Paper 085-2009

Mission Possible: Putting a Table and Multiple Graphs on a Single-Page PDF with ODS and Basic GOPTIONS

Jennifer S. Harper, Baylor Health Care System, Dallas, TX

ABSTRACT

The mission: Report on four cardiac surgery metrics at six different hospitals and summarize the data at the surgeon and system level. Also, show current values as well as trends over time and benchmark them against national averages. Oh, and do this all in a single view of each metric.

The Tools: A simple (or at least straightforward) combination of ODS, GOPTIONS, MACROS, ANNOTATE, PROC REPORT, GPLOT, and GCHART.

The Approach:

- Write table and graph code to make the needed output
- Write Annotate Code to put benchmark data on the graphs
- Use GOPTIONS to resize and position the graphs
- Control page breaks with ODS options

The Output: A PDF document with each metric on a single page with tabular and graphic output that is clear, concise, and not all that hard to make!

Audience: Programmers with familiarity with macros and SAS/Graph

INTRODUCTION

Disclaimer: After the abstract was submitted, I realized I can't publish the real data I used in my project, so instead I'll do the same idea using the sashelp.snacks data set.

The goal of this project is to produce a one-page report that includes a data table, several similar graphs, and one different graph. The approach I took was to break this into two general pieces - write code for each of the individual parts – the table, the repeated graph, the final graph – then figure out how to combine them and make a single-page report.

The "assignment" this paper works on is a fictional request from the manager of the snack food division. We've been asked to produce a one-page PDF report that shows the quarterly income values for each of our four delicious snacks (a table). Also, show me the monthly trends in income for each food type over the past two years (4 similar graphs). And also show me how many units of each we're selling (one more graph).

This paper focuses on the ODS and graphics options that work to assemble the report more than the Report or SAS/Graph code. Really, the approach is robust enough that most any graph code could be plugged in to meet your report needs.

THE CODE PIECES

The first step I took was to do the basic SAS code that would create the pieces. I need to pull data, make the data table, make trend graphs, and a bar chart. After those pieces are done, I'll work on getting them all onto one page.

SET GLOBAL OPTIONS AND DATA SOURCE (SETUP CODE)

My first step was to set some basic global options like suppressing the date and page number from my output. I also took this chance to clear any existing titles and footnotes as to reset all my graphics options. As for the data set, in this example, I select 4 particular snack types, limit the date range, calculate a new variable that we'll be graphing and keep just the few variables I need. In the remainder of this paper, we'll refer to this block of code as the "SETUP CODE".

```
options nodate nonumber ;
title; footnote; goptions reset=all;

data snack_subset; set sashelp.snacks;
    where product in
       ("Buttery popcorn" "Cheese puffs" "Saltine crackers" "Tortilla chips")
        and date<='31dec03'd ;
    income=price*qtysold;
    keep qtysold price date product income;
run;
```

MAKE THE DATA TABLE (PROC REPORT CODE)

My next step was to write some code to make the data table. This example uses basic PROC REPORT code to show the quarterly income for the four product types over the dates included in the data set. In the future, I will refer to this block of code as my “PROC REPORT CODE”.

```
proc report data=snack_subset nowd;
    columns product date,income;
    define product / group " ";
    define date / across format=yyq4. "Annual Income (US$) by Product Type";
    define income / analysis sum f=dollar8.2 " ";
run;
```

MAKE THE INCOME TREND CHARTS (GPLOT TRENDS CODE)

The next part of the report I wrote was the code to produce the trend charts. The task here was to produce four simple charts of the income over time for each of the product types and include a reference line showing management’s target for each product.

These graphs are all identical to each other with only the product name and the sales target changing between graphs. When you repeat code, it usually makes sense to wrap up that code in a macro. In this case, I wrapped up PROC GPLOT code in a macro that has two parameters passed into it – the product name (`&producttype.`) and the target value (`&incometarget.`). I can then call this macro four times to produce the graphs I need. The macro also includes a symbol statement to make the plot show the data points as dots connected by a line. The axis statements are designed to minimize the clutter and make the smaller size hard to read. The symbol and axis statements are included inside the macro as insurance that when I make the trend lines the appropriate symbol and axis definitions are in place. The title statement also includes a reference to the `&producttype.` macro variable so that each graph title, which will be part of the graph, lets the reader know which product we are reporting on.

At this point, we are not worried about the size or position of the charts. We just want to know the code plots the data we are interested in. Later we’ll refer to this code as the “GPLOT TREND CODE”.

```
%macro make_trendgraph(producttype=,incometarget=);
       symbol1 interpol=join value=dot height=.5;
       * x-axis; axis1 label=none minor=none;
       * y-axis; axis2 label=(angle=90 "Total Income") minor=none major=(n=4);
       title1 height=10pt "&producttype.";

       proc gplot data=income_byproduct;
           where product="&producttype.";
           plot income*date /
               vref=&incometarget.
               haxis=axis1 vaxis=axis2;
           format date monyy5. income dollar8.;
       run; quit;
%mend;
```

MAKE THE TOTAL UNITS SOLD CHART (GCHART UNITS CODE)

The final piece of our report that we need to develop is the bar chart along the bottom showing the total number of units sold. This is simple GCHART code and two axis statements. Because we are only calling this code once, we do not need to repeat it so it does not need to be in a macro like the trend code did. Also notice that I chose to use new axis names (axis3 and axis4) rather than re-using axis1 and axis2. This is just good practice and makes sure that the different graphs are kept independent of each other. Like we did in the GPLOT TREND CODE above, notice we include a title here that will stay inside the graph image and serve as the label for this graph.

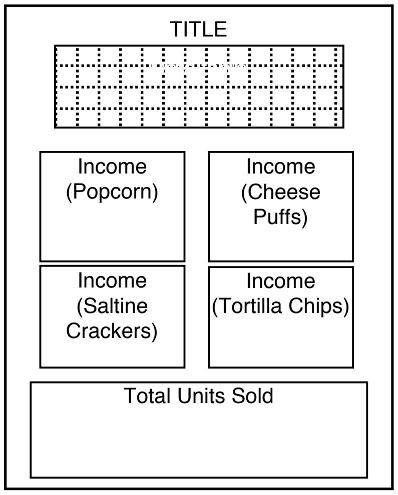
Again, we do not yet need to worry about the size or position of the graph, just that we know the graph is showing our data like we want.

```
* x-axis; axis3 label=none;
* y-axis; axis4 minor=none major=(n=4) label=(angle=90 "Units Sold");
title1 h=10pt  "Total Units Sold for All Product Types";

proc gchart data=snack_subset;
    vbar product / sumvar=qtysold
        noframe outside=sum
        maxis=axis3 raxis=axis4;
run;quit;
```

HOW WILL IT FIT?

Now that we know our data is good to go and we have the code pieces we need to produce the parts of the report, we need arrange those pieces. To figure out how to squeeze all this onto one piece of paper, I sat down with an old-fashioned paper and pencil and simply sketched boxes about the size I wanted things to be. When I had a feel for how to arrange things, I got out a ruler and measured two things for each graph. I need to know (1) the approximate size for my output and (2) where the bottom-left corner of each graph would start. When I came up with a page layout I thought would work and had the measurements I needed, then I moved on to writing the SAS create all the pieces I'd need to include.



In this example, based on experience, I estimated the size of the table resulting frm the PROC REPORT CODE. Based on that I picked a size for the four trend charts that let them be as large as possible while also using some white space between them to visually organize the report. I decided to use 2.5" high x3.75" wide for the four income charts. For the GCHART UNITS SOLD output I decided to use just a little less than the width of the page and then make it the same height as the trend charts – starting with 2.5" high x 8" wide.

To note the location of the charts on the page, you want to measure the distance horizontally and vertically between the bottom-left corner of each graph area and the bottom-left corner of the page. The key here is to remember that you are measuring from the corner of the sheet of paper, not the margins that you may have set up. Also, remember you are measuring to the bottom-left corner of the graphical output area, not the corner of the axis lines. To make the appearance of the 2x2 grid layout, each row of graphs will have the same vorigin (vertical distance from the bottom of the page to the top) and each column will have the same horigin (horizontal distance from the left edge of the page).

In reality the sizing and positioning is probably best accomplished by an iterative process. As I develop a report, I will make and remake a report page many times trying out various sizing, spacing, and positioning options to find a layout that actually fits and is most readable for the target audience.

ASSEMBLING THE PIECES TO MAKE THE GRAPH

With the code pieces completed and tested and a layout idea sketched out and measured, I'm now ready to begin putting the pieces together and including the code that handles the sizing and placement of the graphs.

The approach here is that I will use ODS statements to make a PDF file, choose a style, and suppress default page breaks. With page breaks off, I'll simply rotate through a set of goptions to arrange and size the output.

Once I open the ODS destination, in this case PDF, I make a page title and print the data table using the PROC REPORT CODE.

```
ods pdf file="c:\temp\Snack Food Report.pdf" style=festival;
goptions device=sasprtc target=sasprtc;
ods pdf startpage=never;

title h=14pt "Snack Food Report for SGF2009";
/* PROC REPORT CODE GOES HERE */
```

I then make each of the four graphs. Since all four will have the same size, I use a single goptions statement to set the horizontal (hsize) and vertical (vsize) dimensions of the graph area. I leave those goptions in place until I am ready to make the differently sized GCHARTS UNITS SOLD graph. With the size for the four graphs set, I then use horigin and vorigin options in a goptions statement to set the location for each graph. I will alternate an origin statement with a macro call that puts each graph in a different space. The pattern is to define the size and placement, then make the graph via a macro call. This pattern is repeated four times – each time changing the origin (location) and product names and targets.

```
goptions hsize=3.75in vsize=2.5in ;

goptions horigin=0in vorigin=6in; * top left;
%make_trendgraph(producttype=Buttery popcorn, incometarget=425);
goptions horigin=4.25in vorigin=6in; * top right;
%make_trendgraph(producttype=Cheese puffs, incometarget=375);
goptions horigin=0in vorigin=3in; * bottom left;
%make_trendgraph(producttype=Saltine crackers, incometarget=200);
goptions horigin=4.25in vorigin=3in; * bottom right;
%make_trendgraph(producttype=Tortilla chips, incometarget=400);
```

Now that the four graphs have been made, I change the goptions for size and placement to match what I decided I wanted the final graph to look like. After setting those options I call my GPLOT UNITS CODE and close the PDF destination.

```
goptions hsize=8in vsize=2.5in horigin=0in vorigin=0in;
/* GCHART UNITS CODE GOES HERE */

ods pdf close;
```

That's really all there is to making this report. The secret to getting all the graphs on one page is suppressing the default page break behavior, in using hsize/visze and horigin and vorigin graphics options.

STRENGTHS AND WEAKNESSES OF THIS APPROACH

PLUS: WORKS WITH ANY GRAPH OUTPUT

Because this code uses goptions to size and place a graphics space, you can do whatever SAS/GRAPH procedures you want in the assigned spaces. I've used this model to plot various SPC charts from SAS/QC and also used the new ODS GRAPHICS procedures like SGPLOT. Although I used a macro to put four similar graphs in this report, they could just as easily been four different graphs. The principle is just to size the output with HSIZE and VSIZE and arrange the output with HORIGIN VORIGIN.

PLUS: EXTENSIBLE TO PRODUCE A MULTI-PAGE DOCUMENT

This example focused on putting several graphs on a single page. However if you need to produce a multi-page document that is certainly possible. By putting the statement "`ods pdf startpage=now;`" you can tell SAS to begin a new page of output. After that just begin a new set of graphics options and procedures to meet your needs.

If your goal is to make several pages with the same layout, you can take the approach of wrapping up a page of output such as I have above in a macro. You can pass this macro the parameters you need to change the output. In my example above, you might want to make one page for two different regions. To do this, you'd put a "`startpage=now`" between the two macro calls as shown below.

```
%macro make_page(region=);
    /* Single page output here */
    /* but modified to where region=&region. as appropriate.*/
%mend;
ods pdf file="c:\temp\snack food report by region.pdf";
ods pdf startpage=never;
%make_page(region=East); /* make one page of output */
ods pdf startpage=now; /* put a page break */
%make_page(region=West); /* make a second page of output */
ods pdf close;
```

MINUS: NOT FLEXIBLE FOR A CHANGING OR UNKNOWN NUMBER OF GRAPHS

A key limitation of this approach is that it assumes you already know how many graphs you are going to be placing. This code does not handle fitting an unknown number of graphs into a given area. If my number of snack food products varied each month then it would be more challenging to get this code to work for you. In this case, perhaps the new Graphics Template Language (GTL) or ODS LAYOUT would be worth researching. SAS-L has had a Rumsfeld-ian (see Recommended Reading) solution of making the unknown known by using data or SQL steps to count how many graphs you'll need the dynamically designing a template based on that information. This obviously can get quite complex, but certainly SAS is capable of doing this.

MINUS: NOT GOOD FOR MOVING THE TABLE OR HAVING MULTIPLE DATA TABLES

Another unfortunate limitation is that this approach seems to limit me to putting the data table at the top of a page. If I needed the graph elsewhere on the page or arranged between graphs, it becomes more challenging to rearrange the pieces. In this situation, I think ODS LAYOUT becomes a better option – or perhaps a hybrid of ODS LAYOUT and the goptions used in this approach.

ALTERNATIVE SOLUTION WITH ODS LAYOUT

As with nearly any problem in SAS, there are a few alternative solutions that you may wish to consider based on the needs.

ODS layout has great promise and is probably will become the best way to approach a project like this in the future. However, currently ODS LAYOUT is not well documented in SAS and is inconsistent between versions of SAS. However, if your environment is exclusively V9.2 and above, then ODS LAYOUT merits a good look, especially as SAS improves the documentation of ODS LAYOUT.

CONCLUSION

Like any SAS coding task, there are several ways to solve the problem of multiple graphs on a page. This paper shows that the use of graphics options HSIZE/VSIZE and HORIGIN/VORIGIN is one simple approach to sizing and positioning multiple pages on a single PDF page.

RECOMMENDED READING

Multiple Graphs on One Page Using SAS/GRAPH® V9; Hany Aboutaleb, Biogen Idec, Cambridge, MA; SAS Global Forum 2008 (Paper 222-2008). http://www2.sas.com/proceedings/forum2008/222-2008.pdf

Using SAS®9 ODS Features to Present Table and Graph Data in an Adobe PDF File; Lori S. Parsons, Ovation Research Group, Seattle, WA; SUGI 30 (Paper 172-30). http://www2.sas.com/proceedings/sugi30/172-30.pdf

PDF Can be Pretty Darn Fancy: Tips and Tricks for the ODS PDF Destination; Pete Lund, Looking Glass Analytics, Olympia, WA; SAS Global Forum 2008 (Paper 033-2008) http://www2.sas.com/proceedings/forum2008/033-2008.pdf

Search SAS-L on Google Groups for “Proc Greplay with by groups” for a discussion of an approach to plotting an unknown number of graphs on a single page.

“The unknown”. Department of Defense news briefing (Feb 12, 2002) by Donald Rumsfeld. http://www.slate.com/id/2081042/

Usage Note 19331: Recommended graphics options when creating PDF; SAS Support website (http://support.sas.com/kb/19/331.html)

CONTACT INFORMATION

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

Name: Jennifer Harper, MSPH
Enterprise: Baylor Health Care System
Address: 8080 N Central Expwy, Suite 500
City, State ZIP: Dallas, TX 75206
Work Phone: 214-265-3673
Fax: 214-265-3626
E-mail: jen.harper@baylorhealth.edu

COMPLETE PROGRAM CODE

```
title; footnote;
options nodate nonumber ;
goptions reset=all;

/* Make a small subset of a sashelp data set that picks 4 categories
    the two measures we'll report on are income (price*quantity) and
    quantity */;
data snack_subset; set sashelp.snacks;
    where product in ("Buttery popcorn" "Cheese puffs" "Saltine crackers" "Tortilla
chips")
        and date<='31dec03'd ;
    income=price*qtysold;
    keep qtysold price date product income;
run;

/* Since the graphs in the middle of the page will be identical for each product type,
we can figure out how to make that graph and then just wrap it up in a macro so we can
call it 4 times for each  product type.  The parts that change between graphs are just
the product type that we are  looking at and the income target (reference line);  We
control those by changing the incoming macro parameters */;
%macro make_trendgraph(producttype=,incometarget=);

/* summarize the data to show trends over time for each product */;
proc report data=snack_subset out=income_byproduct nowd;
    where product="&producttype.";
    columns product date income;
    define product / group noprint;
    define date / group format=monyy5. noprint; * groups data by month and year;
    define income / analysis sum noprint;
run;
* sort the data so that the dates are in proper order for graphing;;
proc sort data=income_byproduct; by product date; run;

* a symbol statement sets up what the point markers and lines will look like;
symbol1 interpol=join value=dot height=.5;
* axis statements are used to control the formatting and labels of the x and y axis as
    well as the reference lines;
* x-axis; axis1 label=none minor=none;
* y-axis; axis2 label=(angle=90 "Total Income") minor=none major=(n=4);
```

```
title1 height=10pt "&producttype.";
proc gplot data=income_byproduct;
    where product="&producttype."; * remember to use double quotes around a macro!;
    plot income*date /
        vref=&incometarget.
        haxis=axis1 vaxis=axis2
    ;
    format date monyy5. income dollar8.;
run; quit;
%mend;


ods pdf file="c:\temp\Snack Food Report.pdf" style=festival;
goptions device=sasprtc target=sasprtc;

* startpage=never tells SAS to suppress the default behavior of ODS where each
new piece of output (table/graph) would show up on a new page;
ods pdf startpage=never;

* make a large title that will appear as the page title;
title h=14pt "Snack Food Report for SGF2009";
proc report data=snack_subset nowd
    style(report)={font_size=12pt};
    columns product date,income;
    define product / group " ";
    define date / across format=yyq4. "Annual Income (US$) by Product Type";
    define income / analysis sum f=dollar8.2 " ";
run;

* in this case we have 4 graphs all the same size so we will just set the horizontal
and vertical sizes one time here;
goptions hsize=3.75in vsize=2.5in ;

* use horizontal and vertical origin options to control where the bottom left of
each graph starts - take into consideration your page size and the graph size(s);
goptions horigin=0in vorigin=6in; * top left;
%make_trendgraph(producttype=Buttery popcorn, incometarget=425);
goptions horigin=4.25in vorigin=6in; * top right;
%make_trendgraph(producttype=Cheese puffs, incometarget=375);
goptions horigin=0in vorigin=3in; * bottom left;
%make_trendgraph(producttype=Saltine crackers, incometarget=200);
goptions horigin=4.25in vorigin=3in; * bottom right;
%make_trendgraph(producttype=Tortilla chips, incometarget=400);

* this graph is sized differently so we need to declare new hsize/vsize values
and also tell SAS where the bottom left corner of this graph goes;
goptions hsize=8in vsize=2.5in horigin=0in vorigin=0in;

* x-axis; axis3 label=none;
* y-axis; axis4 minor=none major=(n=4) label=(angle=90 "Units Sold");

title1 h=10pt  "Total Units Sold for All Product Types";
proc gchart data=snack_subset;
    vbar product / sumvar=qtysold
        noframe outside=sum
        maxis=axis3 raxis=axis4
    ;
run;quit;

ods pdf close;
```