Paper 234-2010

SGPANEL: Telling the Story Better

Chuck Kincaid, COMSYS, Portage, MI

ABSTRACT

SAS® has a new set of graphics procedures called Statistical Graphics. They are built upon the Graphics Template Language (GTL) in order to make the powerful GTL easily available to the user. SGPANEL is one of the procedures that can be used to produce powerful graphics that used to require a lot of work. This upgrade is similar to the ease-of-use upgrade in output manipulation when ODS was first published. This paper will introduce the reader to PROC SGPANEL and the new capabilities it provides beyond the standard plot. By using these new capabilities, any user can tell the story better.

INTRODUCTION

SAS/Graph® has been a standard in analytical reporting since the early days. Over the years it has improved in many ways. One of the recent improvements has been the addition of the Graphics Template Language (GTL). Based on the templates used for the Output Delivery System, the GTL gives users amazing control over the look and feel of the graphics that they create. However, the GTL is not necessarily for the faint of heart. To help people use this capability more easily, SAS has created the Statistical Graphics (SG) procedures: SGPLOT, SGPANEL, SGSCATTER, and SGRENDER. With thoughtful default values for color and layout, these procedures allow the user to create powerful graphics pretty easily.

The SGPANEL procedure, in particular, introduces the new capability of the *Panel*. The concept of the panel is not new and many SAS Global Forum papers have been written over the years to implement the panel concept, usually using PROC GREPLAY. SGPANEL removes the headaches that have been associated with creating multiple, related graphs. This paper will explain the concepts, the capabilities, the do's and the don'ts with using SGPANEL and particularly its PANELBY statement. Once the concepts and techniques are mastered you'll never look at graphs the same way again.

COPLOTS

WHAT IS A COPLOT?

The Coplot is a graphical device that has its origins in E. R. Tufte's *small multiples*. Tufte discusses this graphic technique in his book The Visual Display of Quantitative Information (1983). He compares them to frames of a movie: "a series of graphics, showing the same combination of variables, indexed by changes in another variable." By keeping the design of the graphic constant throughout all the frames, the viewer can focus on the changes in the data as they visually move from one frame to another.

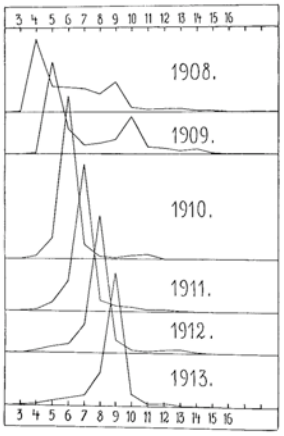


Figure 1 Tufte's small multiples example

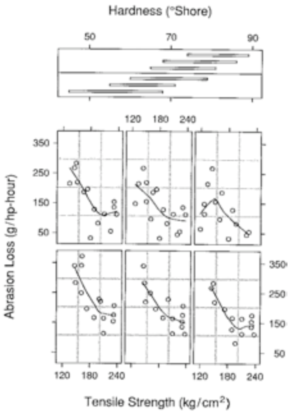


Figure 2 Cleveland's coplot example

Figure 1 above shows an example of small multiples that compares the distribution of *Age of Herring Catches* each year from 1908 to 1913. The shift of the peak (representing the tremendous number of herring that were spawned in 1904) throughout the years can be easily seen, showing the power of this type of plot.

William S. Cleveland brought his ideas together with other research and introduced the concept of the *conditioning plot* or *coplot* in Visualizing Data (1993). Figure 2 shows the first coplot Cleveland introduces. This figure has the characteristic *given panel* across the top of the graph, and the *dependence panels* in the center.

Each dependence panel graphs, in this case, two variables, *Abrasion Loss* vs *Tensile Strength* for a specific range of values for *Hardness*. One can easily see the conditional dependence of the relationship between *Abrasion Loss* and *Tensile Strength* upon *Hardness*. Most of the panels show a particular nonlinear “hockey-stick” relationship, except for the top-right panel which shows an inversion of the relationship.

Different software applications each have a different way of presenting this powerful graphing technique. In this paper we will look at what SAS does in SGPANEL and what you can do with it.

SGPANEL VS SGPLOT

SGPLOT is available for creating the typical one panel plot. It is a step above the GPLOT procedure in the way that ODS is a step above the DATA _NULL_. It is based on the Graphics Template Language (GTL) which became production in SAS 9.2. Two of the nice things about it are the wide variety of plot capabilities that can be easily implemented, and the well-thought out defaults for colors, fonts, line sizes, etc. You can find many good papers from the User Group meetings about SGPLOT and related topics. Delwiche and Slaughter (WUSS 08) have a good overview along with an example for each type of plot.

All of the plot types that SGPLOT can create, except the ELLIPSE plot, are available in SGPANEL. Not all of them work with each other, though. See the SAS Documentation for specifics.

The primary difference between SGPLOT and SGPANEL is that the former gives you one plot in one panel, while the latter gives you multiple plots laid out in multiple panels. The way you control the layout is with the PANELBY statement which we’ll go into below. Other documented differences include

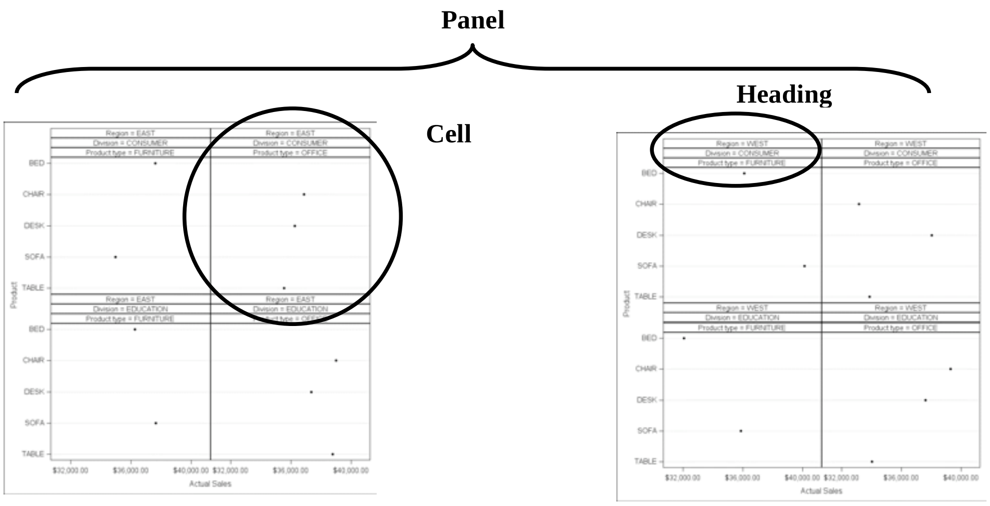
- **No TMPLOUT=”filename” option for SGPANEL.** This option writes the GTL code to a file for you to modify and use later on. However, even though it’s not documented, it does work if you use it.
- **No UNIFORM= option.** This option specifies how to control axis and markers with the BY statement. You can use the BY statement with SGPANEL and there are some times where it’s better to do so. However, the UNIFORM option is not accepted. Because of the *panel* concept, SGPANEL makes the group values and axis scaling consistent by default. However, you can control the scaling as described below.
- **KEYLEGEND.** The user has less control over the legend, with KEYLEGEND, in SGPANEL. The legend in SGPLOT can be placed either inside or outside the graph. For SGPANEL the legend is relevant to all panels, so it only makes sense to have it “outside” all of the panels, but within the plot region. Also, the possible positions for the legend in SGPANEL are BOTTOM, LEFT, RIGHT, and TOP, whereas SGPLOT can use some combinations.

SGPANEL AND PANELBY

SGPANEL displays the relationship among specified variables (cf. dependence panels, above) at given *levels* of another variable (cf. given panel, from above). The variable being conditioned is expected to be a class variable where the *levels* are its unique values. Numeric variables can be used, but the procedure will attempt to make a chart for each unique value, so be careful!

First let’s review the terminology of the SG Procedures. There are potentially three levels of SG output. (Not to be confused with levels of a class variable.) The most basic is the *cell*. The *cell* is what has been traditionally called a graph such as created with PLOT, GPLOT or even SGPLOT. The *panel* is the set of all cells that the procedure generates. Finally, the *graph* is one page of output. For example, when a panel is split across multiple pages, each page is a *graph*.

The labels at the top, in this case, of each cell are the *headings* or *header*. Figure 2 provides visual examples for each of these terms.



Page 1 == Graph 1 **Page 2 == Graph 2**

Figure 2. Terminology Example

THE PANELBY STATEMENT

The PANELBY statement is required and must be the first statement before any plot, axis or legend statements. The procedure will use what you specify in the PANELBY statement to define the overall layout of the panel. The syntax for PANELBY is

PANELBY variable(s) </ option(s)>;

option(s) can be one or more of the following:

BORDER | NOBORDER

COLHEADERPOS= TOP | BOTTOM | BOTH

COLUMNS= n

LAYOUT= LATTICE | PANEL | ROWLATTICE | COLUMNLATTICE

MISSING

NOVARNAME

ONEPANEL

ROWHEADERPOS= RIGHT | LEFT | BOTH

ROWS= n

SPACING= n

SPARSE

START= TOPLEFT | BOTTOMLEFT

UNISCALE= ROW | ALL

In this paper we will cover many of these options and explore what benefit they provide the user.

LAYOUT= LATTICE I PANEL I ROWLATTICE I COLUMNLATTICE

One of the more important options is the LAYOUT. The two primary forms of the panel layout are LATTICE and PANEL (the default). Here is an example of the PANEL layout using the Orion Sales data in SASHELP.

```
   proc sgpanel data=sashelp.orsales;
```

```
      panelby product_line;
      dot year / response=profit;
   run;
```

This default layout arranges the cells by row from left to right and top to bottom with the headings placed at the top of each cell. This layout can be controlled with the START= option below. The number of columns per row is determined by the number of levels of the grouping variable. Right now the procedure uses each level of the grouping variable to create a cell. Eventually, it may take numeric variables and create intervals that it could use for the grouping. Cleveland called this *slicing*.

The PROC tries to make intelligent decisions about how to divide the panels into rows and columns. Table 1 shows the default numbers of rows, columns and pages given the number of panels. As you can see, unless you override the number of rows or columns, the procedure keeps them to 3 and under.

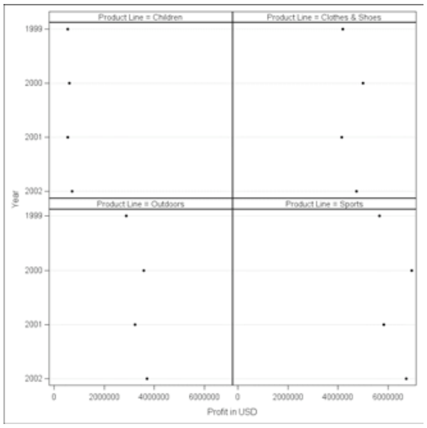


Figure 3 Default Panel Layout

Table 1 Default Layout for PANEL Option

# of Panels	# of Rows	# of Columns	# of Pages	# of Panels	# of Rows	# of Columns	# of Pages
2	1	2	1	8	2	2	2
3	2	2	1	9	1	2	5
4	2	2	1	10	1	2	5
5	2	3	1	11	2	3	2
6	2	3	1	12	2	2	3
7	1	2	4				

The layout can be controlled by the options, COLUMNS, ONEPANEL, and ROWS, as we'll see. The panel layout is useful for single PANELBY variables. The PANELBY could be used with two variables and controlling the number of columns or rows, however it's easier to do this with the LATTICE layout. The LATTICE layout uses the first variable to define the columns and the second to define the rows. Thus, if the first variable has 2 levels and the second has 3 levels, then SGPANEL will create a layout with 2 columns and 3 rows.

By default the following number of cells gives the corresponding numbers of rows, columns and pages.

Table 2 Default Layout for LATTICE Option

# of Levels 1st Var	# of Levels 2nd Var	# of Rows	# of Columns	# of Pages	# of Levels 1st Var	# of Levels 2nd Var	# of Rows	# of Columns	# of Pages
2	2	2	2	1	4	2	2	2	2
2	3	2	3	1	4	3	3	2	2
2	4	2	2	2	4	4	1	2	8
2	5	1	2	5	4	5	1	2	10
3	2	2	3	1	5	2	2	1	5
3	3	1	3	3	5	3	3	1	5
3	4	2	3	2	5	4	2	1	10
3	5	1	3	5	5	5	2	3	6

Note that certain layouts can leave some cells blank depending on the number of levels of the given variables. For example, dividing five columns into two pages will mean three columns on each page. (It tries to keep things uniform). This arrangement will then give one blank column on the second page.

As mentioned the layout can be controlled by using the COLUMNS, ROWS or ONEPANEL option in the following manner.

COLUMNS= N

ROWS= N

These options control the number of columns and/or rows in the layout. You don't have complete control with these options. The procedure will still make its own decisions about the layout if the settings don't fit well into the page. Also, these options have no effect if you use the ONEPANEL option and the LATTICE layout. Only one of them will have an effect if you use the ONEPANEL and the PANELBY layout. In that case, if you specify both COLUMNS= and ROWS=, then the COLUMNS= value will be used and the ROWS= value ignored.

ONEPANEL

The ONEPANEL option on the PANELBY statement makes the entire panel fit into a single graph, which is a single page. If the panel has many cells, the graph is going to be cluttered, so this option is typically recommended when there are only a small number of cells. However, many cells can be okay if the pattern of the relationship in each cell and across cells is what is important and not the actual relationship. An example of this from the web is by Dr. Andrew Gelman, Professor of Statistics and Political Science and Director of the Applied Statistics Center at Columbia University.

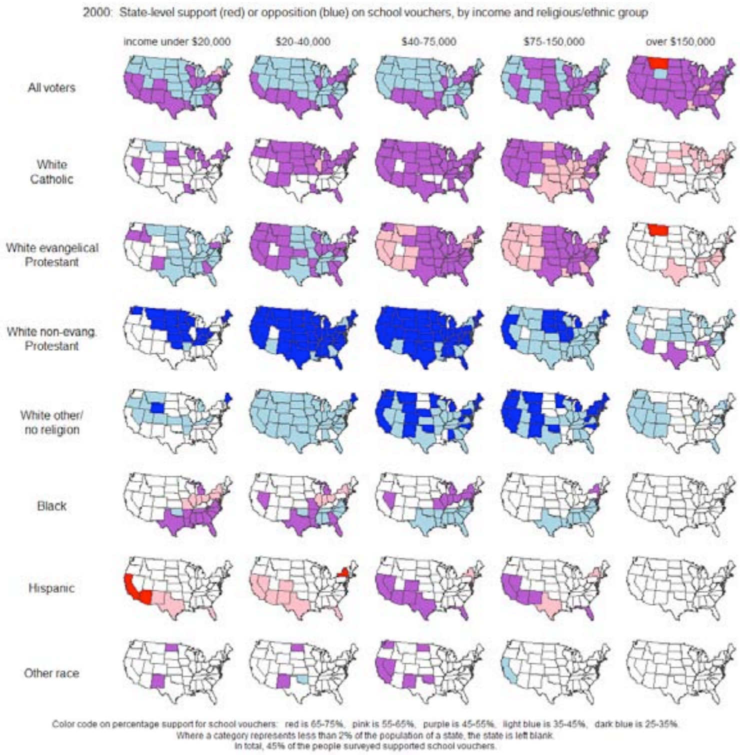

2000: State-level support (red) or opposition (blue) on school vouchers, by income and religious/ethnic group
income under $20,000
$20-40,000
$40-75,000
$75-150,000
over $150,000
All voters
White Catholic
White evangelical Protestant
White non-evang. Protestant
White other/ no religion
Black
Hispanic
Other race
Color code on percentage support for school vouchers: red is 65-75%, pink is 55-65%, purple is 45-55%, light blue is 35-45%, dark blue is 25-35%.
Where a category represents less than 2% of the population of a state, the state is left blank.
In total, 45% of the people surveyed supported school vouchers.

Figure 4 Small Multiples Example 1 (Gelman)

Notice that it's not the individual graphic that we're interested in as much as how the graphic changes as the given variables along the top and side change.

An example of this same thing with SGPANEL can be seen in Figure 5. We could even remove the tick labels here, since they are not of much interest. By using the LOESS statement with NOMARKERS, we can see the overall relationship between the Systolic and Diastolic variables for each of the levels of *Cause of Death* and *Smoking Status*. Most seem similar to each other, but there are a few that stand out.

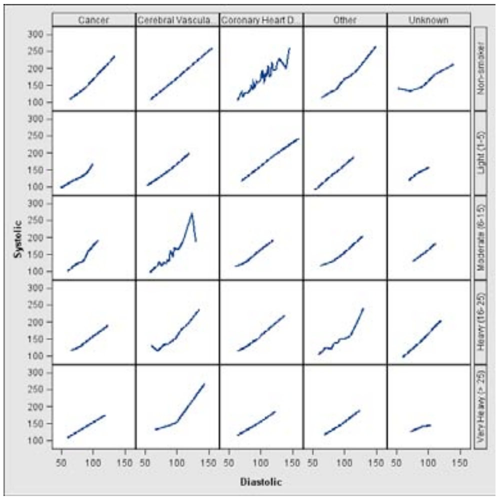


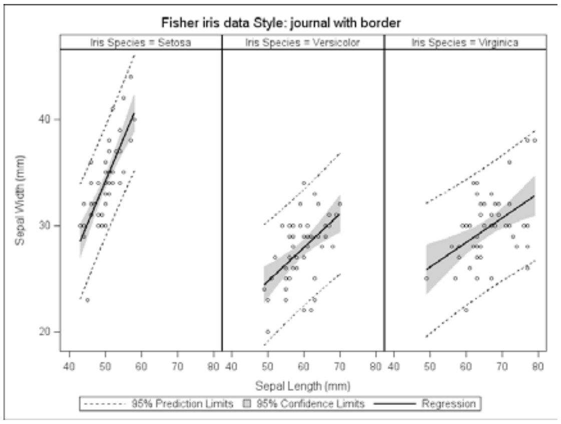
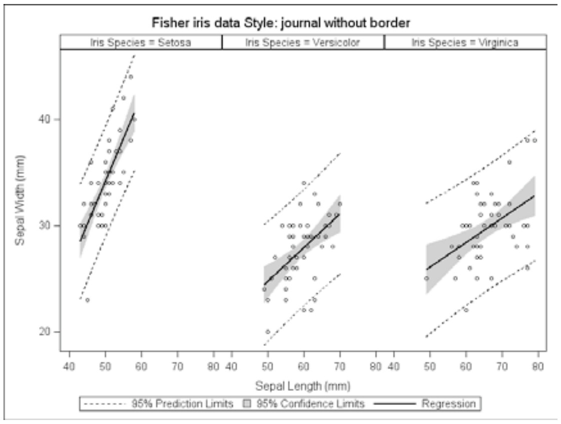
Figure 5 SGPANEL Small Multiples

When used with discretion, this option can be a valuable tool either with many or few cells in your graph. Note, however, that if the panel becomes too large for the output image, then a blank image is created. Finally, with this option and the PANEL layout, you cannot specify both the ROWS= and COLUMNS= values. If you do, then COLUMNS= will be used and ROWS= will be ignored. Neither of these, ROWS= or COLUMNS=, have an effect with ONEPANEL and the LATTICE layout.

BORDER | NOBORDER

This option is pretty straightforward depending on the ODS style in use at the time. The typical styles (Analysis, Statistical, Listing, Journal, Journal2) all have borders by default. BORDER adds borders around each <u>cell</u> in the panel and NOBORDER removes them. Note that this is around each cell and not around the graph as a whole. Here is an example of each with the *Journal* style.

Table 3 BORDER vs NOBORDER

Border with Journal Style	No Border with Journal Style
	

COLHEADERPOS= TOP | BOTTOM | BOTH

ROWHEADERPOS= RIGHT | LEFT | BOTH

These options, which (obviously) control the position of the column and row headings, can be used with the LATTICE layout. For the PANEL layout, the headings are always across the top. The default positions (TOP and RIGHT) are the most useful, because the axes can “interfere” with the BOTTOM and LEFT positions. BOTTOM and LEFT can be useful with small multiples if the axes are not displayed as we can see with the small multiples heart data example above (see Figure 6)

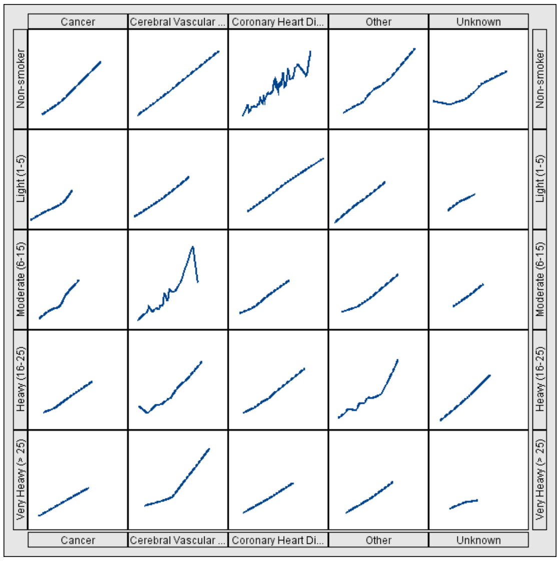


Figure 6 Cell Headers on BOTH Sides

MISSING

Most procedures by default can not use records with missing values. PROC REG, for example, will drop observations that have missing values for any of the variables in the model. However, sometimes it’s helpful to explore the behavior of the non-missing variables for these records. The missing option can help by treating the missing values of the PANELBY classification variable as a valid level and a cell will be drawn.

NOVARNAME

The NOVARNAME option is used in almost every graph the author creates, because it removes the variable name and the “=” symbol from the cell headings. This style provides clean heading labels by avoiding the redundant information. If there are two PANELBY variables that have the same levels, however, then the variable name may be necessary.

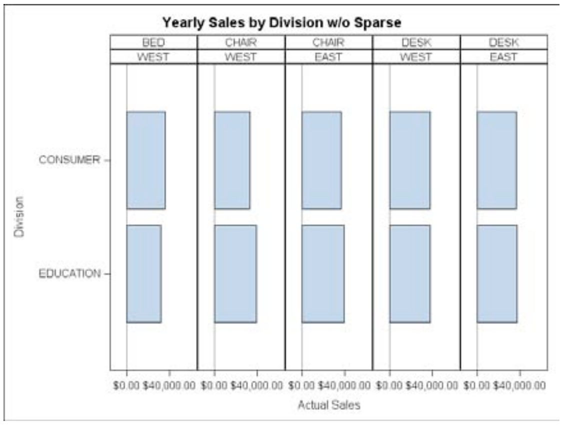
SPACING= N

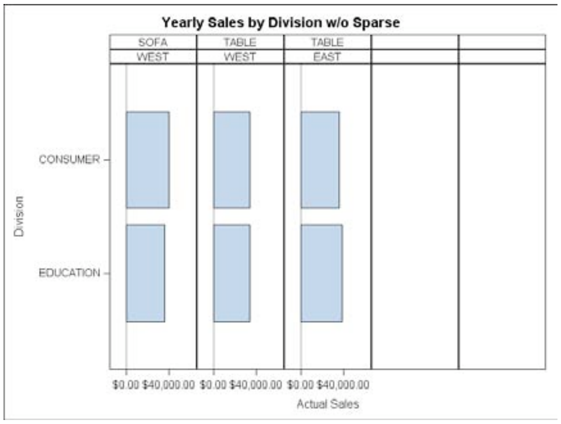
This option controls the number of pixels between the cells. The author hasn’t used this option.

SPARSE

The SPARSE option is useful when not all combinations of the PANELBY classification variables are there. Figure 7 shows our sales data with the BED & EAST and SOFA & EAST combinations deleted (PRDSALE_SUB). Without the SPARSE option the levels are all adjacent.

```
proc sgpanel data=prdsale_sub;
  title "Yearly Sales by Division w/o Sparse";
  panelby product region / novarname columns=5;
  hbar Division / response=actual;
run;
```





```
proc sgpanel data=prdsale_sub;
  title "Yearly Sales by Division w/ Sparse";
  panelby product region / sparse novarname columns=5;
  hbar division / response=actual;
run;
```

With the SPARSE option the missing combinations are included in the grid.

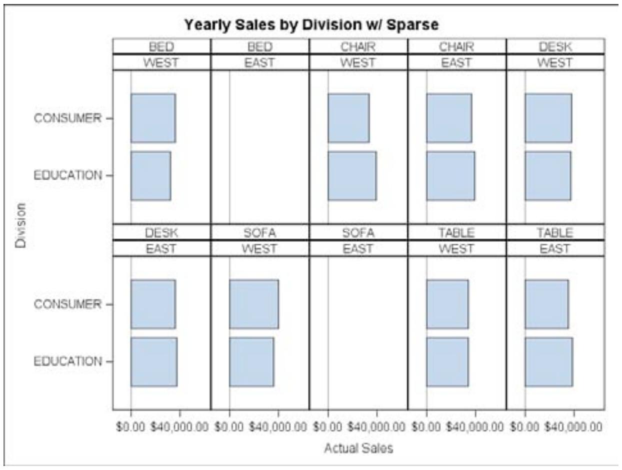


Figure 8 Missing Levels with SPARSE

This layout is similar to the LATTICE layout which shows all combinations by default.

```
proc sgpanel data=prdsale_sub;
  title "Yearly Sales by Division w/ Lattice";
  panelby product region / layout=lattice novarname columns=5;
  hbar Division / response=actual;
run;
```

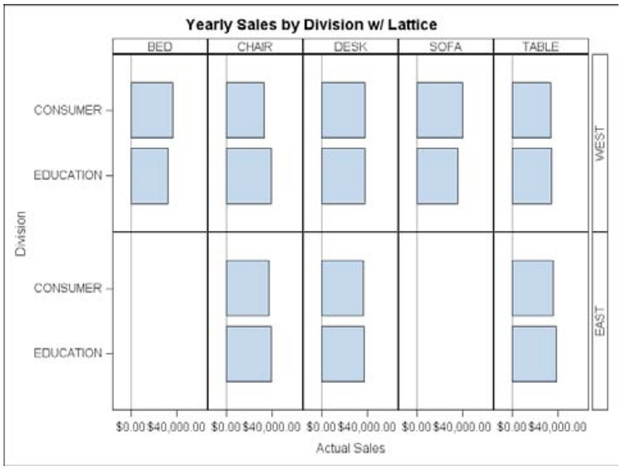


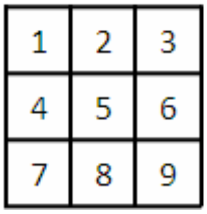
Figure 9 Missing Levels with Lattice Layout

Note that the order of the cells with the PANEL layout is normally alphabetical within each classification variable. This changes when there are missing values and the SPARSE option is used. In Figure 8 the order is WEST then EAST because the BED x EAST combination is missing. Since that occurred under BED (the first level of PRODUCT) all of the other PRODUCTS are WEST then EAST. If instead, the CHAIR x EAST combination had been missing (or any of the PRODUCTS other than the first), then all of the PRODUCTS would have had EAST then WEST.

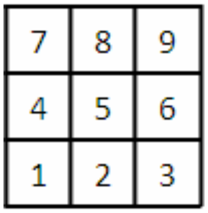
The MISSING option and the SPARSE option sound similar to one another and they are. The difference is in what's missing. Suppose we have a data set with classification variables *A* and *B* that we'll use for defining our cells which will plot *X* vs *Y*. The MISSING option would be useful when we have data records with values for *X* and *Y*, but the values for *A* and *B* are missing. The SPARSE option is useful when no records exist (i.e. no data for *X* and *Y*) for some combinations of *A* and *B*.

START= TOPLEFT I BOTTOMLEFT

This option determines the order of the cells in the graphs by specifying the location of the first cell. TOPLEFT is how we read tables and BOTTOMLEFT is graphical order. From the SAS Documentation we have this pictures.



1	2	3
4	5	6
7	8	9



7	8	9
4	5	6
1	2	3

UNISCALE= COLUMN I ROW I ALL

The default value for this option is ALL and most of the time that's the appropriate one. Specifying either COLUMN or ROW scales just the column axes or row axes, respectively, to be identical. These may be helpful if the scales for the levels of the other dimension are radically different.

COLAXIS AND ROWAXIS

Most of the axis options behave the same with PROC SGPANEL as they do with SGPLOT. We mentioned the DISPLAY=NONE option above when discussing the ROWHEADERPOS and COLHEADERPOS options.

The ALTERNATE option will "interfere" with the row and column headings when using the LATTICE layout. The PANEL layout can benefit from this option, though, particularly if there are a large number of columns or rows. This option will add reference ticks to each side (Top and Bottom, or Left and Right) of the panel and alternates the tick values between the two sides, depending on whether you use COLAXIS or ROWAXIS.

CONCLUSION

The SGPANEL procedure is a very powerful part of the Statistical Graphics family. It allows the intuitive display of complex information, high-dimensional information. Knowing how to use the many options will help you customize your graphs to best present your data. I hope this paper will help you in doing so.

REFERENCES

- Edward R. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, 1983.
- William S. Cleveland. *Visualizing Data*. Hobart Press. 1993.
- Lora D. Delwiche, Slaughter, Susan J. *Using PROC SGPLOT for Quick High Quality Graphs*. WUSS 2008. http://www.wuss.org/proceedings08/08WUSS%20Proceedings/papers/how/how05.pdf
- Andrew Gelman. *Who wants school vouchers?* June 8, 2009 http://www.stat.columbia.edu/~cook/movabletype/archives/2009/06/estimated_suppo.html

ACKNOWLEDGMENTS

Thank you to the SAS ODS Development team for their ideas in the conception of this paper and for answering questions along the way. Thank you, also, to Jack Fuller for his careful review and suggestions.

CONTACT INFORMATION

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

Chuck Kincaid
COMSYS Business Analytics Practice
5220 Lovers Lane
Portage, MI 49002
269-553-5140
ckincaid@comsys.com
www.comsys.com/analytics