

**Visual + Detail = Effective Communication:
Web-enabled Graph + Spreadsheet,
Using SAS/GRAPH®, ODS, PROC PRINT, and Excel**
LeRoy Bessler, Assurant Health, Milwaukee, Wisconsin, USA

Abstract

This is a tutorial on communication-effective design, construction, and delivery of visual reports, for new or experienced users of either Version 9.1.3 or Version 9.2 of SAS/GRAPH.

Graph = quick, easy inference

Precise Numbers = correct inference

Excel = data recipient self-directed post-processing

For some graphs, SAS/GRAPH can automatically provide readable precise values for the graphic elements. For others, there is a possibility of label overlap. Plot point labels can collide with each other or with the plot line(s). Pie slice labels can overlap with each other. In a dense graph, labels might have to be too small to be readable. After 29 years of addressing this challenge of delivering both visual and detail, I've come up with my favorite solution.

Data created with SAS software is frequently packaged in a spreadsheet for users who want to manipulate it further with Excel.

You can tap the power of SAS/GRAPH and ODS to deliver your SAS data in a web-enabled graph linked to a SAS-created spreadsheet. The web-enabled graph includes a hyperlink to the spreadsheet. Furthermore, the web-enabled graph provides pop-up text so that the user can read off EXACT values for plot points, rather than estimate them, without being forced to jump over to the spreadsheet. Finally, the spreadsheet is hyperlinked back to the graph.

Another option covered is to imbed the graph and its companion table in the same Excel worksheet. However, for a web-deployed reporting package of multiple graphs, with drillable menus and possible links between graphs, the two-part cross-linked solution for visual plus detail is the better choice.

The step-by-step presentation of coding for the Web-enabled Graph + Spreadsheet assumes no prior experience with SAS/GRAPH or ODS. The paper also includes past alternatives developed in the quest to find the best way to deliver a trend plot.

Introduction

When data is prepared with SAS software, the ultimate user often insists on having it delivered via an Excel spreadsheet. Then, its next destination might be a regrettable Excel graph.

This understandable fixation with getting data in Excel inspired me to solve a longstanding problem in a new way, or at least in a way new to me.

The problem is this.

Graphs facilitate and accelerate communication and decisions, but detail is needed for reliable communication and decisions. What is the best way to deliver both a graph and the supporting detail? Here I insist that the detail be delivered in a way that is both precise and dependably readable. Furthermore, the graph and the supporting detail must be optionally printable.

To accomplish this, I decided to use Excel as part of the solution.

The structure of the paper is as follows.

I open with a brief discussion about design, which I have covered in greater depth, with abundant examples, in prior papers. Then I move on to the main purpose of this paper.

I present my preferred solution, with pictures, the code used, and some comments. After that, I proceed stepwise, with explanation, through development of the result, for users who are not familiar with the coding used.

I follow that with some enhancements to the basic preferred solution.

Then I present an alternate solution.

I close with pictures of other of my past adventures in use of PROC GPLOT, with references to papers that contain the code for them.

Communication-Effective Design – A Too Short Introduction

Software defaults are designed to give you results, not necessarily the best results, or the results that you really want. Someone else's idea of what is communication-effective might not agree with your own standards of communication effectiveness, and, as implemented, might actually be counter to the intended objective.

The Power to Show™

Show them what is important. That is one of my personal aphorisms, which has been used more than once in the titles of some of my papers.

I believe in The Power of Simplicity™.

My guiding principle in design is elegance—

Everything that is needed, but only what is needed.

One thing that some people seem to fancy are background images, or textured backgrounds, or color gradient backgrounds. Or bar charts where the bars are composed of little images, rather than simple solid color bars. These techniques and the like confuse the possible with the necessary.

There is no benefit to area fill under a plot line. If the graph is a multi-line plot, one plot line's area fill has the potential to overlay part of one or more other plot lines.

Make all text and numbers readable: sufficient size with high contrast on a plain background.

Your graph should not be a vision test. Use big fonts wherever practical.

If you are using color, realize that if lines or characters are too thinly drawn, their color is hard to distinguish. Similarly, plot point markers must be sufficiently big. One of the communication defects in Excel is that the color swatches in legends are so small that some colors are hard to distinguish, which defeats the purpose of the legend. The sizes of the color swatches in Excel are not controllable. In SAS/GRAPH, everything is controllable. (We enjoy the mixed blessing of what I call "Options Over-choice".)

If your graph does not require color (e.g., to distinguish different plot lines, pie slices, or bars), the best color combination is black and white. For more about communication-effective color combinations or communication with color in other respects, see Reference 4, my most comprehensive paper on this topic.

If possible, make every graph title a headline—TELL viewers the implication or revelation, rather than only what variables are being graphed. However, this principle is not exemplified in this paper.

Make a reasonable best effort to keep all of your text horizontal.

We read (English) text left-to-right, not bottom-to-top. I think that this slide image makes the point well:

Axis Labels Are Rarely Needed

- Put the info in graph title or subtitle
- Avoid use of vertical anti-readable label for the vertical axis—a common way try to save display space width is with vertical text
- We read left-to-right, not bottom-to-top

Is text at left OK?

Tell me the difference between it and vertical labels.

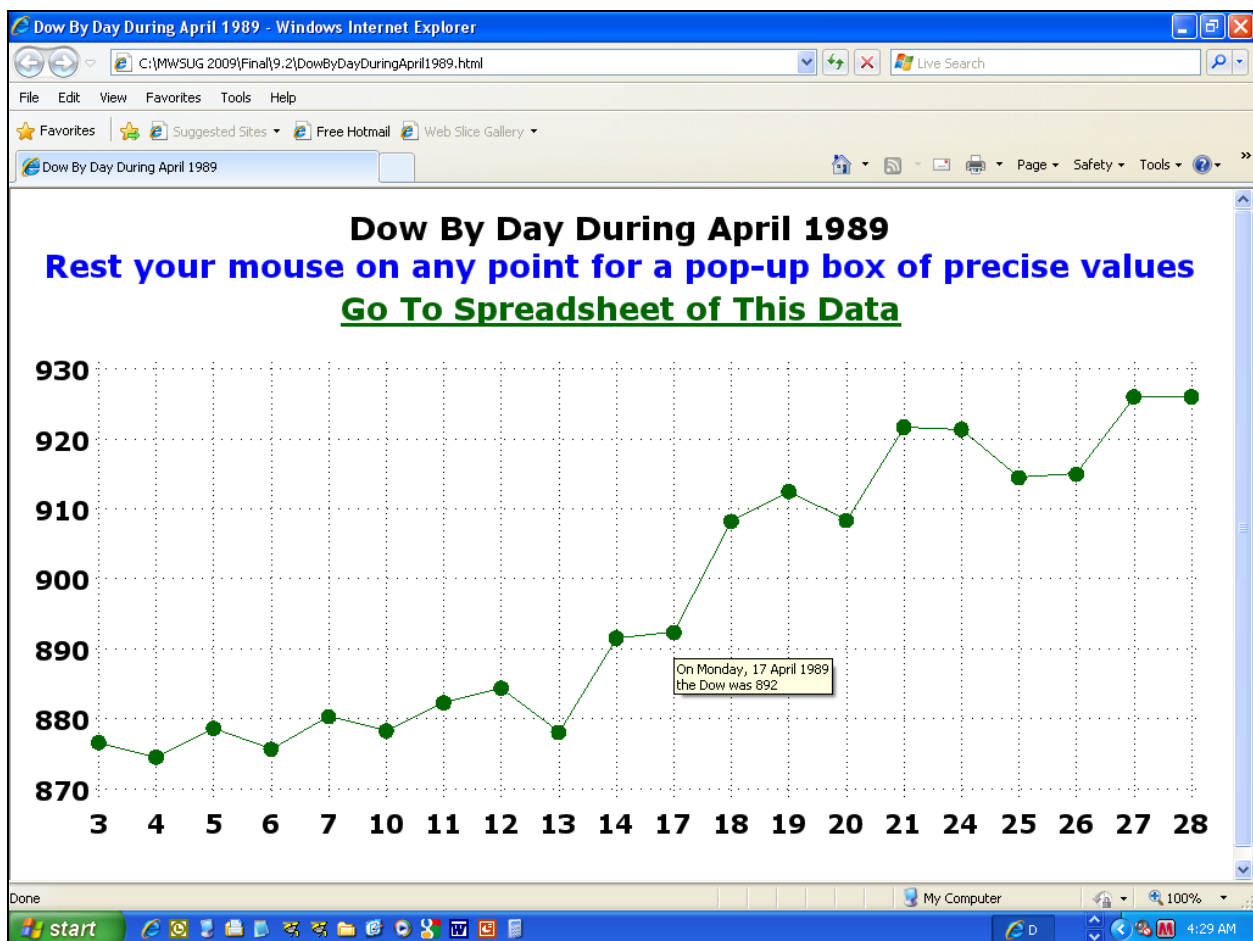
Also, I do not recommend vertical or tilted labels for horizontal axis tick marks.

Preferred Solution - Overview

I will show you how to package your results so that the user can view it as a maximally communication-effective web graph, but can toggle to a spreadsheet to inspect the supporting data as a unit, and manipulate it further non-graphically with Excel. For someone wanting data in Excel, but appreciating a better and finished graph, this is the best deliverable.

Web deployment permits imbedding hyperlinks in a graph, plus ALT text (also variously called pop-up text, flyover text, floatover text, hover text, or a tool tip). A graph most likely to benefit from ALT text is a plot. Even with tick marks and/or reference/grid lines, it is impossible to know the precise coordinates for a plot point. With ALT text, simply resting the mouse on a plot point displays its x and y values to any precision you choose to provide, in any format you like, and with whatever description you like for the x and y variables. (For bar charts and pie charts where, IF you use my design for them, ALT text is never necessary to get at the precise detail, see my other work.)

Here is the graph web page, with the mouse resting on a point:



Here is an enlargement of the ALT text that popped up:

On Monday, 17 April 1989
the Dow was 892

HTML does not allow specification of font, size, or color for ALT text, only line breaks.

Here is the spreadsheet after being linked to from the graph:

DowByDayDuringApril1989.xls - Microsoft Internet Explorer

File Edit View Insert Format Tools Data Go To Favorites Help

Address C:\FolderForMW5UGpaper1\DowByDayDuringApril1989.xls

Obs	Date	Dow
1	Monday, 3 April 1989	877
2	Tuesday, 4 April 1989	875
3	Wednesday, 5 April 1989	879
4	Thursday, 6 April 1989	876
5	Friday, 7 April 1989	880
6	Monday, 10 April 1989	878
7	Tuesday, 11 April 1989	882
8	Wednesday, 12 April 1989	884
9	Thursday, 13 April 1989	878
10	Friday, 14 April 1989	892
11	Monday, 17 April 1989	892
12	Tuesday, 18 April 1989	908
13	Wednesday, 19 April 1989	913
14	Thursday, 20 April 1989	908
15	Friday, 21 April 1989	922
16	Monday, 24 April 1989	921
17	Tuesday, 25 April 1989	915
18	Wednesday, 26 April 1989	915
19	Thursday, 27 April 1989	926
20	Friday, 28 April 1989	926

Here is an enlarged picture of the upper left corner of the spreadsheet:

	A	B	C	D
1		Dow By Day During April 1989		
2		Go To Graph of This Data		
3				
4	Obs	Date of Observation	Dow	
5	1	Monday, 3 April 1989	877	
6	2	Tuesday, 4 April 1989	875	

Why the titles must start in Column B, rather than Column A, I do not know.

Here is the code used:

```
DATA work.ToReport
  (KEEP=date snydjcm Day AltTextVar);
LENGTH AltTextVar $ 200 Day $ 2;
SET sashelp.citiday (KEEP=date snydjcm
  WHERE=(YEAR(date) EQ 1989
    AND MONTH(date) EQ 4));
AltTextVar='alt="On ' ||
  TRIM(LEFT(PUT(date,weekdatx.))) || '0D'X ||
  'the Dow was ' ||
  TRIM(LEFT(PUT(snydjcm,5.))) || '""';
/* '0D'X forces a line break */
Day = PUT(DAY(date),2.);
RUN;

%LET Path = C:\MWSUG 2009\Final\9.2;
%LET Title = Dow By Day During April 1989;
%LET GraphDescription = &Title;
%LET BodyFileName = %SYSFUNC(COMPRESS(&Title));
%LET GraphFileName = %SUBSTR(&BodyFileName,1,8);
/* See note later about filename length */

ODS NORESULTS;
ODS LISTING CLOSE;
GOPTIONS RESET=ALL;
GOPTIONS DEVICE=GIF; /* In V9.2,
  ODS HTML would select PNG as default. */
GOPTIONS TRANSPARENCY;
/* LET web page background show through. */
/* PNG does not support transparency. */
GOPTIONS XPIXELS=980 YPIXELS=525; /* size to fit in a 1024 X 768 screen
  with an extra tool bar using Internet Explorer 8 as the web browser */
GOPTIONS FTEXT='Verdana/Bold' HTEXT=4PCT;

PROC CATALOG CAT=work.gseg KILL; RUN; QUIT;

ODS HTML PATH="&Path" (URL=NONE) BODY="&BodyFileName..html" (TITLE="&Title")
  STYLE=Styles.Minimal;
PROC GPLOT DATA=ToReport;
TITLE FONT='Verdana/Bold' HEIGHT=5PCT "&Title"
  JUSTIFY=CENTER COLOR=CX0000FF
  "Rest your mouse on any point for a pop-up box of precise values"
  JUSTIFY=CENTER COLOR=CX006600 UNDERLIN=2
  LINK="&BodyFileName..xls" 'Go To Spreadsheet of This Data';
SYMBOL1 INTERPOL=JOIN LINE=1 WIDTH=1 VALUE=DOT HEIGHT=1.5 COLOR=CX006600;
AXIS1 LABEL=NONE MAJOR=NONE MINOR=NONE STYLE=0;
PLOT snydjcm*Day / NAME="&GraphFileName" DESCRIPTION="&GraphDescription"
  HTML=AltTextVar NOFRAME VAXIS=AXIS1 HAXIS=AXIS1
  AUTOVREF LAUTOVREF=33 WAUTOVREF=1
  AUTOHREF LAUTOHREF=33 WAUTOHREF=1;
RUN; QUIT;
ODS HTML CLOSE;
```

```

GOPTIONS RESET=ALL;
/* Prior GOPTIONS can affect PROC PRINT. */

/* Use COLSPAN= so that TITLE lines do not get put only into column 1 and expand its
width to length of longest TITLE line. Spread them across four columns, which expand,
if needed */

ODS HTML PATH="&PATH" (URL=NONE)
  BODY="&BodyFileName.xls";
TITLE JUSTIFY=LEFT "<td COLSPAN=4>&Title</td>"
  JUSTIFY=LEFT "<td COLSPAN=4><a href='"
    "&BodyFileName.html"
    "'>Go To Graph of This Data</a></td>";
PROC PRINT DATA=ToReport LABEL;
VAR date snydjcm;
LABEL snydjcm='Dow';
FORMAT date weekdatx. snydjcm 5.;
RUN;
ODS HTML CLOSE;

```

Version 9.2 was used here. The results for 9.1.3 would be identical except that the fonts would not be as thickly drawn, and these HEIGHT=1 plot symbols would be smaller. If you do not specify DEVICE=, then ODS HTML in 9.1.3 selects GIF, but in 9.2 selects PNG, which does not support transparency, which is used here to let the web page background show through the graph. Also, Version 9.2 permits thickening reference lines. However, line type 33 becomes very thick short dashes at a thickness of 2.

Preferred Solution – Step by Step

If I have done this right, this section should require absolutely no prior knowledge of or experience with SAS/GRAPH or ODS. Not only is there an explanation of every piece of code used in the above section Preferred Solution – Overview, but also coding starts with the defaults and evolves to the final customization. The graph and spreadsheet results are presented at steps in that evolution.

The first step is to get the input data:

```
DATA work.ToReport
  (KEEP=date snydjcm Day AltTextVar);
LENGTH AltTextVar $ 200 Day $ 2;
/* Defining AltTextVar longer than it needs to be will do no harm.
   The Alt Text pop-up box will be sized only as big as it really needs to be. */
SET sashelp.citiday (KEEP=date snydjcm
  WHERE=(YEAR(date) EQ 1989
    AND MONTH(date) EQ 4));
AltTextVar='alt="On ' ||
  TRIM(LEFT(PUT(date,weekdatx.))) || '0D'X ||
  'the Dow was ' ||
  TRIM(LEFT(PUT(snydjcm,5.))) || '""';
/* '0D'X forces a line break. */
/* You can do line breaks,
   but HTML does not support font, size, or color for ALT text. */
Day = PUT(DAY(date),2.);
/* Day is numeric day of the month.
   There will be gaps since these are trading days,
   not every possible calendar in the month. */
RUN;
```

Now I want to set up some symbolic variables, after first introducing the concept.

Symbolic variables are also known as macro variables. They allow you to define parts of your program at the top (or to package your program inside a macro, which is outside the scope of this paper). If you want to adapt the program to a new use, or just change it for a rerun, any changes that are supported by set-up at the top are easier to accomplish.

Symbolic variables are assigned, e.g., with %LET, as shown below.

To check them, you can use %PUT to get a display in your SAS log. To see every macro variable that you have assigned so far in a program or session, use %PUT _USER_;

Before doing the real set-up, consider the following example. If you use a symbolic variable to define a title (or anything else whose use requires quotes), the TITLE statement that references the symbolic variable for all or part of its content must use double quotes, not single quotes.

```
%LET YourSymVar = Whatever you like;  
TITLE "&YourSymVar";
```

at run time becomes

```
TITLE "Whatever you like";
```

but

```
TITLE '&YourSymVar';
```

at run time remains

```
TITLE '&YourSymVar';
```

With %LET the symbolic variable content assigned begins with the first non-blank and ends with the last non-blank. To provide leading or trailing blanks, use %str with %LET as in:

```
%LET SomeTextWithBlanksLeftAndRight = %str(  Whatever is your text  );
```

Here is the symbolic variable set-up for web and graphs.

```
%LET Path = C:\MWSUG 2009\Final\9.2;
/* Will use this folder for all output */

%LET Title = Dow By Day During April 1989;
/* Will use this for plot title in the graph and
   for web page title at the top of the browser window */

%LET GraphDescription = &Title;
/* Will be the pop-up description when resting the mouse on the web graph,
   but not on a plot point */

%LET BodyFileName = %SYSFUNC(COMPRESS(&Title));
/* Remove blanks to avoid problems for links.
   Will be the filename for the web page with file extension html. */

%LET GraphFileName = %SUBSTR(&BodyFileName,1,8);
/* Will be the filename for the graph with file extension gif */
```

Now more set-up for web and graph.

```
ODS NORESULTS;  
  /* At end of run, do NOT automatically open the web page  
    in SAS Display Manager or Enterprise Guide. */  
  
ODS LISTING CLOSE;  
  /* We are not producing any text output at this phase. */  
  
GOPTIONS RESET=ALL;  
  /* Turn off all pre-existing GOPTIONS.  
    Also turns off any TITLE or FOOTNOTE statements. */
```

**NOTE: If you are using SAS Enterprise Guide,
go to Tools > Options > Results > Result Formats
and turn off HTML and all other Result Formats.**

```
GOPTIONS DEVICE=GIF; /* In V9.2,  
  ODS HTML would select PNG as default. */  
  
GOPTIONS TRANSPARENCY;  
  /* LET web page background show through. */  
  /* PNG does not support transparency. */  
  
GOPTIONS XPIXELS=980 YPIXELS=525; /* Size to fit in a 1024 X 768 screen  
  with an extra tool bar using Internet Explorer 8 as the web browser.  
  This assures no need to scroll. */  
  
GOPTIONS FTEXT='Verdana/Bold' HTEXT=4PCT;  
  /* Font and Size of any text in graph for which they are not explicitly assigned,  
    or for which no direct, explicit controls are provided in SAS/GRAPH. */
```

Verdana is my recommended sans serif font. It is best for small to medium text. Georgia is my recommended serif font. It is useful for large text. These fonts were developed for Microsoft expressly for web and other non-print display readability.

If using the GIF (or PNG) device driver, the font need not be available on the displaying PC: the font is used by SAS/GRAPH to draw the image. In effect, it is imbedded.

```
/* Setting the height of text in SAS/GRAPH */
```

I recommend using PCT (percent) of graph area height with

```
goptions htext=N.NNPCT;
```

or

height=N.NNPCT on other SAS/GRAPH statements.

This removes dependence on the unknown algorithm used to translate point size to displayed size, or on knowing the number of rows in the graph area.

```
PROC CATALOG CAT=work.gseg KILL;
RUN; QUIT;
```

The above code **PREVENTS** other messages during reruns of your code during the same session, **BUT** the **FIRST** run of your code will produce **THESE** messages in your SAS log:

```
ERROR: Catalog WORK.GSEG does not exist.
WARNING: Command CATALOG not processed because of errors noted above.
```

To create a web page in the folder previously defined with a %LET statement for Path, use this code:

```
ODS HTML PATH="%Path" (URL=NONE)
  BODY="%BodyFileName..html" (TITLE="%Title")
  STYLE=Styles.Minimal;

/* You can put any SAS/GRAPH (or reporting) PROC step here. */

ODS HTML CLOSE;
```

The URL=NONE assures that the web page and graph are portable. They can be moved to a different folder and/or can be attached to an email message to be filed off by the recipient, and the graph will still display in the web page.

You can omit STYLE= , or specify any style that you prefer.

Recall that symbolic variable BodyFileName was defined earlier during the web and graph set-up. You might wonder why the double dot .. in:

```
BODY="%BodyFileName..html"
```

The first dot signals the end of the symbolic variable. The second dot is needed, as usual, between filename and file extension. The resolved result at execution time is:

```
BODY="WhateverWasYourAssignedBodyFileName.html"
```

Without the second dot, the resolved result would be:

```
BODY="WhateverWasYourAssignedBodyFileNamehtml"
```

Here is the minimum code to produce a graph, using the previously supplied set-up:

```
PROC GPLOT DATA=ToReport;
TITLE; /* no title */
SYMBOL1 INTERPOL=JOIN
  LINE=1 /* type of line, 1 is default, 1 = solid line */
  WIDTH=1 /* width of line, 1 is same as default */
  VALUE=DOT HEIGHT=1.5
  COLOR=CX006600; /* browser-safe green */
PLOT snydjcm*Day /
  NAME="%GraphFileName"
  DESCRIPTION="%GraphDescription"
  HTML=AltTextVar;
RUN; QUIT;
```

Before displaying the disappointing result, I want to discuss some of the code above.

If you omit the QUIT statement, the graph PROC will not terminate, and the consequences might be not be what you expect. The reason for the QUIT statement is to allow you to do something like this:

```
PROC GPLOT ...;  
< some statements in common here >  
PLOT ...;  
RUN;  
PLOT ...;  
RUN;  
QUIT;
```

The SYMBOL statement defines all aspects of the plot line, not just the plot markers. There is a POINTLABEL option available on the SYMBOL statement, but the labels for the points are not reliably located off the plot line. See Reference 2 for a collision-resistant alternative to the POINTLABEL option.

I want to explain the purpose of the parameters in the PLOT statement that were assigned via symbolic variables:

```
PLOT snydjcm*Day /  
    NAME="&GraphFileName"  
    DESCRIPTION="&GraphDescription"  
    HTML=AltTextVar;
```

The resolved value of &GraphFileName is used as the FileName for the GIF file, and is also used as the Name of the graph in the graph catalog, which is a piece of SAS/GRAPH infrastructure that you rarely care about.

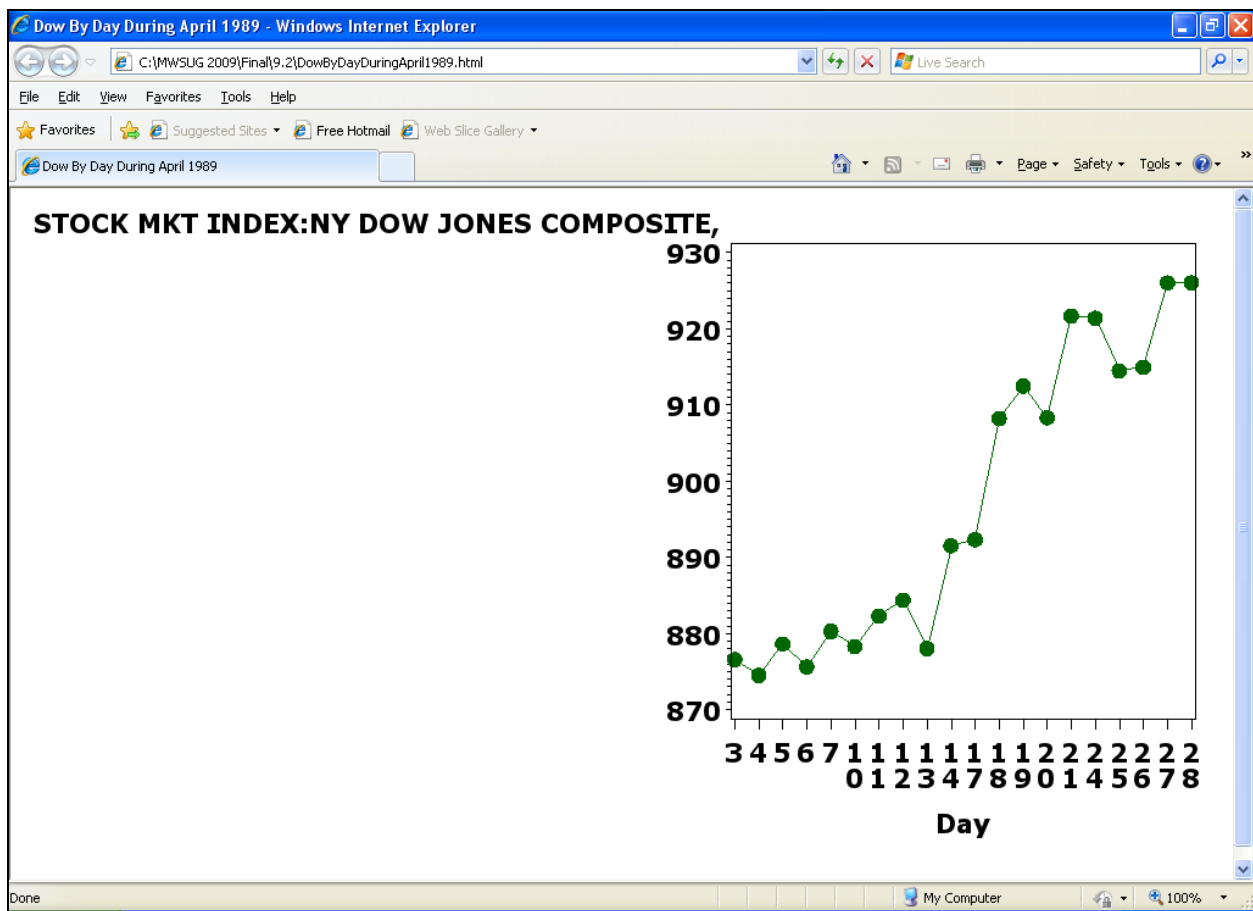
The resolved value of &GraphDescription is used, for a web graph, as the pop-up text description when you rest the mouse anywhere in the field of the graph, other than on a plot point, for which the value of that point's AltTextVar serves as the pop-up text. Though not the case in this example, AltTextVar can also contain HTML for a hyperlink to any web page that is deemed relevant.

For your possible future reference, the generic form of the PLOT statement for a multi-line plot is:

```
PLOT yVariable*xVariable=zVariable / . . . ;
```

where zVariable is a categorical variable. For each distinct value of zVariable, it is necessary to specify a SYMBOLN statement (unless you want to take the defaults). You can use N sequence suffix of the SYMBOLN statements to correlate to the sort sequence of the values of the zVariable. E.g., for a three-line plot where zVariable has values of Oranges, Apples, and Bananas, their associated SYMBOL statements would be SYMBOL3, SYMBOL1, and SYMBOL2.

Here is a picture of what the code presented so far can create:

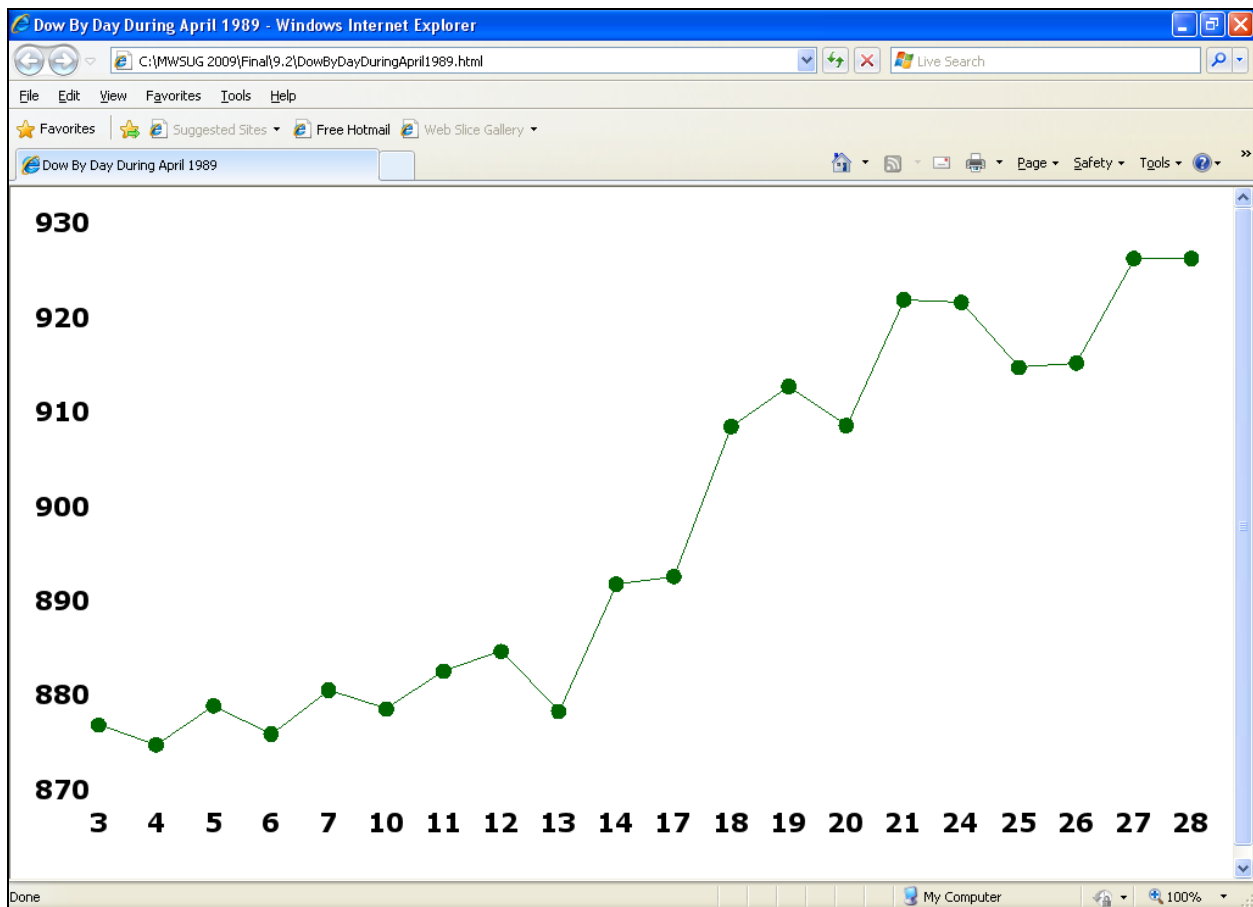


The most obvious problem with this graph is the default decision to display the data set's label for the y-variable as an axis label. Describing what is being plotted, on both axes, can easily be handled in the graph title or subtitle. More often than not, a graph's design will waste the always limited display space to repeat the same information both in the title or the subtitle and in superfluous axis labels.

Let's remove useless axis paraphernalia:

```
PROC Gplot DATA=ToReport;  
TITLE;  
SYMBOL1 ...;  
  
AXIS1 LABEL=NONE /*          no axis label */  
      MAJOR=NONE MINOR=NONE /* no tick marks */  
      STYLE=0; /* line style 0: no axis line */  
  
PLOT snydjcm*Day / NAME="&GraphFileName"  
      DESCRIPTION="&GraphDescription"  
      HTML=AltTextVar NOFRAME /* no graph frame */  
      VAXIS=AXIS1 HAXIS=AXIS1;  
  
RUN; QUIT;
```

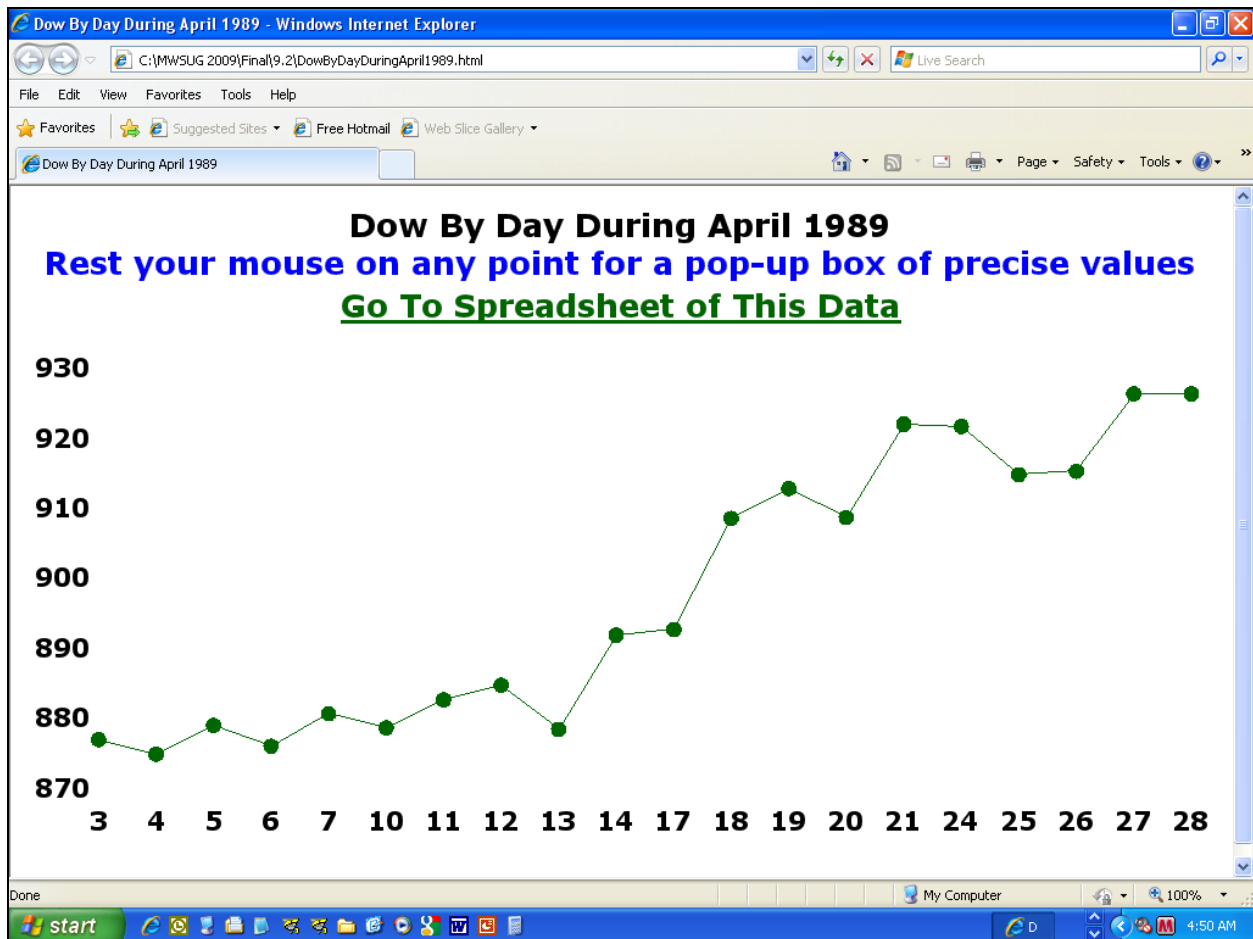
The suppressed graph frame would be on the top and the right side only. The horizontal and vertical axes, which are also being suppressed here, would supply the bottom and left side of the frame.



Now supply the graph title and a hyperlink to the companion spreadsheet:

```
PROC GGPLOT DATA=ToReport;  
  
TITLE FONT='Verdana/Bold' HEIGHT=5PCT "&Title"  
      JUSTIFY=CENTER COLOR=CX006600  
      'Rest your mouse on any point for ...'  
      JUSTIFY=CENTER COLOR=CX0000FF UNDERLIN=2  
      LINK="&BodyFileName.xls"  
      'Go To Spreadsheet of This Data';  
  
      /* JUSTIFY=CENTER forces a link break */  
  
SYMBOL1 ...;  
AXIS1 ...;  
PLOT ...;  
RUN; QUIT;
```

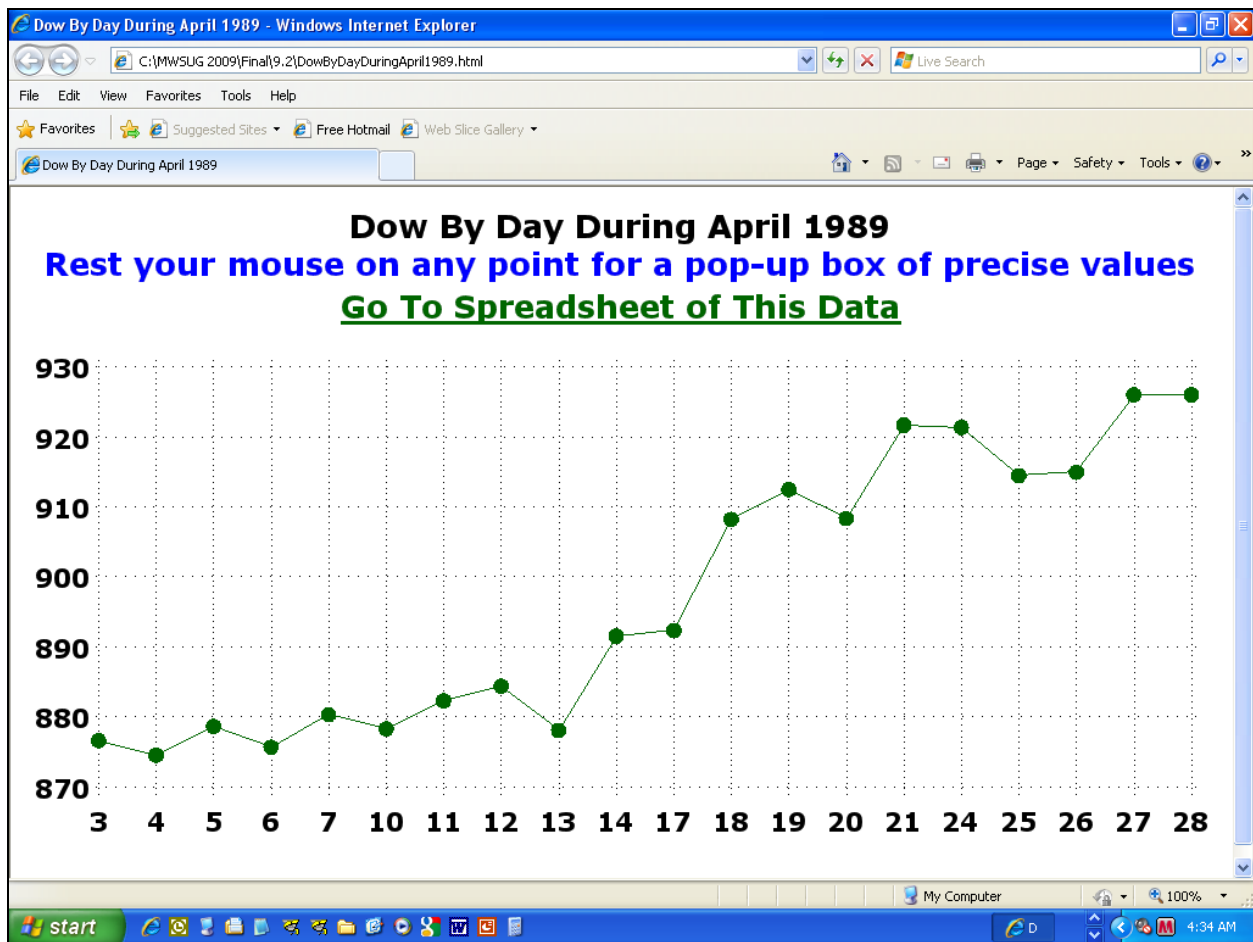
Since the hyperlink is inside the graph area, a visit to the link does not change the color displayed inside the graph, upon return to the graph.



I am adding reference lines, for demonstration purposes, but they are not really needed. The viewer can use pop-up text for precise detail, rather than try to estimate point values based on the grid.

```
PLOT snydjcm*Day / NAME="&GraphFileName" DESCRIPTION="&GraphDescription"
  HTML=AltTextVar NOFRAME VAXIS=AXIS1 HAXIS=AXIS1
  AUTOVREF LAUTOVREF=33 /* for vertical axis */
  AUTOHREF LAUTOHREF=33; /* horizontal axis */
```

AUTOVREF and AUTOHREF produce a reference line at every major vertical and horizontal axis tick mark, even if the tick mark and tick mark value are suppressed. Line type 33 is the finest dotted line. Line type 35 produces equally fine dots, but with the widest spacing for that dot size.



NOTES: In Version 9.2, WAUTOREF is available to specify reference line width. Why one would want to make reference lines thicker is not clear to me, and providing an option to make them more conspicuous is curious in light of a concurrent decision by SAS software developers to make their color less conspicuous. In Version 9.2, if you use STYLE=Styles.Default (or omit to specify STYLE=), the default color of reference lines is very light grey (nearly white). You can overcome that by, in the PLOT statement, adding:

```
CAUTOVREF=BLACK CAUTOHREF=BLACK
```

Now create the companion spreadsheet with:

```
GOPTIONS RESET=ALL;
/* Prior GOPTIONS can affect
   non-SAS/GRAPH PROC output. */

ODS HTML PATH="&Path"
  (URL=NONE) /* portable web page, graph, & hyperlink */
  BODY="&BodyFileName..XLS";

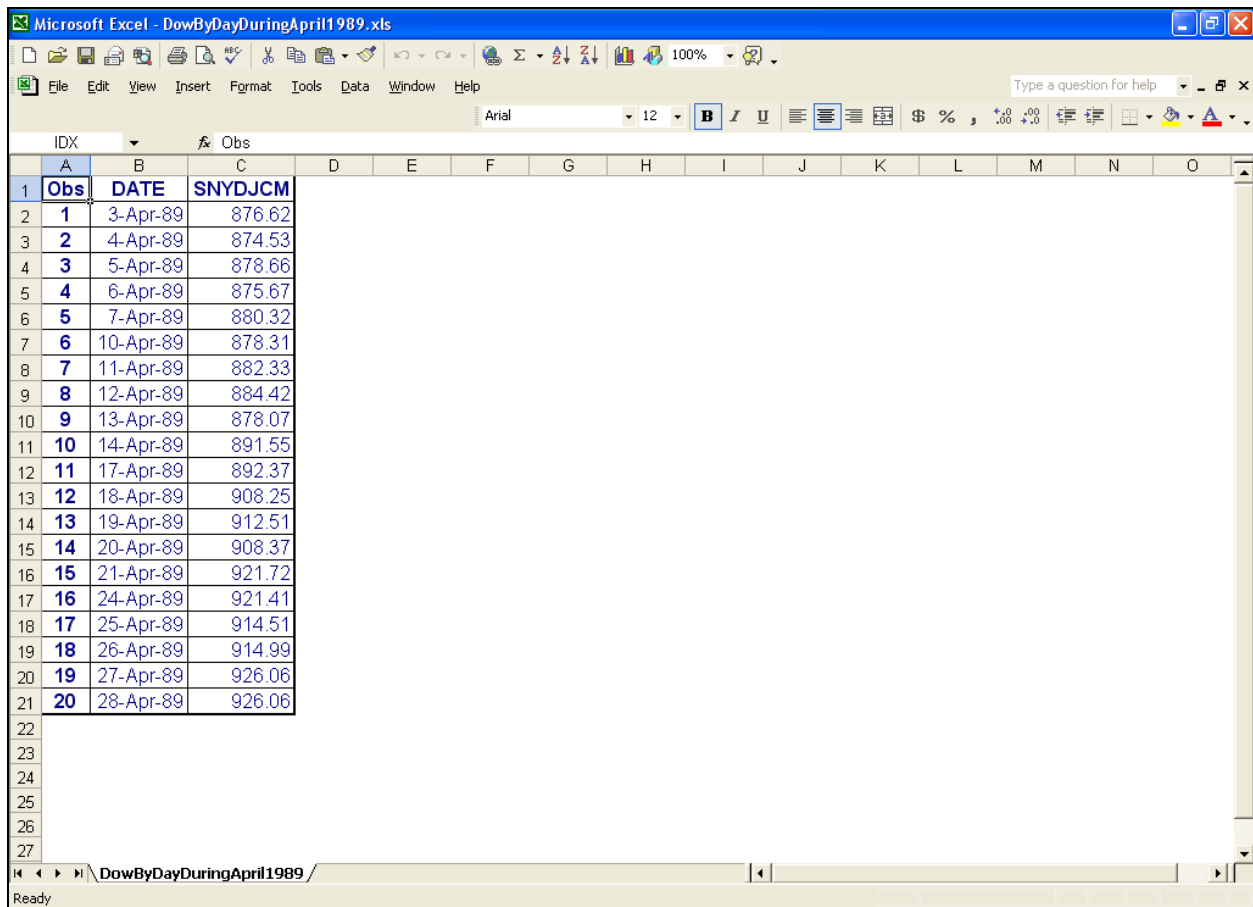
/* You can put any reporting PROC step here. */

ODS HTML CLOSE;
```

Here is the minimum PROC step to produce a spreadsheet:

```
TITLE;
PROC PRINT DATA=ToReport;
VAR date snydjcm;
RUN;
```

If these were the only columns in the rows of the ToReport table, and if their order in the source data set were as desired, then the VAR statement could be omitted.



Obs	DATE	SNYDJCM
1	3-Apr-89	876.62
2	4-Apr-89	874.53
3	5-Apr-89	878.66
4	6-Apr-89	875.67
5	7-Apr-89	880.32
6	10-Apr-89	878.31
7	11-Apr-89	882.33
8	12-Apr-89	884.42
9	13-Apr-89	878.07
10	14-Apr-89	891.55
11	17-Apr-89	892.37
12	18-Apr-89	908.25
13	19-Apr-89	912.51
14	20-Apr-89	908.37
15	21-Apr-89	921.72
16	24-Apr-89	921.41
17	25-Apr-89	914.51
18	26-Apr-89	914.99
19	27-Apr-89	926.06
20	28-Apr-89	926.06

Now customize the column labels and formats. (Use format 5. to prepare for when the Dow breaks 10000.)

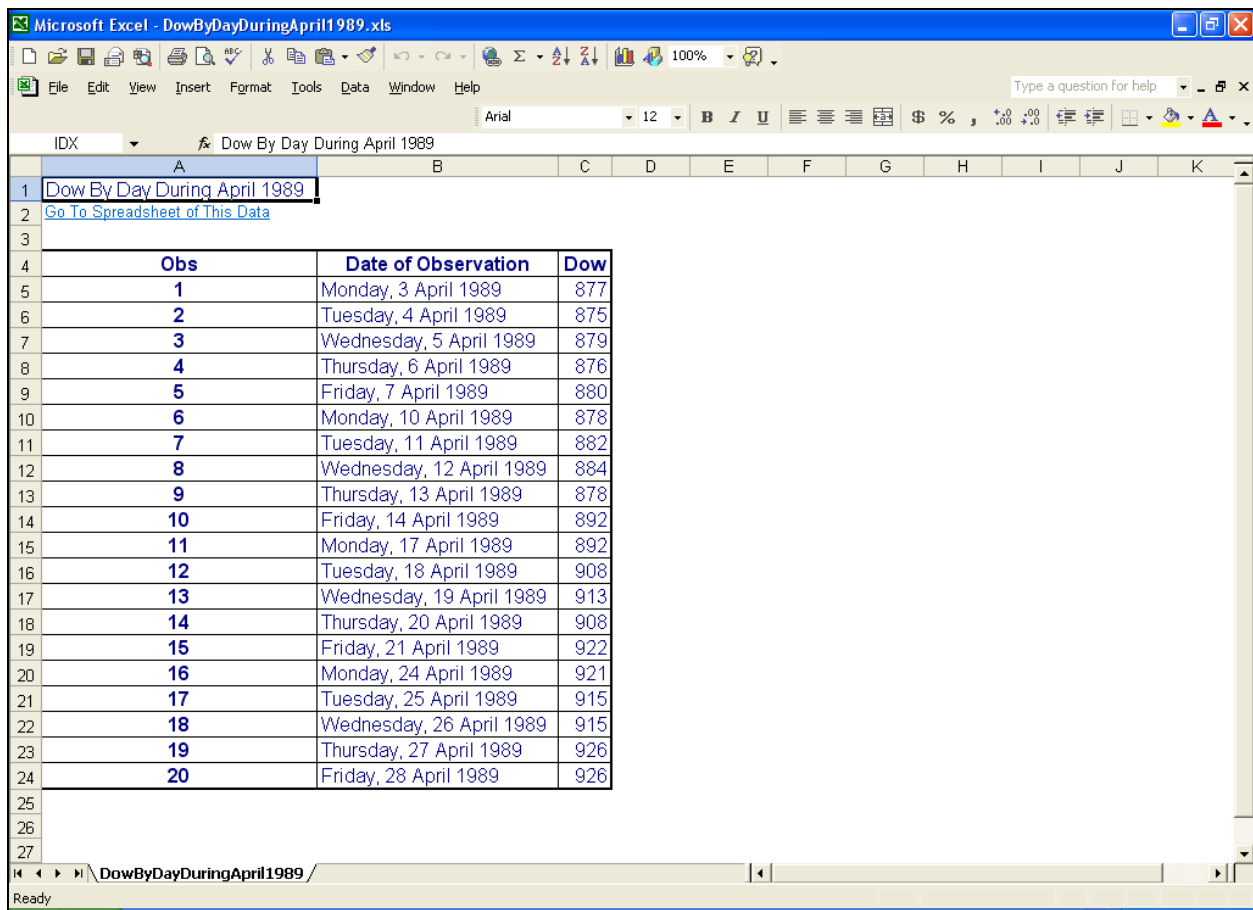
```
PROC PRINT DATA=ToReport LABEL;
VAR date snydjcm;
LABEL snydjcm='Dow';
FORMAT date weekdatx. snydjcm 5.;
RUN;
```

Obs	Date of Observation	Dow
1	Monday, 3 April 1989	877
2	Tuesday, 4 April 1989	875
3	Wednesday, 5 April 1989	879
4	Thursday, 6 April 1989	876
5	Friday, 7 April 1989	880
6	Monday, 10 April 1989	878
7	Tuesday, 11 April 1989	882
8	Wednesday, 12 April 1989	884
9	Thursday, 13 April 1989	878
10	Friday, 14 April 1989	892
11	Monday, 17 April 1989	892
12	Tuesday, 18 April 1989	908
13	Wednesday, 19 April 1989	913
14	Thursday, 20 April 1989	908
15	Friday, 21 April 1989	922
16	Monday, 24 April 1989	921
17	Tuesday, 25 April 1989	915
18	Wednesday, 26 April 1989	915
19	Thursday, 27 April 1989	926
20	Friday, 28 April 1989	926

A title and hyperlinked subtitle are supplied in an obvious way:

```
PROC PRINT DATA=ToReport LABEL;  
  
TITLE1 "&Title";  
  
TITLE2 LINK="&BodyFileName.xls"  
'Go To Spreadsheet of This Data';  
  
VAR date snydjcm;  
LABEL snydjcm='Dow';  
FORMAT date weekdatx. snydjcm 5.;  
  
RUN;
```

Here is the unexpected result:



IDX	A	B	C	D	E	F	G	H	I	J	K
1	Dow By Day During April 1989										
2	Go To Spreadsheet of This Data										
3											
4	Obs	Date of Observation	Dow								
5	1	Monday, 3 April 1989	877								
6	2	Tuesday, 4 April 1989	875								
7	3	Wednesday, 5 April 1989	879								
8	4	Thursday, 6 April 1989	876								
9	5	Friday, 7 April 1989	880								
10	6	Monday, 10 April 1989	878								
11	7	Tuesday, 11 April 1989	882								
12	8	Wednesday, 12 April 1989	884								
13	9	Thursday, 13 April 1989	878								
14	10	Friday, 14 April 1989	892								
15	11	Monday, 17 April 1989	892								
16	12	Tuesday, 18 April 1989	908								
17	13	Wednesday, 19 April 1989	913								
18	14	Thursday, 20 April 1989	908								
19	15	Friday, 21 April 1989	922								
20	16	Monday, 24 April 1989	921								
21	17	Tuesday, 25 April 1989	915								
22	18	Wednesday, 26 April 1989	915								
23	19	Thursday, 27 April 1989	926								
24	20	Friday, 28 April 1989	926								
25											
26											
27											

To prevent the anomaly above, you can revise the specification of title and hyperlink to use the HTML column spanning function:

```
TITLE1 "<td COLSPAN=4>&Title</td>";

TITLE2 "<td COLSPAN=4>
      <a href='\" \"&BodyFileName..html\" \"'>
      Go To Graph of This Data
      </a>
      </td>";
```

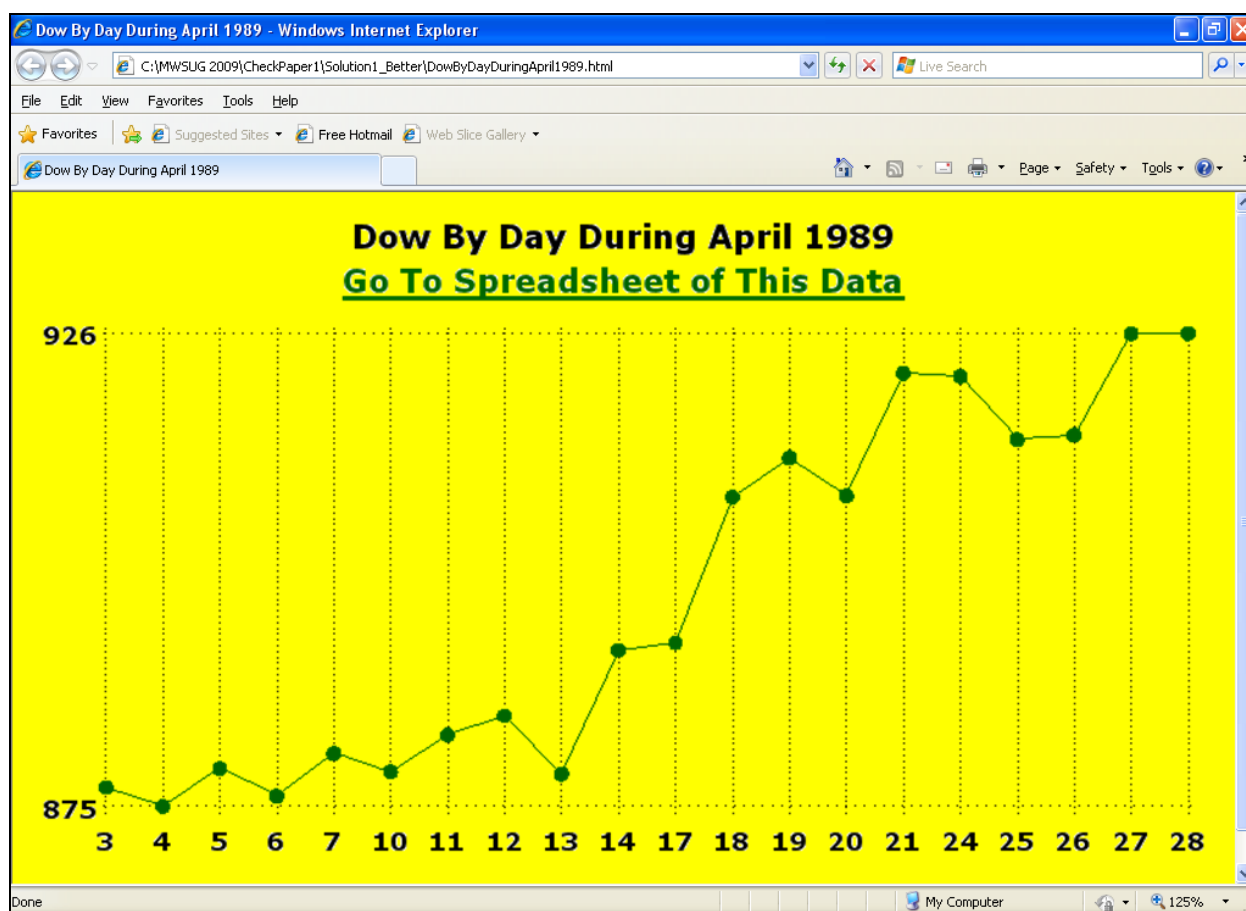
I can not understand or explain why the column spanning begins in Column B, rather than A. It has nothing to do with the fact that Column A is the Observation number variable. This presumably is either inherent to HTML, or specific to some aspect of the HTML created by ODS.

The screenshot shows a Microsoft Excel window titled "DowByDayDuringApril1989.xls". The spreadsheet contains a table with the following data:

Obs	Date of Observation	Dow
1	Monday, 3 April 1989	877
2	Tuesday, 4 April 1989	875
3	Wednesday, 5 April 1989	879
4	Thursday, 6 April 1989	876
5	Friday, 7 April 1989	880
6	Monday, 10 April 1989	878
7	Tuesday, 11 April 1989	882
8	Wednesday, 12 April 1989	884
9	Thursday, 13 April 1989	878
10	Friday, 14 April 1989	892
11	Monday, 17 April 1989	892
12	Tuesday, 18 April 1989	908
13	Wednesday, 19 April 1989	913
14	Thursday, 20 April 1989	908
15	Friday, 21 April 1989	922
16	Monday, 24 April 1989	921
17	Tuesday, 25 April 1989	915
18	Wednesday, 26 April 1989	915
19	Thursday, 27 April 1989	926
20	Friday, 28 April 1989	926

Better Version of the Trend Chart

A better solution is one where the trend chart only includes what is really needed. Most of the vertical axis tick marks and reference lines have no real practical value. The precise y values can only be obtained from the pop-up ALT text or from the companion spreadsheet. Though it is true that the precise x values can be obtained from the ALT text or the spreadsheet, the horizontal axis tick marks and reference lines do deliver the precise x values. Their retention has value. For the best trend chart, it is sufficient that the vertical axis tick marks reveal the precise minimum and maximum y values. Reference lines for them are really superfluous, but can be retained without much clutter effect as shown in the chart below.



The following additional preparatory code is required:

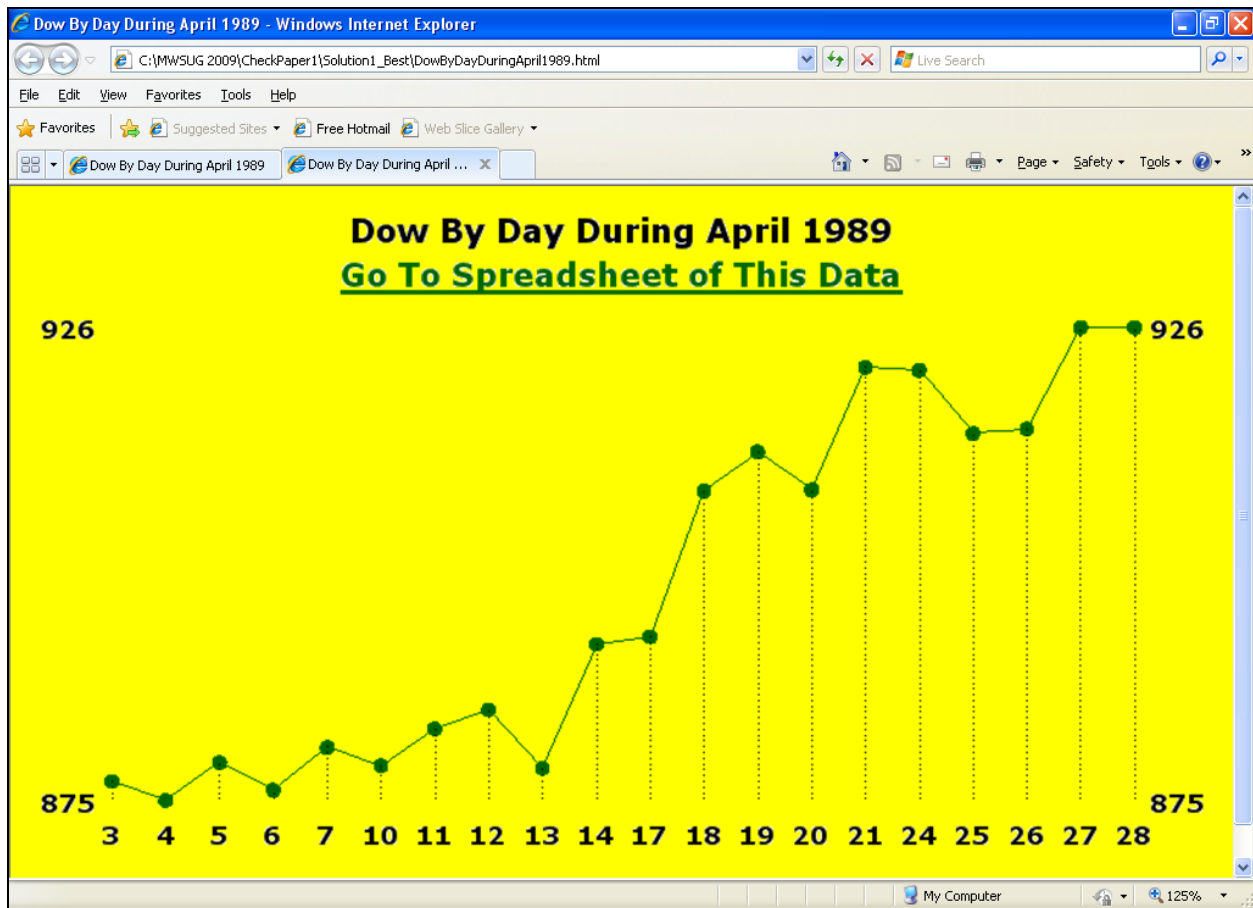
```
PROC MEANS DATA=work.ToReport NOPRINT MIN MAX;
VAR snydjcm;
OUTPUT OUT=work.MinMax MIN=MIN MAX=MAX;
RUN;
%GLOBAL Ymin Ymax;
DATA _NULL_;
SET work.MinMax;
CALL SYMPUT('Ymin',MIN); CALL SYMPUT('Ymax',MAX);
CALL SYMPUT('YminDisplay',TRIM(LEFT(PUT(MIN,5))))); /* display as integer */
CALL SYMPUT('YmaxDisplay',TRIM(LEFT(PUT(MAX,5))))); /* display as integer */
RUN;
```

The PROC GPLOT code must be changed as shown below with **bold red** text, by adding an AXIS2 statement and pointing the VAXIS parameter at it.

```
PROC GPLOT DATA=ToReport;
TITLE FONT='Verdana/Bold' HEIGHT=5PCT "&Title"
  JUSTIFY=CENTER COLOR=CX006600 UNDERLIN=2
  LINK="&BodyFileName..xls"
  'Go To Spreadsheet of This Data';
SYMBOL1 INTERPOL=JOIN LINE=1 WIDTH=1
  VALUE=DOT HEIGHT=1.5 COLOR=CX006600;
AXIS1 LABEL=NONE MAJOR=NONE MINOR=NONE STYLE=0;
AXIS2 LABEL=NONE MAJOR=NONE MINOR=NONE STYLE=0
ORDER=(&Ymin &Ymax) VALUE=("&YminDisplay" "&YmaxDisplay");
PLOT snydjcm*Day / NAME="&GraphFileName"
  DESCRIPTION="&GraphDescription"
  HTML=AltTextVar NOFRAME
VAXIS=AXIS2 HAXIS=AXIS1
  AUTOVREF LAUTOVREF=33 WAUTOVREF=1
  AUTOHREF LAUTOHREF=33 WAUTOHREF=1;
RUN; QUIT;
```


Most Elegant Version of the Trend Chart

The best solution relies on a trend chart that includes the minimum needed. The two vertical axis reference lines are superfluous. The viewer can guess that the vertical axis tick mark values are the minimum and maximum y values without these lines to make that fact unmistakable. There is no reason for the horizontal axis reference lines to stretch all the way to the top of the plot area frame. They can be replaced with needle lines that extend only from the horizontal axis to the points.



Let me explain the required code changes.

The PROC GGPLOT code must be changed as shown below. A PLOT2 statement and its SYMBOL2 statement are added. The OFFSET= is added to the AXIS1 statement for the horizontal axis. The code for the horizontal axis reference lines, which have been replaced by the needle lines, must be removed. The code for the vertical axis reference lines was also removed.

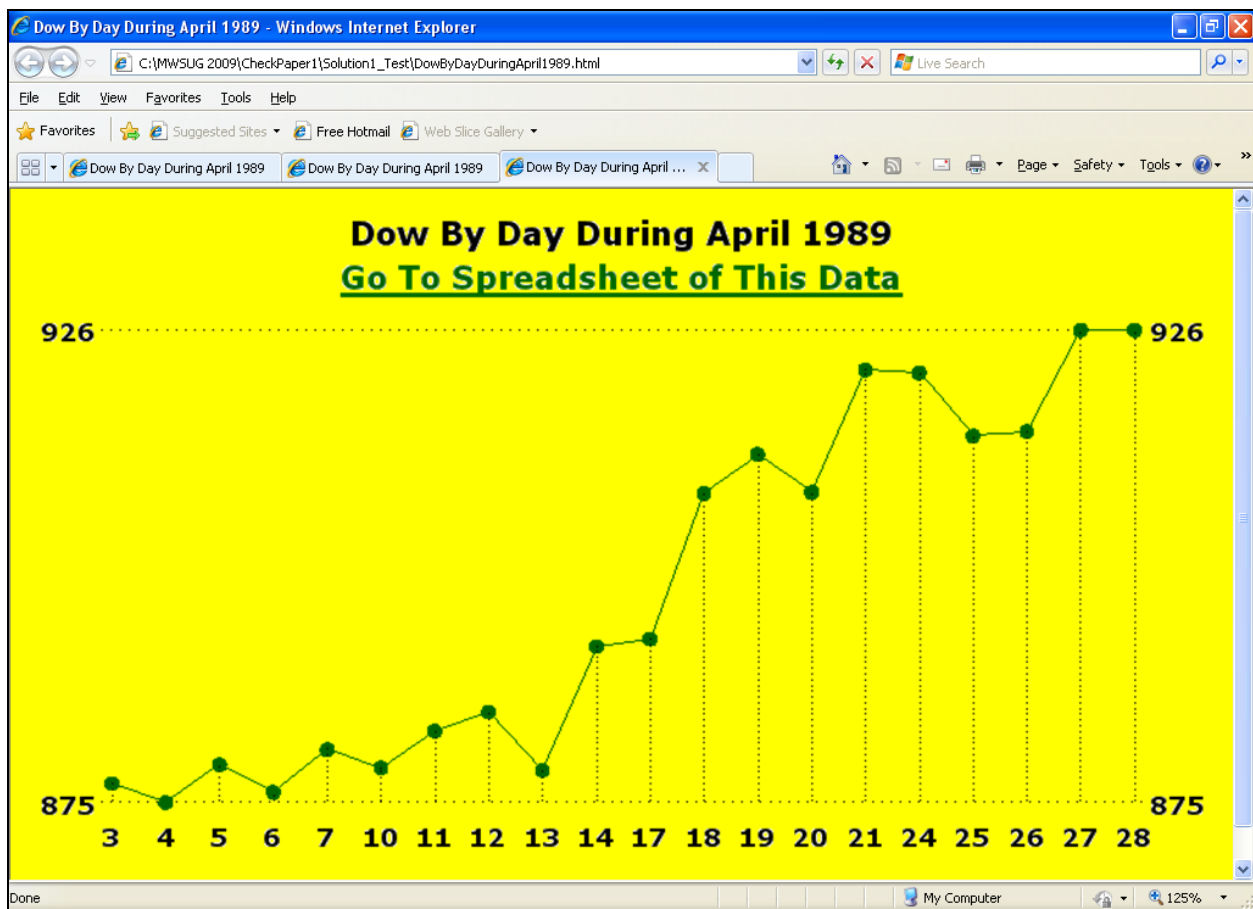
```
PROC GGPLOT DATA=ToReport;  
TITLE FONT='Verdana/Bold' HEIGHT=5PCT "&Title"  
JUSTIFY=CENTER COLOR=CX006600 UNDERLIN=2  
LINK="&BodyFileName..xls"  
'Go To Spreadsheet of This Data';  
SYMBOL1 INTERPOL=JOIN LINE=1 WIDTH=1 COLOR=CX006600 VALUE=DOT HEIGHT=1.5;
```

```

SYMBOL2 INTERPOL=NEEDLE LINE=33 WIDTH=1 COLOR=CX000000 VALUE=NONE;
AXIS1 LABEL=NONE MAJOR=NONE MINOR=NONE STYLE=0
      OFFSET=(1 PCT,1 PCT); /* With the vertical axis reference lines removed,
      the vertical axis tick mark values are displayed too close to the plot area.
      Use OFFSET to insert some space. */
AXIS2 LABEL=NONE MAJOR=NONE MINOR=NONE STYLE=0
      ORDER=(&Ymin &Ymax) VALUE=( "&YminDisplay" "&YmaxDisplay" );
PLOT snydjcm*Day / NAME="&GraphFileName" DESCRIPTION="&GraphDescription"
      HTML=AltTextVar NOFRAME VAXIS=AXIS2 HAXIS=AXIS1;
PLOT2 snydjcm*Day /
      HTML=AltTextVar NOFRAME VAXIS=AXIS2 HAXIS=AXIS1;
RUN; QUIT;

```

For this particular trend, the starting point happens to be very close to the minimum y value and the ending point happens to be at the maximum value, the y axis minimum and maximum values could be mistaken as annotation of those points rather than recognized as the bounds of the y axis. However, the presence of the maximum y value at the left side, clearly disconnected from any point, and the presence of the minimum y value at the right side, clearly disconnected from any point, should diminish the likelihood of such a misinterpretation. Nevertheless, below are the same trend chart with vertical axis reference lines restored and the code used to accomplish that.

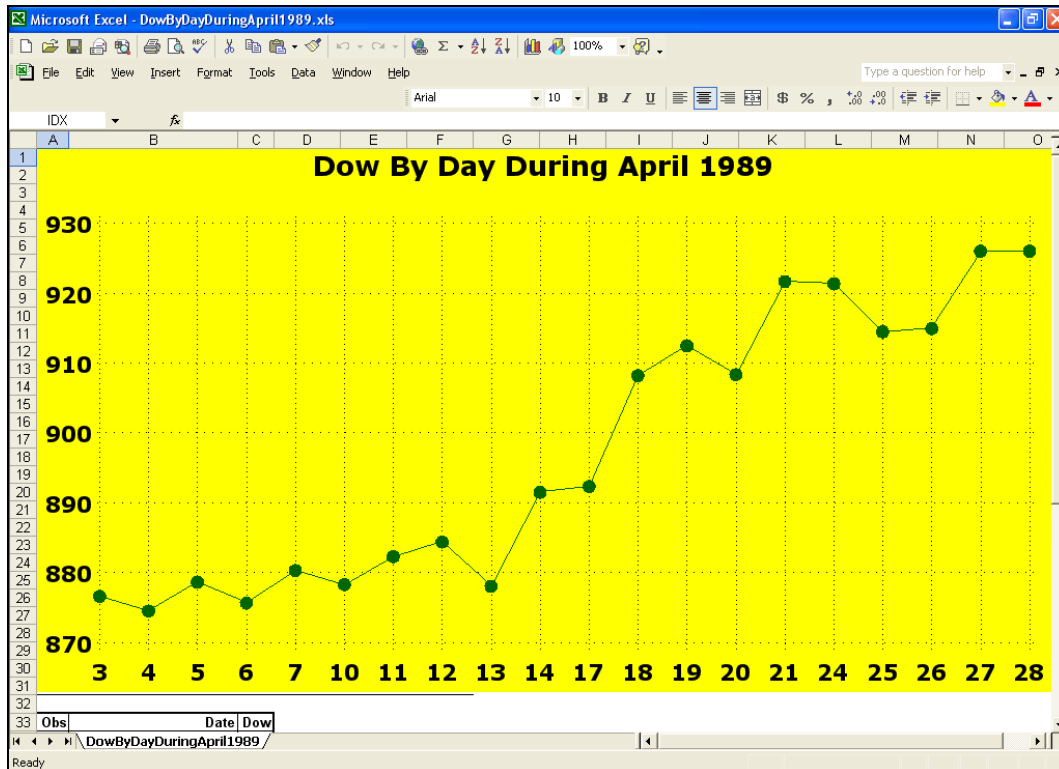


```

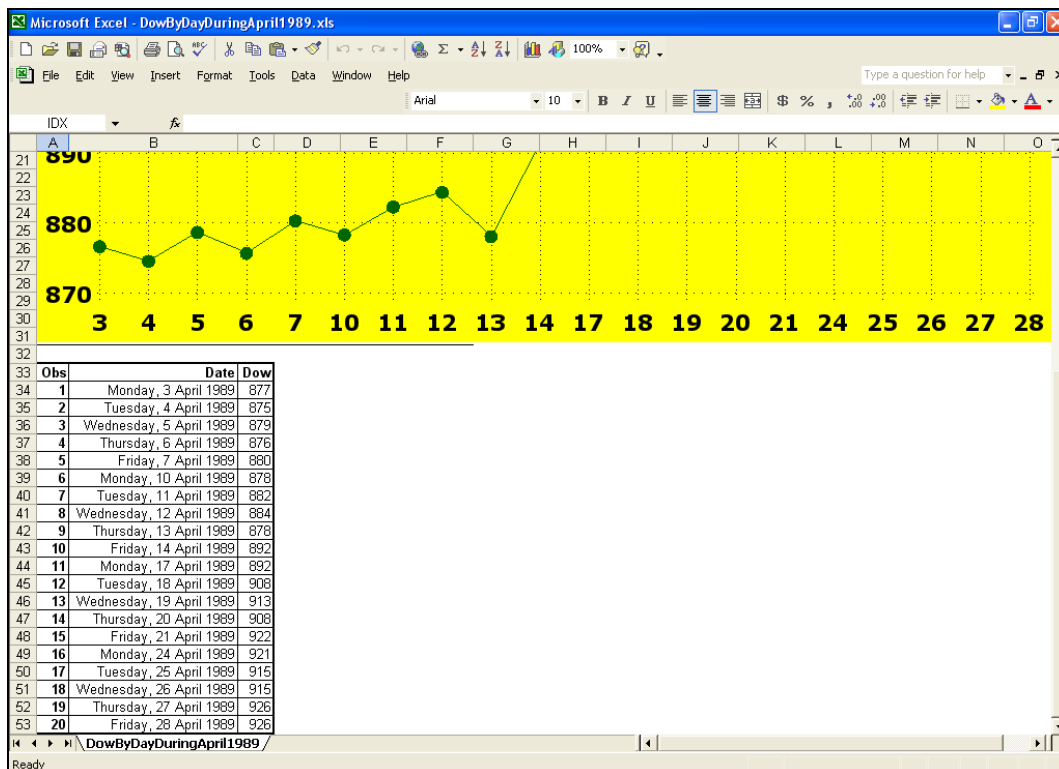
PLOT snydjcm*Day / NAME="&GraphFileName" DESCRIPTION="&GraphDescription"
      HTML=AltTextVar NOFRAME VAXIS=AXIS2 HAXIS=AXIS1 AUTOVREF LAUTOVREF=33 WAUTOVREF=1;
PLOT2 snydjcm*Day /
      HTML=AltTextVar NOFRAME VAXIS=AXIS2 HAXIS=AXIS1 AUTOVREF LAUTOVREF=33 WAUTOVREF=1;

```

Alternate Solution: Graph and Table on One Excel Worksheet



Scrolling down:



Now, I show you how to package both the graph and the table on one Excel worksheet as presented above.

If your user prefers everything in a single package and does not regard web-enabled pop-up plot point descriptions for plot points as necessary and does not appreciate the convenience of simple printability on separate pages of a stand-alone graph and a stand-alone spreadsheet, this solution will suffice.

After the code, I will explain important SAS version differences.

Here is the code used:

```
DATA work.ToReport(KEEP=date snydjcm Day);
LENGTH Day $ 2;
SET sashelp.citiday
  (KEEP=date snydjcm WHERE=(YEAR(date) EQ 1989 AND MONTH(date) EQ 4));
Day = PUT(DAY(date),2.);
RUN;

%LET Path = C:\MWSUG 2009\Final\9.2;
%LET Title = Dow By Day During April 1989;
%LET BodyFileName = %SYSFUNC(COMPRESS(&Title));

ODS NORESULTS;
ODS LISTING CLOSE;
GOPTIONS RESET=ALL;
GOPTIONS DEVICE=GIF;
GOPTIONS CBACK=CXFFFF00; /* yellow background color */
GOPTIONS FTEXT='Verdana/Bold' HTEXT=4PCT;
GOPTIONS XPIXELS=980 YPIXELS=525; /* size to fit on a 1024 X 768 screen
                                   inside an Excel 2003 window */

PROC CATALOG CAT=work.gseg KILL; RUN; QUIT;

ODS TAGSETS.MSOFFICE2K PATH="&Path" (URL=NONE)
  BODY="&BodyFileName.xls" STYLE=Styles.Minimal;
/* (URL=NONE) is used to make the worksheet and its imbedded GIF file relocatable */

PROC GPLOT DATA=ToReport;
TITLE FONT='Verdana/Bold' HEIGHT=5PCT "&Title";
SYMBOL1 INTERPOL=JOIN LINE=1 WIDTH=1 VALUE=DOT HEIGHT=1.5 COLOR=CX006600;
AXIS1 LABEL=NONE MAJOR=NONE MINOR=NONE STYLE=0;
PLOT snydjcm*Day /
  NOFRAME VAXIS=AXIS1 HAXIS=AXIS1 AUTOVREF LAUTOVREF=33 AUTOHREF LAUTOHREF=33;
RUN; QUIT;

GOPTIONS RESET=ALL;
/* Prior GOPTIONS can affect PROC PRINT. */

TITLE1;
PROC PRINT DATA=ToReport LABEL;
VAR date snydjcm;
LABEL date='Date' snydjcm='Dow';
FORMAT date weekdatx. snydjcm 5.;
RUN;

ODS TAGSETS.MSOFFICE2K CLOSE;
```

Version 9.2 was used here. The results for 9.1.3 would be identical, except that the fonts would not be as thickly drawn, and these HEIGHT=1 plot symbols would be smaller.

In both SAS 9.2 and 9.1.3, omitting STYLE=Styles.Minimal (instead accepting Styles.Default) has several noteworthy effects. In both versions, it yields colored area fills for the table, and seven useless empty rows between graph and table. If you omit GOPTIONS CBACK=CXFFFF00, which provides the yellow background color for the graph, in SAS 9.2 the graph has a light blue background, whereas in SAS 9.1.3 the graph has a white (no) background. The light blue background is due to the fact that SAS/GRAPH 9.2 uses a default graph style. All colors, fonts, symbols, and graph sizes are derived from whatever is the style, default or custom, currently in effect in your program or session. Procedure statement options and GOPTIONS can be used to override individual elements of that style. In 9.2, to revert to the simpler 9.1.3 environment, with no graph style, specify OPTIONS NOGSTYLE.

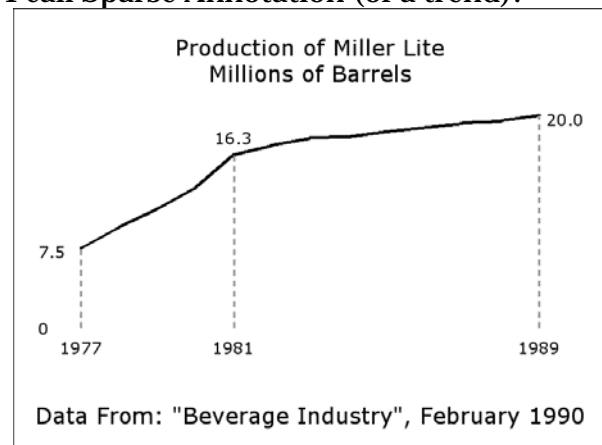
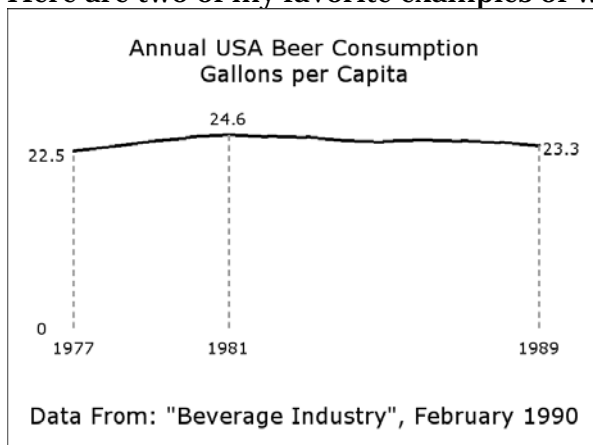
Also, in SAS 9.2 with Styles.Default the dotted reference lines are imperceptible. If you use solid reference lines by specifying LAUTOVREF=1 and LAUTOHREF=1, you will find that the lines are white (or very light grey). Thin black dotted lines are a better design choice if you want less conspicuous reference lines that are nevertheless visible. The color of the reference lines can be controlled with CAUTOVREF and CAUTOHREF.

Other Trend Plots - Examples to deliver varying levels of detail

Case 1. Sparse Annotation = Visual + Only Essential Detail for a Trend Plot

I put the word “Essential” into the title of this section for this reason. I am convinced that most of the time, there are only four points in a trend that are of real interest: Start, End, Minimum, and Maximum. There are, of course, other points of potential interest: (a) points when there was some significant environmental event which could be expected (or hoped?) to influence the trend; and (b) points not a minimum or maximum but where the trend experienced a permanent change in slope. Case (b) includes a point of inflection, of which there are four possibilities: (1) increase at an increasing rate; (2) increase at a decreasing rate; (3) decrease at a decreasing rate; and (b) decrease at an increasing rate. For purposes of computer-generated trend plots, it is trivially easy with SAS software to programmatically identify Start, End, Minimum, and Maximum. For me, at least so far, it has not been possible to programmatically discover points of inflection in trend data. I suppose that if one knew how to reliably fit a continuous function to a set of data points, there might be some way to discover points of inflection.

Here are two of my favorite examples of what I call Sparse Annotation (of a trend):



As you can guess, I coined my graphic design term “Sparse Annotation” a long time ago, but the utility of these two examples does not diminish with age. The national trend has only three points of interest: Start, End, and Maximum. The Miller Lite trend has only three points of interest: Start, End, and a point where the rate of growth permanently, at least over the range of this graph, changes.

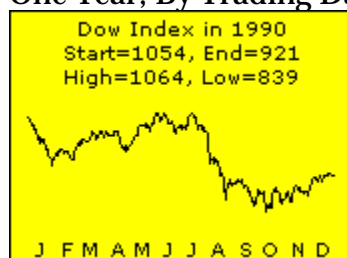
These two designs also embody one of my preferred design rules for trends: whenever your audience will let you, start the vertical axis at zero. Yes, this does diminish the viewer’s ability to discern small changes. However, at the same time it diminishes the propensity for needless elation or despair about what might be insignificant fluctuations. The flat appearance of the national trend is not a failure of graphic communication—industry observers at the time described the trend as “flat”. The best way to assess the significance of a change is to look at the precise numbers and the impact of that on the business, the organization, or the process.

You can find the code for the examples above in Reference 3.

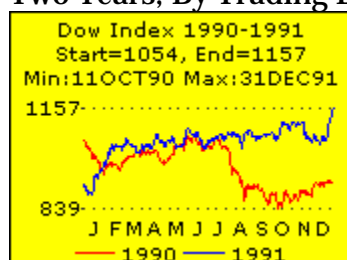
Case 2. When Only the Essential Will Fit

Two years ago I faced the challenge of presenting readable trends on the small screen—BlackBerry, iPhone, etc. The size of the graph in pixels must be limited to 172 X 129. The results are presented below, in Actual Size. Code for these examples is in Reference 4.

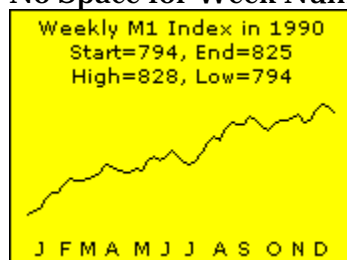
One Year, By Trading Day



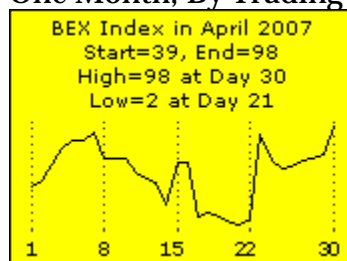
Two Years, By Trading Day



No Space for Week Numbers



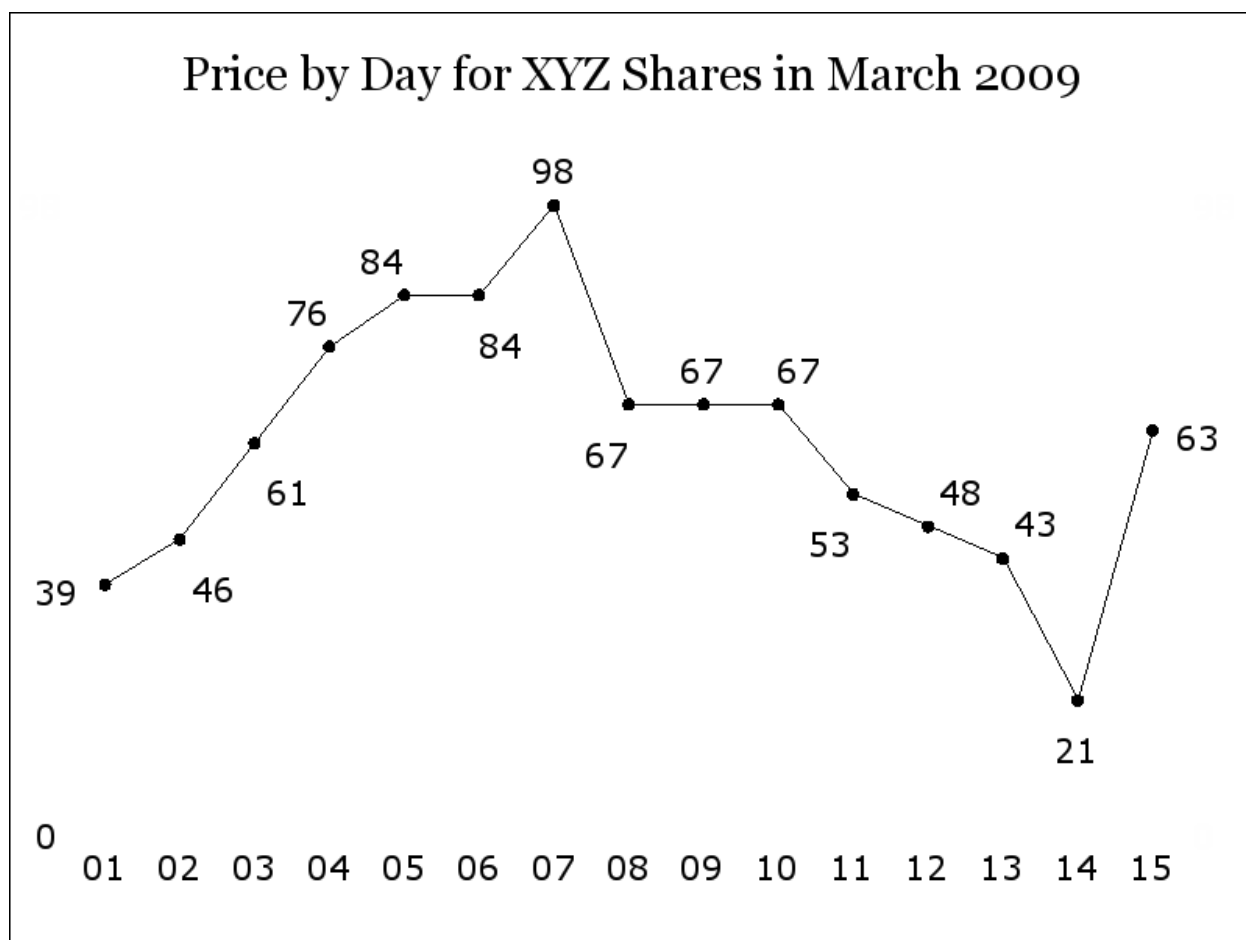
One Month, By Trading Day



Case 3. Complete Detail for a Single-line Plot

This solution is based on a macro first developed by me in 2001. One of its later publications is Reference 2. Its algorithm makes single-line plots resistant to the collisions between line or point and the annotated value to which the native POINTLABEL option on the SAS/GRAPH SYMBOL statement is vulnerable. I describe it as collision-resistant, not collision-proof, since it will break down when the plot line is too dense with points.

Here is what the result looks like when applied to a test data set contrived to produce a trend line with every possible three-point, two-segment slope transition:



For an illustration of what happens instead when using the POINTLABEL option on the SYMBOL statement, please see Reference 2.

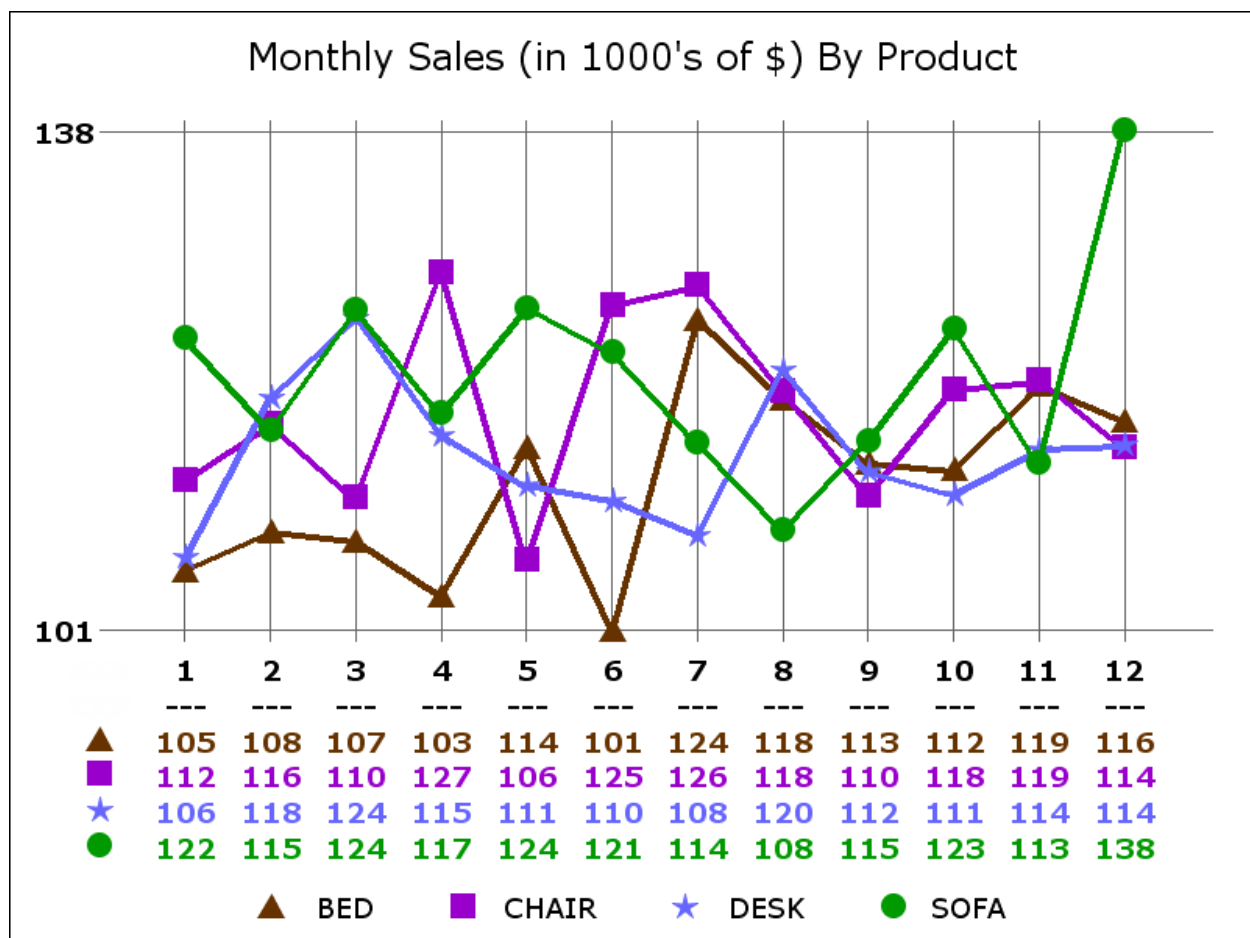
Case 4. Complete Detail for Any Multi-line Plot

This solution is complex for a multi-line plot, but simple for a single-line plot. It relies on converting the horizontal axis tick mark area into a table.

I have used it successfully with as many as 37 horizontal tick marks values—i.e., for a trend chart of three years back to the same month as ending month. I like the concept of a monthly trend chart ending on the same month as the starting year, to be able to do a month-to-same-month-N-years-ago chart.

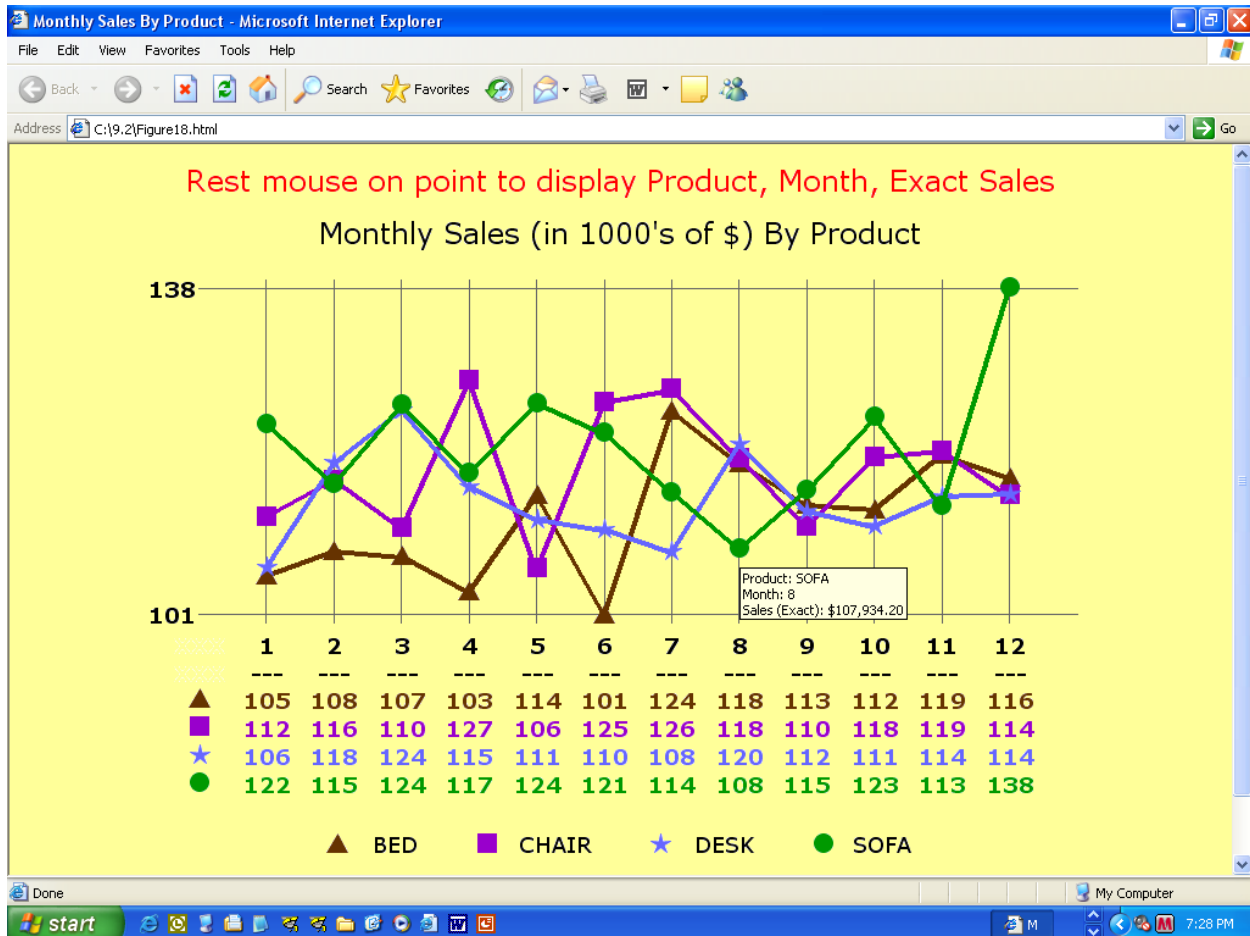
On a display, the multiple colors are another legend aid, but the differing symbols for each line's marker assure that the graph and its internal table still work if printed (or photocopied) in black-and-white.

For code to create this example see Reference 3.



Case 4A. Complete Detail for Any Multi-line Plot – Web-enabled

This is the same as Case 4, except that the background is colored, and pop-up text is available for the points. A multi-line plot like this can, of course, be cross-linked to a companion spreadsheet. For code to create this example, see Reference 3.



Bibliography of Related Work (By This Author, Unless Otherwise Noted)

1. “Communication-Effective Use of Color for Web Pages, Graphs, Tables, Maps, Text, and Print”, *Proceedings of the Twenty-Ninth Annual SAS Users Group International Conference*, SAS Institute Inc. (Cary, NC), 2004. Find it on the web at <http://www2.sas.com/proceedings/sugi29/176-29.pdf>.
2. “%TREND: A Macro to Produce Maximally Informative Trend Charts with SAS/GRAPH, SAS, and ODS for the Web or Hardcopy”, *Proceedings of the Twenty-Seventh Annual SAS Users Group International Conference*, SAS Institute Inc. (Cary, NC), 2002. Co-author: Francesca Pierri. Find it on the web at <http://www2.sas.com/proceedings/sugi27/p093-27.pdf>.
3. “Chart Smart: Design and Build SAS Graphs That Inform and Influence”, *Proceedings of Midwest SAS Users Group Conference 2007*, MWSUG (Des Moines, IA), 2007. Find it on the web at www.lexjansen.com/mwsug/2007/DataVisualization/D01-2007.pdf.
4. “SAS Graphs for a BlackBerry, iPhone, or Other Small Email Screen: Extreme SAS/GRAPH and the Necessity and Power of Simplicity”, *Proceedings of SAS Global Forum 2008*, SAS Institute Inc. (Cary, NC), 2008. Find it on the web at <http://www2.sas.com/proceedings/forum2008/034-2008.pdf>.

About the Author

A SAS user since 1978, Dr. LeRoy Bessler has shared his knowledge and experience with other users at conferences throughout the USA and in Montreal, London, Heidelberg, and Dublin. At Assurant Health, he supports the SAS users, server, software, and data, provides internal consulting and occasional training, and gets to do some programming—especially if it involves SAS/GRAPH, ODS, highly formatted Excel reporting, web information delivery, automated dynamic HTML-formatted email, or tools to support the SAS server and its users. LeRoy is a regular contributor to *VIEWS News*, an electronic web newsletter of the VIEWS International SAS Programmer Community. He has organized one regional and numerous subregional SAS users conferences in the north central USA since 1989. Next year, Dr. Bessler will serve with Craig Wildeman and Alexandra Riley as a co-chair for the Midwest SAS Users Group Conference 2010 in Milwaukee.

Contact Information, Etc.

I am always interested in user suggestions, comments, and questions about visual communication with SAS, SAS/GRAPH, and ODS.

LeRoy Bessler PhD
Le_Roy_Bessler@wi.rr.com

The Power to Show™
The Power of Simplicity™

Zum sehen geboren, zum schauen bestellt.
—Goethe

SAS is a registered trademark or trademark of SAS Institute Inc. in the USA and other countries.
® indicates USA registration. Other product and brand names are trademarks or registered trademarks of their respective owners.

The Power to Show and The Power of Simplicity are trademarks of LeRoy Bessler PhD.