Paper 230-2010

Integrating Tables and Graphs with ODS LAYOUT

Steven Feder, Federal Reserve Board, Washington, DC

ABSTRACT

ODS layout of PDF output can arrange multiple tables and graphs on the same page, and even overlay tables onto graphs and vice versa. This can replace GREPLAY coding of multiple graphs with reduction of the most tedious aspects. It also provides easier and more flexible coding of certain elements of graph annotation. Matching style elements such as colors, lines, and symbols of graphs with text helps integrate the appearance for easier user interpretation.

DESTINATIONS

Layout works essentially only to the PDF at this time. The RTF destination by its nature does not handle tables within tables, which is the basis of LAYOUT. The HTML destination will write files when LAYOUT is active, but does not work for absolute layout. The PRINTER destination works with LAYOUT, but this paper will focus exclusively on PDF and assumes throughout that an ODS PDF statement has already started writing to that destination. This paper also assumes that the creation of the test data used in these examples is trivial and does not show the code. The data is not the point.

VERSION AND PLATFORM

All of the output in this paper was produced using SAS® Version 9.2 stage 2 on the Windows platform.

BASIC LAYOUT

For a simple demonstration of a graph followed by a few tables, start LAYOUT and define the REGION for the plot. The LAYOUT statement turns on/off LAYOUT as a whole without defining the output. The REGION statement specifies the dimensions of the writable area. The starting point of the X and Y coordinates is the upper left corner of the output. The following plot area will start 4.25in from the top of the page and 1in from the left margin, with the specified height and width. The code is called within a MACRO to illustrate switching color schemes in the next section.

```
%macro all(c1=,c2=,c3=,b1=,b2=,b3=);
ods layout start;
ods region width=6.5in height=4.25in y=0in x=1in;
```

Symbol statements set the colors of the plot lines. The symbol is varied for visual interest, but this is incidental. The interpolation is for a join line. The PROC GPLOT is simple, three variables plotted against a date:

```
symbol1 color=&c1 f=marker v='C' interpol=join value=dot;
symbol2 color=&c2 f=marker v='U' interpol=join value=dot;
symbol3 color=&c3 interpol=join value=dot;
proc gplot data=in.t;
  plot (b c d)*date/overlay haxis=axis1 vaxis=axis2;
run;
```

To continue with a few tables, specify a REGION for each table. This code lays out three tables at the same vertical location at the bottom of the page:

```
ods region width=1.83in height=3in y=5.75in x=1in;
title 'one';
proc report data=in.t nowd style(report)=[background=&b1.];
  columns date b;
  define b/display;
run;
```

```
ods region width=1.83in height=3in y=5.75in x=3.65in;
  title 'two';
proc report data=in.t nowd style(report)=[background=&b2.];
  columns date c;
  define c/display;
run;

ods region width=1.83in height=3in y=5.75in x=6.5in;
title 'three';
proc report data=in.t nowd style(report)=[background=&b3.];
  columns date d;
  define d/display;
run;
ods layout end;
%mend all;
```

Calling this code with a simple black and white scheme:

```
%all(c1=black,c2=black,c3=black,b1=white,b2=white,b3=white);
```

produces Appendix Layout 1.

BASIC COLOR MATCHING

Setting a basic color scheme that is applied to both the plot symbols (lines) and the tables:

```
%all(c1=blue,c2=green,c3=yellow,b1=blue,b2=green,b3=yellow);
```

produces coordinated output in Appendix Layout 2 with the plot lines the same color as the data that produced them

Or, set the color scheme within a single PROC REPORT table with a color for each DEFINE statement:

```
ods region width=5in height=3in y=5.75in x=1in;
proc report data=in.t nowd;
  columns date b c d;
  define date/display;
  define b/display style(column)=[background=&c1];
  define c/display style(column)=[background=&c2];
  define d/display style(column)=[background=&c3];
run;
```

The same color scheme and MACRO call as above produces a single multi-colored table coordinating the plot lines with the data in Appendix Layout 3.

OVERLAY A TABLE ONTO A GRAPH

To overlay the multi-colored table just created onto the graph, switch the REGION to print to a smaller section of the the same area as the graph:

```
ods region width=2in height=3in y=.5in x=5in;
```

Then continue with PROC REPORT code as before. The output in Appendix Layout 4 is obviously custom designed to fit the table into the plot white space. This sort of trick will not work for all applications.

FURTHER INTEGRATE A TABLE WITH A GRAPH

If the plot has a horizontal reference line:

```
plot (b c d)*date/overlay haxis=axis1 vaxis=axis2
```

```
  href=('03JUN07'd) chref=red;
```

coordinate the line color with the data by defining the color for an individual value in the PROC REPORT with a COMPUTE statement:

```
compute date;
  if put(date,date7.)='03jun07' then do;
    call define('date',"style","style=
      [font_style=italic background=red]");
  end;
endcomp;
```

This produces Appendix Layout 5.

ARRANGE A PIE CHART AT THE CENTER

Next, using a different test data set, put a pie chart at the center of four tables listing the data summarized on the pie chart. The LAYOUT code begins similarly, but a PATTERN statement controls the colors of the pie slices:

```
%macro all(b1=,b2=,b3=,b4=);
ods layout start;
ods region width=6.5in height=4.25in y=2in x=1in;

pattern1 color=&b1;
pattern2 color=&b2;
pattern3 color=&b3;
pattern4 color=&b4;

title font=swiss height=.3in 'graph title';
footnote font=swiss height=.25in 'footnote';
proc gchart data=in.t;
  pie type/sumvar=v1;
run;
quit;
```

The four PROC REPORTS surrounding the chart read the same data, with the WHERE statement selecting the observations of the corresponding value of TYPE. The coding of the background color is the same as before:

```
ods region width=1.83in height=3in y=.5in x=.4in;
title 'one';
proc report data=in.t nowd style(report)=[background=&b1.];
  columns type v1;
  define v1/display;
  where type='a';
run;
ods region width=1.83in height=3in y=.5in x=6in;
title 'two';
proc report data=in.t nowd style(report)=[background=&b2.];
  columns type v1;
  define v1/display;
  where type='b';
run;
ods region width=1.83in height=3in y=5.75in x=.4in;
title 'three';
proc report data=in.t nowd style(report)=[background=&b3.];
  columns type v1;
  define v1/display;
  where type='c';
run;
ods region width=1.83in height=3in y=5.75in x=6in;
title 'three';
proc report data=in.t nowd style(report)=[background=&b4.];
  columns type v1;
  define v1/display;
```

```
  where type='d';
run;

 footnote;
 ods layout end;
 %mend;

 %all(b1=blue,b2=yellow,b3=green,b4=purple);
```

See the output in Appendix Star.

MAP EXAMPLE

Assuming as an example a U.S. map data set with response data for only Pennsylvania, California, and Florida, again use the PATTERN statement described above to match the colors on the map with the PROC REPORT colors set in the COMPUTE statement:

```
ods layout start;
ods region width=4.5in height=4.25in y=0in x=.5in;
pattern1 color=blue;
pattern2 color=yellow;
pattern3 color=green;
proc gmap data=in.t4 map=in.lower48;
  id state;
  choro b_mean / discrete nolegend;
run;
quit;

ods region width=5in height=3in y=1in x=4in;
proc report data=in.t6 nowd;
  columns state b_mean;
  define state/display;
  define b_mean/display format=2.; * style(column)=[background=&b1];
  where state in('ca','pa','fl');
  compute state;
    if state='CA' then do;
      call define('state',"style","style [background=blue]");
    end;
    if state='PA' then do;
      call define('state',"style","style=[background=yellow]");
    end;
    if state='FL' then do;
      call define('state',"style","style=[background=green]");
    end;
  endcomp;
run;
```

See Appendix Map 1 for the output.

Next, as in previous examples, overlay the tables onto the graph by keeping the GMAP code above and substituting for the above PROC REPORT one table for each state written onto the assigned REGION:

```
ods region width=2in height=2in y=1in x=1.3in;
proc report data=in.t6 nowd  style(report)=[background=&c1] ;
  columns state b_mean;
  define state/display;
  define b_mean/display format=2. style(column)=[background=&c1];
  where state in('ca');
run;

ods region width=2in height=2in y=.5in x=4.75in;
proc report data=in.t6 nowd  style(report)=[background=&c2];
  columns state b_mean;
  define state/display;
  define b_mean/display format=2. style(column)=[background=&c2];
  where state in('pa');
```

```
run;

ods region width=2in height=2in y=2.3in x=3.5in;
proc report data=in.t6 nowd style(report)=[background=&c3];
  columns state b_mean;
  define state/display;
  define b_mean/display format=2.;
  where state in('fl');
run;
```

See Appendix Map 2 for the output.

WATCH FOR THIS WARNING

This warning may appear in the log when using layout:

WARNING: THE ABSOLUTE REGION WAS TOO SMALL TO ACCOMMODATE THE TEXT SUPPLIED. OUTPUT WAS LOST.

This occurs when the output from any PROC does not fit into the REGION defined for it. This may not be a problem because the lost output is white space or otherwise non-essential. Or, it might not be a problem because it is obvious and corrected during development. In a production job, however this could be a concern. If the number of observations in a PROC PRINT increased, code that had been running without a problem could begin generating a warning--producing code that is truncated in a way that is not obvious. This is a good reason to check production code logs, whether manually or with a utility program.

CONCLUSIONS

LAYOUT is not particularly complicated, but it does provide substantial new capabilities. Using LAYOUT to integrate graphics and text may be a functionality in search of a use. However, it does look good, and users will probably find a use eventually. They're like that. In the interim, it may substitute nicely for GREPLAY in a few instances.

REFERENCES

Schellenberger, Brian T. "ODS LAYOUT: Arranging ODS Output as You See Fit." SUGI 28.

ACKNOWLEDGMENTS

CONTACT INFORMATION

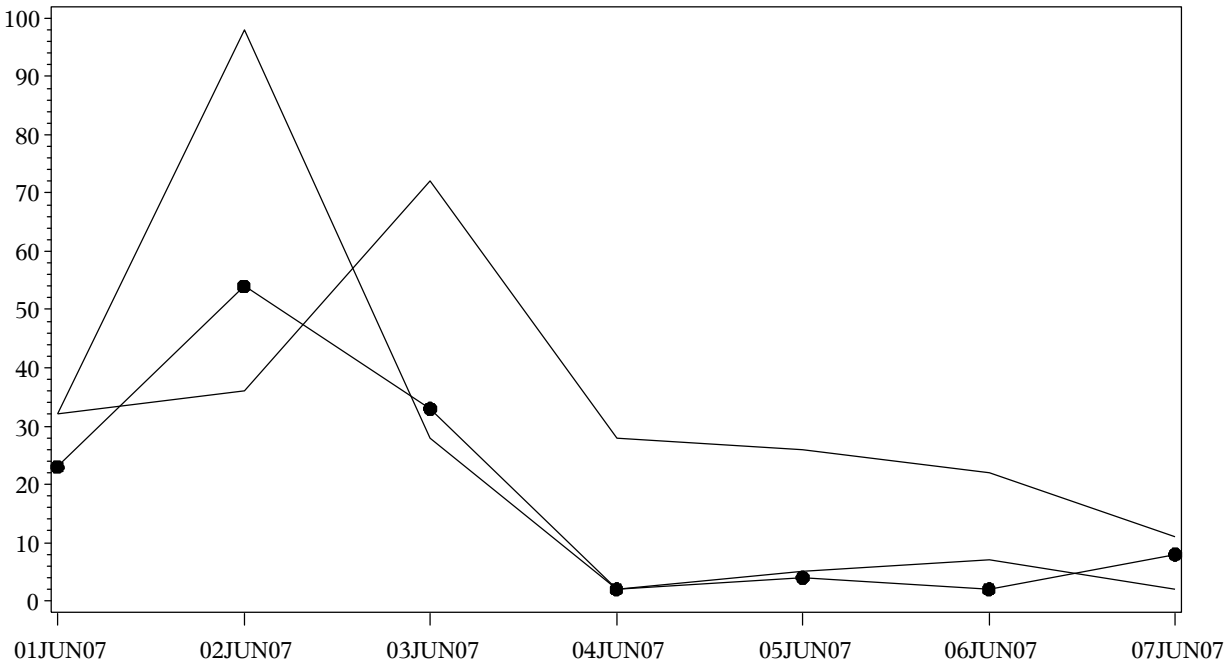
Your comments and questions are valued and encouraged. Contact the author at:

Steven Feder
Federal Reserve Board
20th & C Streeets
Washington, DC 20551
202-452-3144
Steven.h.feder@frb.gov

:

Layout 1

Second Title



Axis Label 1

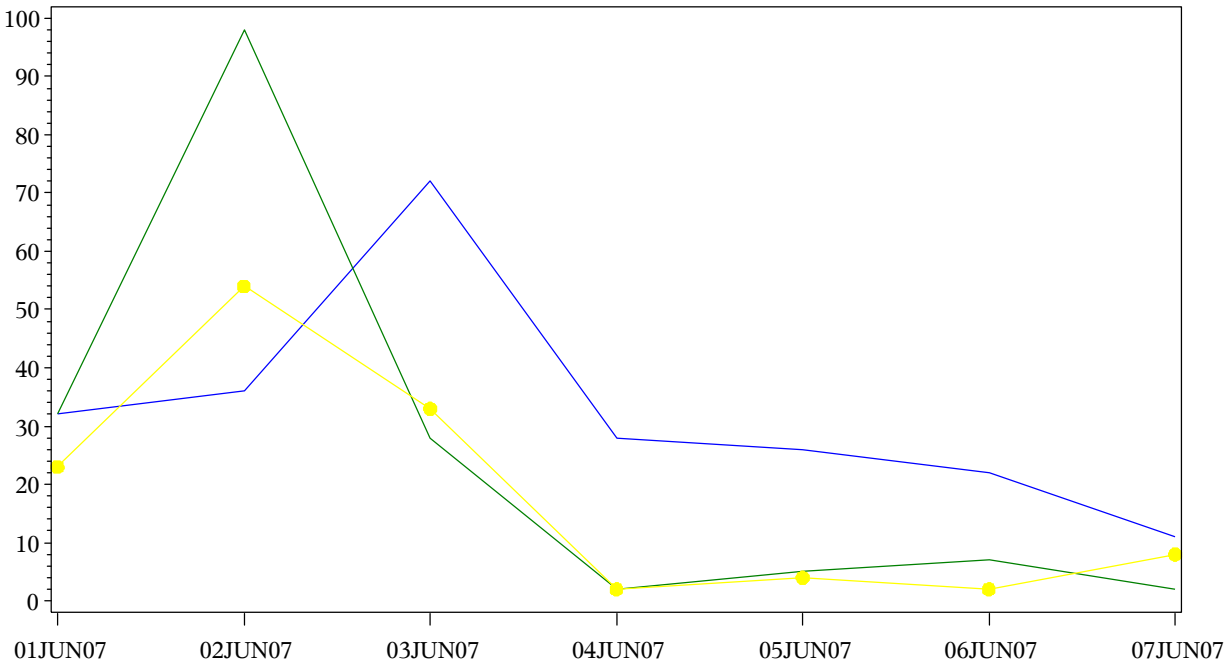
date	b
01JUN07	32
02JUN07	36
03JUN07	72
04JUN07	28
05JUN07	26
06JUN07	22
07JUN07	11

date	c
01JUN07	32
02JUN07	98
03JUN07	28
04JUN07	2
05JUN07	5
06JUN07	7
07JUN07	2

date	d
01JUN07	23
02JUN07	54
03JUN07	33
04JUN07	2
05JUN07	4
06JUN07	2
07JUN07	8

Layout 2

Second Title



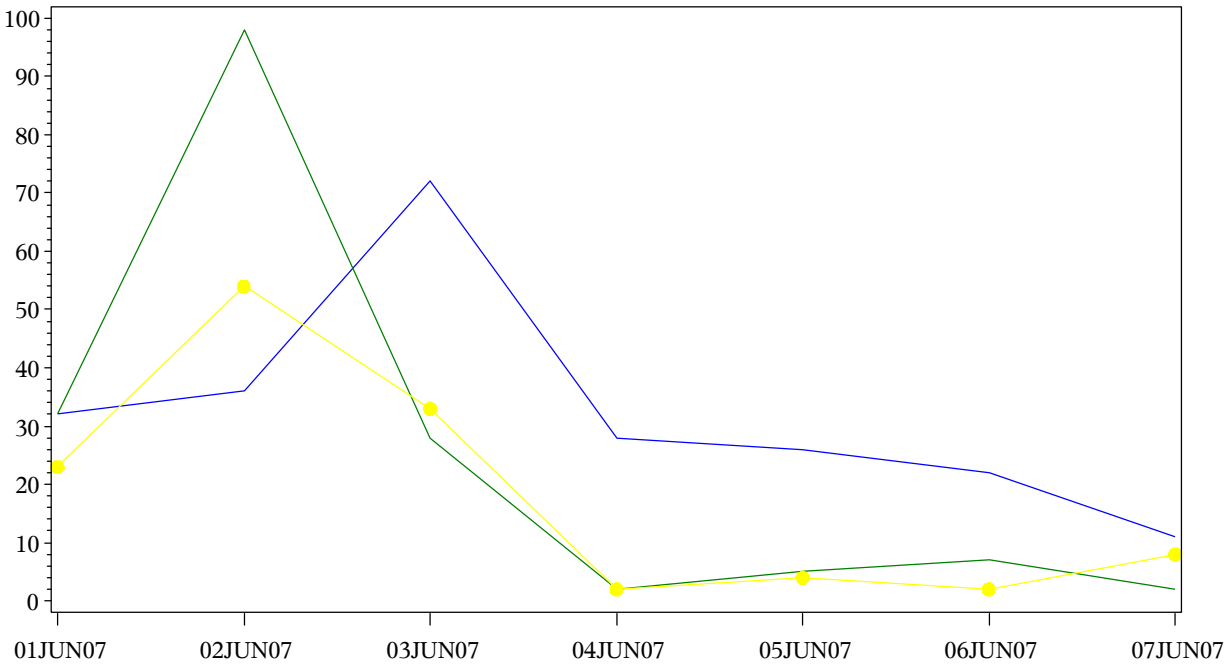
date	b
01JUN07	32
02JUN07	36
03JUN07	72
04JUN07	28
05JUN07	26
06JUN07	22
07JUN07	11

date	c
01JUN07	32
02JUN07	98
03JUN07	28
04JUN07	2
05JUN07	5
06JUN07	7
07JUN07	2

date	d
01JUN07	23
02JUN07	54
03JUN07	33
04JUN07	2
05JUN07	4
06JUN07	2
07JUN07	8

Layout 3

Second Title

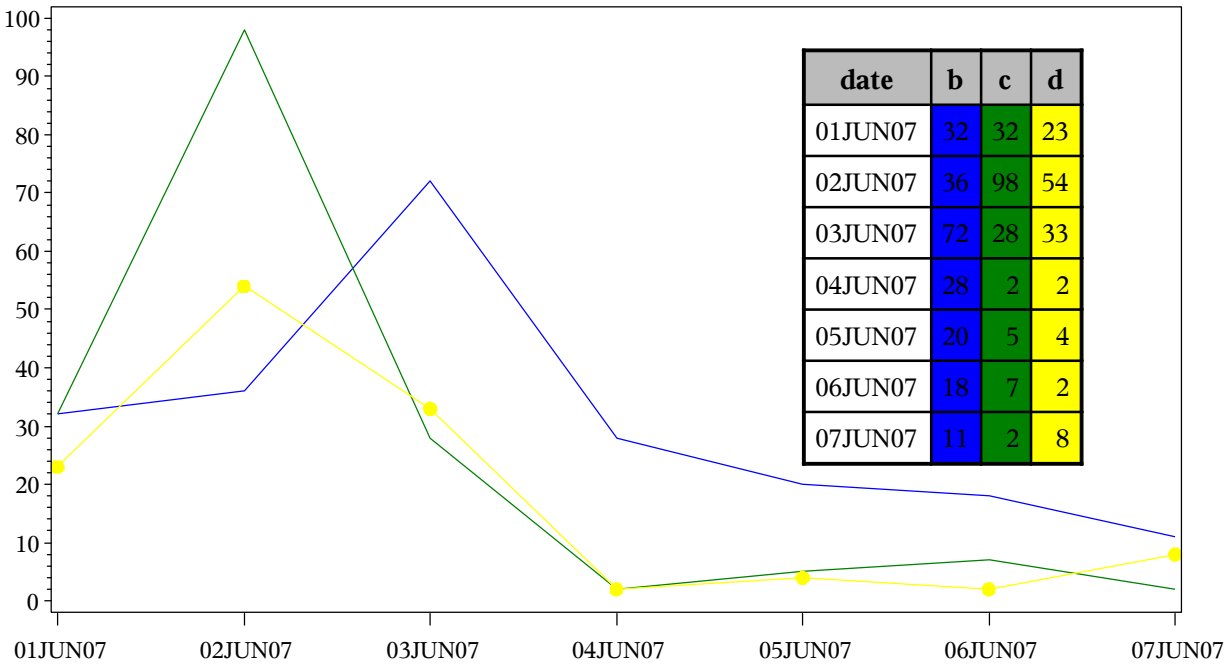


Axis Label 1

date	b	c	d
01JUN07	32	32	23
02JUN07	36	98	54
03JUN07	72	28	33
04JUN07	28	2	2
05JUN07	26	5	4
06JUN07	22	7	2
07JUN07	11	2	8

Layout 4

Second Title

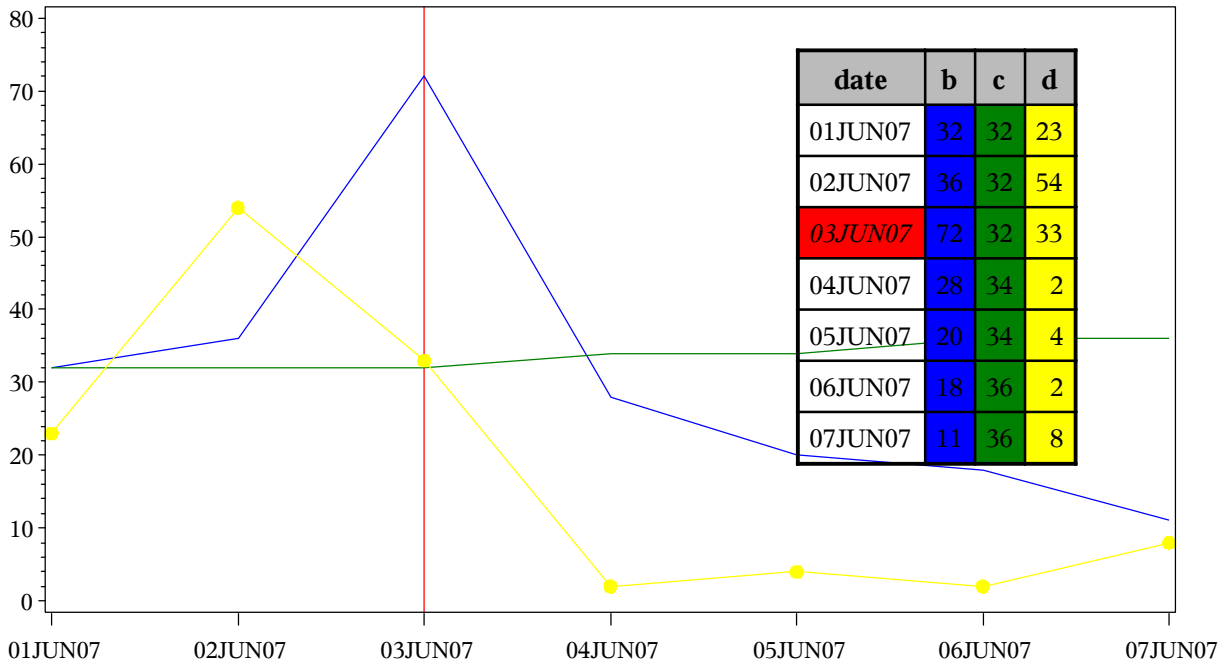


date	b	c	d
01JUN07	32	32	23
02JUN07	36	98	54
03JUN07	72	28	33
04JUN07	28	2	2
05JUN07	20	5	4
06JUN07	18	7	2
07JUN07	11	2	8

Axis Label 1

Layout 5

Second Title



date	b	c	d
01JUN07	32	32	23
02JUN07	36	32	54
03JUN07	72	32	33
04JUN07	28	34	2
05JUN07	20	34	4
06JUN07	18	36	2
07JUN07	11	36	8

Axis Label 1