me in prior work and Reference 2, but not here) to use only enough ranked observations to account for at least P% of the grand total of the measure of interest. The last method answers this question: “Which of the most impactful of these entities are responsible for, say, 90% of the measure that I am concerned about?”

Many years ago, when I was agonizing over preparing a report for executive management, Kenneth J. Wesley advised me thus:

If it doesn't fit on one page, they won't read it.

Subsetting the data lets you present your data on one paper page certainly, and, if sufficiently subsetting, in one web browser window without scrolling.

Communication-Effective Graphic Design Principle 4: Provide a reliably usable legend

In graphs, legends are typically used to decode the identity of colored plot lines and markers or of color-filled bars or pie slices.

The usability of a color-based legend is related to four insufficiently recognized facts of color communication.

NOTE: If you want to be able to reliably distinguish colors, then

- 1. colored text must be thick enough**
- 2. colored lines must be thick enough**
- 3. colored plot point markers must be large enough**
- 4. colored legend samples, whether area fill, line segment, or plot point marker, must be large enough**

ODS Graphics does provide controls for thickness of lines and size of plot point markers. And that thickness and size also carry over into the legend for a plot.

Unfortunately, there are no ODS Graphics controls for the size of the colored sample patches in bar chart legends. SAS/GRAPH provides that support. ODS Graphics does not. This same functional deficiency has always been present in Excel pie chart legends. The lack of control of legend color samples in ODS Graphics is unacceptable.

Communication-Effective Graphic Design Principle 5: Suppress and avoid graphic frills

Let your data talk.

ODS Graphics, unlike SAS/GRAPH, somewhat limits the user's ability to inflict needless non-communicative or anti-communicative decoration (e.g., no 3D features for what are really two-variable 2D graphs). However, it does retain the standard graphic paraphernalia from the days when laboratory data was plotted with ink on grid paper. Coding to remove that paraphernalia is presented here later.

Note about Color. Though it might not be thought of as such, a too common, and often counter-productive, graphic frill is color.

Why should you care about clutter? Here are miniature excerpts of some of my favorite slides to answer this.

**Complexity
Distorts, Impedes, Delays
Communication**

**Simplicity
- Powerful
- Elegant**

**Elegant
- everything needed
- only the needed**

**sparse graph
more easily and
more quickly interpreted**

**sparse image
focuses attention**

The last slide above is a powerful demonstration of what it says.

CLUTTER SUPPRESSION

Before we FINALLY get to some graphs, let me quickly cover the easy how-to technical details of clutter suppression.

Non-communicative graphic elements that I always remove are:

- axis labels that can be in title or subtitle
- axis lines
- tick marks for tick values, especially if using grid lines
- frame around graph detail area (which is not the Border around the entire graph)

An axis of dates, times, or datetimes (unless exotically rendered in a hard-to-interpret format) NEVER needs a label.

ODS Graphics coding methods to accomplish removal are:

```
yaxis display=(nolabel); /* no axis label */
yaxis display=(noline) ; /* no axis line */
yaxis display=(noticks); /* no tick marks */

yaxis display=none;      /* no label, no line, no tick marks */
```

The above display options also apply to xaxis, rowaxis, and colaxis.

```
hbar . . . / . . . nooutline ; /* no bar outline */
```

The above also applies to the vbar statement.

The choice `display=none` might seem counterintuitive, but, if you have a bar chart with each bar-end labeled with the response value (using the simple option of `DATALABEL`), there is absolutely No Reason to include a response axis. Nevertheless, this superfluity commonly appears in bar charts.

ODS Graphics automatically encases the bars of a bar chart or the plot area of a plot in a frame. Unlike SAS/GRAPH, in the current version (9.3) of ODS Graphics there is no way to turn off the frame. Until the option is provided, if you wish to suppress the frame, you must use a customized ODS style. If you do not routinely specify an ODS style for your output, you need to know which default style is being applied by SAS to your ODS output:

```
HTML – Styles.Default (formerly)
HTML – Styles.HTMLblue (V9.3)
RTF – Styles.Rtf
PDF – Styles.Printer
LISTING – Styles.Listing (NEW)
```

If you are creating a graph on disk for later use, you need to use the ODS LISTING destination. If your base style is one of the above or one of the non-default choices supported by ODS, to be able to remove the graph work area frame, you need to first create a derived style with code like this:

```
proc template;
define style styles.YourStyleName;
  parent=styles.SomePreExistingStyle;
  class graphwalls / frameborder=off;
end;
run;
```

When creating the graph, you must apply the style like this (for a graph on disk):

```
ODS LISTING GPATH="YourTargetFolderLocation" STYLE=styles.YourStyleName;
ODS GRAPHICS ON / . . . IMAGENAME="YourDesiredFileName";
/* your SG procedure code goes here */
ODS LISTING CLOSE; ODS LISTING;
```

NOTE: It is important to always close the LISTING destination (and to reopen it as shown). If you omit the close, the graph will be written to disk, but there can be unexpected effects in the next graph created in the same session.

Communication-Effective Graphic Design Principle 6: Make Wise Use of Web Enablement

Whenever Possible, Prevent the Need for Scrolling

Please see the discussion in Appendix A on this important topic. *Graph viewers want to see The Whole Picture.*

Easy Linking of ODS-Created Web Pages (See also the note at the bottom of this page.)

When creating a table or graph, in its TITLE statement or FOOTNOTE statement use this code:

```
LINK="FileNameToLinkTo.html" "Your Description"
```

Image and Numbers

For certain kinds of graphs, the web is mandatory if you want to get at the detail numbers. A graph enables quick and easy inferences from or revelations about data. However, for reliable inferences or revelations, you also need the precise numbers, not just the picture.

As you will see, for horizontal bar charts and for pie charts, it is always possible to deliver both visuals and numbers in the same graphic image. For a simple, single-line trend chart, with not too many data points, hard (i.e., permanent) annotation of the plot points is usually possible. However, with multiple lines and/or with data points too close together, attempts to annotate suffer overlaps between labels and the plot line(s) and/or between labels and labels.

For trend lines that cannot be annotated, the simplest solution is ALT text (also known as data tips, mouseover text, floatover text, hover text, flyover text, tool tips, or pop-up text). There are several limitations of ALT text:

- You cannot inspect more than one point at a time.
- They can require dexterity to get it to pop up as desired, especially if the plot points are small. When they are too close together, you can wonder: "Which point am I looking at? Is this the point that I want?"
- They are unavailable if you print the web page (or the graph image file it presents).
-

As a side note, you should be aware that while SAS/GRAPH provides you as much absolute control over content, layout, and formatting of the box of ALT text as you are willing to code, ODS Graphics makes it very easy to get ALT text with minimal coding, but permits no control over what is delivered. ALT text is discussed again later in this paper, and several ODS Graphics examples of its use are presented.

The ability to hyperlink from a trend chart to a table (or, better, to a spreadsheet) makes the precise numbers available. If provided in a spreadsheet (which can be hyperlinked BACK to the chart), the data can then be reformatted, and/or used further in any way desired, with a tool that is universally available and familiar.

Easiest Tip to Implement: Use TITLE=

```
ODS HTML . . . BODY="YourWebPageFileName.html"(TITLE="Your web page title text");
```

If you omit TITLE=, the default is "SAS Output". The title text that you supply above is used in these ways:

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

If You Want To Keep Control of Fonts in Graph Titles and Footnotes

If you let ODS put your graph titles and footnotes in the HTML source code that imbeds your graph image file, what it will look like in your or someone else's web browser is unpredictable. The font face might not be what you expect and the size will be whatever. The HTML source in this case contains the name of the font you specified, but it might not be available on the viewing computer. The HTML source will contain an HTML font size (a digit in the range 1 to 7), selected by ODS using an algorithm that you do not control, and the web browser user has access (assuming the browser is Internet Explorer, but other browsers presumably have something similar or identical) to a choice of five different sizes to rescale the HTML font size (Largest, Larger, Medium, Smaller, Smallest). On Internet Explorer, this size scaling is accessed on the Menu Bar with View > Text Size.

To imbed your title and footnote text in your graph image file, use this: `ODS HTML ... GTITLE GFOOTNOTE;`

NOTE: If you wish to use LINK= in a title (footnote) to link the web page to another web page, and you are using ODS GRAPHICS (an SG procedure), then you must use NOGTITLE (NOGFOOTNOTE) instead.

Communication-Effective Graphic Design Principle 7: Use color communication-effectively

This deserves to be the subject of its own paper. I've written about the subject repeatedly since 1995.

See Reference 3 for my latest paper, or ask for a pre-publication version of the update that I have in progress.

See also the Note under Design Principle 4.

PIE CHARTS: A Deservedly Popular Tool for Graphic Communication

I understand that some statisticians, as well as some critics of graphic style, object to pie charts.

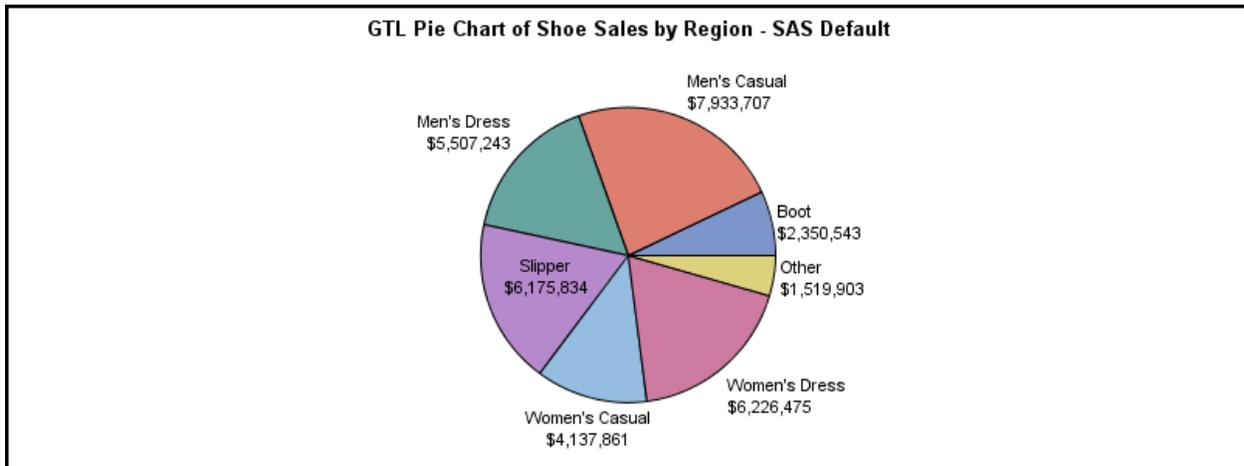
Well, it is a fact that pie charts are one of the commonest graphic delivery instruments, regardless of ideological abstention in some venues and despite misconceptions about what is possible.

**If a pie chart is designed and executed well,
there is no better way to visually compare the relative sizes of shares of the whole.**

Let's proceed to achieve this, but starting from defaults, which DO deliver sub-optimal results.

NOTE: For my most recent prior work on pie charts, but in the context of SAS/GRAPH, see Reference 5.

Unacceptable Default Pie Chart: NOT Communication-Effective for four reasons enumerated below.



1. Pie slices are ordered by label name, not by size.
2. "Other" withholds, rather than delivers, information.
3. The numeric values of percent share of the whole are not shown.
4. Some labels are outside, one label is inside.

Failing to be able to order by size makes it cumbersome to identify the relative significance of the shares of the whole. This is NOT an inherent pie chart defect. It was a choice of the SAS ODS GRAPHICS software developer.

The presence of an "Other" slice is an all too common pie chart practice. Any graph or other report should answer questions, not prompt them (like "What is IN 'Other'?") But see "Exploiting the Extremes of 'Other'" in Reference 5.

Graphs accelerate inference. Numbers are necessary for reliable inference. Pie chart slices should be accompanied by the numeric percents of the whole.

Putting labels inside the pie slices causes two problems:

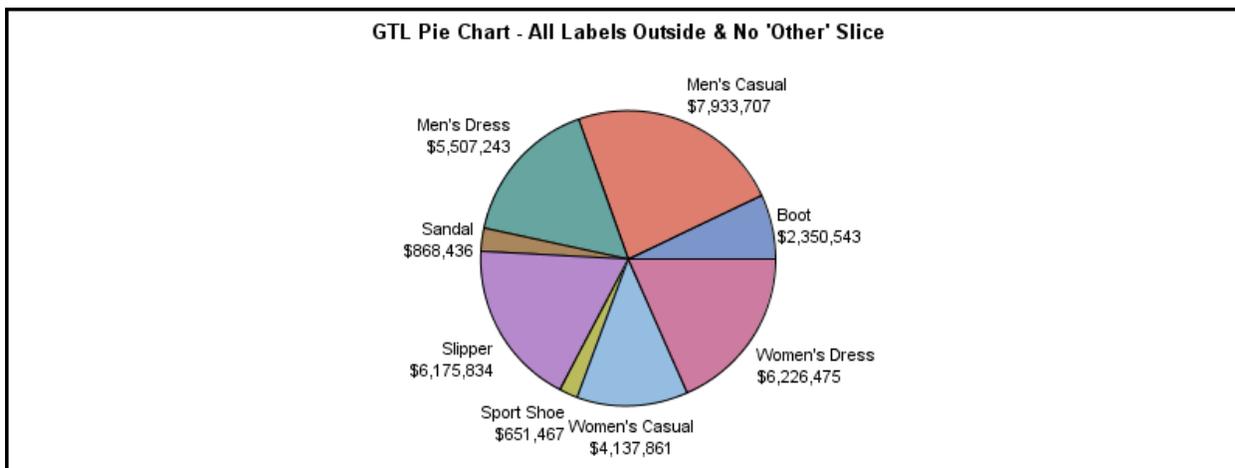
- (a) Black text on white background is maximally readable. Text on a color background is more difficult to read.
- (b) Slices are always more constricted than the white space outside the pie. This increases the difficulty of also supplying the numeric percent of the whole.

Software Problem: Instead of providing an SG procedure, or an SG procedure feature, to DIRECTLY create a pie chart, the software developers require users to get involved with GTL (Graphic Template Language), which is a less user-friendly tool.

Here is the code used to create the pie chart above:

```
proc template;
define statgraph SASDefaultPieChart;
  begingraph;
    entrytitle "GTL Pie Chart of Shoe Sales by Region - SAS Default";
    layout region;
      piechart category=Product response=Sales;
    endlayout;
  endgraph;
end; run;
ods listing gpath="D:\!PharmaSUG2013\Results";
ods graphics on / reset=all border=on height=300px width=800px
  imagename='SGRENDER_GTL_PieChart_SASDefault';
proc sgrender data=sashelp.shoes template=SASDefaultPieChart;
run;
ods listing close; ods listing;
```

Better, But Not Best, Pie Chart: Pie slices are still ordered by label name, not size, and percent of whole is missing.



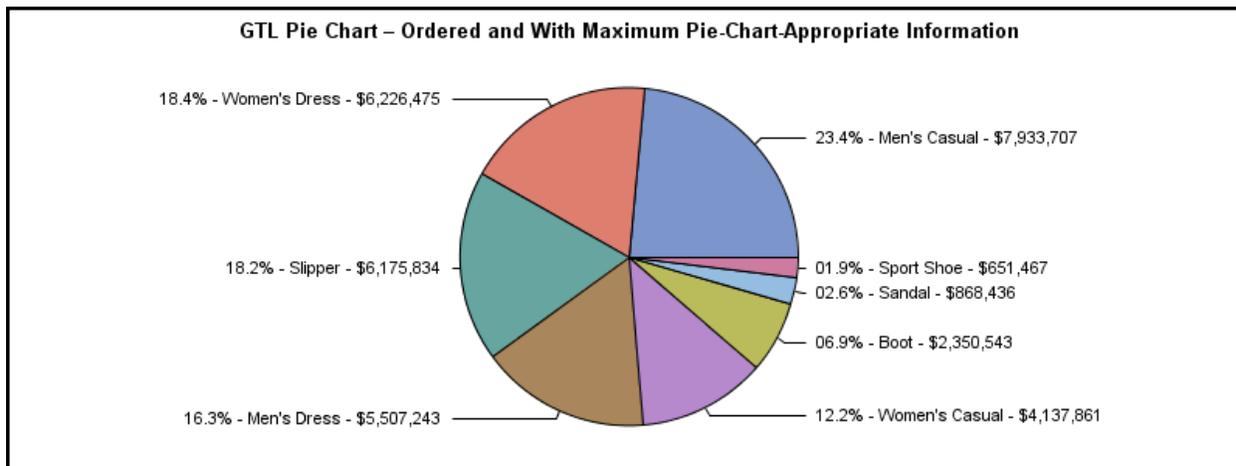
Better: All slice labels are now outside the pie, and all shoe styles are now explicitly identified—no “Other” permitted.

Here is the code, now including two added simple parameters, used to create the pie chart above:

```
proc template;
define statgraph BesslerNearDefaultPieChart;
  begingraph;
    entrytitle "GTL Pie Chart - All Labels Outside & No 'Other' Slice";
    layout region;
      piechart category=Product response=Sales /
        datalabellocation=outside
        otherslice=FALSE
      ;
    endlayout;
  endgraph;
end;
run;

ods listing gpath="D:\!PharmaSUG2013\Results";
ods graphics on / reset=all border=on height=300px width=800px
  imagename='SGRENDER_GTL_PieChart_BesslerNearDefault';
proc sgrender data=sashelp.shoes template=BesslerNearDefaultPieChart;
run;
ods listing close; ods listing;
```

Best Pie Chart: Pie slices are ordered by size, and numeric percent of whole is now included in the slice labels.



Below is the more complicated code needed to create the pie chart above. In Reference 2, a macro solution is provided to make this easier and less vulnerable to possible error.

```

proc summary data=sashelp.shoes nway;
class Product;
var Sales;
output out=ToPrep sum=TotalByClass;
run;

proc sql noprint;
select sum(TotalByClass) into :GrandTotal from ToPrep;
quit;

data ToChart;
length SliceNameWithPercentAndValue $ 256;
set ToPrep;
SliceNameWithPercentAndValue =
  trim(left(
    put(((TotalByClass / &GrandTotal) * 100),z4.1)
  )) ||
  '% - ' || trim(left(Product)) ||
  ' - ' ||
  trim(left(put(TotalByClass,dollar10.)));
run;
proc sort data=ToChart;
by Descending TotalByClass;
run;

proc template;
define statgraph BesslerBestPieChart31May2012;
  beginngraph;
  entrytitle
    "GTL Pie Chart - Ordered and With Maximum Pie-Chart-Appropriate Information";
  layout region;
  piechart category=SliceNameWithPercentAndValue
    response=TotalByClass /
    datalabelcontent=(category)
    datalabellocation=callout
    otherslice=FALSE;
  endlayout;
  endngraph;
end;
run;

```

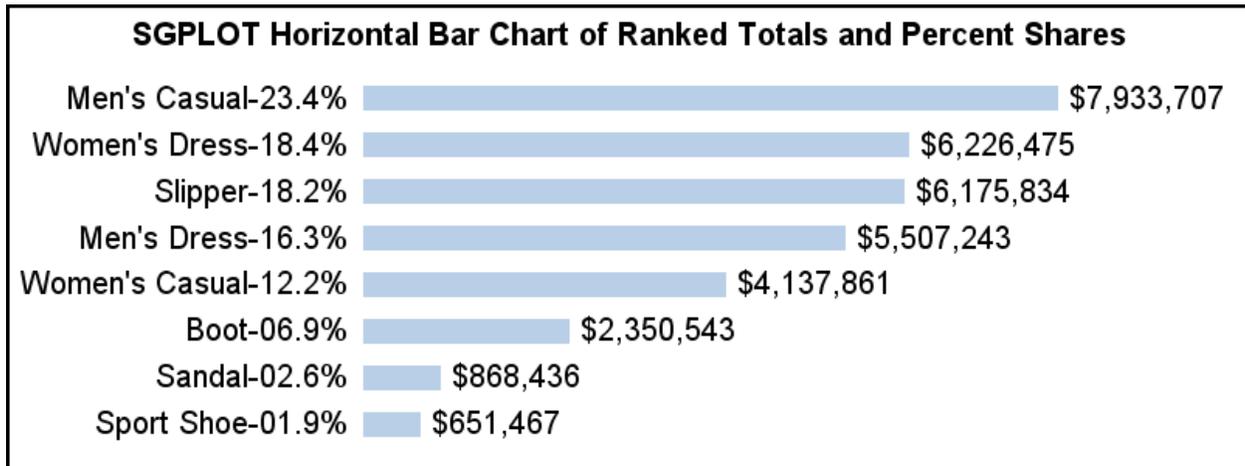
```
ods listing gpath="D:\!PharmaSUG2013\Results";

ods graphics on / reset=all
                 border=on
                 height=300px
                 width=800px
                 imagename="BesslerBestPieChartCreatedNotUsingMacro";

proc sgrender data=ToChart template=BesslerBestPieChart31May2012;
run;
ods listing close; ods listing;
```

BAR CHART OF SUMS WITH PERCENT SHARES: ALTERNATIVE FOR WHEN THE PIE CHART IS INFEASIBLE OR UNACCEPTABLE

I have long been an advocate for horizontal bar charts rather than vertical bar charts. Vertical bar charts work well only when the bar labels are short. Tilted, or worse, vertical labels for vertical bars are somewhere between inelegant and outright anti-communicative. With V9.2, the length on labels was extended to 256, which is always adequate, and, for horizontal bar charts, always useful for longer labels that are, in fact, often needed. 256 would be impractical (no space left for the bar—unless you expand the image width, which IS possible), but it's a welcome, friendly limit.



Below is the code used to create the chart above. In Reference 2, a macro solution is provided to make this easier and less vulnerable to possible error.

```
proc summary data=sashelp.shoes nway;
class Product;
var Sales;
output out=ToPrep sum=TotalByClass; run;

proc sql noprint;
select sum(TotalByClass) into :GrandTotal from ToPrep; quit;

data ToChart;
length BarNameWithPercent $ 256;
set ToPrep;
BarNameWithPercent = trim(left(Product)) ||
  '-' || trim(left(put(((TotalByClass / &GrandTotal) * 100),z4.1))) || '%'; run;

proc template;
define style styles.ListingWithNoFrame; /* remove a useless box around the bars */
  parent=styles.Listing;
  class graphwalls / frameborder=off;
end; run;

ods listing gpath="D:\!PharmaSUG2013\Results" style=styles.ListingWithNoFrame;
ods graphics on / reset=all border=on height=300px width=800px
  imagename="SGPLOTHorizontalBarChartOfRankedTotalsAndPercentShares";

title height=16pt "SGPLOT Horizontal Bar Chart of Ranked Totals and Percent Shares";
proc sgplot data=ToChart;
hbar BarNameWithPercent / response=TotalByClass categoryorder=respdesc
  datalabel datalabelattrs=(size=16pt) barwidth=0.5 nooutline;
yaxis display=(nolabel noline noticks) valueattrs=(size=16pt);
xaxis display=none;
run;
ods listing close; ods listing;
```

BAR CHART OF SUMS WITH TWO LEVELS OF CATEGORIZATION

The SASHELP.SHOES data set classifies sales not only by Product, but also by Region.

Here is the code used to handle that situation, and to easily create the chart on the following page:

```
ods listing gpath="D:\!PharmaSUG2013\Results";

ods graphics on / reset=all border=on height=1600px width=1155px
                 imagename='SGPANELhbar';

title1 height=16pt "SGPANEL Sales by Product within Region in SASHELP.SHOES";
title2 height=16pt
      "Easy To Create, but no control available for size of panel cell header text,";
title3 height=16pt
      "and the shared row header for columns makes CATEGORYORDER unusable.";
proc sgpanel data=sashelp.shoes;
panelby region / novarname rows=5 columns=2;
rowaxis display=(nolabel noline noticks) valueattrs=(size=12pt);
colaxis display=none;
hbar Product / response=Sales barwidth=0.5 nooutline
      datalabel datalabelattrs=(size=12pt);
run; quit; title;
ods listing close; ods listing;
```

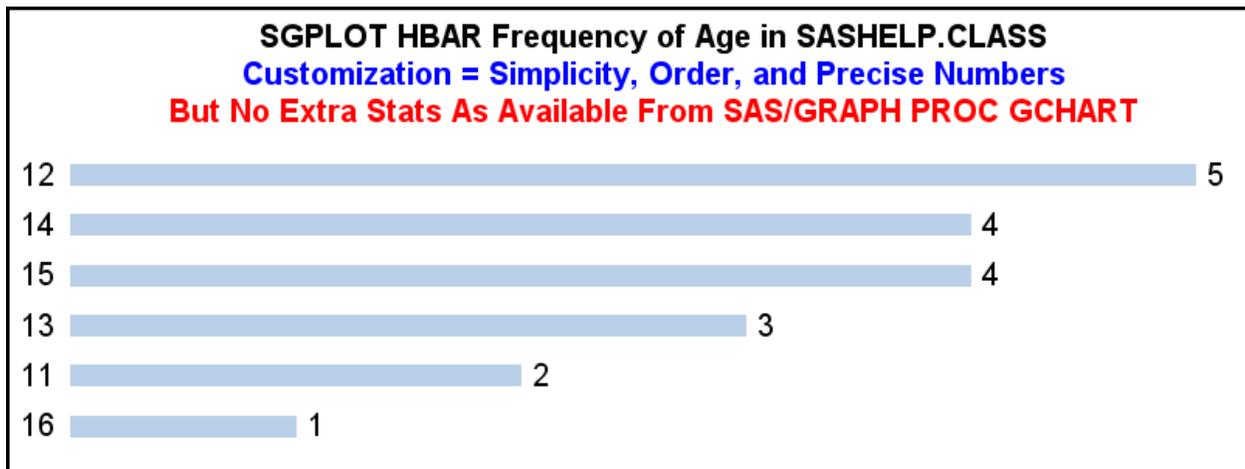
The main disadvantage of this graphic solution is that it is unable to order the bars within each region based upon the Sales By Product. This is due to the fact that the bar labels are provided only at the left margin, rather than for each cell of the panel.

Also, the ability, if it were available in PROC SG PANEL, to remove the line that separates the panel cell header from the graphic content of the cell would make it unambiguous as to whether the descriptor pertains to the graph below or the graph above.

SGPANEL Sales by Product within Region in SASHELP.SHOES
Easy To Create, but no control available for size of panel cell header text,
and the shared row header for columns makes CATEGORYORDER unusable.

	Africa	Asia
Boot	\$119,835	\$62,708
Men's Casual	\$562,794	\$11,754
Men's Dress	\$318,500	\$119,366
Sandal	\$190,409	\$8,208
Slipper	\$337,076	\$152,032
Sport Shoe	\$22,150	\$2,092
Women's Casual	\$417,516	\$25,837
Women's Dress	\$374,308	\$78,234
	Canada	Central America/Caribbean
Boot	\$385,613	\$190,743
Men's Casual	\$441,903	\$756,513
Men's Dress	\$920,101	\$404,895
Sandal	\$14,798	\$378,382
Slipper	\$952,751	\$883,181
Sport Shoe	\$140,389	\$26,964
Women's Casual	\$410,807	\$399,357
Women's Dress	\$989,350	\$617,718
	Eastern Europe	Middle East
Boot	\$306,785	\$171,282
Men's Casual	\$576,396	\$2,058,254
Men's Dress	\$335,761	\$839,571
Sandal	\$3,716	\$35,186
Slipper	\$509,698	\$662,480
Sport Shoe	\$91,202	\$4,007
Women's Casual	\$209,256	\$748,792
Women's Dress	\$362,126	\$1,112,207
	Pacific	South America
Boot	\$123,575	\$245,675
Men's Casual	\$662,368	\$544,950
Men's Dress	\$426,191	\$425,669
Sandal	\$48,424	\$165,925
Slipper	\$390,740	\$462,651
Sport Shoe	\$26,169	\$33,061
Women's Casual	\$219,886	\$179,227
Women's Dress	\$399,441	\$377,625
	United States	Western Europe
Boot	\$448,296	\$296,031
Men's Casual	\$1,372,527	\$946,248
Men's Dress	\$969,271	\$747,918
Sandal	\$12,039	\$11,349
Slipper	\$967,927	\$857,298
Sport Shoe	\$104,403	\$201,030
Women's Casual	\$541,536	\$985,647
Women's Dress	\$1,087,987	\$827,479

BAR CHART OF FREQUENCY DISTRIBUTION



```
proc template;
define style styles.ListingWithNoFrame; /* remove a useless box around the bars */
  parent=styles.Listing;
  class graphwalls / frameborder=off;
end; run;

ods listing gpath="D:\!PharmaSUG2013\Results" style=styles.ListingWithNoFrame;
ods graphics on / reset=all border=on height=300px width=800px
  imagename='SGPLOT hbar Freq Chart_Bessler_Customization';
title1 height=16pt "SGPLOT HBAR Frequency of Age in SASHELP.CLASS";
title2 height=16pt
  color=blue "Customization = Simplicity, Order, and Precise Numbers";
title3 height=16pt
  color=red "But No Extra Stats As Available From SAS/GRAPH PROC GCHART";
proc sgplot data=sashelp.class;
yaxis display=(nolabel noline noticks) valueattrs=(size=16pt); xaxis display=none;
hbar Age / stat=freq categoryorder=respdesc datalabel datalabelATTRS=(size=16pt)
  barwidth=0.4 nooutline; run;
ods listing close; ods listing;
```

With PROC SGPLOT, it is advantageous to have the counts automatically appended to the bar end without having to do extra programming to annotate the bars. In SAS/GRAPH PROC GCHART, it is easy to get a listing of the values for all of the bars in a column at the right margin, but with a large number of bars, especially when some of them are short, it becomes ambiguous as to which value belongs to which bar. In that case, the extra effort of annotating the bar ends is necessary, and the column is suppressed. PROC GCHART actually provides three other additional statistics at the right margin: Percent of Total Frequencies, Cumulative Frequency, and Cumulative Percent. You can turn off whatever you do not wish to present. For a large number of bars, doing the needed preprocessing of the data to compute them and providing all of the four statistics via annotation is possible, but probably not desirable. Instead, after the same preprocessing, all of the statistics could be imbedded in expanded bar labels as demonstrated in the Subsetted and Ranked Horizontal Bar Chart later in this paper.

TIME PLOTS, TIME SERIES GRAPHS, TREND LINES, LINE CHARTS, etc.

In SAS/GRAPH, the GPLOT procedure can visually present the evolution of data values over time. In ODS GRAPHICS, the SG procedures provide many more ways.

In References 6 and 7, I compared a variety of ways to visually present time series data using the old and new technology. In Reference 8, in collaboration with Alexandra Riley, I focused on three cases from Reference 6 and provided all of the code. Reference 7 is a comparison of a variety of graph types, not just time series.

One of the tools omitted in those prior works is the SERIES plot, which is the focus here. I use SERIES plots both with PROC SGPLOT and PROC SGPANEL.

These examples use web-deployed graphs so that ALT text can be provided, rather than forcing the viewer to guess y and x values based on tick mark values at the axes. Another solution for time series, first developed with SAS/GRAPH in Reference 4 and adapted to ODS GRAPHICS later in this paper, is to link the plot forwards and backwards with a spreadsheet.

NOTE 1: Most of the illustrations in this section are inserted PNG files, not screen prints of the web graphs from a web browser window. However, all code does create the web-enabling HTML for the graphs.

NOTE 2: These examples require that the code in Appendix B be run first to prepare the input data.

In SAS/GRAPH PROC GPLOT, DESCRIPTION=' ' can be used on the PLOT statement in order to prevent the display of the default pop-up text that tries to describe the graph when you rest the mouse anywhere in the graph area on the web page. Or you can use it to provide a custom description to replace the default. For Statistical Graphics procedures, the default description can neither be nullified nor customized. These pop-up descriptions can be a nuisance when you want to instead display the ALT text for the plot points.

The ALT text for PROC GPLOT is provided via the PLOT statement's HTML parameter, which identifies a variable on the plot input data set. The variable can be customized with anything the designer/programmer desires, including line breaks, any text, the plot point values in any format to any precision, and times, datetimes, or dates in any format that you prefer (e.g., dates formatted with day of the week name, as well as month, day, year). For Statistical Graphics procedures, ALT text is triggered by IMAGEMAP=ON in an ODS GRAPHICS statement, as in this section. The text displayed is simply a list of the form Variable Name (or Label) = Value, and cannot be customized.

For these graphs, the data used is SASHELP.CITIDAY, which is shipped with SAS/ETS. It contains daily history for financial market and other economic information. If you don't have SAS/ETS, you can ask SAS Technical Support how to download the data set. The data used here is limited to year 1990.

Since the best way to determine the precise value of the Dow index in these web-enabled plots is from the pop-up ALT text, in these graphs I display along the y axis only the minimum and maximum values for the Dow index.

INTRODUCTION TO THE TIME SERIES EXAMPLES: DISCUSSION OF STATEMENTS AND PARAMETERS

Statements Common to All of the Time Series Charts

```
ods graphics on /
  reset=all      /* Suppress any leftovers from prior code runs */
  border=off     /* Not usually desired on a web page
                  but turn on if wanting to see how much extra space
                  might be available in the browser window
                  for a wider image */
  antialiasmax=2500 /* antialiasing is on by default,
                       but antialiasmax is defaulted to 600.
                       If that is insufficient,
                       then antialiasing will be incomplete */
  tipmax=2500 /* This is the maximum number of distinct mouse-over areas allowed
                before data tips are disabled. Default is 400. */
  imagemap=on /* This turns on the data tips.
                SAS/GRAPH allows customization of the data tips,
                but ODS GRAPHICS does not. */
  imagename='DesiredFileNameForThePNGImageFile';

yaxis display=(nolabel) /* the yaxis variable is identified in the plot title */
  values=(&Ymin_1990 &Ymax_1990); /* show only the yaxis minimum and maximum,
  available as symbolic (or macro) variables from setup processing. */

xaxis display=(nolabel) /* the xaxis variable is identified in the plot title */
  grid /* provide reference lines for each xaxis tick mark value */
  . . . /* here specify a list of values or (see below) Interval= */
```

Example Assignments for Parameters Added to ODS GRAPHICS Statement for Some Series Charts

```
width=800px /* width of image in pixels, default is 640 */
height=600px /* height of image in pixels, default is 480 */
```

If your target is HTML (web page), your choice of dimensions should rarely, if ever, exceed the dimensions of the available live space in the browser and monitor likely to be used. Live Space is the viewable display area WITHOUT scrolling. The usability of a graph is diminished if it needs to be scrolled.

Typical (but not All) Parameters Used on SERIES Statement in This Paper

```
series y=YourYAXISvar x=YourXAXISvar /
  markers /* turns on plot-point markers
           default is nomarkers */
  markerattrs=(size=7
               symbol=circlefilled
               color=red /* But DO NOT specify a color
                           if using GROUP= on the SERIES statement
                           to produce a multi-line overlay */
               )
  lineattrs=(thickness=N /* where, in this paper, N is 2 or 3.
                          3 is used when chart has multiple lines
                          (as in the twelve months of 1990 overlaid)
                          so that line colors are more easily distinguished */
             pattern=solid /* If you do not specify your pattern preference,
                             the software might make a decision for you,
                             which you might not like. */
             color=blue /* But DO NOT specify a color
                           if using GROUP= on the SERIES statement
                           to produce a multi-line overlay */
             );
```

Parameters of Special Interest on the XAXIS Statement

For Date, Time, or DateTime Variables:

Interval= can have numerous different values.
Used here are **MONTH**, **SEMIMONTH**, and **WEEK**.

In some graphs here, the XAXIS variable is Day Of Month, which is not a Date.
There Interval= does not apply.

For Any Variable:

FitPolicy= defaults to **THIN**, which means that an ODS GRAPHICS SG procedure will OMIT any tick mark values that it cannot fit (based on its estimates, which CAN BE wrong). Also, it does this omission with no message in the SAS log, even if you have explicitly specified a list of tick mark values to be displayed with the **VALUES=** parameter or have certain expectations based on what you might have specified with the **INTERVAL=** parameter.

When the displayed tick mark values are fewer than what you would like, there are five options:

1. increase the width of the image if the xaxis has the problem
2. specify **FitPolicy=Stagger**
3. specify **FitPolicy=StaggerRotate** (first try Stagger, if no help, use Rotate)
4. specify **FitPolicy=RotateStagger** (first try Rotate, if no help, use Stagger)
5. specify **FitPolicy=Rotate** (least desirable)

NOTE: When a yaxis set of values is thinned, if that causes a problem, your only option is to increase the height of the image. **If the yaxis variable has discrete character values, thinning is absolutely unacceptable.**

TIME SERIES EXAMPLES: ALL USING THE SERIES STATEMENT, WITH PROC SG PANEL OR PROC SG PLOT

All examples use Dow Index data for one year of trading days in 1990

One of these uses PROC SG PANEL which creates an array of subplots for segments of the time period. In the case presented here, it is an array of the twelve months of year 1990.

The remaining examples all use PROC SG PLOT.

If GROUP=MONTH is added to the SERIES statement, the twelve months are plotted as separate lines in an overlay.

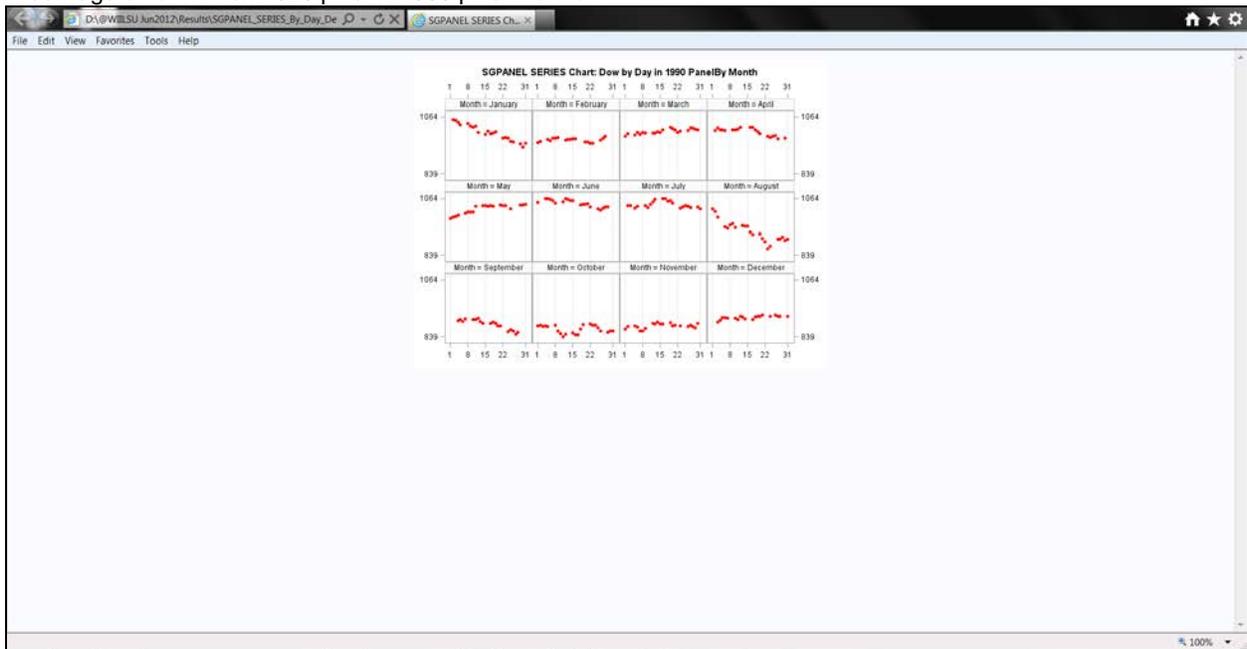
Without GROUP=MONTH, the data is plotted as one line.

The last example in this section demonstrates a plot and spreadsheet hyperlinked to each other forwards and backwards. ALT text (aka "data tips") on the web-deployed plot DOES make precise numbers available for temporary look-up, but the spreadsheet makes them available both for inspection as a whole, and for reuse with Excel however the user of the deliverable sees fit to explore or further manipulate the data.

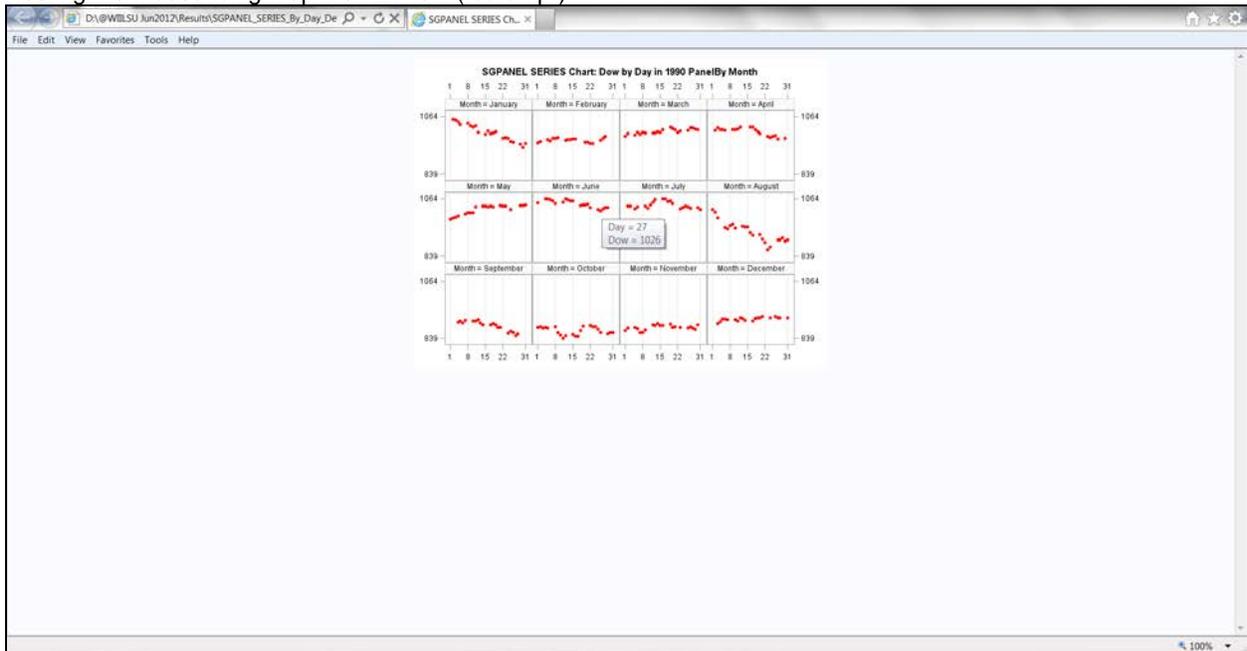
For non-web-enabled plots, the section after this one shows how to imbed the precise y values inside a plot image area by using SG PLOT with a VLINE statement instead of the SERIES statement, which is the focus of this section. The y values can be imbedded as either a table between the plot lines and x axis or as direct plot-point annotation.

SGPANEL SERIES Plot By Day (default size) Panelled By Month

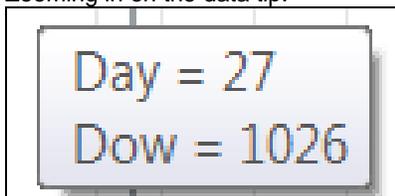
Web Page on a 17-inch 1920 pixel X 1090 pixel Monitor:



Resting the mouse to light up the ALT text ("data tip")

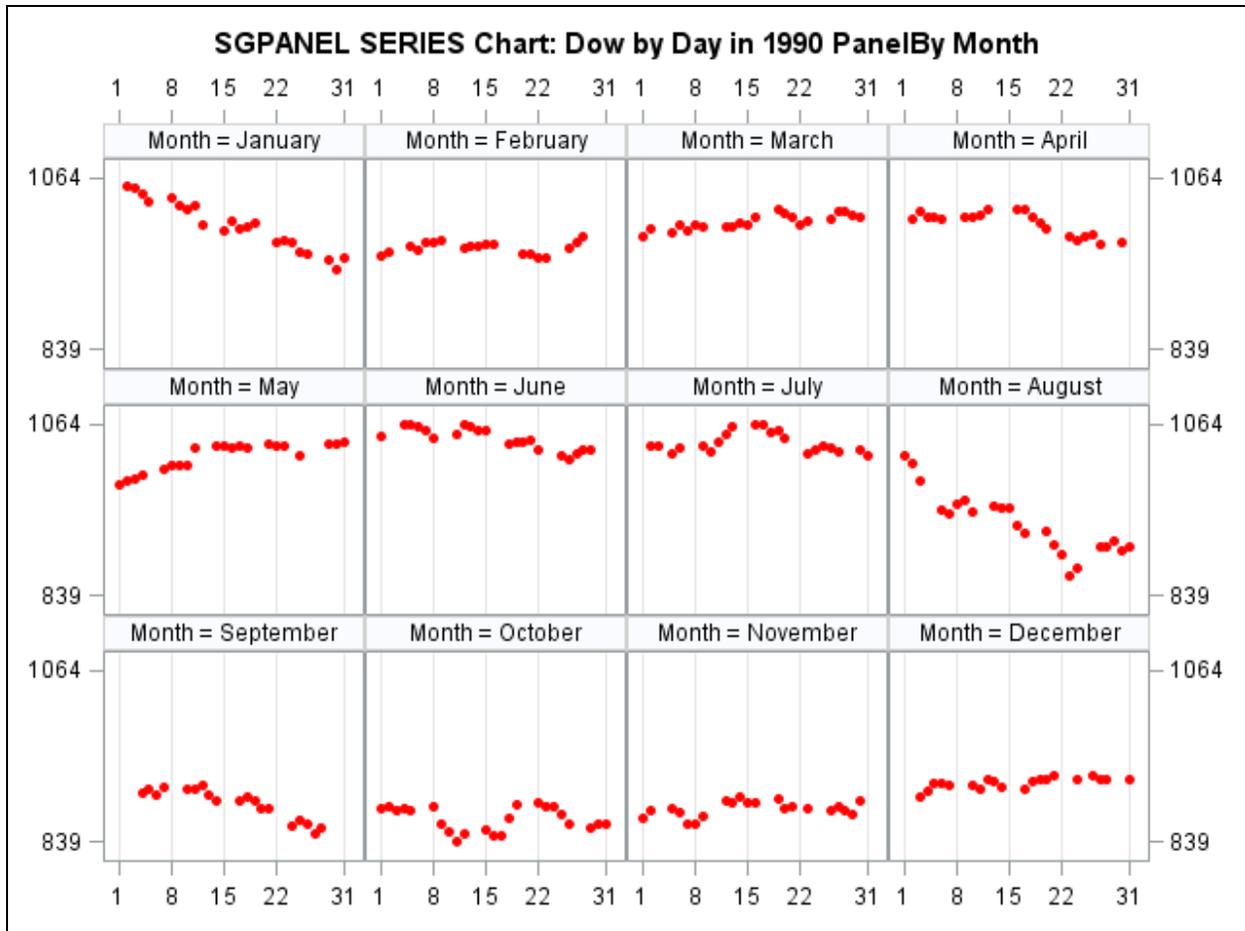


Zooming in on the data tip:



The Image File At Actual Size

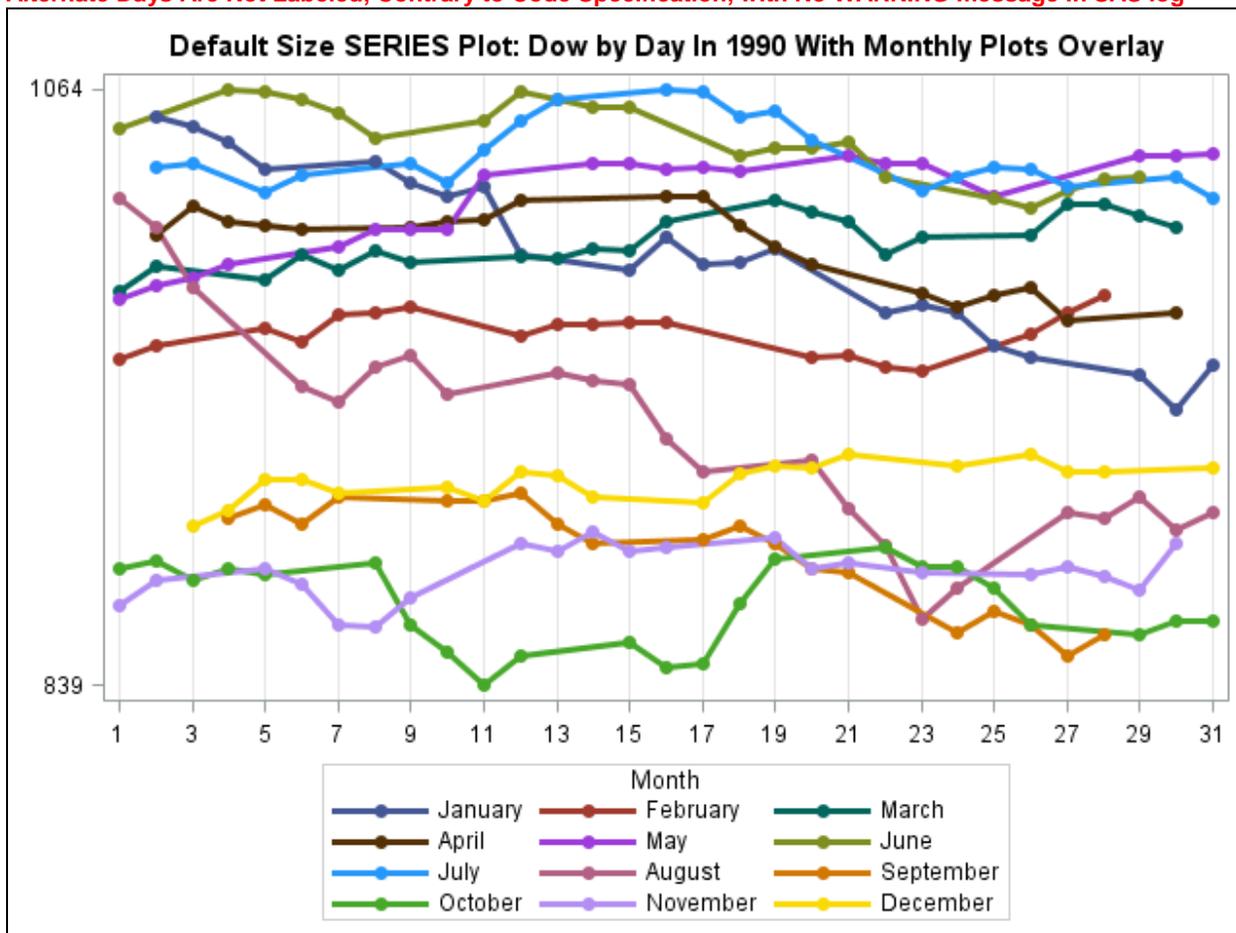
In this context, it is easy to reliably associate the Month=*MonthName* labels with the appropriate subplot. However, in a very complex array, a viewer unfamiliar with this format might be uncertain as to whether a label goes with the subplot above or the subplot below. A better frame structure would clearly associate each label with its subplot.



Below is the code to create this web-deployed graph. See predecessor processing in Appendix B.

```
ods noresults; ods listing close;
ods graphics on / reset=all border=off antialiasmax=2500 tipmax=2500
  imagemap=on imagename='SGPANEL_SERIES_By_Day_DefaultSize_PanelBy_Month';
ods html path="D:\!PharmaSUG2013\Results"
  (url=none) /* make the combination of HTML file and PNG image file portable */
  body='SGPANEL_SERIES_By_Day_DefaultSize_PanelBy_Month.html'
  (title='SGPANEL SERIES Chart Dow by Day in 1990 PanelBy Month');
title 'SGPANEL SERIES Chart: Dow by Day in 1990 PanelBy Month';
proc sgpanel data=work.DowByDayIn1990;
  panelby Month / columns=4 rows=3;
  series y=Dow x=Day / markers markerattrs=(size=7 symbol=circlefilled color=red)
    lineattrs=(color=white); /* hide the line which adds no value */
  rowaxis display=(nolabel) refticks=(values) values=(&Ymin_1990 &Ymax_1990);
  colaxis display=(nolabel) refticks=(values) values=(1 8 15 22 31) grid;
format Month MonthNm.;
format Dow 5. Day 2.;
run;
ods html close; ods listing;
```

Alternate Days Are Not Labeled, Contrary to Code Specification, with No WARNING message in SAS log

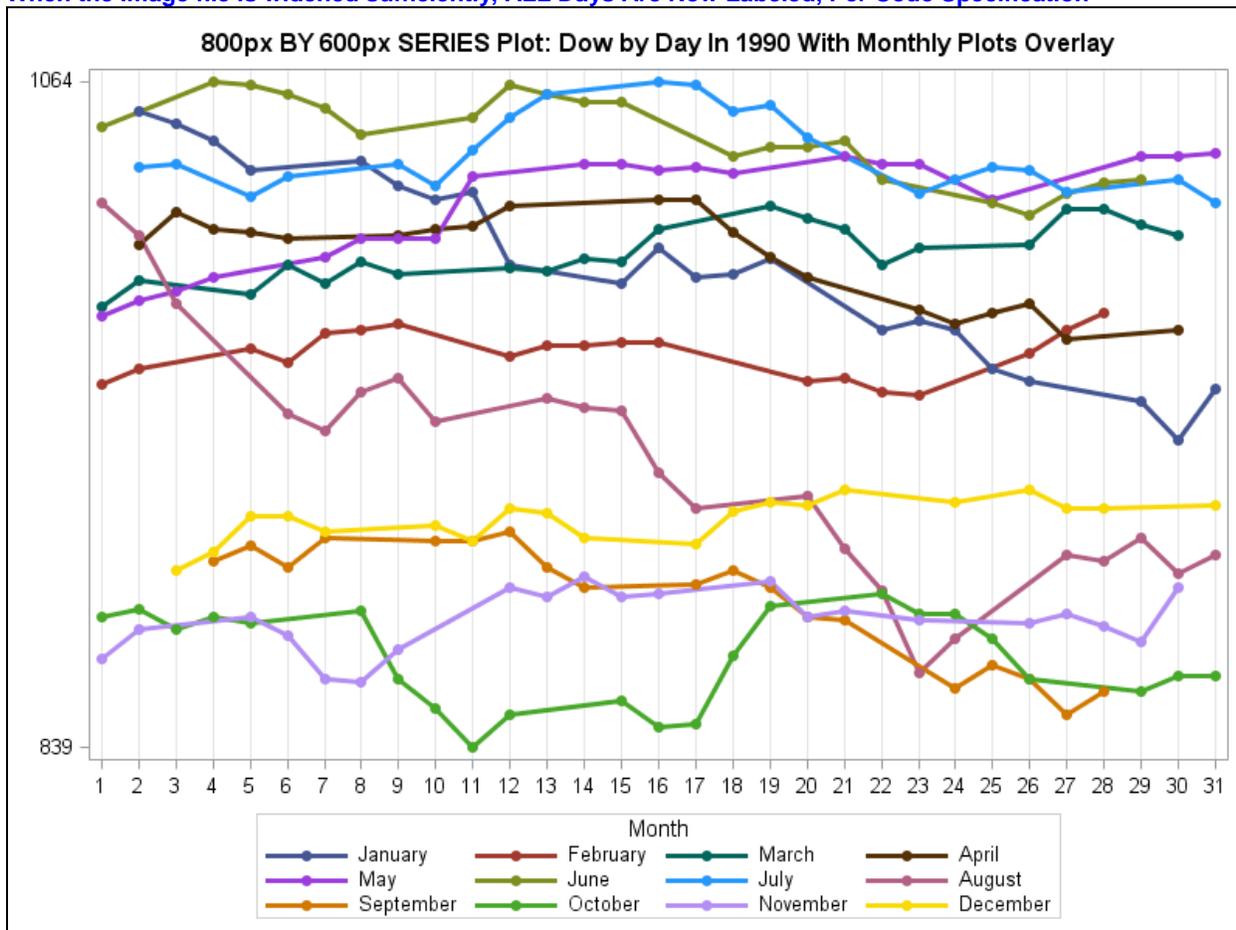


Below is the code to create this web-deployed graph.

```
ods noresults; ods listing close;
ods graphics on / reset=all border=off antialiasmax=2500 tipmax=2500
  imagemap=on imagename='SGPLOT_SERIES_By_Day_With_Months_Overlaid_DefaultSize';
ods html path="D:\!PharmaSUG2013\Results" (url=none)
  body='SGPLOT_SERIES_By_Day_With_Months_Overlaid_DefaultSize.html'
  (title='Default Size SERIES Plot - Dow by Day In 1990 With Monthly Plots
Overlay');
title 'Default Size SERIES Plot: Dow by Day In 1990 With Monthly Plots Overlay';
proc sgplot data=work.DowByDayIn1990;
  series y=Dow x=Day /
    group=Month /* creates the overlay of monthly lines */
    markers markerattrs=(size=7 symbol=circlefilled)
    lineattrs=(thickness=3 /* thicken the lines for color distinguishability */
              pattern=solid);
  yaxis display=(nolabel) values=(&Ymin_1990 &Ymax_1990);
  xaxis display=(nolabel)
    values=(1 to 31 by 1) /* NOT DELIVERED IN THE RESULT */
  grid;
format Month MonthNm.;
format Dow 5. Day 2.;
run;
ods html close; ods listing;
```

Image File (**Shrunk by Microsoft Word To Fit Page Width**) for SGPANEL SERIES Plot By Day (custom 800px X 600px) With Monthly Lines Overlaid

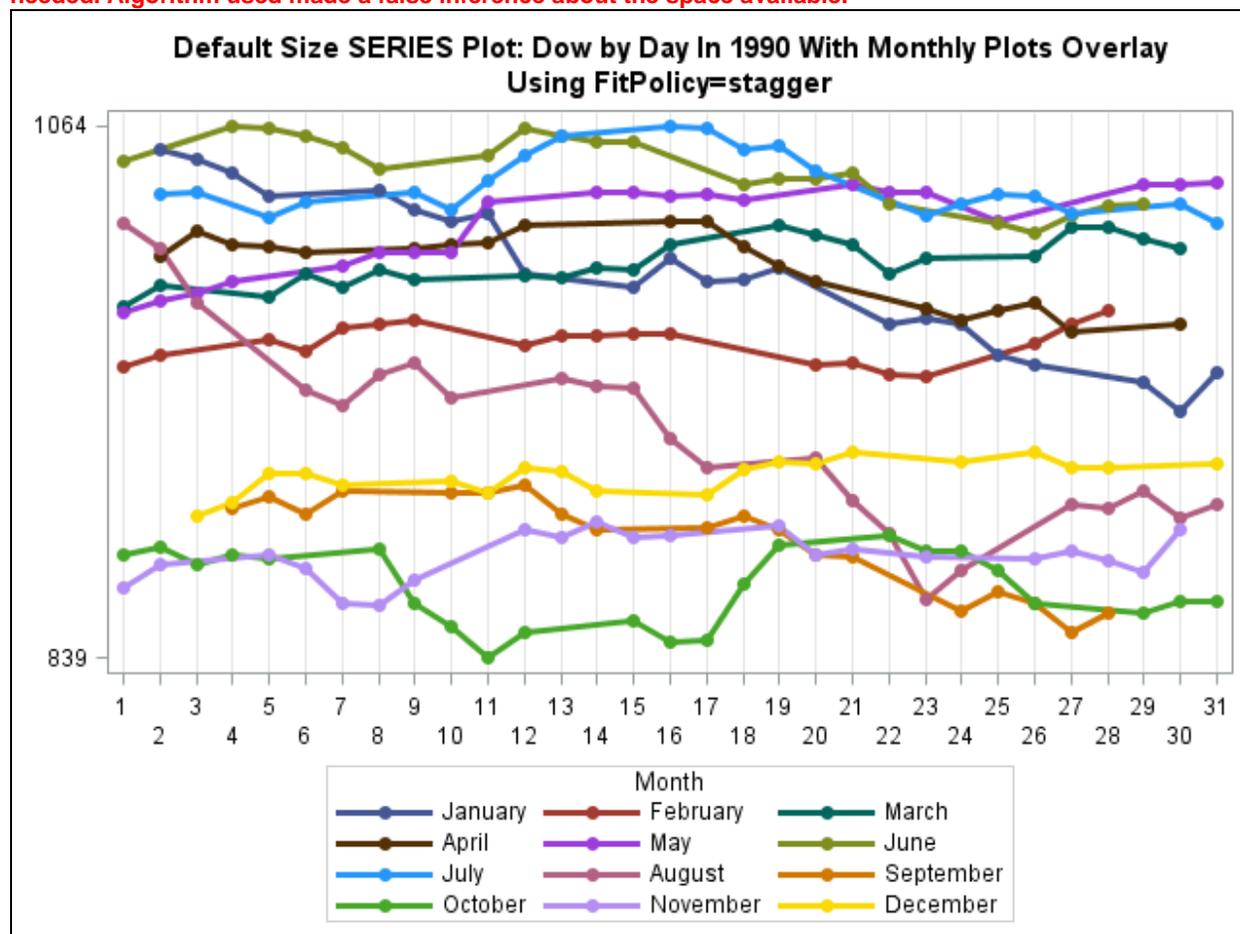
When the image file is widened sufficiently, ALL Days Are Now Labeled, Per Code Specification



```
ods noresults; ods listing close;
ods graphics on / reset=all border=off antialiasmax=2500 tipmax=2500
width=800px height=600px /* override default size 640px X 480px */
imagemap=on imagename='SGPLOT_SERIES_By_Day_With_Months_Overlaid_800pxBY600px';
ods html path="D:\!PharmaSUG2013\Results" (url=None)
body='SGPLOT_SERIES_By_Day_With_Months_Overlaid_800pxBY600px.html'
(title='800px BY 600px SERIES Plot - Dow by Day In 1990 With Monthly Plots
Overlay');
title '800px BY 600px SERIES Plot: Dow by Day In 1990 With Monthly Plots Overlay';
proc sgplot data=work.DowByDayIn1990;
series y=Dow x=Day / group=Month markers markerattrs=(size=7 symbol=circlefilled)
lineattrs=(thickness=3 pattern=solid);
yaxis display=(nolabel) values=(&Ymin_1990 &Ymax_1990);
xaxis display=(nolabel) values=(1 to 31 by 1) grid;
format Month MonthNm.;
format Dow 5. Day 2.;
run;
ods html close; ods listing;
```

Image File (**Actual Size:** default 640px X 480px) for SG PANE L SERIES Plot By Day With Monthly Lines Overlaid

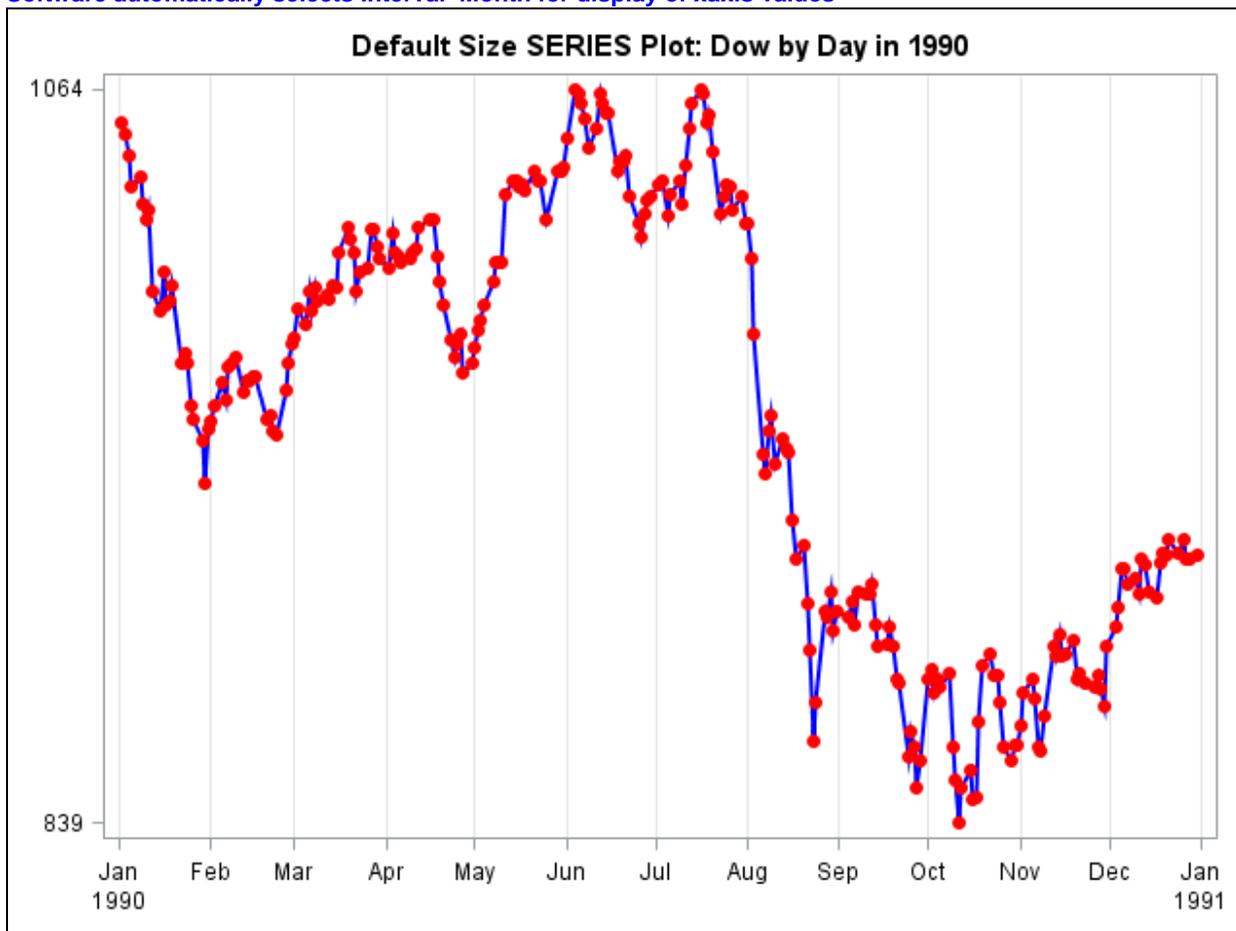
Even for the default width image file, ALL Days Can Be Labeled, But With Alternate Day Values Staggered On Two Lines, Per FitPolicy=Stagger. Therefore, trimming at default width, without Stagger, obviously was NOT needed. Algorithm used made a false inference about the space available.



```
ods noresults; ods listing close;
ods graphics on / reset=all border=off antialiasmax=2500 tipmax=2500
  imagemap=on
imagenname='SGPLOT_SERIES_By_Day_With_Months_Overlaid_DefaultSize_FitPolicyEqStagger';
ods html path="D:\!PharmaSUG2013\Results" (url=None)
  body='SGPLOT_SERIES_By_Day_With_Months_Overlaid_DefaultSize_FitPolicyEqStagger.html'
  (title='Default Size SERIES Plot FitPolicy=stagger - Dow by Day In 1990 With Monthly
Plots Overlay Using FitPolicy=stagger');
title height=11pt
  'Default Size SERIES Plot: Dow by Day In 1990 With Monthly Plots Overlay';
title2 height=11pt 'Using FitPolicy=stagger';
proc sgplot data=work.DowByDayIn1990;
  series y=Dow x=Day / group=Month
    markers markerattrs=(size=7 symbol=circlefilled)
    lineattrs=(thickness=3 pattern=solid);
  yaxis display=(nolabel) values=(&Ymin_1990 &Ymax_1990);
  xaxis display=(nolabel) values=(1 to 31 by 1) grid
    FitPolicy=stagger; /* stagger alternate xaxis values on two lines*/
format Month MonthNm.;
format Dow 5. Day 2.;
run;
ods html close; ods listing;
```

Image File (**Actual Size:** default 640px X 480px) for SGPLOT SERIES Plot By Day

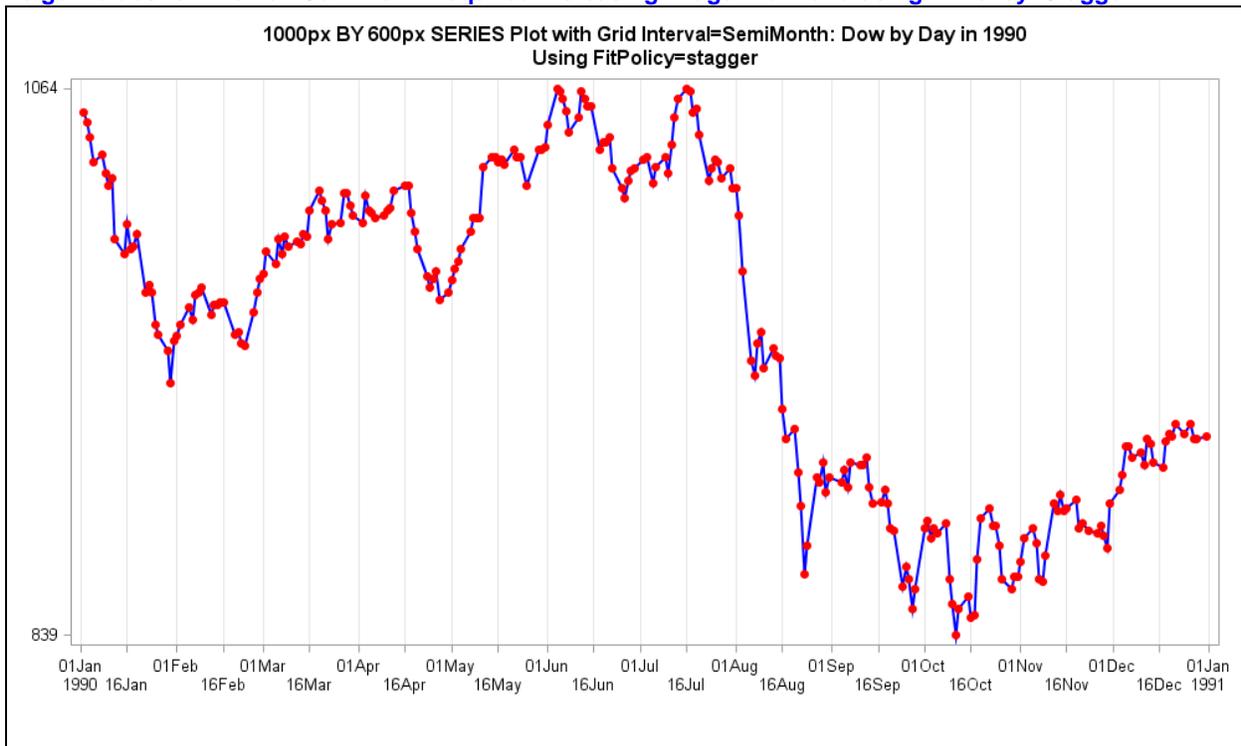
Software automatically selects Interval=Month for display of axis values



```
ods noresults; ods listing close;
ods graphics on / reset=all border=off antialiasmax=2500 tipmax=2500
  imagemap=on imagename='SGPLOT_SERIES_By_Day_DefaultSize';
ods html path="D:\!PharmaSUG2013\Results" (url=none)
  body='SGPLOT_SERIES_By_Day_DefaultSize.html'
  (title='Default Size SERIES Plot - Dow by Day in 1990');
title 'Default Size SERIES Plot: Dow by Day in 1990';
proc sgplot data=work.DowByDayIn1990;
  series y=Dow x=Date /
    markers markerattrs=(size=7 symbol=circlefilled color=red)
    lineattrs=(thickness=2 pattern=solid color=blue);
  yaxis display=(nolabel) values=(&Ymin_1990 &Ymax_1990);
  xaxis display=(nolabel) grid;
format Dow 5.;
run;
ods html close; ods listing;
```

Image File (**Shrunk To Fit Page Width**) for SGPLOT SERIES Plot By Day (Custom Size 1000px X 600px)

To get values for Interval=SemiMonth required increasing image width and using FitPolicy=Stagger



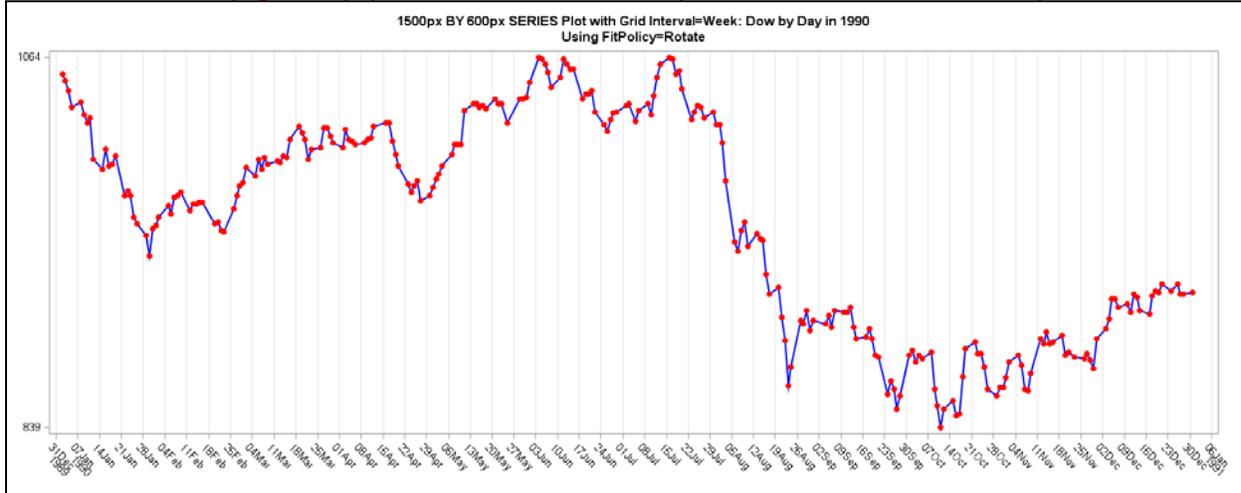
```
ods noresults; ods listing close;
ods graphics on / reset=all border=off antialiasmax=2500 tipmax=2500
width=1000px height=600px
imagemap=on
imagenname='SGPLOT_SERIES_By_Day_1000pxBY600px_IntervalEqualSemiMonth_FitPolicyEqualStagger';
ods html path="D:\!PharmaSUG2013\Results" (url=none)

body='SGPLOT_SERIES_By_Day_1000pxBY600px_IntervalEqualSemiMonth_FitPolicyEqualStagger.html'
(title='1000px BY 600px SERIES Plot with Grid Interval=SemiMonth - Dow by Day in 1990 Using FitPolicy=stagger');
title height=11pt '1000px BY 600px SERIES Plot with Grid Interval=SemiMonth: Dow by Day in 1990';
title2 height=11pt 'Using FitPolicy=stagger';
proc sgplot data=work.DowByDayIn1990;
series y=Dow x=Date /
markers markerattrs=(size=7 symbol=circlefilled color=red)
lineattrs=(thickness=2 pattern=solid color=blue);
yaxis display=(nolabel) values=(&Ymin_1990 &Ymax_1990);
xaxis display=(nolabel) grid interval=semimonth FitPolicy=stagger;
format Dow 5.;
run;
ods html close; ods listing;
```

Image File (**Shrunk To Fit Page Width**) for SGPLOT SERIES Plot By Day (Custom Size 1500px X 600px)

To get values for Interval=Week required increasing image width and using FitPolicy=Rotate

At narrower widths (e.g., 1400px) the first & second values (and the last & second from last) values collided

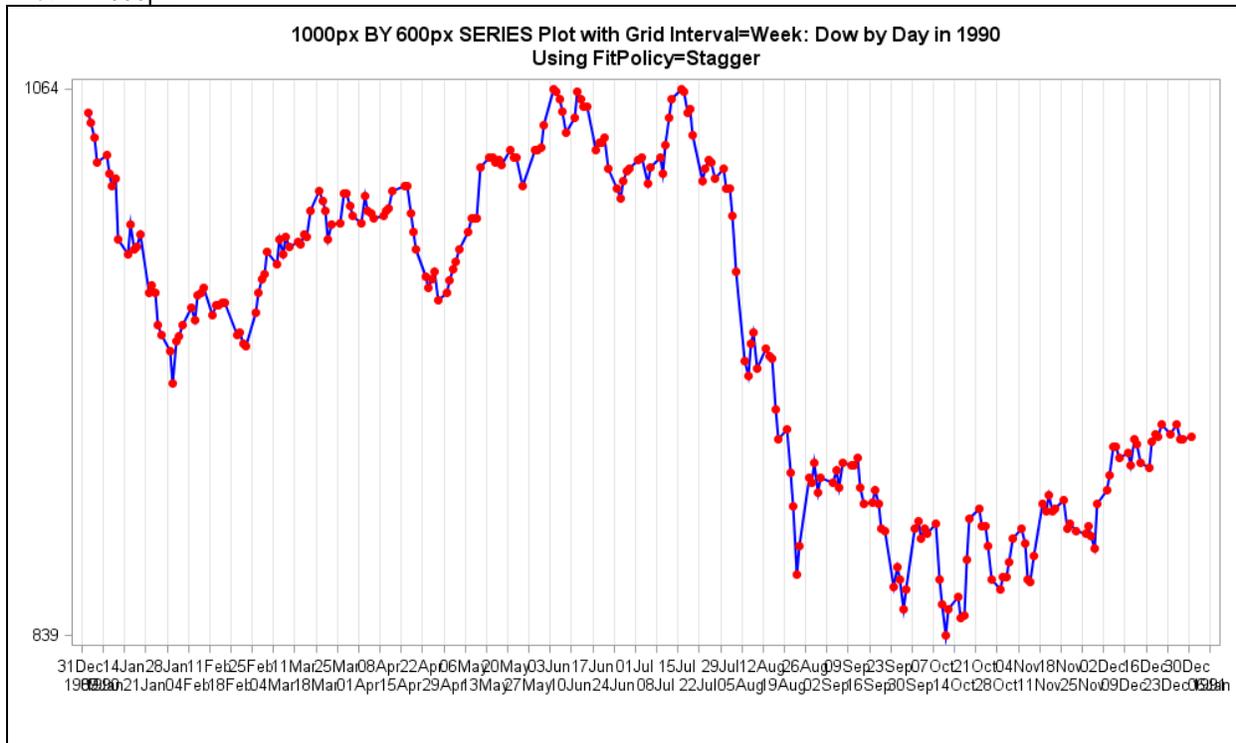


```
ods noresults; ods listing close;
ods graphics on / reset=all border=off antialiasmax=2500 tipmax=2500
  width=1500px height=600px
  imagemap=on imagename=
    'SGPLOT_SERIES_By_Day_1500pxBY600px_IntervalEqualWeek_FitPolicyEqualRotate';
ods html path="D:\!PharmaSUG2013\Results" (url=none)
  body='SGPLOT_SERIES_By_Day_1500pxBY600px_IntervalEqualWeek_FitPolicyEqualRotate.html'
  (title='1500px BY 600px SERIES Plot with Grid Interval=Week - Dow by Day in 1990');
title height=11pt
  '1500px BY 600px SERIES Plot with Grid Interval=Week: Dow by Day in 1990';
title2 height=11pt 'Using FitPolicy=Rotate';
proc sgplot data=work.DowByDayIn1990;
  series y=Dow x=Date /
    markers markerattrs=(size=7 symbol=circlefilled color=red)
    lineattrs=(thickness=2 pattern=solid color=blue);
  yaxis display=(nolabel) values=(&Ymin_1990 &Ymax_1990);
  xaxis display=(nolabel) grid interval=week FitPolicy=Rotate;
format Dow 5.;
run;
ods html close; ods listing;
```

Image Files (**Shrunk To Fit**) for SGPLOT SERIES Plot By Day With Interval=Week and FitPolicy=Stagger

NOTE: With Interval=Week there is overlap between axis values when using FitPolicy=Stagger. At greater widths (at least as wide as 2500px) overlaps still persist, at least for first & second values (and last & second from last) values. Failure to recognize Stagger overlap for Interval=Week and to thin the values is a known software problem.

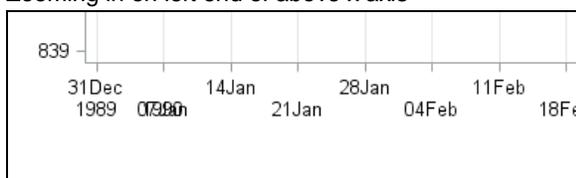
Width = 1000px



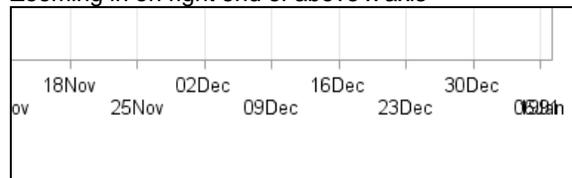
Width = 2500px



Zooming in on left end of above x axis



Zooming in on right end of above x axis



Code used to create the above example is the same as that on the following page except for different image width specified in the ODS GRAPHICS statement and mentioned in the title.

Image File (**Shrunk To Fit Page Width**) for SGPLOT SERIES Plot By Day (Custom Size 2600px X 600px)

Non-overlapping values for Interval=Week and FitPolicy=Stagger requires increasing image width to 2600px, but at this width the software does not need to bother to stagger the values.



```
ods noresults; ods listing close;
ods graphics on / reset=all border=off antialiasmax=2500 tipmax=2500
width=2600px height=600px
imagemap=on imagename=
'SGPLOT_SERIES_By_Day_2600pxBY600px_IntervalEqualWeek_FitPolicyEqualStagger';
ods html path="D:\!PharmaSUG2013\Results" (url=None)
body='SGPLOT_SERIES_By_Day_2600pxBY600px_IntervalEqualWeek_FitPolicyEqualStagger.html'
(title='2600px BY 600px SERIES Plot with Grid Interval=Week - Dow by Day in 1990');
title height=11pt
'2600px BY 600px SERIES Plot with Grid Interval=Week: Dow by Day in 1990';
title2 height=11pt 'Using FitPolicy=Stagger';
proc sgplot data=work.DowByDayIn1990;
series y=Dow x=Date /
markers markerattrs=(size=7 symbol=circlefilled color=red)
lineattrs=(thickness=2 pattern=solid color=blue);
yaxis display=(nolabel) values=(&Ymin_1990 &Ymax_1990);
xaxis display=(nolabel) grid interval=week FitPolicy=Stagger;
format Dow 5.;
run;
ods html close; ods listing;
```

Interval=Week at image width 2600px and accepting default FitPolicy yields same result as above.



This was created with same code as above, except different title2 text, filename changes, and this XAXIS statement:

```
xaxis display=(nolabel) grid interval=week;
```

Simplifying the Appearance of the SERIES Plot

Though necessary tick mark values, and sometimes grid lines, provide communication value, the little tick marks themselves, the axis lines, and any framing of the plot area add NO communication value.

In PROC SGPLOT it is easy to turn off the tick marks, and OSTENSIBLY, the axis lines, but it is cumbersome to remove the frame. With the frame still present, the suppressed axis lines are overlaid with the four-sided frame, in which case the absence of axis lines is not apparent.

It is my understanding that in SAS Version 9.4 it will be possible to remove the frame around the graph display area with an option, so that it will no longer be necessary to create a customized style to accomplish that. That capability is not be confused with **ODS GRAPHICS ON / BORDER=OFF** which is used to turn off the border around the entire image, not around the display area that contains the plot.

Here is the code used to do the frame removal for the web graph displayed on the following page:

```
proc template;
define style styles.HTMLblueWithNoFrame; /* remove useless box around the plot area */
  parent=styles.HTMLblue;
  class graphwalls / frameborder=off;
end; run;

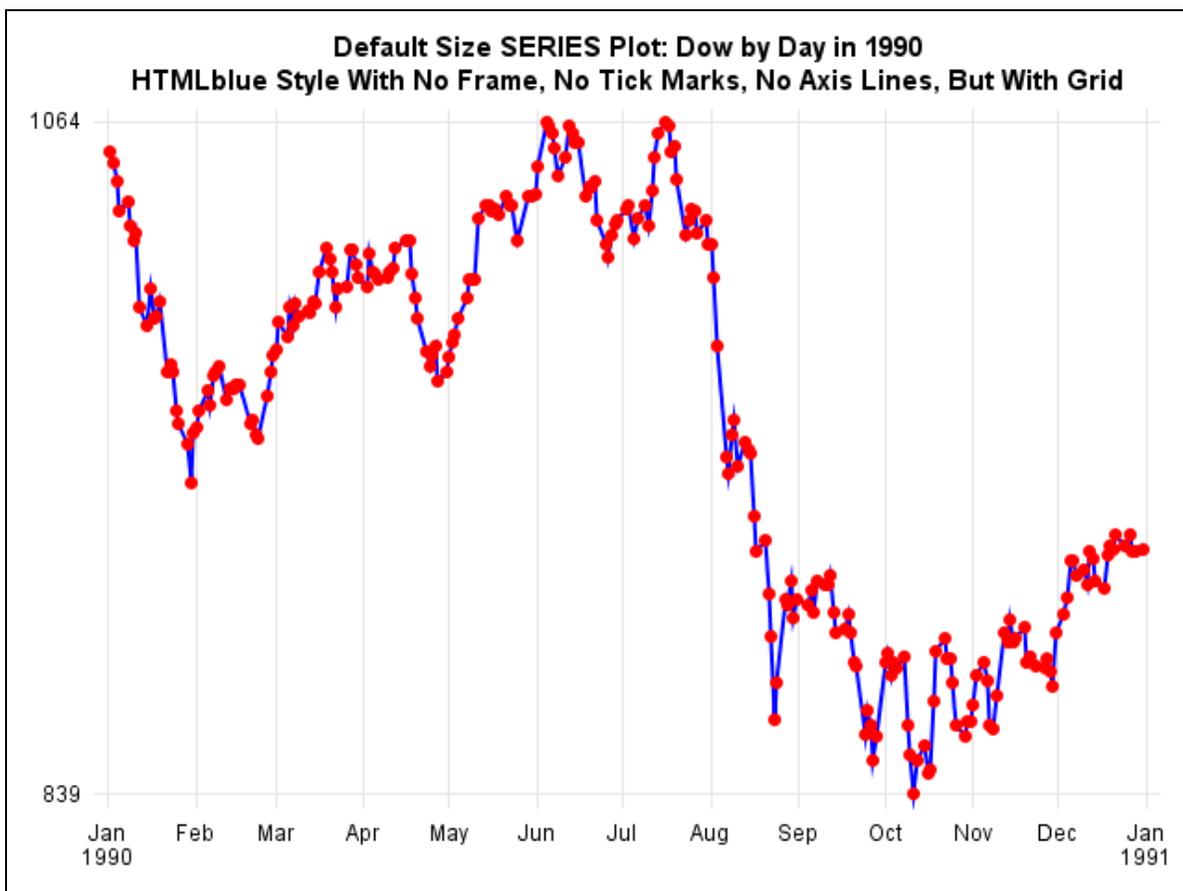
ods noresults; ods listing close;
ods graphics on / reset=all border=off antialiasmax=2500 tipmax=2500
  imagemap=on
  imagename='SGPLOT_SERIES_By_Day_DefaultSize_HTMLblueStyle_WithNoFrame';
ods html path="D:\!PharmaSUG2013\Results" (url=none)
  style=styles.HTMLblueWithNoFrame
  body='SGPLOT_SERIES_By_Day_DefaultSize_HTMLblueStyle_WithNoFrame.html'
  (title='Default Size SERIES Plot - Dow by Day in 1990');
title1 height=11pt 'Default Size SERIES Plot: Dow by Day in 1990';
title2 height=11pt 'HTMLblue Style With No Frame, No Tick Marks, No Axis Lines, But
With Grids';
proc sgplot data=work.DowByDayIn1990;
  series y=Dow x=Date /
    markers markerattrs=(size=7 symbol=circlefilled color=red)
    lineattrs=(thickness=2 pattern=solid color=blue);
  yaxis display=(nolabel noticks noline) grid values=(&Ymin_1990 &Ymax_1990);
  xaxis display=(nolabel noticks noline) grid;
format Dow 5.;
run;
ods html close; ods listing;
```

Web Page (**Shrunk To Fit**) for SGPLOT SERIES Plot By Day With Frame Around Plot Display Area Suppressed

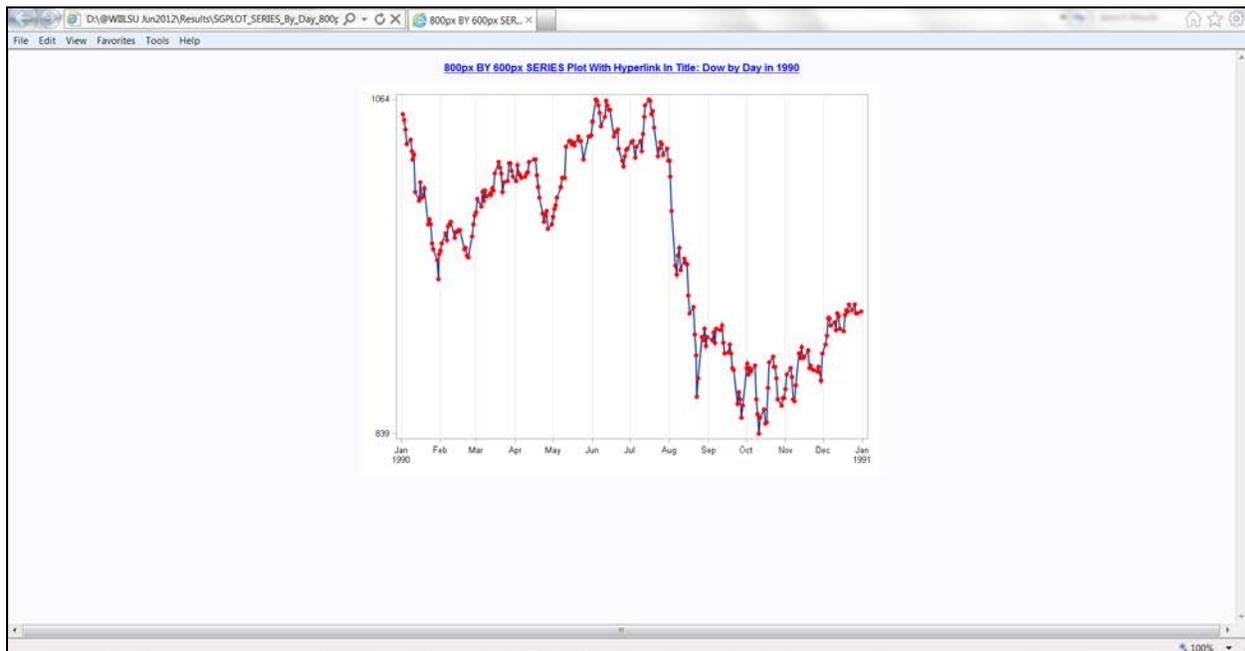
Other default features are also suppressed. ODS style HTMLblue is the V9.3 default style for ODS HTML.



For A Better View: The Image File Itself (**Actual Size:** default 640px X 480px)



The Most Communication-Effective, Most Usable Information Delivery: Web-Enabled Graph + Spreadsheet Linked Forwards and Backwards



Obs	DATE	Dow
1	Tuesday, 2 January 1990	1054
2	Wednesday, 3 January 1990	1050
3	Thursday, 4 January 1990	1044
4	Friday, 5 January 1990	1034
5	Monday, 8 January 1990	1037
6	Tuesday, 9 January 1990	1029
7	Wednesday, 10 January 1990	1024
8	Thursday, 11 January 1990	1027
9	Friday, 12 January 1990	1002
10	Monday, 15 January 1990	996
11	Tuesday, 16 January 1990	1008
12	Wednesday, 17 January 1990	998
13	Thursday, 18 January 1990	999
14	Friday, 19 January 1990	1004
15	Monday, 22 January 1990	980
16	Tuesday, 23 January 1990	983
17	Wednesday, 24 January 1990	980
18	Thursday, 25 January 1990	967
19	Friday, 26 January 1990	963
20	Monday, 29 January 1990	956
21	Tuesday, 30 January 1990	943
22	Wednesday, 31 January 1990	960
23	Thursday, 1 February 1990	962
24	Friday, 2 February 1990	967
25	Monday, 5 February 1990	974
26	Tuesday, 6 February 1990	969
27	Wednesday, 7 February 1990	979
28	Thursday, 8 February 1990	980
29	Friday, 9 February 1990	982

Below is the code used to create the interlinked plot and spreadsheet. The ODS HTML code block to create the graph accepts the default style (HTMLblue), rather than using the custom HTMLblueWithNoFrame style used in the preceding example. Choice of style has no material effect on the interlinking function being demonstrated here.

```
ods noreresults; ods listing close;
ods graphics on / reset=all border=off antialiasmax=2500 tipmax=2500
width=800px height=600px
imagemap=on
imagenname='SGPLOT_SERIES_By_Day_800pxBY600px_WithHyperLinkInTitle';
```

```

ods html path="D:\!PharmaSUG2013\Results" (url=none) nogtitle
  body='SGPLOT_SERIES_By_Day_800pxBY600px_WithHyperLinkInTitle.html'
  (title='800px BY 600px SERIES Plot With Hyperlink In Title - Dow by Day in 1990');
title
link=
'D:\!PharmaSUG2013\Results\SGPLOT_SERIES_By_Day_SpreadSheet_WithHyperLinkInTitle.xls'
'800px BY 600px SERIES Plot With Hyperlink In Title: Dow by Day in 1990';
proc sgplot data=work.DowByDayIn1990;
  series y=Dow x=Date / markers markerattrs=(size=7 symbol=circlefilled color=red)
    lineattrs=(thickness=2 pattern=solid);
  yaxis display=(nolabel) values=(&Ymin_1990 &Ymax_1990);
  xaxis display=(nolabel) grid;
format Dow 5.;
run;
ods html close;
ods html path="D:\!PharmaSUG2013\Results"
  body='SGPLOT_SERIES_By_Day_SpreadSheet_WithHyperLinkInTitle.xls';
title justify=left "<td COLSPAN=4>Dow by Day in 1990</td>"
justify=left
"<td COLSPAN=4>
<a href='SGPLOT_SERIES_By_Day_800pxBY600px_WithHyperLinkInTitle.html'>
Go To Graph of This Data</a></td>";
proc print data=work.DowByDayIn1990;
var Date Dow;
format Dow 5. Date weekdatx.;
run;
ods html close; ods listing;

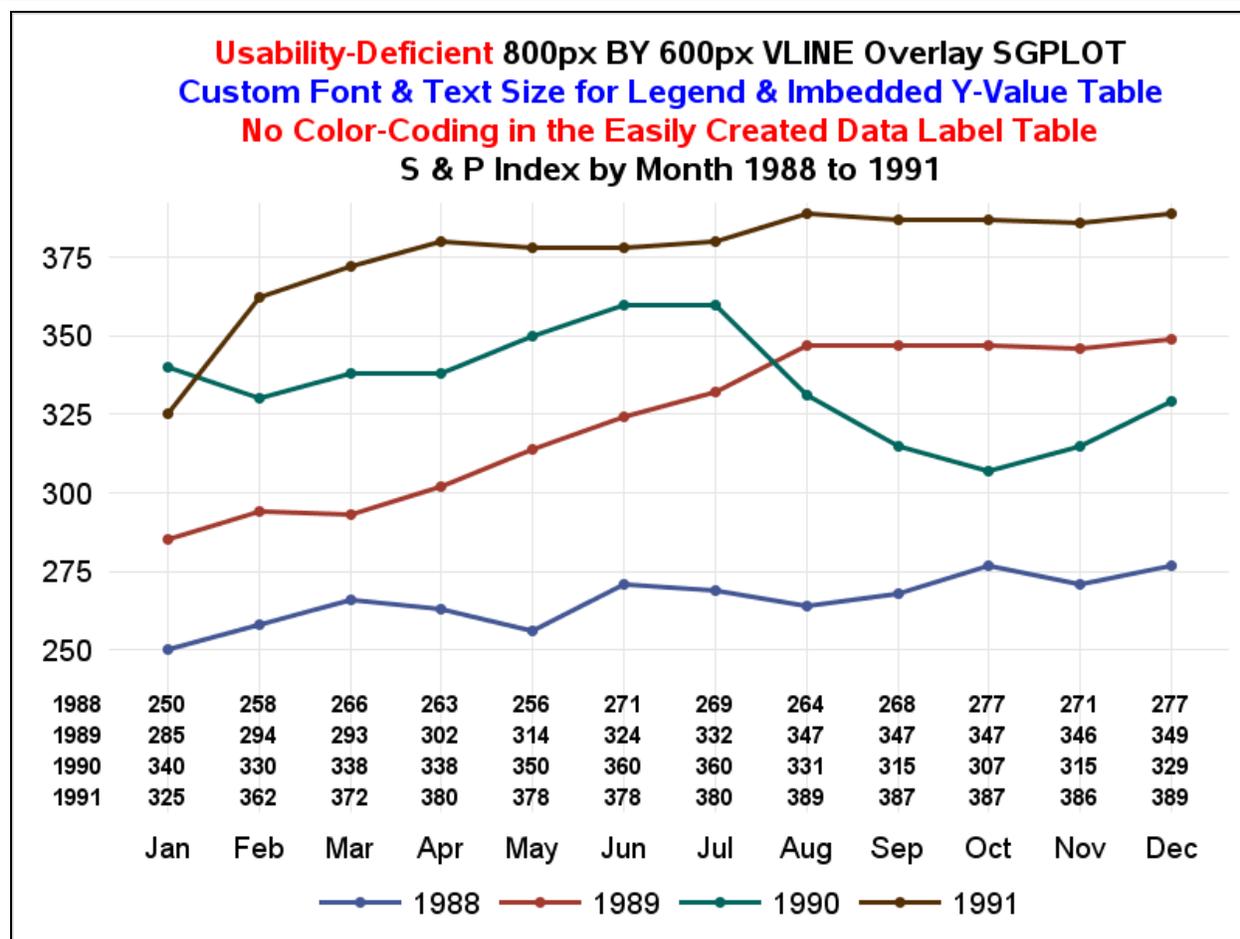
```

COMMUNICATION-EFFECTIVE MULTI-LINE PLOT WITH DETAIL DATA IMBEDDED: HOW TO PROVIDE IMAGE PLUS DETAIL FOR RTF/Word, PDF, PowerPoint, or PRINT

The y values for a plot are easily imbedded in the plot with the DATALABEL option. You are able to control their text characteristics (font, size, color, etc.) with the DATALABELATTRS option. The labels are automatically annotated adjacent to the plot points, but they can sometimes, as you will see, overlap the plot line. (For SAS/GRAPH, in Reference 9, I presented an algorithm to reliably annotate a plot line without such overlaps.) However, when the lines and/or the plot points per line are too numerous and too close, collisions between labels and/or between labels and lines are inevitable. A reliably successful solution removes the y value labels out of the plot area and puts them either in a table separately created with, say, PROC PRINT, and ODS-appended below the graph, or in a table that is part of the graph but below the plot area. **PROC SGPLOT supports time series plots with either the SERIES statement or the VLINE statement, but only the VLINE statement supports the positioning option of DATALABELPOS=BOTTOM, which produces the table that you see in the graph below.**

In earlier work with SAS/GRAPH, I showed that you can use tick mark labels to create, at each x value, a stack of the y values for a multi-line plot. In production, that solution worked well for as many as seven lines and 37 values of x. The solution has no theoretical limit to the number of plot lines or x values, but sufficiently many x values will require the y values to be so small as to be unreadable. That limitation would also apply to the method used below.

NOTE: The usability deficiency in the solution below is that the table entries are not color-coded to match the plot lines (unlike my SAS/GRAPH solution). To associate a y value with a plot point means going from the plot point to the legend to translate color into year and then going to that year's entry under that plot point, keeping in mind which is the month of interest. However, if you do not create the table with DATALABELPOS=BOTTOM, the plot-point y value annotations are color-coded to match the line and marker. So, it is obvious that the software could easily be enhanced to likewise do the color-coding here. Complex coding alternatives for better results are provided later.



NOTE: Titles are centered by the software above the plot area, not within the full graph image width. It is interesting to note that though the starting y axis tick mark value is less than or equal to the lowest y value, the

ending y axis tick mark value is NOT greater than or equal to the highest y value. This does make better use of the available image height, and does not make the graph unusable.

NOTE: The title for the legend is absent because I omitted use of TITLE= in the KEYLEGEND statement below. I used the KEYLEGEND statement only to remove the legend frame and to control the font and size of the legend text. When TITLE= is omitted, the legend title is not defaulted to the variable name or variable label for the GROUP variable. The SAS OnlineDoc describes the TITLE parameter on the KEYLEGEND statement as optional. It is optional, in the sense that omitting it causes no ERROR message, but its omission has the unexpected result of no title text at all. If you omit using KEYLEGEND, then the legend title does default to "Year", the font and text size for title and legend values are defaults, and the legend is displayed in a box.

Code in this section outputs the graphs to disk. From disk, graphs can be manually inserted in Word, PowerPoint, or Excel files. For output direct to RTF or PDF, needed ODS wrapping statements are standard and well explained in other published resources, including those at support.sas.com or in SAS OnlineDoc.

Here is the code used to create the graph above:

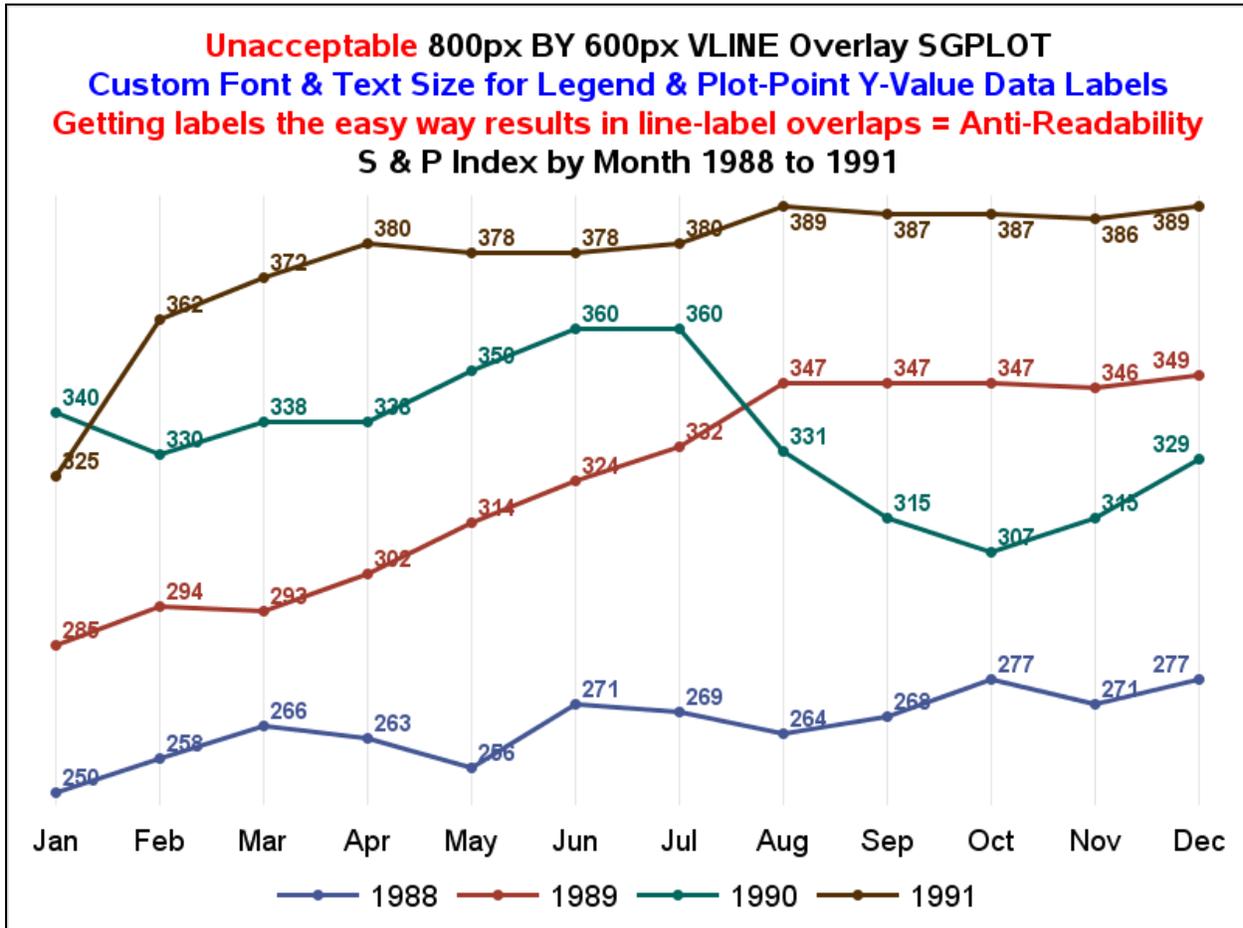
```
ods listing gpath="D:\!PharmaSUG2013\Results" style=Styles.htmlblueWithNoFrame;
ods graphics on / reset=all border=on labelmax=48
  width=800px height=600px /* override default size 480px X 360px */
  imagename=
'SGPLOT_VLINE_By_Month_OneResponseVar_YearAsGroupVarForOverlay_WithYvalueDataLabelsInT
able_AutomaticColorsForLinesAndMarkersAndLabels';
title1 FONT='Albany AMT/Bold' HEIGHT=14 PT
  COLOR=Red 'Usability-Deficient '
  COLOR=Black '800px BY 600px VLINE Overlay SGPLOT';
title2 FONT='Albany AMT/Bold' HEIGHT=14 PT COLOR=Blue
  'Custom Font & Text Size for Legend & Imbedded Y-Value Table';
title3 FONT='Albany AMT/Bold' HEIGHT=14 PT COLOR=Red
  'No Color-Coding in the Easily Created Data Label Table';
title4 FONT='Albany AMT/Bold' HEIGHT=14 PT
  'S & P Index by Month 1988 to 1991';
proc sgplot data=work.SandPindexByMon1988to1991;
vline Month / response=SandPindex group=Year
  markers markerattrs=(size=7 symbol=circlefilled)
  lineattrs=(thickness=3 pattern=solid)
  datalabel=SandPindex
  datalabelattrs=(family='Albany AMT/Bold' size=10 PT weight=Bold)
  datalabelpos=bottom;
keylegend / noborder
  titleattrs=(family='Albany AMT/Bold' size=12 PT)
  valueattrs=(family='Albany AMT/Bold' size=12 PT);
yaxis display=(nolabel noticks noline) grid
  offsetmin=0.05 /* space between top of datalabel table & bottom of plot area */
  valueattrs=(family='Albany AMT/Bold' size=12 PT);
xaxis display=(nolabel noticks noline) grid
  valueattrs=(family='Albany AMT/Bold' size=12 PT) values=(1 to 12 by 1);
format Month MonthAbbrev.;
format SandPindex 5.;
run;
ods listing close; ods listing;
```

NOTE: This particular data requires that the code in Appendix C be run first to prepare the input data from the SASHELP.CITIMON data set.

Below is the result of omitting DATALABELPOS=BOTTOM, changing the titles, using HEIGHT=13pt for TITLE3, and changing the YAXIS statement to

```
yaxis display=none;
```

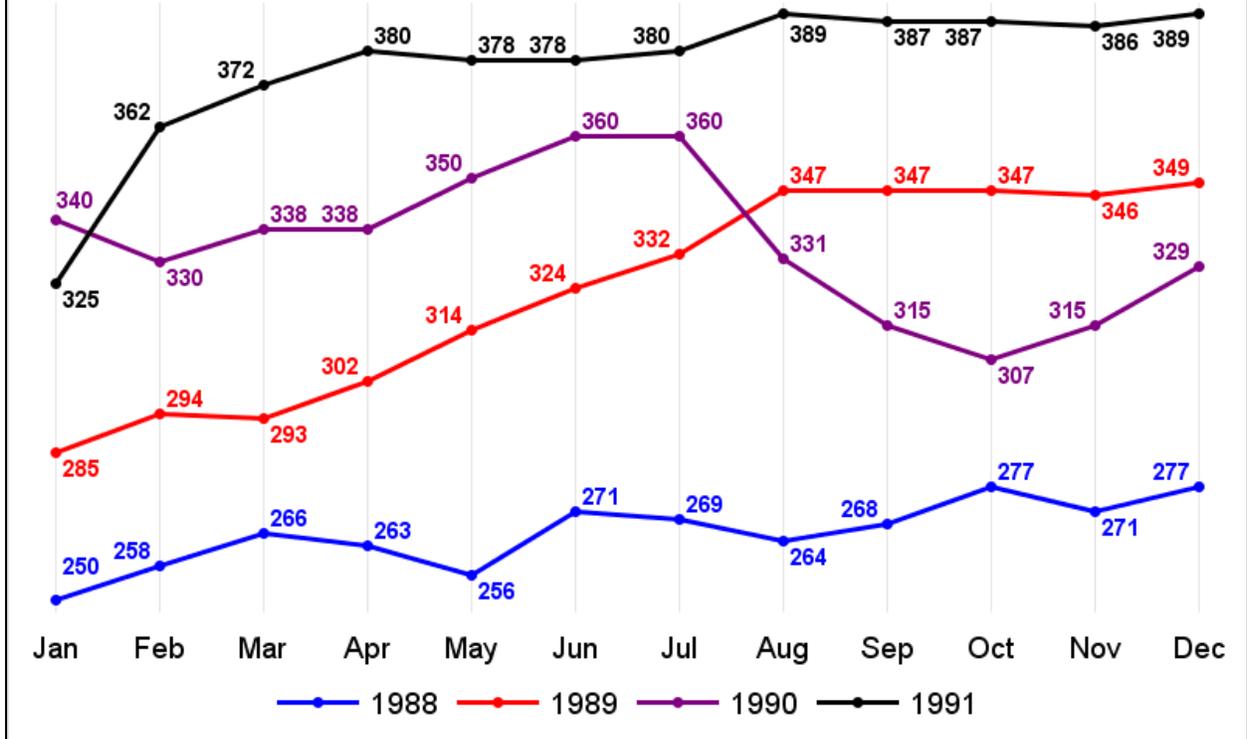
but otherwise using the same code as for the data label table example above. Since all of the y values are annotated in the plot area itself, there would be no benefit in providing a y axis and tick mark values. **With the y axis omitted, unlike the prior example, the titles are centered within the full graph width.**



When experimenting with the TRANSPARENCY option that is available on the VLINE statement, which documentation says will lighten the lines and markers, I found that the data label text was also lightened to the same degree, thus preserving impaired readability.

Good News: If you do some extra work with preprocessing the data and do some extra coding in the SGPLOT PROC step, it IS possible to remove the line-label overlaps, as is shown on the next page. **What is surprising about the solution is that it also uses PROC SGPLOT and the VLINE statement, but the VLINE DATALABEL option delivers line-label collisions if you create the four lines using one VLINE statement with the GROUP=Year option, but no collisions if the lines are created with four separate, more complicated, VLINE statements with no GROUP option.**

**Acceptable Non-Web-Enabled 800px BY 600px VLINE Overlay SGPLOT
 Custom Font & Text Size for Legend & Plot-Point Y-Value Data Labels
 Getting labels the hard way eliminates the line-label overlaps
 S & P Index by Month 1988 to 1991**



NOTE: It should be realized that if there are enough lines intersecting often enough, or nearly parallel and close enough to each other, and/or sufficiently many data points within the lines, collisions between labels and/or labels and lines are inevitable. The above solution is Acceptable, for this particular set of data, but not Most Reliable for the general case. An imbedded data label table is more likely to be successful, but the text size needs to be reduced sufficiently as the number of y values per line increases.

Here is the code used to create the graph above:

```
data work.SandPindexByMonFourYearVars(drop=Year SandPindex);
set work.SandPindexByMon1988to1991;
if Year EQ 1988
then SandPindex_1988 = SandPindex;
else
if Year EQ 1989
then SandPindex_1989 = SandPindex;
else
if Year EQ 1990
then SandPindex_1990 = SandPindex;
else SandPindex_1991 = SandPindex;
run;

ods listing gpath="D:\!PharmaSUG2013\Results" style=Styles.htmlblueWithNoFrame;
ods graphics on / reset=all border=on labelmax=48
width=800px height=600px /* override default size 480px X 360px */
imagename=
'SGPLOT_VLINE_By_Month_FourResponseVariableOverlay_WithYvaluePlotPointDataLabels_Custo
mColorsForLinesAndMarkersAndLabels';
```

```

title1 FONT='Albany AMT/Bold' HEIGHT=14 PT
      COLOR=Blue 'Acceptable Non-Web-Enabled '
      COLOR=Black '800px BY 600px VLINE Overlay SGPLOT';
title2 FONT='Albany AMT/Bold' HEIGHT=14 PT COLOR=Blue
      'Custom Font & Text Size for Legend & Plot-Point Y-Value Data Labels';
title3 FONT='Albany AMT/Bold' HEIGHT=14 PT COLOR=Blue
      'Getting labels the hard way eliminates the line-label overlaps';
title4 FONT='Albany AMT/Bold' HEIGHT=14 PT
      'S & P Index by Month 1988 to 1991';
proc sgplot data=work.SandPindexByMonFourYearVars;
vline Month / response=SandPindex_1988 nostatlabel legendlabel='1988'
      markers markerattrs=(size=7 symbol=circlefilled color=blue)
      lineattrs=(thickness=3 pattern=solid color=blue)
      datalabel=SandPindex_1988
      datalabelattrs=(family='Albany AMT/Bold' size=10 PT weight=Bold color=blue);
vline Month / response=SandPindex_1989 nostatlabel legendlabel='1989'
      markers markerattrs=(size=7 symbol=circlefilled color=red)
      lineattrs=(thickness=3 pattern=solid color=red)
      datalabel=SandPindex_1989
      datalabelattrs=(family='Albany AMT/Bold' size=10 PT weight=Bold color=red);
vline Month / response=SandPindex_1990 nostatlabel legendlabel='1990'
      markers markerattrs=(size=7 symbol=circlefilled color=purple)
      lineattrs=(thickness=3 pattern=solid color=purple)
      datalabel=SandPindex_1990
      datalabelattrs=(family='Albany AMT/Bold' size=10 PT weight=Bold color=purple);
vline Month / response=SandPindex_1991 nostatlabel legendlabel='1991'
      markers markerattrs=(size=7 symbol=circlefilled color=black)
      lineattrs=(thickness=3 pattern=solid color=black)
      datalabel=SandPindex_1991
      datalabelattrs=(family='Albany AMT/Bold' size=10 PT weight=Bold color=black);
keylegend / noborder
      titleattrs=(family='Albany AMT/Bold' size=12 PT)
      valueattrs=(family='Albany AMT/Bold' size=12 PT);
yaxis display=none;
xaxis display=(nolabel noticks noline) grid
      valueattrs=(family='Albany AMT/Bold' size=12 PT) values=(1 to 12 by 1);
format Month MonthAbbrev.;
format SandPindex_1988 SandPindex_1989 SandPindex_1990 SandPindex_1991 5.;
run;
ods listing close; ods listing;

```

The above code is more complex. In principle, one could convert it to a macro. It would be straightforward, but a lot of code. I leave it as an exercise for the interested and motivated reader, and would be grateful for a copy of the macro.

As a next step, let's see what happens if one simply adds "datalabelpos=bottom" to each of the VLINE statements above, changes the graph titles and filename, provides LABEL statements for each response variable:

```

label SandPindex_1988='1988';
label SandPindex_1989='1989';
label SandPindex_1990='1990';
label SandPindex_1991='1991';

```

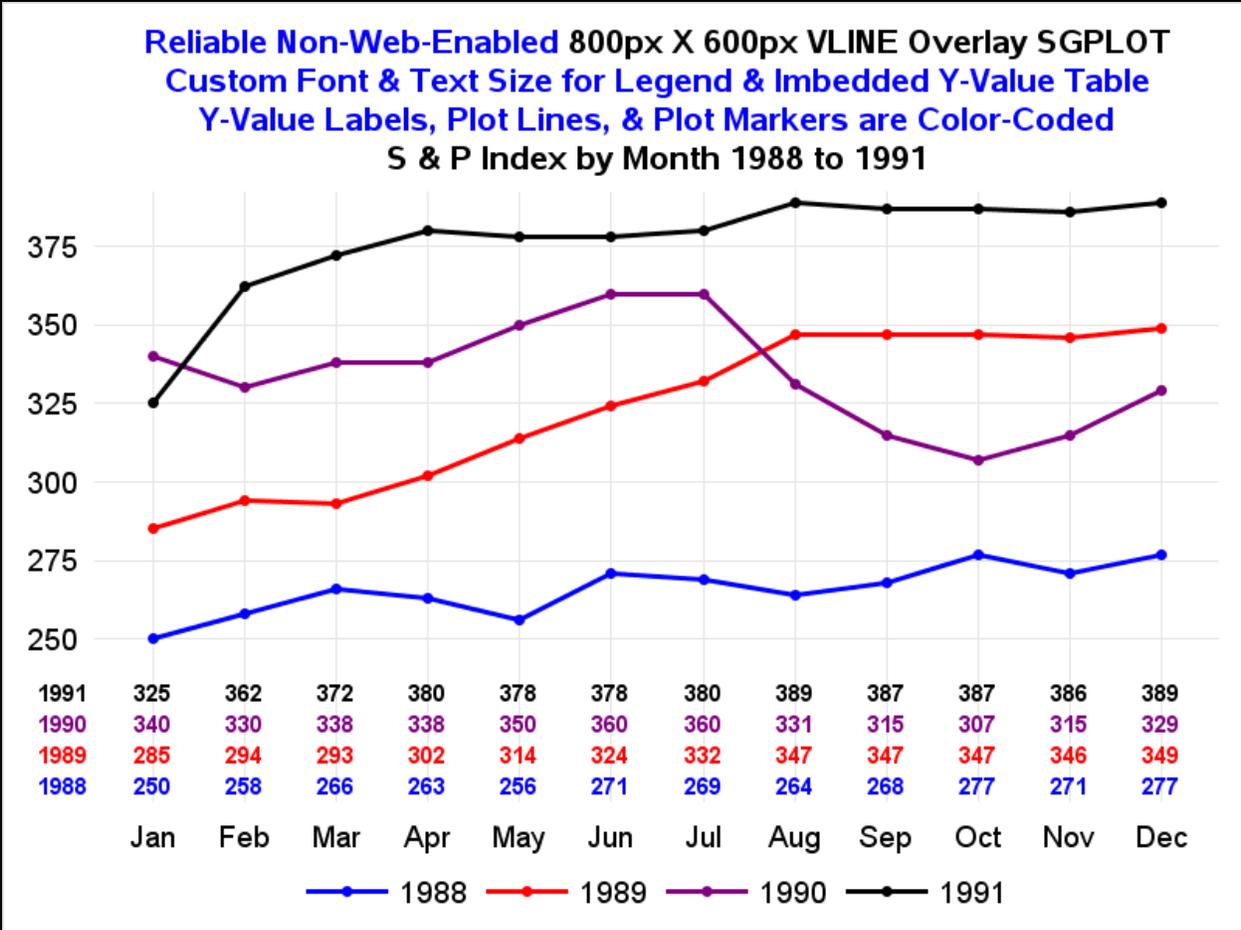
and restores the y axis tick mark values by replacing

```
yaxis display=none;
```

with this statement

```
yaxis display=(nolabel noticks noline) grid offsetmin=0.05
      valueattrs=(family='Albany AMT/Bold' size=12 PT);
```

The result is on the next page.



Here is the code used to create the graph above:

```

data work.SandPIndexByMonFourYearVars(drop=Year SandPIndex);
set work.SandPIndexByMon1988to1991;
if Year EQ 1988
then SandPIndex_1988 = SandPIndex;
else
if Year EQ 1989
then SandPIndex_1989 = SandPIndex;
else
if Year EQ 1990
then SandPIndex_1990 = SandPIndex;
else SandPIndex_1991 = SandPIndex;
run;

ods listing gpath="D:\!PharmaSUG2013\Results" style=Styles.htmlblueWithNoFrame;
ods graphics on / reset=all border=on labelmax=48
width=800px height=600px /* override default size 480px X 360px */
imagename=
'SGPLOT_VLINE_By_Month_FourResponseVariableOverlay_WithYvalueDataLabelsInTable_CustomC
olorsForLegendAndLinesAndMarkersAndDataLabels';
title1 FONT='Albany AMT/Bold' HEIGHT=14 PT
COLOR=Blue 'Reliable Non-Web-Enabled '
COLOR=Black '800px X 600px VLINE Overlay SGPLOT';
title2 FONT='Albany AMT/Bold' HEIGHT=14 PT COLOR=Blue
'Custom Font & Text Size for Legend & Imbedded Y-Value Table';
title3 FONT='Albany AMT/Bold' HEIGHT=14 PT COLOR=Blue
'Y-Value Labels, Plot Lines, & Plot Markers are Color-Coded';

```

```

title4 FONT='Albany AMT/Bold' HEIGHT=14 PT
      'S & P Index by Month 1988 to 1991';
proc sgplot data=work.SandPindexByMonFourYearVars;
vline Month / response=SandPindex_1988 nostatlabel
      markers markerattrs=(size=7 symbol=circlefilled color=blue)
      lineattrs=(thickness=3 pattern=solid color=blue)
      datalabel=SandPindex_1988
      datalabelattrs=(family='Albany AMT/Bold' size=10 PT weight=Bold color=blue)
      datalabelpos=bottom;
vline Month / response=SandPindex_1989 nostatlabel
      markers markerattrs=(size=7 symbol=circlefilled color=red)
      lineattrs=(thickness=3 pattern=solid color=red)
      datalabel=SandPindex_1989
      datalabelattrs=(family='Albany AMT/Bold' size=10 PT weight=Bold color=red)
      datalabelpos=bottom;
vline Month / response=SandPindex_1990 nostatlabel
      markers markerattrs=(size=7 symbol=circlefilled color=purple)
      lineattrs=(thickness=3 pattern=solid color=purple)
      datalabel=SandPindex_1990
      datalabelattrs=(family='Albany AMT/Bold' size=10 PT weight=Bold color=purple)
      datalabelpos=bottom;
vline Month / response=SandPindex_1991 nostatlabel
      markers markerattrs=(size=7 symbol=circlefilled color=black)
      lineattrs=(thickness=3 pattern=solid color=black)
      datalabel=SandPindex_1991
      datalabelattrs=(family='Albany AMT/Bold' size=10 PT weight=Bold color=black)
      datalabelpos=bottom;
keylegend / noborder
      titleattrs=(family='Albany AMT/Bold' size=12 PT)
      valueattrs=(family='Albany AMT/Bold' size=12 PT);
yaxis display=(nolabel noticks noline) grid
      offsetmin=0.05 /* space between top of datalabel table & bottom of plot area */
      valueattrs=(family='Albany AMT/Bold' size=12 PT);
xaxis display=(nolabel noticks noline) grid
      valueattrs=(family='Albany AMT/Bold' size=12 PT) values=(1 to 12 by 1);
format Month MonthAbbrev.;
format SandPindex_1988 SandPindex_1989 SandPindex_1990 SandPindex_1991 5.;
label SandPindex_1988='1988';
label SandPindex_1989='1989';
label SandPindex_1990='1990';
label SandPindex_1991='1991';
run;
ods listing close; ods listing;

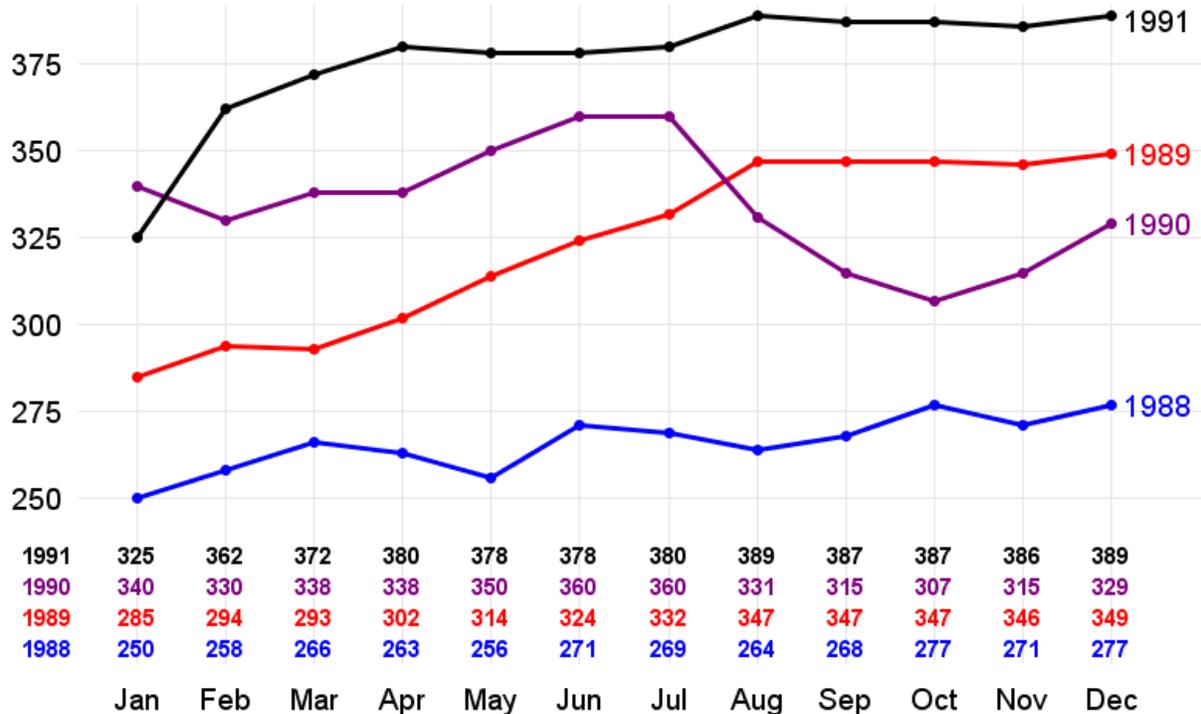
```

As was the case for the improved solution for y-value plot-point data labels, the above code is complex. My quest for a way to create a multi-line plot with an imbedded color-coded y-value table was inspired by seeing Philip R. Holland's black-and-white multi-line plot with imbedded table, which relies on use of Graph Template Language (GTL) and PROC SGRENDER. See Reference 10.

I recently heard Sanjay Matange emphasize the benefits of using the `curvelabel` option to replace the legend. The `curvelabels` eliminate the need for eye travel from a legend to the plot lines to identify the plots. Of course, if the legend entry text is long, then `curvelabels` would steal display horizontal space from the plot lines. With a history plot, it is the horizontal space for the time dimension that is usually the critical constraint, unless there are only a small number of dates, times, or datetimes.

On a multi-line plot, putting a label anywhere inside the plot area would incur the risk of overlaps between labels for different curves or between a curve label and one of the other curves. I prefer the option `curvelabelpos=max`, which, per the documentation, "places the label at the part of the curve closest to the maximum X axis value". I initially tried `curvelabelpos=end`, but the label of each curve was too close to its last plot point. If you have a plot where too many of the last plot points are too close together, overlap of the curve labels might be inevitable, in which case `curvelabelpos=min` or use of a legend might be the required alternative. Also, multi-line plots with `curvelabels` can be interpreted even if printed (or rendered) in black-and-white without using different symbol types by line and a legend.

**Easier-To-Use Non-Web-Enabled 800px X 600px VLINE Overlay SGPLOT
 Custom Font & Text Size for Curve Labels & Imbedded Y-Value Table
 Needs no legend and is effective even if printed black & white
 S & P Index by Month 1988 to 1991**



Here is the final code used to create the graph above:

```
data work.SandPIndexByMonFourYearVars(drop=Year SandPIndex);
set work.SandPIndexByMon1988to1991;
if Year EQ 1988
then SandPIndex_1988 = SandPIndex;
else
if Year EQ 1989
then SandPIndex_1989 = SandPIndex;
else
if Year EQ 1990
then SandPIndex_1990 = SandPIndex;
else SandPIndex_1991 = SandPIndex;
run;

ods listing gpath="D:\!PharmaSUG2013\Results" style=Styles.HTMLblueWithNoFrame;
ods graphics on / reset=all border=on labelmax=48
width=800px height=600px /* override default size 480px X 360px */
imagename=
'SGPLOT_VLINE_By_Month_FourResponseVariableOverlay_WithYvalueDataLabelsInTableAndCurve
LabelAtEnd_CustomColorsForLinesAndMarkersAndDataLabelsAndCurveLabels';
title1 FONT='Albany AMT/Bold' HEIGHT=14 PT
COLOR=Blue 'Easier-To-Use Non-Web-Enabled '
COLOR=Black '800px X 600px VLINE Overlay SGPLOT';
title2 FONT='Albany AMT/Bold' HEIGHT=14 PT COLOR=Blue
'Custom Font & Text Size for Curve Labels & Imbedded Y-Value Table';
title3 FONT='Albany AMT/Bold' HEIGHT=14 PT COLOR=Blue
'Needs no legend and is effective even if printed black & white';
```

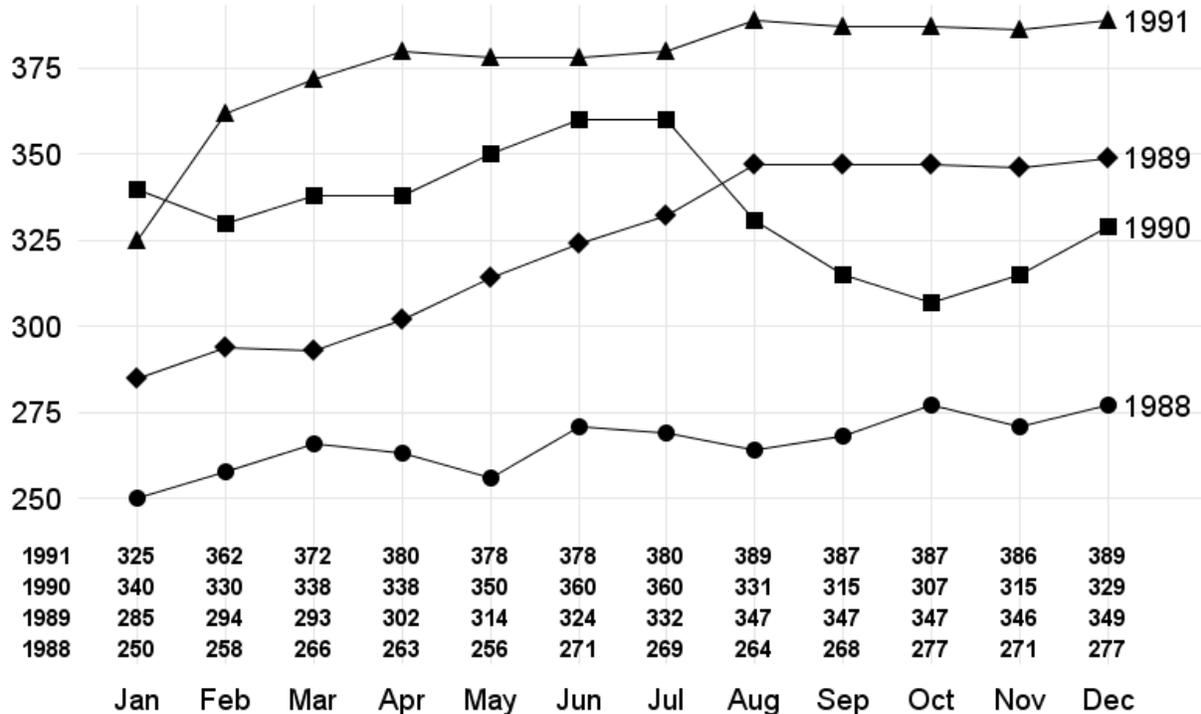
```

title4 FONT='Albany AMT/Bold' HEIGHT=14 PT
      'S & P Index by Month 1988 to 1991';
proc sgplot data=work.SandPindexByMonFourYearVars;
vline Month / response=SandPindex_1988 nostatlabel
      markers markerattrs=(size=7 symbol=circlefilled color=blue)
      lineattrs=(thickness=3 pattern=solid color=blue)
      datalabel=SandPindex_1988
      datalabelattrs=(family='Albany AMT/Bold' size=10 PT weight=Bold color=blue)
      datalabelpos=bottom
      curvelabel='1988'
      curvelabelattrs=(family='Albany AMT/Bold' size=12 PT color=blue)
      curvelabelpos=max;
vline Month / response=SandPindex_1989 nostatlabel
      markers markerattrs=(size=7 symbol=circlefilled color=red)
      lineattrs=(thickness=3 pattern=solid color=red)
      datalabel=SandPindex_1989
      datalabelattrs=(family='Albany AMT/Bold' size=10 PT weight=Bold color=red)
      datalabelpos=bottom
      curvelabel='1989'
      curvelabelattrs=(family='Albany AMT/Bold' size=12 PT color=red)
      curvelabelpos=max;
vline Month / response=SandPindex_1990 nostatlabel
      markers markerattrs=(size=7 symbol=circlefilled color=purple)
      lineattrs=(thickness=3 pattern=solid color=purple)
      datalabel=SandPindex_1990
      datalabelattrs=(family='Albany AMT/Bold' size=10 PT weight=Bold color=purple)
      datalabelpos=bottom
      curvelabel='1990'
      curvelabelattrs=(family='Albany AMT/Bold' size=12 PT color=purple)
      curvelabelpos=max;
vline Month / response=SandPindex_1991 nostatlabel
      markers markerattrs=(size=7 symbol=circlefilled color=black)
      lineattrs=(thickness=3 pattern=solid color=black)
      datalabel=SandPindex_1991
      datalabelattrs=(family='Albany AMT/Bold' size=10 PT weight=Bold color=black)
      datalabelpos=bottom
      curvelabel='1991'
      curvelabelattrs=(family='Albany AMT/Bold' size=12 PT color=black)
      curvelabelpos=max;
yaxis display=(nolabel noticks noline) grid
      offsetmin=0.05 /* space between top of datalabel table & bottom of plot area */
      valueattrs=(family='Albany AMT/Bold' size=12 PT);
xaxis display=(nolabel noticks noline) grid
      valueattrs=(family='Albany AMT/Bold' size=12 PT) values=(1 to 12 by 1);
format Month MonthAbbrev.;
format SandPindex_1988 SandPindex_1989 SandPindex_1990 SandPindex_1991 5.;
label SandPindex_1988='1988';
label SandPindex_1989='1989';
label SandPindex_1990='1990';
label SandPindex_1991='1991';
run;
ods listing close; ods listing;

```

To create a usable multi-line plot in black and white requires use of different plot symbols for each plot line. On the following page is an illustration of such a plot, with curvelabels. (For a black and white plot WITHOUT curvelabels, given the illustration and creation code above for a four-color graph without curvelabels and with a legend, you should be able to visualize and produce one yourself, by modifying the code below the illustration on the next page.)

**Easier-To-Use Non-Web-Enabled 800px X 600px VLINE Overlay SGPLOT
 Custom Font & Text Size for Curve Labels & Imbedded Y-Value Table
 Needs no legend and is designed for and with Black & White
 S & P Index by Month 1988 to 1991**



Here is the code used to create the graph above:

```
data work.SandPIndexByMonFourYearVars(drop=Year SandPIndex);
set work.SandPIndexByMon1988to1991;
if Year EQ 1988
then SandPIndex_1988 = SandPIndex;
else
if Year EQ 1989
then SandPIndex_1989 = SandPIndex;
else
if Year EQ 1990
then SandPIndex_1990 = SandPIndex;
else SandPIndex_1991 = SandPIndex;
run;

ods listing gpath="D:\!PharmaSUG2013\Results" style=Styles.HTMLblueWithNoFrame;
ods graphics on / reset=all border=on labelmax=48
width=800px height=600px /* override default size 480px X 360px */
imagename=
'SGPLOT_VLINE_By_Month_FourResponseVariableOverlay_WithYvalueDataLabelsInTableAndCurve
LabelAtEnd';
title1 FONT='Albany AMT/Bold' HEIGHT=14 PT
COLOR=Blue 'Easier-To-Use Non-Web-Enabled '
COLOR=Black '800px X 600px VLINE Overlay SGPLOT';
title2 FONT='Albany AMT/Bold' HEIGHT=14 PT COLOR=Blue
'Custom Font & Text Size for Curve Labels & Imbedded Y-Value Table';
title3 FONT='Albany AMT/Bold' HEIGHT=14 PT COLOR=Blue
'Needs no legend and is designed for and with Black & White';
```

```

title4 FONT='Albany AMT/Bold' HEIGHT=14 PT
      'S & P Index by Month 1988 to 1991';
proc sgplot data=work.SandPindexByMonFourYearVars;
vline Month / response=SandPindex_1988 nostatlabel
      markers markerattrs=(size=10 symbol=circlefilled color=black)
      lineattrs=(color=black)
      datalabel=SandPindex_1988
      datalabelattrs=(family='Albany AMT/Bold' size=10 PT weight=Bold color=black)
      datalabelpos=bottom
      curvelabel='1988'
      curvelabelattrs=(family='Albany AMT/Bold' size=12 PT color=black)
      curvelabelpos=max;
vline Month / response=SandPindex_1989 nostatlabel
      markers markerattrs=(size=10 symbol=diamondfilled color=black)
      lineattrs=(color=black)
      datalabel=SandPindex_1989
      datalabelattrs=(family='Albany AMT/Bold' size=10 PT weight=Bold color=black)
      datalabelpos=bottom
      curvelabel='1989'
      curvelabelattrs=(family='Albany AMT/Bold' size=12 PT color=black)
      curvelabelpos=max;
vline Month / response=SandPindex_1990 nostatlabel
      markers markerattrs=(size=10 symbol=squarefilled color=black)
      lineattrs=(color=black)
      datalabel=SandPindex_1990
      datalabelattrs=(family='Albany AMT/Bold' size=10 PT weight=Bold color=black)
      datalabelpos=bottom
      curvelabel='1990'
      curvelabelattrs=(family='Albany AMT/Bold' size=12 PT color=black)
      curvelabelpos=max;
vline Month / response=SandPindex_1991 nostatlabel
      markers markerattrs=(size=10 symbol=trianglefilled color=black)
      lineattrs=(color=black)
      datalabel=SandPindex_1991
      datalabelattrs=(family='Albany AMT/Bold' size=10 PT weight=Bold color=black)
      datalabelpos=bottom
      curvelabel='1991'
      curvelabelattrs=(family='Albany AMT/Bold' size=12 PT color=black)
      curvelabelpos=max;
yaxis display=(nolabel noticks noline) grid
      offsetmin=0.05 /* space between top of datalabel table & bottom of plot area */
      valueattrs=(family='Albany AMT/Bold' size=12 PT);
xaxis display=(nolabel noticks noline) grid
      valueattrs=(family='Albany AMT/Bold' size=12 PT) values=(1 to 12 by 1);
format Month MonthAbbrev.;
format SandPindex_1988 SandPindex_1989 SandPindex_1990 SandPindex_1991 5.;
label SandPindex_1988='1988';
label SandPindex_1989='1989';
label SandPindex_1990='1990';
label SandPindex_1991='1991';
run;
ods listing close; ods listing;

```


CONCLUSION

SAS/GRAPH has been progressively improved over 34 years, and many of its features are regrettably missing in the new technology of ODS Graphics and SG procedures. Ways to cope with this situation have been shown, and macro tools to make that even easier for some objectives, as well as macros to implement some powerful and useful data visualization solutions not shown here (mostly using ODS Graphics, but with SAS/GRAPH when that is the better tool), are presented in the companion paper that is Reference 2.

REFERENCES

1. Holland, Philip R. "Anscombe's Quartet", *IEWS News, Issue 54*. UK: IEWS International SAS Programmer Community. 2011. http://www.sascommunity.org/wiki/IEWS_News_backissues
2. Bessler, LeRoy. "Data Visualization Power Tools: Expedite the Easy, Implement the Difficult, or Handle Big Data", *Proceedings of the Pharmaceutical SAS Users Group 2013 Conference*, Chicago, IL, USA: 2013. www.PharmaSUG.org
3. Bessler, LeRoy. "The Art of Being Color-Communication-Effective, Not Just Colorful", *Proceedings of the MidWest SAS Users Group 2010 Conference*, Milwaukee, WI, USA: 2010. <http://www.mwsug.org/proceedings/2010/solutions/MWSUG-2010-169.pdf>
4. Bessler, LeRoy. "The Most Communication-Effective and Most Usable Information Delivery", *Proceedings of the SAS Global Forum 2010 Conference*. Cary, NC: SAS Institute Inc. 2010. <http://support.sas.com/resources/papers/proceedings10/231-2010.pdf>
5. Bessler, LeRoy. "Communication-Effective Pie Charts", *Proceedings of the SAS Global Forum 2007 Conference*. Cary, NC: SAS Institute Inc. 2007. <http://www2.sas.com/proceedings/forum2007/134-2007.pdf>
6. Bessler, LeRoy. "All the Ways to Display Multiple Trend Charts", *IEWS News, Issue 52*. UK: IEWS International SAS Programmer Community. 2011. http://www.sascommunity.org/wiki/IEWS_News_backissues
7. Bessler, LeRoy. "Comparison of SAS Graphic Alternatives, Old and New", *Proceedings of the SAS Global Forum 2012 Conference*. Cary, NC: SAS Institute Inc. 2012. <http://support.sas.com/resources/papers/proceedings12/235-2012.pdf>
8. Bessler, LeRoy and Riley, Alexandra. "All the Ways to Display Multiple Trend Charts: Three Selected Examples and Their Code", *IEWS News, Issue 53*. UK: IEWS International SAS Programmer Community. 2011. http://www.sascommunity.org/wiki/IEWS_News_backissues
9. Bessler, LeRoy. "%TRENDANO Macro for Collision-Free Automatic Annotation of Trend Plots", *Wisconsin Illinois SAS Users Conference Proceedings*. Mequon, WI: Software User Services. 2001. For the latest report on this work, see <http://www2.sas.com/proceedings/sugi27/p093-27.pdf> for which Dr. Francesca Pierri was co-author.
10. Holland, Philip R. "Why Should You Be Using the New SG (Statistical Graphics) Procedures in SAS 9.2?", *Proceedings of the SAS Global Forum 2011 Conference*. Cary, NC: SAS Institute Inc. 2011. <http://support.sas.com/resources/papers/proceedings11/427-2011.pdf>

AUTHOR INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

LeRoy Bessler PhD
Bessler Consulting and Research
Converting Complexity Into Clarity™
Strong Smart Systems™
1-262-512-1729
Le_Roy_Bessler@wi.rr.com

Dr. LeRoy Bessler has presented at software user conferences in the US, Canada, and Europe, on topics such as effective visual communication (using graphs, tables, web pages, or color), highly formatted Excel reporting from SAS, custom-developed tools to assist SAS server administrators, users, and managers, and Software-Intelligent Application Development methods to maximize Reliability, Reusability, Maintainability, Extendibility, and Flexibility. His SAS experience includes application development and supporting users, servers, software, and data.

SAS, SAS/GRAPH, 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 trademarks of their respective companies.

Converting Complexity Into Clarity and Strong Smart Systems are trademarks of LeRoy Bessler PhD.

APPENDIX A. Web Page Scrolling Avoidance: Let them see The Whole Picture

Scrolling

Another concern with trend plots is that a time period which is very, very long (i.e., has too numerous data points) cannot be adequately viewed in a single static display area. A web graph can be scrolled horizontally.

The same consideration applies to a horizontal bar chart with too many bars to fit in the display area. A web graph can be scrolled vertically.

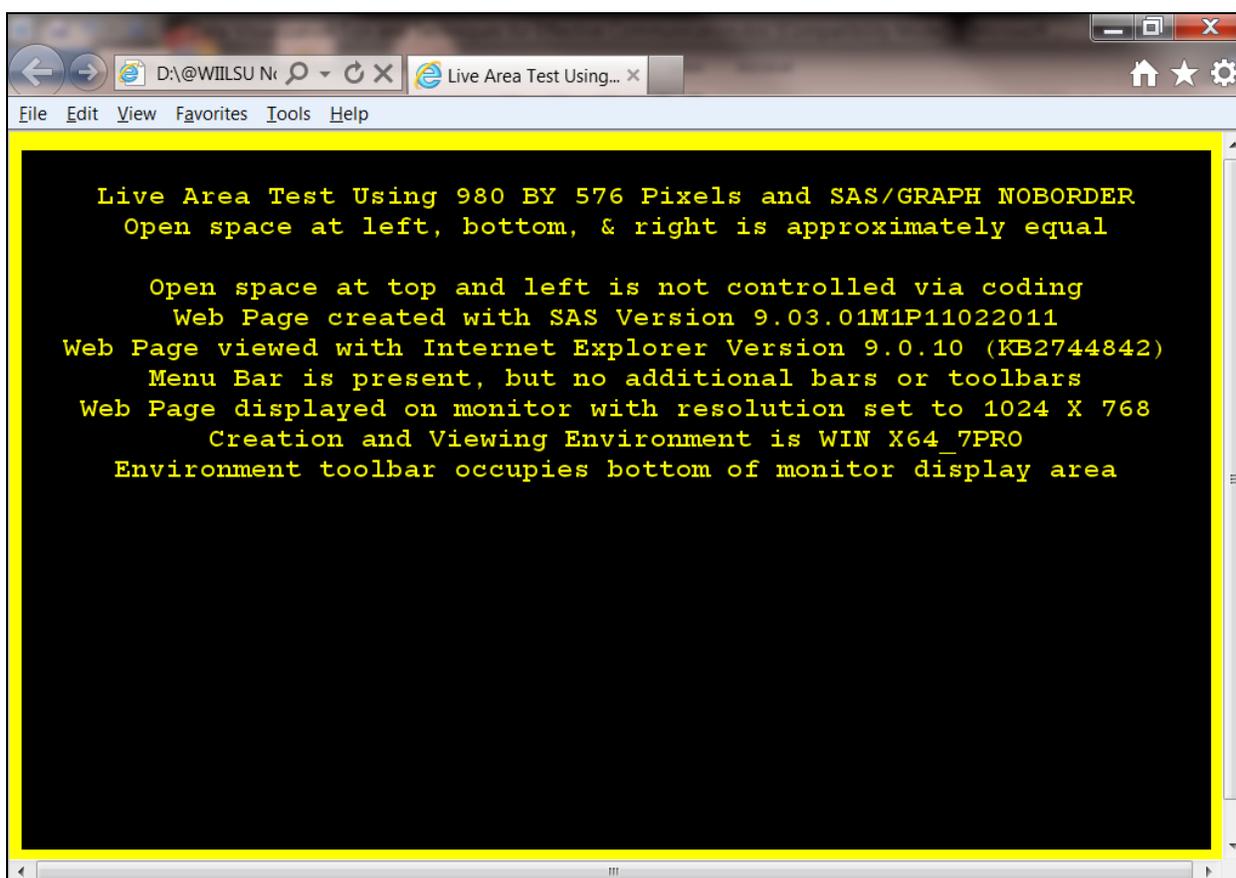
Scrolling Avoidance

Above I emphasized the benefits of scrolling in cases where it is a functional requirement for effective information delivery. However, web design and construction should PREVENT the need for scrolling whenever possible. If unneeded, scrolling is anti-communicative. I.e., it denies the viewer the entire picture, and reliable visual comparison of all of its parts simultaneously is impossible. When picking the pixel dimensions of your output web graphs, you need to make some assumptions about the environment where your graphs will be viewed.

Factors that come into play include screen shape, which was fixed and therefore always known before the arrival of wide screen laptops, which I suspect are optimized for viewing movies. All meeting room/site projection screens are still in the traditional proportions of 4 : 3, being the same as those of old-fashioned laptops and CRT monitors.

An obvious factor is monitor screen resolution. On old laptops, a common setting is 1024 X 768.

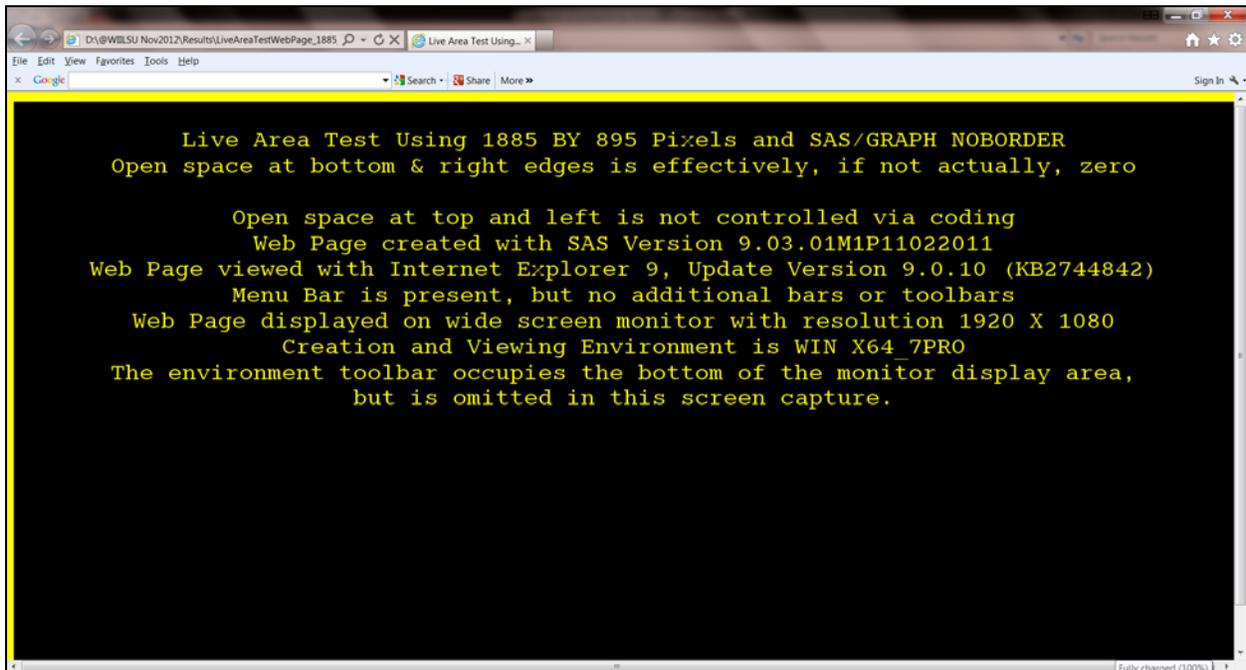
Another factor is web browser. People use Internet Explorer, Google Chrome, etc. The space left when you subtract the Windows framing and the browser framing is the available display space, often called the live space or live area. Below is a demonstration, using SAS/GRAPH PROC GSLIDE, of how to determine the live space for a web graph, with my screen resolution set to 1024 X 768 which is the commonly used resolution for traditional monitor screens with 4 : 3 proportions (wide screens have 16 : 9 proportions). Pixel settings are as described.



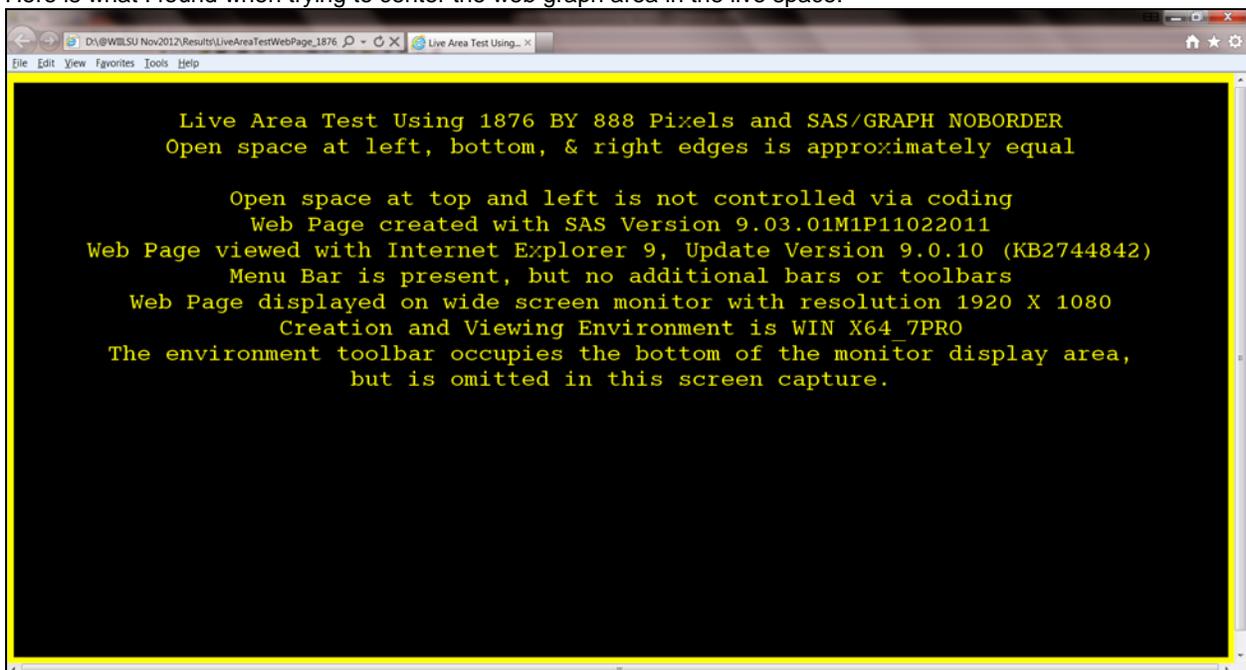
Graph image sizing is a quagmire in my judgment. Your safest policy is to design for the worst case (smallest live space), but you have to decide what that worst case is. I recently did a project to develop web-deployed information delivery where the audience and display environment were known. In another case, I did not have such knowledge, but managed to create a design that drew no complaints, despite being for an important, highly used deliverable.

When you do know your target environment, it is possible and useful to determine the maximum live area. Even if you do not intend to use the maximum, it's important to know what it is so that you have a framework that you can either maximally exploit, or instead within which you can know how much you are safely under-sizing your web graphs.

With my screen resolution set to 1920 X 1080, which is the typical maximum size for a wide screen laptop, below is what I found as maximum graph display capacity:



Here is what I found when trying to center the web graph area in the live space:



If you use GOPTIONS BORDER, you must reduce the GOPTIONS xpixels and ypixels counts to fit it.

If you would like to experiment with determining live space, below is the code I used for the demonstrations above.

NOTE: This cannot be done with ODS Graphics, because it has no PROC GSLIDE analogue.

```
proc template;
edit Styles.Minimal AS Styles.Minimal_CXFFFF00;
style body / background = CXFFFF00; /* browser-safe yellow */
end; run;

%macro LiveAreaTestWebPage(xpixels=,ypixels=,TitleLine2=);

ODS LISTING CLOSE;

ODS HTML PATH="D:\!PharmaSUG2013\Results" (URL = NONE)
FILE="LiveAreaTestWebPage_&xpixels._BY_&ypixels._NOBORDER.html"
(TITLE="Live Area Test Using &xpixels BY &ypixels Pixels and SAS/GRAPH NOBORDER")
STYLE=Styles.Minimal_CXFFFF00
GTITLE GFOOTNOTE;

GOPTIONS RESET=ALL;
GOPTIONS DEVICE=GIF;
GOPTIONS NOBORDER;
GOPTIONS CBACK=CX000000; /* black */
GOPTIONS XPIXELS=&xpixels YPIXELS=&ypixels;

proc gslide;
title1
  height=4 PCT /* each title text line to be four percent of image height */
  color=CXFFFF00 /* browser-safe yellow */
  " " /* blank space at top of web page */
justify=center
  "Live Area Test Using &xpixels BY &ypixels Pixels and SAS/GRAPH NOBORDER"
justify=center "&TitleLine2"
justify=center " "
justify=center "Open space at top and left is not controlled via coding"
justify=center "Web Page created with SAS Version &sysvlong4"
justify=center "Web Page viewed with Internet Explorer 9, Update Version 9.0.10
(KB2744842)"
justify=center "Menu Bar is present, but no additional bars or toolbars"
justify=center "Web Page displayed on wide screen monitor with resolution 1920 X 1080"
justify=center "Creation and Viewing Environment is &sysscp &sysscpl"
justify=center
  "The environment toolbar occupies the bottom of the monitor display area,"
justify=center "but is omitted in this screen capture.";
run; quit;

ODS HTML CLOSE;

ODS LISTING;

%mend LiveAreaTestWebPage;

%LiveAreaTestWebPage(
xpixels=1876
,ypixels=888
,TitleLine2=%str(Open space at left, bottom, & right edges is approximately equal));

%LiveAreaTestWebPage(
xpixels=1885
,ypixels=895
,TitleLine2=
  %str(Open space at bottom & right edges is effectively, if not actually, zero));
```

APPENDIX B. Set-Up Code for SGPLOT Time Series Graphs Using CITIDAY Data Set

```
proc format library=work;
  value MonthNm
    1 = 'January'
    2 = 'February'
    3 = 'March'
    4 = 'April'
    5 = 'May'
    6 = 'June'
    7 = 'July'
    8 = 'August'
    9 = 'September'
   10 = 'October'
   11 = 'November'
   12 = 'December';
run; quit;

libname CITIHELP "D:\SASHELP Sample data sets from SAS ETS";

/* There are five cyclic data sets of financial and economic data,
   citiday, citiwk, citimon, citiqtr, and citiyr,
   that are shipped when your site licenses SAS/ETS.
   If you do not have SAS/ETS, SAS Technical Support can tell you
   how to download them. */

data work.DowByDayIn1990;
keep Year Month Day Dow date;
format Dow 5.;
set CITIHELP.citiday(keep=date snydjcm where=(snydjcm ne .));
Year = year(date); if Year EQ 1990;
if 1988 LE year LE 1991;
Month = month(date);
Day = day(date);
Dow=round(snydjcm,1);
run;

proc means data=work.DowByDayIn1990 min max noprint;
var Dow;
output out=minmax min=DowMin max=DowMax;
run;

data _null_;
set minmax;
call symput('Ymin_1990',trim(left(put(DowMin,5))));
call symput('Ymax_1990',trim(left(put(DowMax,5))));
run;
```

APPENDIX C. Set-Up Code for SGPLOT Time Series Graphs Using CITIMON Data Set

```
/* Style-related code below has nothing to do with use of CITIMON data set per se */

%macro RemoveCraphAreaFrameFromStyle
(ParentStyle=,StyleWithNoFrame=);
PROC TEMPLATE;
DEFINE STYLE &StyleWithNoFrame;
  PARENT=&ParentStyle;
  CLASS GRAPHWALLS / FRAMEBORDER=OFF;
  /* remove a useless box around the bars */
END; RUN;
%mend RemoveCraphAreaFrameFromStyle;

%RemoveCraphAreaFrameFromStyle
(ParentStyle=Styles.htmlblue
,StyleWithNoFrame=Styles.htmlblueWithNoFrame);

/* Style-related code above has nothing to do with use of CITIMON data set per se */

proc format library=work;
  value MonthAbbrev
    1='Jan' 2='Feb' 3='Mar' 4='Apr' 5='May' 6='Jun'
    7='Jul' 8='Aug' 9='Sep' 10='Oct' 11='Nov' 12='Dec';
run; quit;

libname CITIHELP "D:\SASHELP Sample data sets from SAS ETS";

/* There are five cyclic data sets of financial and economic data,
citiday, citiwk, citimon, citiqtr, and citiyr,
that are shipped when your site licenses SAS/ETS.
If you do not have SAS/ETS, SAS Technical Support can tell you
how to download them. */

data work.SandPindexByMon1988to1991;
keep Year Month SandPindex date;
format SandPindex 5.;
set CITIHELP.citimon(keep=date FSPCOM where=(FSPCOM ne .));
Year = year(date);
if 1988 LE Year LE 1991;
Month = month(date);
SandPindex=round(FSPCOM,1);
run;
```