Simple Steps for Great Graphics:
Beyond the Default Settings for More Effective Data Visualization

Kristina B. Metzger, Metzger Consulting, Austin, Texas

ABSTRACT

Graphs are an efficient way to present large quantities of data. The human brain is able to recognize patterns in pictures more easily than numbers or text. In oral presentations and written reports, researchers often use graphic displays of data to help tell the story of their research. Data visualization expert Stephanie Evergreen has cited that the default settings in statistical software packages are one of the biggest obstacles to effective data visualization. Features of better graphics, such as more white space, clean lines, and judicious use of color, are not automatic with the default settings in SAS. A few extra lines of code in your graphing programs can easily produce more effective data visualization graphics. Maximize white space by reducing clutter, such as by eliminating borders and removing default text like titles and labels. The graph axes can be streamlined by using appropriate ranges, intervals, and scales. Axes labels and value labels should be used appropriately, but sparingly. The default colors in SAS can be distracting; better use of color can enhance the readability of graphs. Graph colors may be chosen based on a company brand, journal requirements, or other objective, including accessibility for color-blind persons. Color may also be used to highlight a particular data point. These simple steps can improve your SAS graphs in order to better communicate your story.

INTRODUCTION

One main purpose of data analysis is to synthesize and interpret large amounts of data into summary measures in order to communicate the main findings. Tables can be useful in a report but can be overwhelming to a reader and nearly impossible to see when included in a presentation. Furthermore, the human brain has developed to recognize patterns rather than numbers. Thus, researchers often turn to graphs to help show the story of their data. In order to concisely demonstrate your main point, choose a type of graphic that is best suited to highlight the message. A graph that is easy to view, through judicious use of color, text, and data elements, will help focus your story. A graph should only present information relevant to understanding its purpose. By using a more effective graph, you can spend time telling your story rather than explaining its construction. Your audience will appreciate the time you spend considering the presentation of your data. Your coworkers will appreciate your effort to produce coherent and understandable graphs even when you're still in the draft stages.

I'll present samples of code that you'll be able to include in every graphics program in order to create effective graphs. My code will generate a separate PDF document for each graph. If you prefer to output graphs into a different format, you'll need to modify the ods destination statement. Each figure includes two graphs—a series graph on the left and a bar graph on the right. To create the examples in this paper, two of the SAS data sets available in the *SASHelp* library were used. The series graph uses data from *SASHelp.Retail* and displays the quarterly earnings of a retail establishment over four years. The bar graph uses data from *SASHelp.Baseball* and displays the proportion of field position players with annual earnings of at least $500,000. The Appendix includes code used to modify the *SASHelp.Baseball* data for the graphs, along with formats used.

THE INADEQUACY OF DEFAULT SETTINGS

Stephanie Evergreen, a data visualization guru, has stated that the default software settings of statistical programs are one of the biggest obstacles to effective data visualization (see Research in Action podcast). While SAS graphics have dramatically improved over the years, the default settings remain distracting. For example, here is the code used to generate the two default graphs shown in Figure 1:

```
goptions reset=goptions ;

ods listing close;
ods graphics on;

ods pdf file="C:\retail_series_default.pdf";

proc sgplot data=sashelp.retail ;
where 1980<=year<=1983;
series x=month y=sales / group=year;
run;

ods pdf close;

ods pdf  file="C:\baseball_bar_default.pdf";

proc sgplot data=baseballfreq ;
vbarparm category=position response=percent;
run;

ods pdf close;

ods graphics off;
ods listing;
```

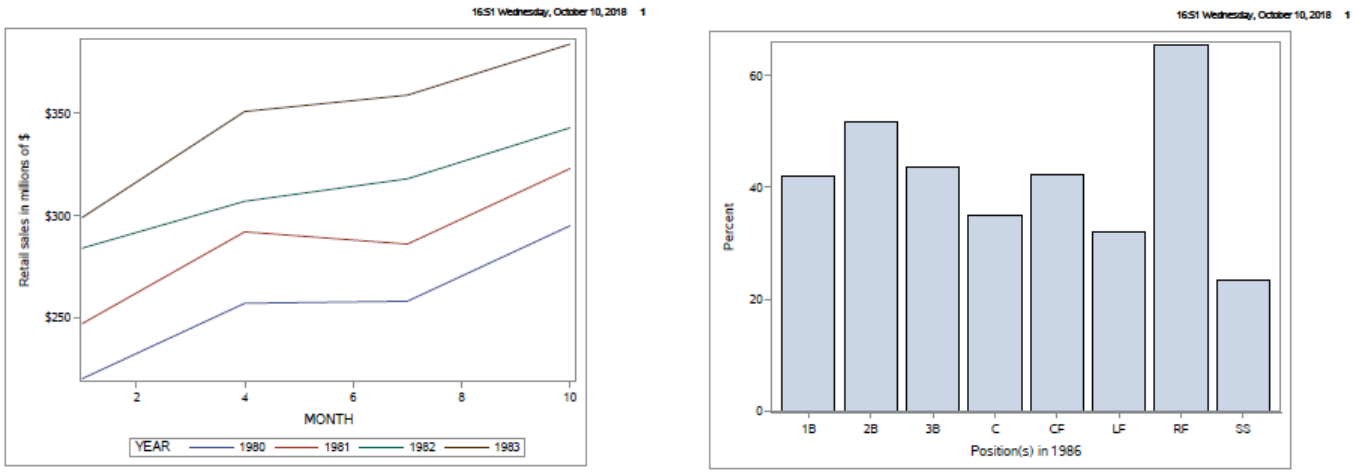

Figure 1. SAS graph default settings.

The default graphics code generates unnecessary borders that enclose both the entire graphic and the graph itself as extensions of the x- and y- axes. Additionally, SAS automatically includes the date generated and page numbers on the output. The axis labels, tick-mark values, and legend use the labels, values and names of the variables specified in the graphics code. The line colors rotate through a SAS default color list that does not appear to have any inherent order. The color of the bar graph is a pleasant, but arbitrary, light blue. The

order of the bars in the bar graph is determined by the unformatted values of the categorical variable plotted on the x-axis.

MAXIMIZE WHITE SPACE

First, excess content should be removed from the graph in order to maximize its white space. Essentially, any parts of the graphic that are unnecessary and distracting should be taken away. This includes removing borders, frames, and automatically generated text. The following code is modified from the default code and generates the graphs in Figure 2.

```
options nodate nonumber;   ①
goptions reset=goptions ;

proc template;   ②
    define style styles.noframe;
        parent=styles.pearl;
        class graphwalls / frameborder=off;
    end;
run;

ods listing close;
ods graphics on / noborder;   ③


ods pdf style=noframe file="C:\retail_series_whitespace.pdf";  ④
proc sgplot data=sashelp.retail ;
where 1980<=year<=1983;
series x=month y=sales / group=year;
keylegend / noborder;
run;

ods pdf close;


ods pdf style=noframe file="C:\baseball_bar_whitespace.pdf";   ④

proc sgplot data=baseballfreq ;
vbarparm category=position response=percent;
Run;

ods pdf close;

ods graphics off;
ods listing;
```

The automatically generated SAS-date and page number can be removed in the options statement, by specifying *nodate* and *nonumber* (①). Each type of ODS destination has its own default style template. For the PDF destination, the default style is called *pearl*. In the PROC TEMPLATE procedure (②), the PDF default style (designated as the “parent” style) is modified to create a new style named *noframe* in the define style statement. The class statement indicates that the frame border, which is the border that completes the rectangle partially formed by the x- and y-axes, should be turned off. Then, the newly created *noframe* style is

referred to in the ods pdf statements (④). The ods graphics option indicates that there should be no border surrounding the entire graphics output (③).

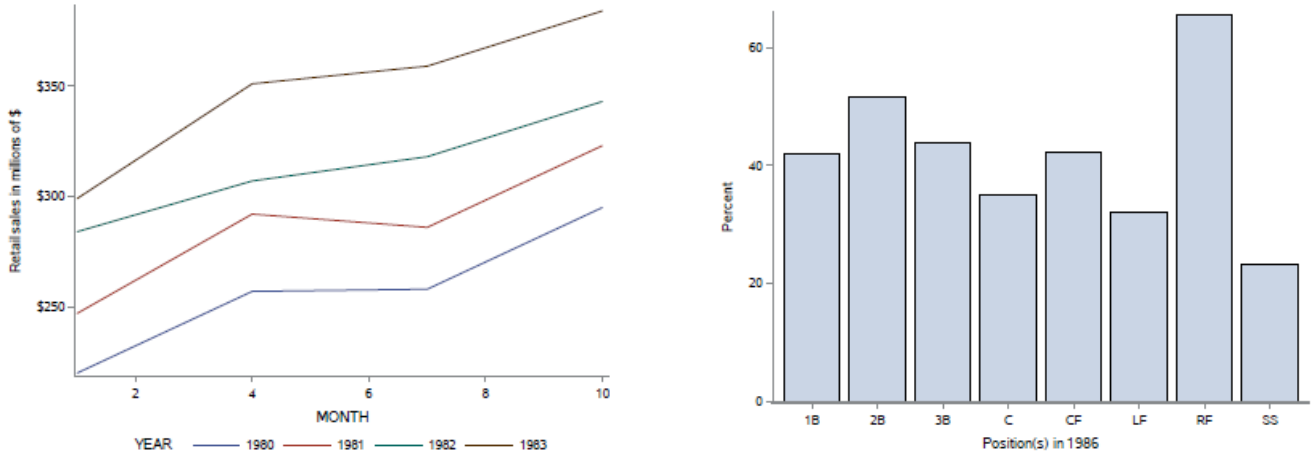

Figure 2. Maximizing white space in your graphs.

USE YOUR OWN COLORS

As seen in the graphs above, the default colors SAS automatically cycles through do not have an apparent order. Some circumstances will limit your use of color to black or grayscale, such as publication requirements for particular journals, but often the use of a variety of colors is preferred. The choice of an appropriate color palate can enhance the effectiveness of your graph. Certain colors, like black, dark blue, and dark gray, are easier to read as text than lighter or brighter colors. The use of color gradients (e.g., moving through light blue to medium blue to dark blue) can be used to help illustrate sequential data, such as an outcome of increasing severity. Divergent color schemes that shift from one distinct color through a neutral color to another distinct color are often used to demonstrate distinct outcomes or variations on either side of a null value. Other color choice considerations may include color-blind safe palettes and those that can be distinguished if printed in black and white. Additionally, your organization may have specific branding colors for use in graphics. In these examples, I will be using the corporate branding colors for the Children's Hospital of Philadelphia (CHOP), the organization for which I do much of my consulting. The following code can be run prior to the graphing code to change the default color scheme and produces the graphs in Figure 3.

```
proc template;
    define style mycolors;
        parent=styles.pearl;
        class graphwalls / frameborder=off;
        style GraphAxisLines "dark gray"/ Color = cx786452 ContrastColor = cx786452 ;
        style GraphDataDefault "dark blue" / Color = cx005587 ContrastColor = cx005587 ;
        style GraphOutlines "dark blue" / Color = cx005587 ContrastColor = cx005587 ;
        style GraphData1 "dark blue" / Color = cx005587 ContrastColor = cx005587 ;
        style GraphData2 "olive green" / Color = cx91A01E ContrastColor = cx91A01E ;
        style GraphData3 "bright blue" / Color = cx41B6E6 ContrastColor = cx41B6E6 ;
        style GraphData4 "pink" / Color = cxED1F7F ContrastColor = cxED1F7F ;
    end;
run;
```

Using PROC TEMPLATE, the *pearl* parent style has been modified again, creating a new style called *mycolors.* The style statements specify the CHOP colors to be used for each style element (e.g., GraphDataDefault, GraphOutlines, GraphData1, etc.). In this example, the colors are produced using SAS hex codes, which are specified by the 6-character hexidecimal format with the prefix of “cx”. As a reminder of which color is designated by which code, the color name for each style element has been provided in quotation marks before the slash. Then, the graphics code is submitted again, now replacing style=noframe with style=mycolors. In the series graph, the first line (for year 1980, the bottom line here) is dark blue, the second line (for year 1981) is olive green, the third line (for year 1982) is bright blue, and the fourth line (for year 1983) is pink; each line 1 through 4 correspond to style GraphData1 through GraphData4 in the proc template code. In the bar graph, the color of the bars and the bar outlines are both dark blue, designated by the style GraphDataDefault and GraphOutlines statements, respectively. In both graphs, the axis lines are now dark gray per the style GraphAxisLines.

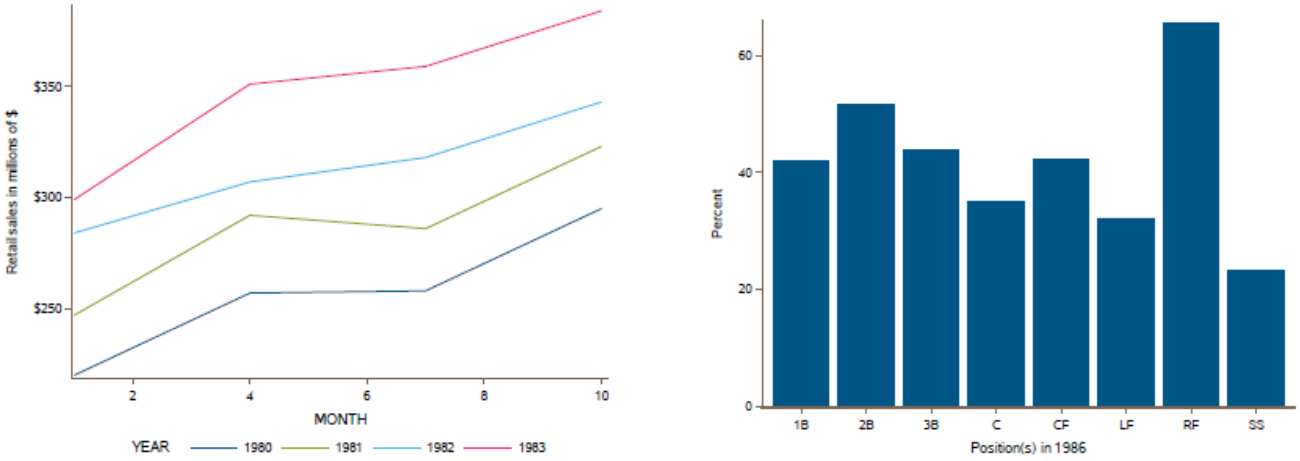

Figure 3: Using your own colors.

SIMPLIFY AXES AND LABELS

In almost every situation, the default axes, which use the variable names, labels, and values, can be improved upon. Axis labels should be concise, but descriptive. In many cases, the axis label can be removed altogether, particularly if a graph title or footnote can be used to augment the information usually given in the axis label. The range of the x- and y-axes should help reflect the main message of the data. In many instances, the axis will reflect a continuous, linear numeric variable, including the y-axes in both examples. In the retail sales graph, the y-axis is in millions of dollars. In the baseball graph, the y-axis shows percent. These type of data benefit from equal spacing of axis values at tick marks and should span at least the range of the data values. Other data that reflect the multiplicative scale, such as risk ratios, should be plotted on a logarithmic scale. The use of the appropriate scale for each axis is important so as to not distort your data. I encourage you to consider the overall dimensions of your graph so that it is not too narrow or too wide, in addition to the orientation on the vertical versus horizontal plane (e.g. should you switch the positions of the x- and y-axes?). The following code demonstrates improvements in the x- and y-axes, including labels and tick values, in addition to the inclusion of graph titles, to produce the graphs in Figure 4.

```
ods pdf style=mycolors file="C\retail_series_fixaxis.pdf";

title1 'Retail sales in millions by quarter (1980-1983)';     ⑤
proc sgplot data=sashelp.retail ;
where 1980<=year<=1983;
series x=month y=sales / group=year ;
keylegend / noborder;
xaxis display=(nolabel) type=discrete offsetmin=0.1 ;      ⑥
yaxis display=(nolabel) values=(200 to 400 by 50) ;      ⑦
format month yrq.;    ⑧
run;

ods pdf close;
```

In the retail data graph, the default x-axis used month as a continuous variable that ranged from 1 to 10, rather than as four discrete time points that correspond to annual quarters (e.g., month 1 is Q1, month 4 is Q2, month 7 is Q3, and month 10 is Q4). The tick marks should reflect these time points rather than every other month. In order to specify tick values that are more pertinent to the graph, the option type=discrete (in ⑥) indicates that the data should be plotted on the x-axis as discrete values. The format statement (in ⑧) formats the values to reflect quarters instead of months. The offset option (in ⑥) allows for a space between the axis origin and the first tick, which makes the graph easier to read.

```
proc sort data=baseballfreq out=baseballfreq_sorted;    ⑨
by descending percent ;
run;

ods pdf style=mycolors file="C:\baseball_bar_fixaxis.pdf";

title1 'Proportion of baseball players with a salary of more than $500K';    ⑩
proc sgplot data=baseballfreq_sorted;
hbarparm category=position response=percent / baselineattrs=(thickness=0);  ⑪
xaxis display=(nolabel noticks) values=(0 to 100 by 10);  ⑫
yaxis display=(nolabel noticks) discreteorder=data;    ⑬
format position $position. percent percent_graph.;    ⑭
run;

ods pdf close;
```

In the baseball data graph, the x- and y-axes are switched (using hbarparm instead of vbarparm in ⑪), so that the player positions can be spelled out (using a format in ⑭) with the text displayed horizontally on the y-axis for ease of reading. If the positions were spelled out on the x-axis, they might need to be horizontal or diagonal in order to fit in the space below each bar. Additionally, the data are resorted by the variable percent with the largest percent sorted first (⑨). With the reordered data, the discreteorder=data option in the y-axis statement

(in ⑬), displays the player position with the highest proportion of players with a salary of more than $500,000 at the top of the graph, then continues downward in decreasing order. This display allows for a quicker and easier interpretation of the data, and is more visually pleasing. The tick marks were unnecessary on both the x- and y-axes, and thus have been removed with display=(noticks) in ⑫ and ⑬.

For both graphs in Figure 4, because descriptive graph titles have been added (in ⑤ and ⑩), the axis labels are unnecessary and have been removed altogether, using display=(nolabel) in ⑥, ⑦, ⑫, and ⑬.

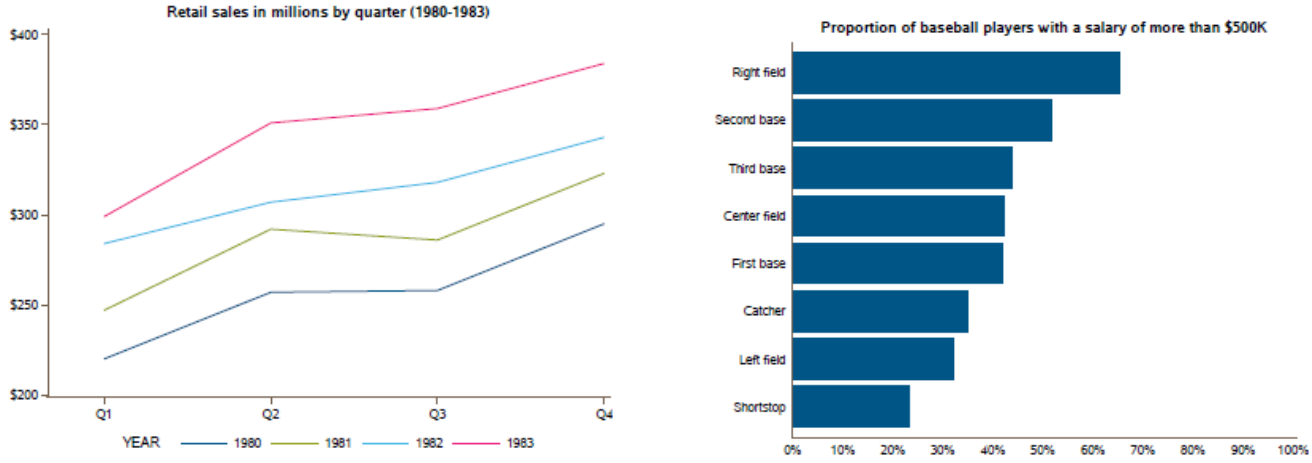

Figure 4. Simplifying axes and labels.

COMMUNICATE YOUR STORY

Finally, each graph can be tweaked to more completely illustrate your results. Effective tweaks include highlighting a specific part of the graph to draw attention to it by adding value labels, changing the dimensions of certain data elements, and using a contrasting color. Titles can be used to explicitly tell your reader the main point. The following code generates the final graphs presented in Figure 5.

```
ods pdf style=mycolors file="C:\retail_series_final.pdf";

title1 'Retail sales (in millions) generally increase quarter-by-quarter and year-to-year';  ⑮
proc sgplot data=sashelp.retail noautolegend;   ⑯
series x=month y=sales / group=year
                         curvelabel curvelabelattrs=(color=black)   ⑰
                         lineattrs=(thickness=4);     ⑱
where 1980<=year<=1983;
xaxis display=(nolabel) type=discrete offsetmin=0.1;
yaxis display=(nolabel) values=(200 to 400 by 50) ;
format month yrq.;
run;

ods pdf close;
```

In the retail graph, the title has been improved upon (⑮) to explain the main point of the graph to the reader. Labels have been added at the end of each line to more clearly indicate the year (⑰). Thus, the legend is redundant and has been removed (⑯) to reduce unnecessary clutter. The thickness of each line is increased for better visibility (⑱).

```
ods pdf style=mycolors file="C:\baseball_bar_final.pdf";

title1 'Only 35% of catchers made more than $500K during the 1986 MLB season';
proc sgplot data=baseballfreq_sorted noautolegend;
hbarparm category=position response=percent
    / baselineattrs=(thickness=0) datalabel datalabelattrs=(size=10);     ⑲

hbarparm category=position response=percent_c         ⑳
    / baselineattrs=(thickness=0) ;
xaxis display=none ;
yaxis display=(nolabel noticks noline) discreteorder=data valueattrs=(size=10);
format position $position. percent percent_graph.;
run;

ods pdf close;
```

For the baseball graph, in addition to adding a more descriptive title, I removed the x-axis altogether and added the text of the percent values at the end of each horizontal bar (⑲). Because I wanted to draw attention to the proportion of catchers who made more than $500,000 annually, I highlighted that bar specifically using a contrasting color. To do this, I added a second hbarparm statement that uses a second variable (percent_c) with just one data point—it has missing values for all positions except catcher. By including this hbarparm statement after the original one, it overlays the one bar it produces in olive green (as the 2nd piece of data being graphed) over the original blue bars (⑳).

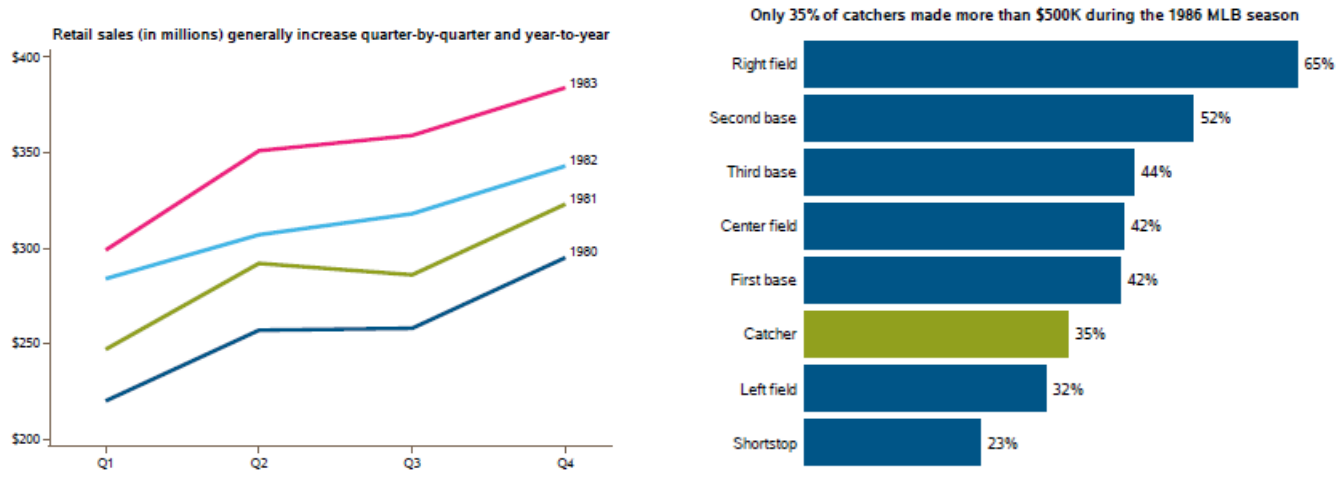

Figure 5. Communicate your story.

CONCLUSION

While I can't help you decide what your data story is, I hope these basic graphing strategies help you tell your data story with a more effective graph. The coding presented in the sections "Maximize White Space" and "Use Your Own Colors" can become standard code that you use in all your graphing programs. The ideas behind the coding in "Simplify Axes and Labels" and "Communicate Your Story" can be adjusted to optimize your own graphs. Effective data visualization is worth the extra effort to help convey your message.

ACKNOWLEDGEMENTS

I'd like to thank Steve Fleming, Brian Metzger, and Melissa Pfeiffer for their thoughtful and constructive comments on this paper. Additionally, I'd like to thank Alli Curry and Meghan Carey for providing the opportunities to help tell important stories with data.

DATA VISUALIZATION RESOURCES

- Color Brewer 2.0. http://colorbrewer2.org
- Evergreen, Stephanie. Evergreen Data: Intentional Reporting and Data Visualization. https://stephanieevergreen.com/
- Research in Action podcast. Dr. Stephanie Evergreen on Data Visualization. Episode #92. Oregon State University. https://ecampus.oregonstate.edu/research/podcast/e92/
- Simmon, Robert. Subtleties of Color. Earth Observatory Blog, NASA. Published 8/5/2013. https://earthobservatory.nasa.gov/blogs/elegantfigures/2013/08/05/subtleties-of-color-part-1-of-6/
- Tufte, Edward. www.edwardtufte.com

SAS GRAPHING RESOURCES

- http://support.sas.com/rnd/app/ODSGraphics/TipSheet_GraphStyle.pdf
- https://www.pharmasug.org/proceedings/2016/DG/PharmaSUG-2016-DG04.pdf
- http://support.sas.com/documentation/prod-p/grstat/9.4/en/PDF/odsbasicg.pdf
- https://support.sas.com/documentation/cdl/en/statug/63347/HTML/default/viewer.htm#statug_odsgraph_sect056.htm
- http://www2.sas.com/proceedings/sugi31/053-31.pdf

CONTACT INFORMATION

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

Kristina B. Metzger
Metzger Consulting
512-825-5543
krisbusmetz@gmail.com

APPENDIX

The following code was used to manipulate *SASHelp.Baseball* for the graph, including limiting the data set to players with a single field polistion only, flagging players with an annual salary of at least $500,000, and calculating the proportion of players in each position with an annual salary of at least $500,000.

```
data baseball;
set sashelp.baseball;
where position in ('1B','2B','3B','C','CF','LF','RF','SS');
if salary>500 then salaryflag=1;
else salaryflag=0;
run;

proc tabulate data=baseball
    out=baseballfreq (keep=position salaryflag pctn_10 where=(position ne ' ' and salaryflag=1)
rename=(pctn_10=Percent)) ;
class position salaryflag;
table salaryflag all,
      all*(n colpctn)
      position*(n colpctn) ;
run;

data baseballfreq;
set baseballfreq;
if position='C' then percent_c=percent;
run;
```

Formats used in this paper:

```
proc format;
value yrq
    1 = 'Q1'
    4 = 'Q2'
    7 = 'Q3'
    10 = 'Q4' ;
value $ position
    '1B' = 'First base'
    '2B' = 'Second base'
    '3B' = 'Third base'
    'C' = 'Catcher'
    'CF' = 'Center field'
    'LF' = 'Left field'
    'RF' = 'Right field'
    'SS' = 'Shortstop' ;
picture percent_graph (round)
    low-high = '009%';
run;
```