Paper 134-2007

Communication-Effective Pie Charts

LeRoy Bessler, Assurant Health, Milwaukee, Wisconsin, USA

Abstract

Pie charts are a popular and simple tool to visually compare the relative size of the parts of a numeric whole. Yet, there is more to getting a SAS/GRAPH® pie chart right than you may think. We start with an eye-opening comparison of 3D and 2D. You will see how 3D can actually undermine visual communication by distorting the apparent relative size of slices. Then we look at ordering pie slices. If you also want to control the color palette for a multi-color SAS/GRAPH pie (i.e., if you care which slice is which color), there is an unexpected challenge to meet. It is also important to consider when and how to use an "Other" slice. To maximally exploit the power of simplicity, you will be shown the author's "Pac-man Pie Chart". The biggest pie chart challenge is labeling. Unless creating a pie of a few slices with all of them about the same size, it may be a problem to get every slice description, value, and percent to fit around the perimeter of the pie without overlapping. SAS/GRAPH provides a legend option, but it does not necessarily eliminate the pie label overlap problem. You will learn how to build a custom legend, which will solve the problem without great difficulty. Also, this paper shows a possible SAS/GRAPH alternative to the pie chart and demonstrates that an Excel pie chart does not reliably deliver a communication-effective legend. No prior SAS/GRAPH experience is required.

Introduction

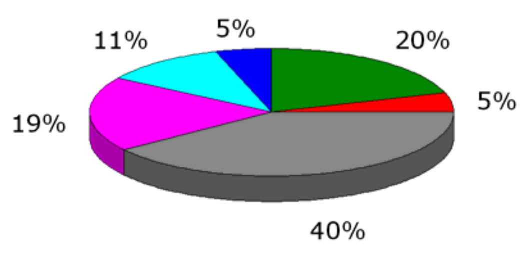
To visually deliver the truth about data, avoid the popular 3D pie chart. To show the viewer what is important, control the order of pie slices. To circumvent the possible problem of label overlap, I provide a custom legend solution. The use of an "Other" collective for data is regrettably common in pie charts. Usually, I recommend against creating questions about what is in that "Other" slice, but here I present two ways to use "Other" to maximize communication effectiveness. I also show you: (a) why Excel is not a good tool for pie charts; and (b) a pie chart alternative. Also, I provide information about fonts and colors. The graphs are made with the GIF driver at 960 pixels wide by 900 pixels high (10 inches by 9.38 inches), but reduced to 3.2 inches by 3 inches after insertion.

This paper is part of a trio. Reference 1 covers communication-effective graph design and web design, and widely usable examples of bar charts and trend charts. Reference 2 covers: ODS destinations; graphic device drivers; preparing graphs for, and using them in, Microsoft Word or PowerPoint; "Accessibility"; etc.

3D Pie Chart Always Distorts Apparent Relative Size of Slices

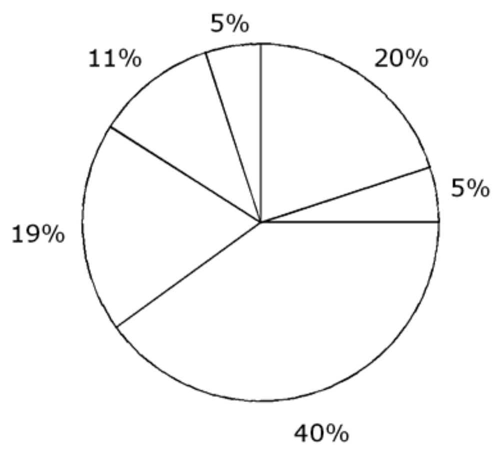
After inspecting Figure 1, why would you ever create another 3D pie chart? The only 3D pie charts that never distort relative size of shares of the whole are: (a) one with two equal slices; and (b) a pie chart with four equal slices, if drawn with one of the two dividers horizontal and the other vertical. However, a pie chart with equal slices is not worth creating. A simple text statement can describe that situation clearly, concisely, and completely.

Figure 1: 3D Pie Chart.
Always Distorts Apparent Relative Size of Slices.



5% slice at rear is twice as big as 5% slice at right
11% slice is almost as big as 19% slice

Figure 2: 2D Pie Chart. Simpler is Better.
Always Presents Accurate Relative Size of Slices.



Here is code used to create Figure 2:

```
  /* Common Preliminary Code presented later in the paper would be inserted here at top of your program. */
title2 height=5 PCT font='Georgia'
justify=LEFT '  Figure 2: ' color=CX0000FF '2D Pie Chart. Simpler is Better.'
justify=LEFT '  Always Presents Accurate Relative Size of Slices.';
footnote height=5 PCT ' ';  /* white space at bottom */
pattern1 v=pempty repeat=6; /* empty pie slices      */
proc gchart data=DataForSimpleCharts;
pie Name / sumvar=Value
      noheading         /* suppress default pie chart heading */
      coutline=CX000000 /* color of pie slice outline is Black.
             The default is COUTLINE=SAME, which means "same color
             as slice fill", presumably controlled by PATTERN statements.
             However, though v=pempty leaves pie slices empty,
             SAS/GRAPH colors the slice outlines using the driver's default
             color list IF you accept the default COUTLINE=SAME. */
      woutline=2        /* thicken the pie outline */
      slice=none        /* no Name labels */
      value=none        /* no Value labels */
      percent=outside;  /* Percent-Of-Whole labels outside the pie */
run; quit;
```

Slice names and slice values are suppressed in Figures 1 and 2 to focus viewer attention on the discrepancy between the actual numerical percent and the misleading pictorial percent.

Overcoming the Labeling Challenge Inherent in All Pie Charts and Taking Control of Slice Order

Figure 3 is a standard use of the SAS/GRAPH 2D pie chart, displaying slice Name, Value, and Percent. You can position this information INSIDE, OUTSIDE, or (outside with a connecting) ARROW. Order of the slices is, by default, alphabetical by slice Name. There is no good way to solve The Overlap Problem without a legend. And, unlike the custom legend shown in Figure 4, the legend provided by SAS/GRAPH includes only the slice name/description, leaving you with the risk of a labeling problem for the value and percent. Note my preference to order slices by size. If bigger (or smaller) is better, order the slices to show the viewer what is important. Ordering the slices in Figure 3 by size would force a label overlap because the two small slices would be adjacent.

Figure 3: Standard Form of 2D Pie Chart. Susceptible to Label Overlaps.

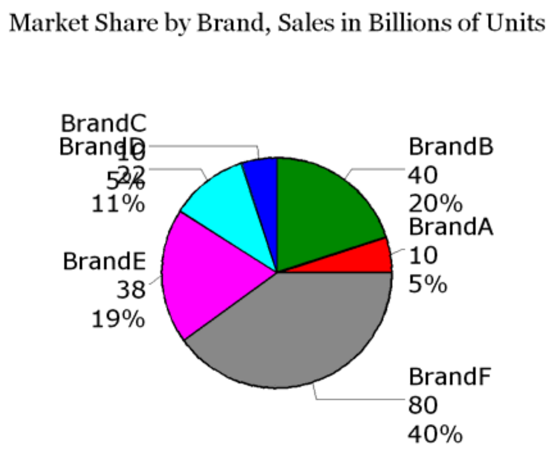
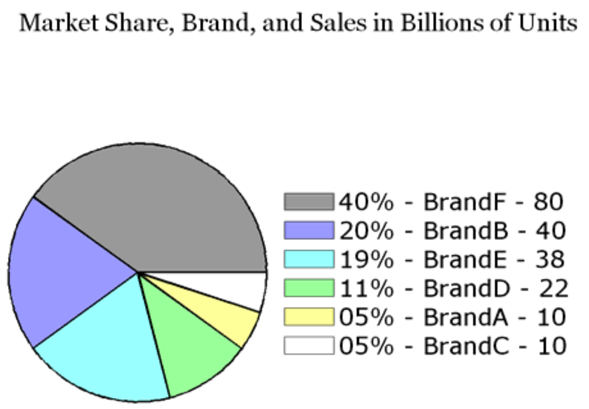


Figure 4: Custom Legend (and Ordering By Size). Immune to label overlaps.



Here is code to create Figure 3:

```
  /* Common Preliminary Code presented later in the paper would be inserted here at top of your program. */
title2 height=5 PCT font='Georgia'
justify=LEFT '  Figure 3: Standard Form of 2D Pie Chart.'
justify=LEFT color=CXFF0000 '  Susceptible to Label Overlaps.';
title3 height=2.5 PCT ' ';
title4 height=5 PCT font='Georgia' justify=CENTER
'Market Share by Brand, Sales in Billions of Units';
footnote;
proc gchart data=DataForSimpleCharts;
pie Name / sumvar=Value
    noheading
    woutline=2
    slice=arrow
    value=arrow
    percent=arrow;
run; quit;
```

Here is part of the SAS® log from the execution of this code:

```
WARNING: Text boundaries for the pie/donut slices overlap.
NOTE: Two or more pie/donut slice labels have been overwritten. This can occur if there are
several adjacent small slices. To avoid overwriting labels you can use one or more of the
following techniques:
      1. Increase GOPTIONS HPOS= and VPOS= to make the text smaller.
      2. Reduce the size of the text with GOPTIONS HTEXT=.
      3. Use a different labeling method and perhaps a legend.
      4. Use the OTHER= option on the PIE/PIE3D/DONUT statement to group small slices into a
         single category.
      5. Use the MIDPOINTS= or ANGLE= options on the PIE/PIE3D/DONUT statement to change the
         order of slices or starting angle for the first slice so that two small slices do not
         fall next to each other.
      6. Use a VBAR or HBAR chart instead of a PIE/DONUT chart.
      7. Use the graphics editor to reposition or resize the labels.
      Consult the proper documentation for details on any of these techniques.
```

Let me assess these options in the light of my design objectives.

I want a pie chart. Option 6 is unacceptable.

I want to be able to get the graph right the first time every time. This is mandatory for ever-changing data input to computer-scheduled production batch, or to real-time online graphs—such as can be produced with SAS/IntrNet applications or Stored Processes. Even when you are producing an ad hoc pie chart for a presentation, it is good practice to adapt, or invent, a solution that can be reused. A well-designed communication tool is likely to be applicable again in the future, and—unless the specifications are extensively different—it is almost always more efficient to reuse, possibly with some adaptation, an existing solution than to code a new solution from scratch. When working under deadline, getting to presentation-grade, communication-effective graphs should not entail an iterative adjustment burden or require development of all new code. Thus, options 1, 2, 5, and 7 are unacceptable.

As a communication tool, your graph should answer any anticipatable questions, not create them. An “Other” slice invites two questions: “What is in OTHER, and how big are the pieces?” Option 4 is unacceptable.

Let us consider Option 3. Though the labeling options are INSIDE, OUTSIDE, and ARROW, experimenting in pursuit of a satisfactory way to display SLICE, VALUE, and PERCENT by some single choice from the three options, or by some fortunate combination, is an unwanted iterative adjustment burden. Moreover, a satisfactory solution for one set of data is not guaranteed, and frankly is unlikely, to also work for a different set of data. So, let us consider the standard legend available with SAS/GRAPH.

SAS/GRAPH can provide a pie legend, but unfortunately it includes only the slice name. Therefore, you are still left with the possibility of label overlaps when trying to fit both VALUE and PERCENT labels on and/or around a pie, which, depending on the data, has an arbitrary number and relative sizing of slices.

Figure 4 is the most problem-resistant pie chart you can build with SAS/GRAPH—if you wisely insist on code that can merely be pointed at the data with no requirement for post-processing, special manipulation, or circumvention. This custom pie chart becomes, in effect, a table with an adjacent visual indicator of percent of whole to complement the explicit percent of whole that is listed. If a share is too small to be visible on the pie, its name/description, value, and percent of whole are still reliably delivered in the legend/table.

Below is code used to create Figure 4. By default, SAS/GRAPH orders its legend entries and applies colors from the device driver default color list in alphabetical order of the pie slice names. I wanted an automated solution that would order the legend and colors by decreasing size of the slices. I also wanted the color list to go from darkest to lightest. Lighter colors help the smaller pie slices become more conspicuous, making them more visible despite their size disadvantage. The Software-Intelligent solution preprocesses the data, stores information in the macro symbol table, and uses custom macros to dynamically build the PATTERN statements and the LEGEND statement.

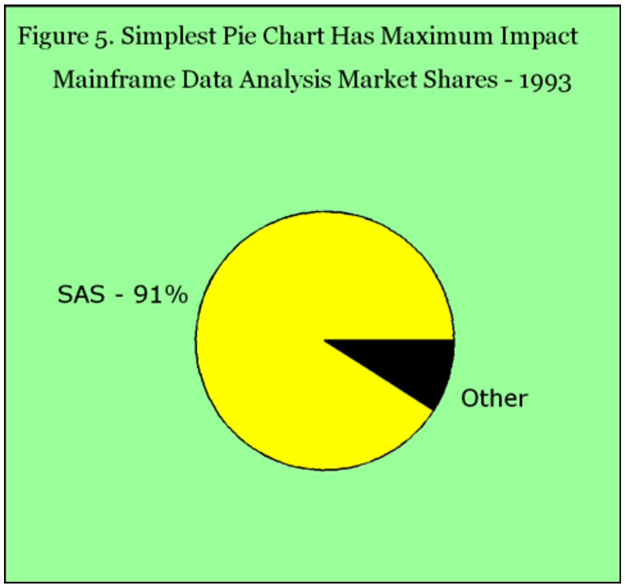
```
%macro PatternStatements;
%do i = 1 %to 6;
pattern&i v=psolid color=&&SliceColor&i repeat=1;
%end;
%mend  PatternStatements;

%macro LegendEntries;
%do i = 1 %to 6;
  "&&LegendEntry&i"
%end;
%mend  LegendEntries;

proc means data=DataForSimpleCharts noprint sum;
var Value;
output out=PieTotal sum=TotalValue;
run;

 /* concatenate Percent and Value with the original Slice Name */
data SliceNameWithPercentAndValue;
length NameWithPercentAndValue $ 17;
set DataForSimpleCharts;
if _N_ eq 1 then set PieTotal;
Percent = (Value / TotalValue) * 100;
NameWithPercentAndValue = trim(left(put(Percent,Z2.)))  || '%' || ' - ' ||
                    trim(left(Name)) || ' - ' || trim(left(put(Value,2.)));
run;

data ColorList; /* Sequence the colors from darkest to lightest,
                   if you want small slices to stand out. */
pctseq = 1; color='CX999999'; output;
pctseq = 2; color='CX9999FF'; output;
pctseq = 3; color='CX99FFFF'; output;
pctseq = 4; color='CX99FF99'; output;
pctseq = 5; color='CXFFFF99'; output;
pctseq = 6; color='CXFFFFFF'; output;
run;
```

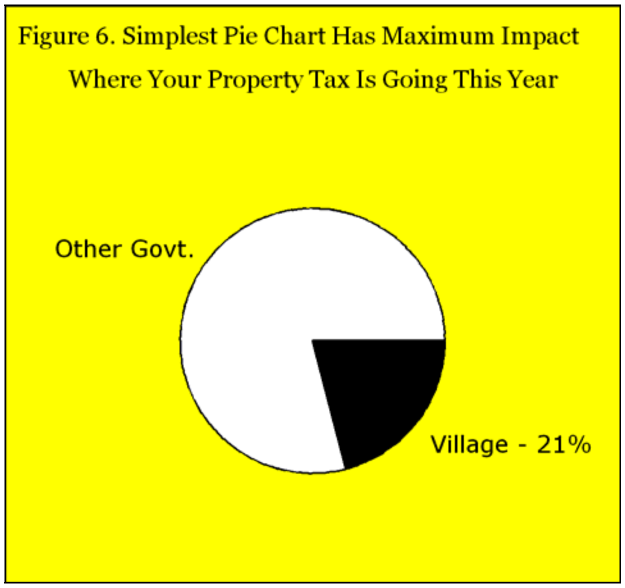
```
proc sort data=SliceNameWithPercentAndValue;
by descending Percent; /* sequence the slices from largest to smallest */
run;

data ToSort;
set SliceNameWithPercentAndValue;
pctseq = _N_; /* add a key to merge this data with the ColorList    */
call
symput('LegendEntry'||trim(left(_N_)),trim(left(NameWithPercentAndValue)));
              /* Store the legend text entries in the symbol table. */
run;

proc sort data=ToSort;
by pctseq;
run;

data SliceWithColor(drop=pctseq);
merge ToSort ColorList;
by pctseq;
run;

proc sort data=SliceWithColor;
by NameWithPercentAndValue; /* because SAS/GRAPH will always apply
  your PATTERN statements in the sort order of the SLICE name text */
run;

data _null_;
set SliceWithColor;
call symput('SliceColor'||trim(left(_N_)),trim(left(color)));
            /* Store the slice colors in the symbol table. */
run;

OPTIONS MPRINT; /* in SAS log, show the code generated by the macros */

  /* Common Preliminary Code presented later in the paper would be inserted here. */
title2 height=5 PCT font='Georgia'
justify=LEFT '  Figure 4: ' color=CX0000FF 'Custom Legend ' color=CX000000
'(and Ordering By Size).'
justify=LEFT color=CX0000FF '  Immune to label overlaps.';
title3 height=2.5 PCT ' ';
title4 height=5 PCT font='Georgia'
justify=CENTER 'Market Share, Brand, and Sales in Billions of Units';
footnote angle=-90 height=1 PCT ' '; /* push pie to the right */
%PatternStatements;
legend1 order=(%LegendEntries) label=none shape=bar(8 PCT,3 PCT)
        across=1 position=(middle right outside) offset=(-5 PCT,0);
proc gchart data=SliceNameWithPercentAndValue;
pie NameWithPercentAndValue / sumvar=Value
    noheading coutline=CX000000 woutline=2 descending
    legend=legend1
    slice=none value=none percent=none; /* turn off all pie labels */
run; quit;
```

To adapt the above code to a different number of pie slices will, of course, require a change to the macro index range maximum and a change to the COLORLIST DATA step. To customize the setup in the `SliceNameWithPercentAndValue` DATA step is straightforward. The most work will be to customize the TITLEs and FOOTNOTE, to position the LEGEND, and to adjust the SHAPE and possibly the OFFSET. It is simpler to move the legend to below the pie, using **`position=(bottom center outside)`**. This is the default when you omit the POSITION parameter, and would allow you to drop the OFFSET parameter. Still on my To Do List: it is possible to imbed all of the pre-processing and the main body of the GCHART step in a macro, which takes a list of colors and the legend BAR width and height as invocation parameter assignments. The TITLE and FOOTNOTE statements and the Common Preliminary Code would precede the macro invocation.

The Irrefutable Power of Simplicity: Exploiting the Extremes of "Other"

A graph should answer questions, not prompt them. However, suppressing some information by use of an "Other" slice can sometimes aid communication, rather than obstruct it. Figure 5 is a case where what is in "Other" just does not matter. All of the remaining market shares are insignificant. Conversely, you can emphasize the smallness of a key share of interest by lumping the remainder into one huge "Other" slice. For Figure 6, you could satisfy interest in the content of "Other" with a companion table. If web-deploying such a graph, you could provide that table as pop-up text for, and/or a hyperlink from, its big slice. I created Figure 6 (and an innovative table) while serving as elected Village Trustee to redirect undeserved complaints from residents about the size of their property tax bills.





Here is code used to create Figure 5:

```
data ForSimplestPie; infile CARDS;
input @1 software $9. @18 PctShare 4.1;
CARDS;
SAS - 91%        91
Other             9
; run;
```

/* Common Preliminary Code presented later in the paper would be inserted here. */

```
title2 height=5 PCT font='Georgia'
justify=LEFT '  Figure 5. Simplest Pie Chart Has Maximum Impact';
title3 height=2.5 PCT ' ';
title4 height=5 PCT font='Georgia'
justify=CENTER 'Mainframe Data Analysis Market Shares - 1993';
footnote1 angle=-90 height=8 PCT ' '; /* shift and compress pie from left  */
footnote2 angle=+90 height=3 PCT ' '; /* shift and compress pie from right */
/* Above adjustments to pie size and position are NOT done for Figure 6. */
goptions cback=CX99FF99;      /* light green background */
pattern1 v=psolid color=CXFFFF00; /* full strength yellow    */
```

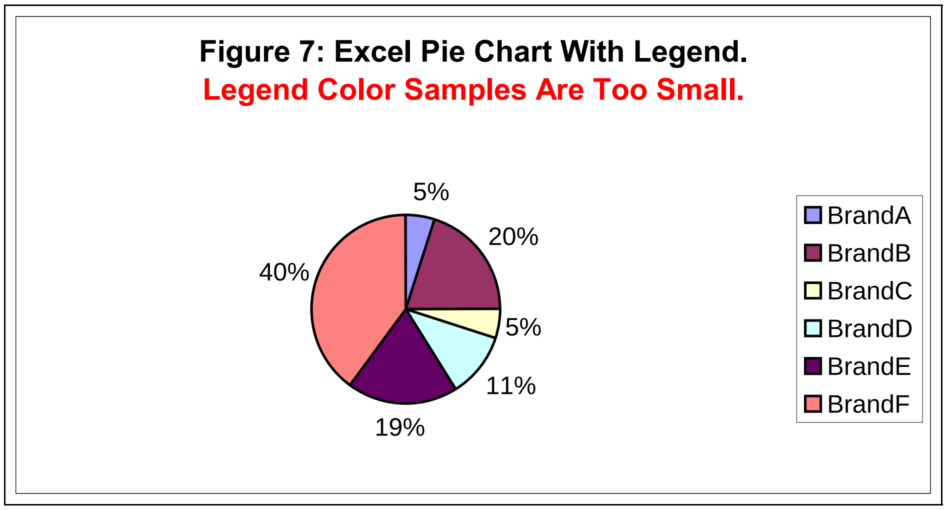
```
pattern2 v=psolid color=CX000000; /* black                    */
proc gchart data=ForSimplestPie;
pie software / sumvar=PctShare noheading coutline=CX000000 woutline=2
    slice=outside value=none midpoints='SAS - 91%' 'Other     ';
run; quit;
```

Common Preliminary Code for the Graphs

```
data DataForSimpleCharts;
infile datalines;
input @1 Name $6. @8 Value 2.;
datalines;
BrandA 10
BrandB 40
BrandC 10
BrandD 22
BrandE 38
BrandF 80
; run;
goptions reset=all;  /* it is best to do this reset before EVERY graph */
goptions device=GIF;
goptions cback=CXFFFFFF;  /* background color is RGB white, EXCEPT for Figures 5 and 6 */
goptions border;  /* Put the graph in a box, to separate it from text when being published in, e.g., Word.
Alternatively, you can apply a border to the inserted graph using Word itself. In this paper, both borders are used. */
goptions xpixels=960 ypixels=900;
goptions htext=5 PCT ftext='Verdana';  /* height and font used for parts of graph for which you do
not make an explicit assignment, or for which no direct controls are available in SAS/GRAPH. */
goptions gsfname=anyname;  /* output the graph as a file on disk with this fileref */
filename anyname "c:\YourFolderName\Figure_NN.GIF";
title1 height=2.5 PCT ' ';  /* empty space between border and TITLE2 */
```

NOTE: Using the PNG driver with the pixel assignments above and XMAX=3.2 IN YMAX=3.0 yielded graphs that could be inserted and used without resizing, but the image quality was not better, and, in my judgment, worse.

The Anti-Communicative Excel Pie Chart

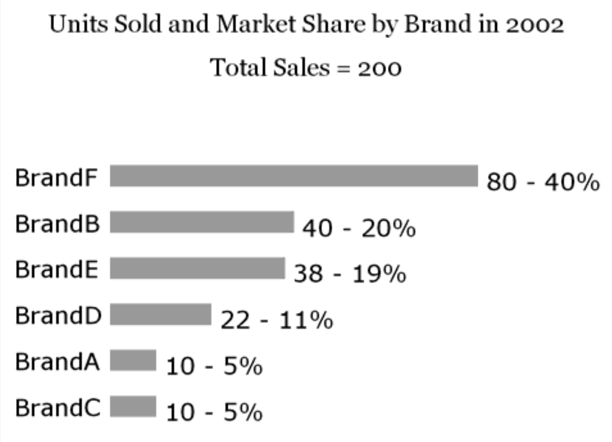


Excel does not allow changing the size of the color samples. (Other design flexibility is very limited as well.) Especially when printed, the colors in the legend are difficult (or impossible) to distinguish for Brands B and E.

Possible Pie Chart Alternative

Since there is a chance that you might have a boss, a client, or a service requestor who does not like pie charts, here is a way to deliver the name/description, the value, and the percent for all of the shares of a whole with a picture that is not a pie chart. The only drawback of the bar chart in Figure 8 is that the comparability of the image elements is possible only on the basis of relative sizes, not on the basis of size as compared to the whole. By the way, a horizontal bar chart is always preferable to a vertical bar chart if you want to be immune to label space problems. The “solution” to label vertical bars with tilted text may be common, but is not my design recommendation. We read English from left to right, not uphill.

Figure 8: Pie Chart Alternative.



Here is code used to create Figure 8:

```
proc means data=DataForSimpleCharts sum noprint;
var Value;
output out=Total sum=TotalValue;
run;

data _null_;
set Total;
call symput('GrandTotal',
trim(left(put(TotalValue,3.))));
run;

goptions gunit=PCT;
data Annotation(drop=Name Value);
length text $ 14 style $ 40 function color $ 8
position xsys ysys hsys when $ 1 size 3;
retain style "'Verdana'" function 'label' color 'CX000000'
       position '6' xsys ysys '2' hsys '4' when 'a' size 2.5;
set DataForSimpleCharts;
yc = Name;
x = Value;
text = ' ' || trim(left(put(Value,2.))) || ' - ' ||
       trim(left(put(((Value/&GrandTotal) * 100),2.))) || '%';
run;
```

```
  /* Common Preliminary Code presented later in the paper would be inserted here. */
title2 height=5 PCT font='Georgia'
justify=LEFT '  Figure 8: ' color=CX0000FF 'Pie Chart Alternative.';
title3 height=2.5 PCT ' ';
title4 height=5 PCT font='Georgia'
justify=CENTER 'Units Sold and Market Share by Brand in 2002';
title5 height=2.5 PCT ' ';
title6 height=5 PCT font='Georgia'
justify=CENTER "Total Sales = &GrandTotal";
footnote1 height=5.0 PCT ' '; /* white space at bottom */
footnote2 angle=+90 height=22 PCT ' '; /* force bar ends to left to leave space for annotation */
footnote3 angle=-90 height=2  PCT ' '; /* white space at left margin */
pattern1 v=solid color=CX999999 repeat=6; /* use light browser-safe gray */
axis1 label=none major=none minor=none style=0; /* if midpoint values are long and variable,
                                                    consider adding value=(justify=RIGHT) to AXIS1 statement */
axis2 label=none major=none minor=none style=0 value=none;
proc gchart data=DataForSimpleCharts
            anno=Annotation;
hbar Name / sumvar=Value
     nostats
     width=1.8
     space=2.2
     maxis=axis1
     raxis=axis2
     descending;
run; quit;
filename anyname clear;
```

NOTE: In Figure 4 it would have likewise been possible to create a dynamically generated subtitle for "Total Sales" using the method shown in the code above.

Fonts and Text Sizes

For titles and footnotes, I recommend the Georgia font. For graph "body" text, which often must be smaller, I recommend the Verdana font. Matthew Carter designed, and Tom Rickner built, these Windows TrueType fonts for Microsoft to provide improved on-screen readability. To override the software defaults, a "preferred default" text font and text height are specified by GOPTIONS FTEXT= HTEXT= in the Common Preliminary Code. (Defaults for the TITLE1 statement are controlled with GOPTIONS FTITLE= HTITLE=.) You can specify the height of text with suffix CELL (the default), IN (inches), CM (centimeters), PT (point size), or PCT (percent of the graphic display area). After over a quarter century of experience using all of the other options, I have finally settled on PCT. It makes your design portable across device drivers that may have different cell sizes and/or different default graphic display area sizes. Moreover, it is absolutely intuitive. As you change a particular height assignment value, the results are never perplexing. Be aware that you can append /BOLD and /ITALIC to any Windows font assignment.

Colors

A software default color list is specific to each device driver. If you do not turn off color fill for the pie slices and do not explicitly specify a color for each, they will be filled in order according to that color list. A "personal default" color is specified by GOPTIONS CTEXT= in the Common Preliminary Code. If you do not specify your preference, for any text for which you do not or can not explicitly assign a color, the current setting of CTEXT= will be used. You may not like that result. (See the SAS/GRAPH Online Doc for a discussion of GOPTIONS CTITLE.)

For pie charts, SAS/GRAPH provides a MATCHCOLOR option, which means the color of slice labels will match the color of area fill. This may be decorative, or it may have some "connection value" if you put the labels outside of the pie slices without an ARROW connector. But, since colored text is more difficult to read than black (presuming that the background is white or sufficiently light), such an attempt to enhance communication is not a good idea.

If you create an empty (i.e., no-area-fill) pie chart by using V=PEMPTY in your PATTERN statements and take no other action, then the slices will be outlined with colors in order according the color list. Whether or not you color the pie slices, I recommend use of COUTLINE=BLACK on the PIE statement. In the case of empty slices, the color of thin lines is hard to distinguish and has no communication value. If the slices are colored, then a black separator can diminish, if not eliminate, any unusual optical/visual effect of side-by-side color pairs.

On a light (dark) background, use black (white) labels for maximum contrast—i.e., readability.

For the web, it is best to use the Browser-Safe subset of RGB colors. For more about them, and for more about effective communication with color, please see Reference 3.

Conclusion

3D pie charts always distort communication.

The only communication-effective pie chart that can be produced without ad hoc iterative adjustments (which are infeasible for production batch or real-time applications) is a Custom Legend pie chart like that shown in Figure 4.

Order pie slices from largest to smallest, or vice versa, depending on whether largeness or smallness is better, to show the viewer what is important.

Usually avoid use of “Other” to avoid creating questions—anticipate and answer them. However, when the most important slice in your pie chart is very big or very small, it can be a powerful message to combine all remaining slices into one “Other” slice, tiny or huge.

Excel is not a suitable tool for communication-effective pie charts.

It is possible to substitute a bar chart for a pie chart. A bar chart can display share name, value, and percent of whole, but visual comparability is only by relative size of bar, not by size of share as compared to the whole.

References (Related Work by the Author)

1. Get the Best out of SAS/GRAPH and ODS, *Proceedings of the SAS Global Forum 2007*.

2. How to Make the “Best Choice” from the Many Ways to Create and Deliver SAS Graphs, *Proceedings of the SAS Global Forum 2007*.

3. Communication-Effective Use of Color for Web Pages, Graphs, Tables, Maps, Text, and Print, *Proceedings of the Twenty-Ninth Annual SAS Users Group International Conference*, 2004.

Acknowledgments

I am grateful to my colleague Alix Riley for her helpful review of this paper, and to SAS Global Forum 2007 Poster Section Chairs David Johnson and Zulfiqar Habib for the opportunity to share my ideas with other SAS users.

Contact Information

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

LeRoy Bessler PhD
Email: bessler@execpc.com
Phone: 1 414 351 6748 (evenings and weekends—time is six hours earlier than Greenwich Mean Time)