

Paper 3713-2019
Using ODS EXCEL to Integrate Tables, Graphics, and Text into
Multi-Tabbed Microsoft Excel Reports
 Caroline Walker, Warren Rogers LLC

ABSTRACT

Do you have a complex report involving multiple tables, text items, and graphics that could best be displayed in a multi-tabbed spreadsheet format? The Output Delivery System (ODS) EXCEL destination, introduced in SAS® 9.4, enables you to create Microsoft Excel workbooks that easily integrate graphics, text, and tables, including column labels, filters, and formatted data values. In this paper, we examine the syntax used to generate a multi-tabbed Excel report that incorporates output from the REPORT, PRINT, SGPLOT, and SGPANEL procedures.

INTRODUCTION

With the introduction of the ODS destination for Excel, creating complex and professional quality Excel workbooks has become as convenient and straightforward as generating PDF reports. This paper will demonstrate the use of essential syntax needed to quickly begin producing multi-tabbed Excel workbooks which incorporate all the niceties you might expect in a professional report such as titles, tab names, plots, and tables with column labels, filters, flyover text, formatted values, and even traffic lighting. The code presented will be used to create an Excel workbook containing the three worksheets shown in Figures 1-3.

	A	B	C	D	E
1	Terminal Comparisons				
2	January 1 - February 1, 2017				
3					
4	Terminal	# Deliveries in Analysis	Shortage Across All Deliveries (Gallons)	Overall % Short This Period	Change Since Last Reported
5	Terminal 1	39	-2,087	-.53%	-.18%
6	Terminal 2	42	-939	-.31%	-.03%
7	Terminal 3	457	-10,524	-.29%	.05%
8	Terminal 4	68	-2,191	-.47%	.03%
9	Terminal 5	605	-6,109	-.13%	.03%
10	Terminal 6	40	-1,498	-.52%	-.06%
11	Terminal 7	374	-17,352	-.49%	.03%
12	Terminal 8	347	-12,253	-.37%	-.06%
13	Terminal 9	73	-132	-.04%	-.08%
14	Terminal 10	36	-1,183	-.27%	.06%
15	Terminal 11	183	-5,322	-.3%	.08%
16	Terminal 12	123	-4,176	-.46%	.01%
17					
18	Overall Shortage				
19					
20	Total Net Deliveries in Analysis (Gallons)	Total Net Shortage in Analysis (Gallons)	Overall % Short		
21	12,404,913	-39,258	-.32%		
22					

Figure 1

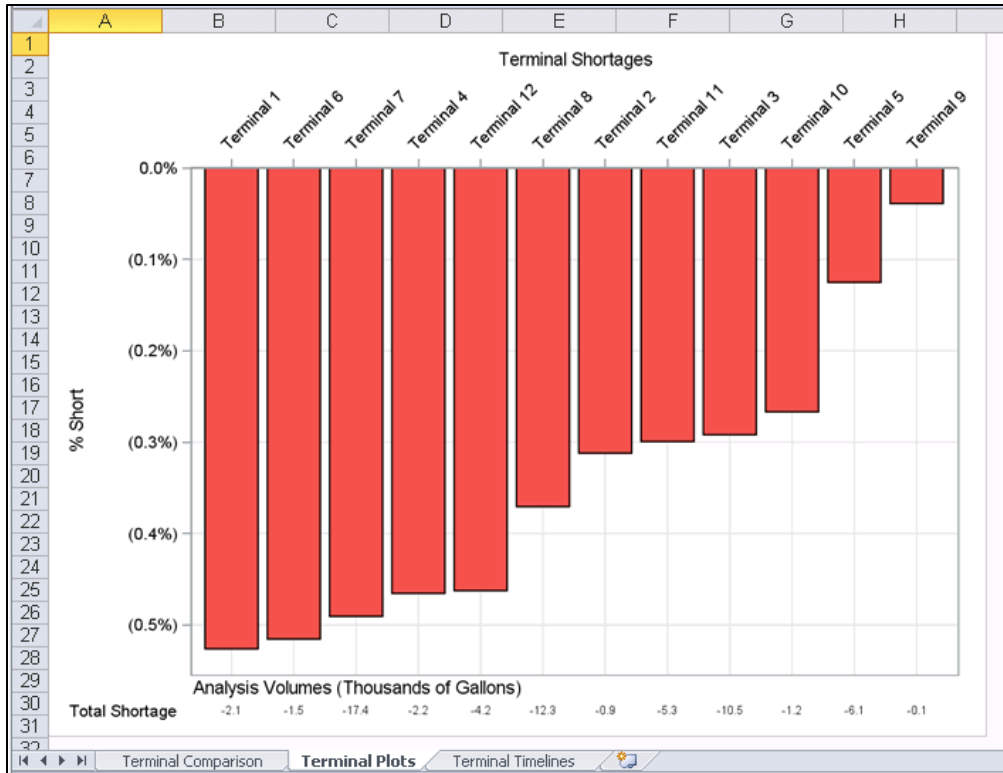


Figure 2

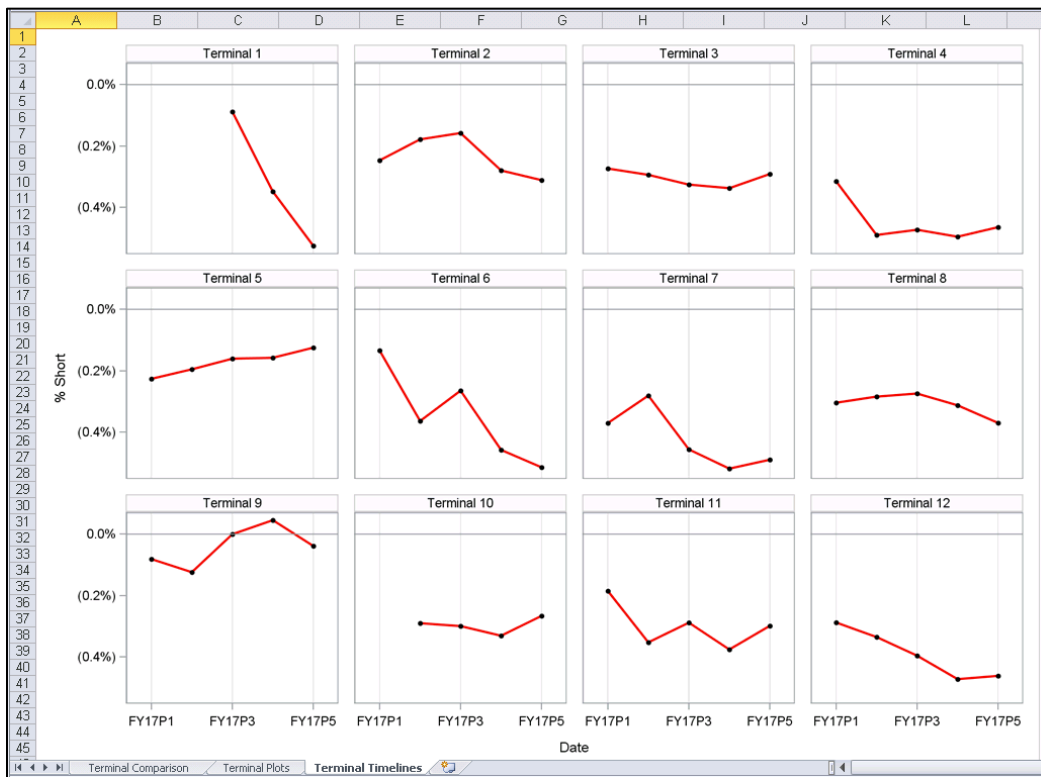


Figure 3

BACKGROUND AND EXAMPLE DATA

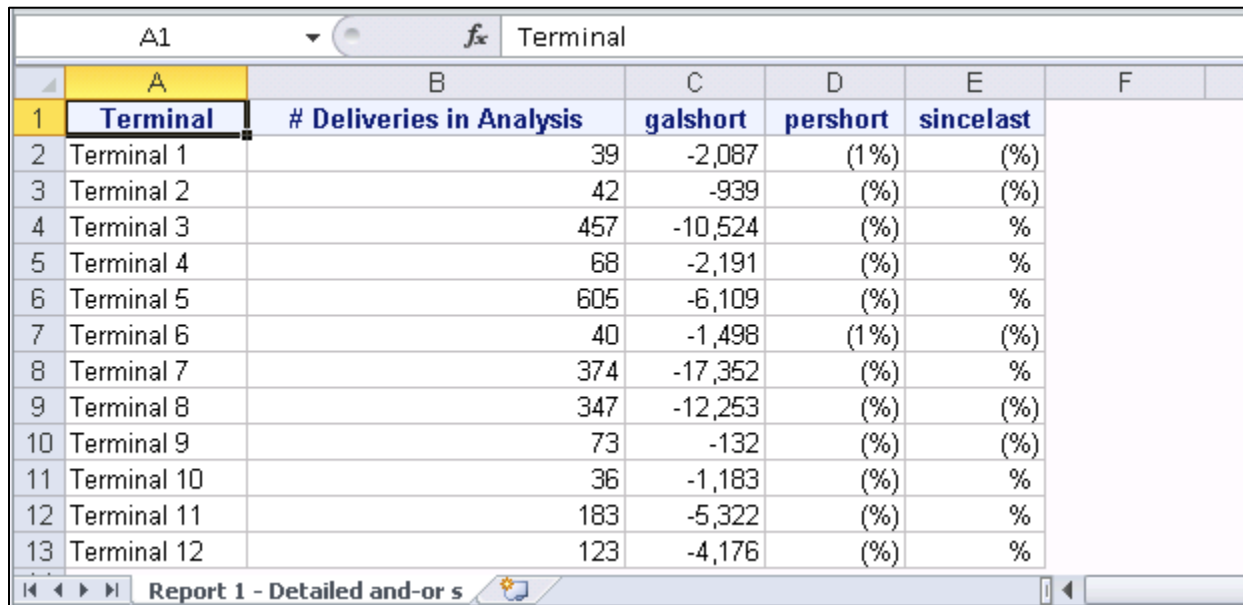
Warren Rogers LLC is a consulting company providing precision fuel system diagnostics to both truck stop and convenience store gas stations. When our clients receive deliveries of fuel into underground storage tanks, the actual amount of fuel delivered may or may not match the amount of fuel for which they were billed. Our sophisticated tank modeling algorithms and precision monitoring capabilities allow us to accurately determine actual delivery amounts. We then provide delivery reconciliation services for our clients, and also terminal analysis reporting, in which we look at patterns in delivery shortages (or overages) specific to each of the many terminals which have supplied the deliveries. The example workbook and data sets shown in this paper compare shortage trends observed in a set of twelve example terminals.

GETTING STARTED: THE ODS SANDWICH AND PROC REPORT

If you have used the ODS destination to produce PDF or HTML files in the past then the syntax for creating an Excel file should look excitingly familiar. If you are new to the ODS destination it may still be exciting to see that you can create your first basic Excel file from SAS with just three lines of code:

```
ods excel file="c:\Terminal Report.xlsx" ; ①
proc report data = demoreport; run;       ②
ods excel close;                          ③
```

That's right, to begin creating an Excel document from SAS you need only place the procedure of your choice (PROC REPORT is used here) between the starting and ending lines above. The first line opens the ODS EXCEL destination and specifies a filename for the workbook to be created. The third line closes the ODS EXCEL destination and marks the end of the file creation. Collectively these two lines of code are sometimes referred to as the "ODS wrapper" or the "bread" of the "ODS sandwich". All output generated between these two statements will be written to the Excel file named in line ①. For simplicity we have included only a single REPORT procedure within the ODS wrapper here. As we progress we will add many additional statements, but the first and the last lines of the code will remain unchanged.



	A	B	C	D	E	F
1	Terminal	# Deliveries in Analysis	galshort	pershort	sincelast	
2	Terminal 1	39	-2,087	(1%)	(%)	
3	Terminal 2	42	-939	(%)	(%)	
4	Terminal 3	457	-10,524	(%)	%	
5	Terminal 4	68	-2,191	(%)	%	
6	Terminal 5	605	-6,109	(%)	%	
7	Terminal 6	40	-1,498	(1%)	(%)	
8	Terminal 7	374	-17,352	(%)	%	
9	Terminal 8	347	-12,253	(%)	(%)	
10	Terminal 9	73	-132	(%)	(%)	
11	Terminal 10	36	-1,183	(%)	%	
12	Terminal 11	183	-5,322	(%)	%	
13	Terminal 12	123	-4,176	(%)	%	

Figure 4

Figure 4 shows the worksheet created by those three lines of code. Although this output still leaves much to be desired, there are a few aspects to be appreciated already. It is worth noting that any variables which had been assigned labels or formats in the data set DEMOREPORT are automatically displayed using those labels and formats in the Excel worksheet. Examples of this can be seen in the variable

GALSHORT, which is displayed using the format comma10.0 in column C of Figure 4. In columns A and B, the column headings reflect the variable labels of the variables displayed, rather than the variable names. For illustrative purposes no other variables were assigned labels in the DEMOREPORT data set, instead additional column headings will be assigned within the REPORT procedure in the next code example.

Although most SAS formats map correctly to Excel formats by default, percent formats which include decimal allowances are an exception in SAS 9.4 Maintenance Release 3. This is evidenced in columns D and E of Figure 4. Although the variables PERSHORT and SINCELAST have the format percent6.4 in the data set DEMOREPORT, they are not displayed with that format in the Excel worksheet. A workaround for this, kindly provided by Chevell Parker at SAS Technical Support, is to specify tagattr="format:###.##\%" for the style of those columns in the PROC REPORT statement itself. This is also illustrated in the next code example. Note, this fix is not needed if you are using SAS 9.4 Maintenance Release 4 or later.

To begin improving the output shown in Figure 4, you can expand the basic REPORT procedure called within the ODS wrapper to take advantage of some of the many data presentation refinements that procedure provides. As illustrated in the code below, the REPORT procedure allows for easy specification of column headings as part of the define statements for each variable. You can also specify individual column widths using the style(column) option within the define statement for each variable:

```
ods excel file="c:\Terminal Report.xlsx" ;

proc report data = demoreport;
column terminal_id ndels galshort pershort sincelast;
define terminal_id /display style(column)=[cellwidth=1in];
define ndels /display style(column)=[cellwidth=1in];
define galshort / display style(column)=[cellwidth=1in] "Shortage
Across All Deliveries (Gallons)";
define pershort / display style(column)=[cellwidth=1.5in
tagattr="format:####.##\%" fontweight = bold] "Overall % Short This
Period";
define sincelast / display style(column)=[cellwidth=1in
tagattr="format:###.##\%"] "Change Since Last Reported";
run;

ods excel close;
```

This code produces the worksheet shown in Figure 5. Note that the data values in columns D and E are now correctly displayed in percent formats with decimal allocations. This is a result of specifying tagattr="format:###.##\%" in the column style specifications for the variables PERSHORT and SINCELAST.

	A	B	C	D	E
1	Terminal	# Deliveries in Analysis	Shortage Across All Deliveries (Gallons)	Overall % Short This Period	Change Since Last Reported
2	Terminal 1	39	-2,087	-.53%	-.18%
3	Terminal 2	42	-939	-.31%	-.03%
4	Terminal 3	457	-10,524	-.29%	.05%
5	Terminal 4	68	-2,191	-.47%	.03%
6	Terminal 5	605	-6,109	-.13%	.03%
7	Terminal 6	40	-1,498	-.52%	-.06%
8	Terminal 7	374	-17,352	-.49%	.03%
9	Terminal 8	347	-12,253	-.37%	-.06%
10	Terminal 9	73	-132	-.04%	-.08%
11	Terminal 10	36	-1,183	-.27%	.06%
12	Terminal 11	183	-5,322	-.3%	.08%
13	Terminal 12	123	-4,176	-.46%	.01%

Figure 5

ADDING FILTERS, TAB NAMES, AND TITLES WITH ODS EXCEL OPTIONS

While the worksheet is already looking much nicer, much more quickly than it might have with an alternate technique, a few simple modifications added via an ODS EXCEL OPTIONS statement can make the output even more polished and useful. Filters, tab names (also called sheet names) and titles can be added to create the output shown in Figure 6.

	A	B	C	D	E
1	Terminal Comparisons				
2	January 1 - February 1, 2017				
3					
4	Terminal	# Deliveries in Analysis	Shortage Across All Deliveries (Gallons)	Overall % Short This Period	Change Since Last Reported
5	Terminal 1	39	-2,087	-.53%	-.18%
6	Terminal 2	42	-939	-.31%	-.03%
7	Terminal 3	457	-10,524	-.29%	.05%
8	Terminal 4	68	-2,191	-.47%	.03%
9	Terminal 5	605	-6,109	-.13%	.03%
10	Terminal 6	40	-1,498	-.52%	-.06%
11	Terminal 7	374	-17,352	-.49%	.03%
12	Terminal 8	347	-12,253	-.37%	-.06%
13	Terminal 9	73	-132	-.04%	-.08%

Figure 6

FILTERS

One of many reasons end-users may prefer to receive reports as Excel worksheets rather than as PDFs is to facilitate easy sorting and sub-setting of the data through the use of column filters. When creating reports with the ODS EXCEL destination, filters can be assigned to variables of interest within SAS, so that end users can immediately begin interacting with the data. Adding filters requires only the addition of

an ODS EXCEL OPTIONS statement, with an AUTOFILTER definition specifying the columns for which filters are desired. Example code is shown below.

```
ods excel file="c:\Terminal Report.xlsx" ;
ods excel options(autofilter="1-5");
run;
proc report ...

ods excel close;
```

In this code, filters are specified for columns 1-5 of the report. Since this specific report has only five columns, the alternative syntax autofilter = 'all' could be used to produce the same results.

TAB NAMES

When generating Excel worksheets using the ODS EXCEL destination SAS will automatically assign a name to each worksheet created. This can be seen in the lower left corner of Figures 4 and 5, where the name supplied by SAS is shown: "Report 1 = Detailed and-or s". Adding a SHEET_NAME specification to the ODS EXCEL OPTIONS statement allows you to change this to a more appropriate name of your own choosing:

```
ods excel file="c:\Terminal Report.xlsx" ;
ods excel options(autofilter="1-5" sheet_name = "Terminal Comparison");
run;

proc report ...

ods excel close;
```

TITLES

Finally, you can specify EMBEDDED_TITLES = 'yes' within the ODS EXCEL OPTIONS statement to allow titles and subtitles to display directly on the worksheet, as shown in Figure 6 (footnotes work as well). Once EMBEDDED_TITLES = 'yes' is specified, TITLE statements can be used to define any titles and subtitles necessary for the report. In the example code below the option "j = l" is used in the TITLE definitions so that the titles will be left justified, which may be preferable for a worksheet format:

```
ods excel file="c:\Terminal Report.xlsx" ;
ods excel options(autofilter="1-5" sheet_name = "Terminal Comparison"
embedded_titles='yes'); run;

title j = 1 "Terminal Comparisons";
title2 j = 1 "January 1 - February 1, 2017";
run;

proc report ...

ods excel close;
```

ADDING TRAFFIC LIGHTING AND POP-UP TEXT WITH PROC REPORT

TRAFFIC LIGHTING

One of many nice functionalities of the REPORT procedure is the ability to assign customized colors to certain columns, rows, or individual cells of a report based on the values of data items within the report. This is known as traffic lighting, and the good news is, the ODS EXCEL destination now enables this to

be achieved as easily in an Excel worksheet as it could previously be done in PDF and other output formats.

For those unfamiliar with traffic lighting in PROC REPORT, the paper "Turn Your Plain Report into a Painted Report Using ODS Styles " by Cynthia Zender and Allison Booth (2013) may be a useful reference to explore. The code below illustrates implementing traffic lighting on the example DEMOREPORT data set. Each row of the worksheet will be color coded according to the value of the variable SINCELAST (shown in the column labeled "Change Since Last Reported"):

```
ods excel file="c:\Terminal Report.xlsx" ;
ods excel options(autofilter="1-5" sheet_name = "Terminal Comparison"
embedded_titles='yes'); run;

title j = 1 "Terminal Comparisons";
title2 j = 1 "January 1 - February 1, 2017";
run;

proc report data = demoreport;
column terminal_id ndels galshort pershort sincelast;
define terminal_id /display style(column)=[cellwidth=1in];
define ndels /display style(column)=[cellwidth=1in];
define galshort / display style(column)=[cellwidth=1in] "Shortage
Across All Deliveries (Gallons)";
define pershort / display style(column)=[cellwidth=1.5in
tagattr="format:####.##\%" fontweight = bold] "Overall % Short This
Period";
define sincelast / analysis sum style(column)=[cellwidth=1in
tagattr="format:####.##\%"] "Change Since Last Reported";
compute sincelast;
    if sincelast.sum ne . then do;
        if sincelast.sum < -0.001 then call define(_row_,"style",
"style={background= light red}");
        else if sincelast.sum < -0.0005 then call
define(_row_,"style", "style={background= very light red}");
        else if sincelast.sum > 0.001 then call
define(_row_,"style", "style={background= light green}");
        else if sincelast.sum > 0.0005 then call
define(_row_,"style", "style={background= very light green}");
    end;
endcomp;
run;

ods excel close;
```

The worksheet generated from this code is shown in Figure 7. Terminals with improved performance are now shown in green while those with deteriorating performance are shown in light or dark red depending on the degree of deterioration. Any terminal whose performance has changed by fewer than .05 percentage points since the last report is left un-highlighted.

	A	B	C	D	E
1	Terminal Comparisons				
2	January 1 - February 1, 2017				
3					
			Shortage Across All Deliveries (Gallons)	Overall % Short This Period	Change Since Last Reported*
4	Terminal	# Deliveries in Analysis			
5	Terminal 1	39	-2,087	-.53%	-.18%
6	Terminal 2	42	-939	-.31%	-.03%
7	Terminal 3	457	-10,524	-.29%	.05%
8	Terminal 4	68	-2,191	-.47%	.03%
9	Terminal 5	605	-6,109	-.13%	.03%
10	Terminal 6	40	-1,498	-.52%	-.06%
11	Terminal 7	374	-17,352	-.49%	.03%
12	Terminal 8	347	-12,253	-.37%	-.06%
13	Terminal 9	73	-132	-.04%	-.08%
14	Terminal 10	36	-1,183	-.27%	.06%
15	Terminal 11	183	-5,322	-.3%	.08%
16	Terminal 12	123	-4,176	-.46%	.01%

Figure 7

POP-UP TEXT

Another nice feature of the REPORT procedure is the ability to create flyover text (also known as pop-up text, or hover over text) that is associated with individual columns of the table. This text is displayed when a user hovers over or clicks on that column of the table. With the advent of the ODS EXCEL destination this feature can now be easily utilized in Excel workbooks.

In PROC REPORT syntax, pop-up text can be added to any column of a table by defining a FLYOVER value in the column style of the DEFINE statement for that variable. The code below illustrates adding the pop-up text "Here is a helpful note" to column C (the GALSHORT variable) of the report:

```
ods excel file="c:\Terminal Report.xlsx" ;
ods excel options(autofilter="1-5" sheet_name = "Terminal Comparison"
embedded_titles='yes'); run;

title j = 1 "Terminal Comparisons";
title2 j = 1 "January 1 - February 1, 2017";
run;

proc report data = demoreport;
column terminal_id ndels galshort pershort sincelast;
define terminal_id /display style(column)=[cellwidth=1in];
define ndels /display style(column)=[cellwidth=1in];
define galshort / display style(column)=[cellwidth=1in flyover = "Here is a
helpful note."] "Shortage Across All Deliveries (Gallons)";
define pershort / display style(column)=[cellwidth=1.5in
tagattr="format:####.##\%" fontweight = bold] "Overall % Short This
Period";
define sincelast / analysis sum style(column)=[cellwidth=1in
tagattr="format:###.##\%"] "Change Since Last Reported";
compute sincelast;
if sincelast.sum ne . then do;
if sincelast.sum < -0.001 then call define(_row_,"style",
"style={background= light red}");
end;
endcompute;
run;
```



```

else if sincelast.sum < -0.0005 then call define(_row_,"style",
"style={background= very light red}");
else if sincelast.sum > 0.001 then call define(_row_,"style",
"style={background= light green}");
else if sincelast.sum > 0.0005 then call define(_row_,"style",
"style={background= very light green}");

end;
endcomp;
run;

ods excel close;

```

The worksheet generated from this code is shown in Figure 8. In that figure, the user has clicked on row 5 of column C, prompting the flyover text for that column to be displayed.

Terminal	# Deliveries in Analysis	Shortage Across All Deliveries (Gallons)	Overall % Short This Period	Change Since Last Reported
Terminal 1	39	-2,087	-.53%	-.18%
Terminal 2	42		-.31%	-.03%
Terminal 3	457		-.29%	.05%
Terminal 4	68		-.47%	.03%
Terminal 5	605	-6,109	-.13%	.03%
Terminal 6	40	-1,498	-.52%	-.06%
Terminal 7	374	-17,352	-.49%	.03%
Terminal 8	347	-12,253	-.37%	-.06%
Terminal 9	73	-132	-.04%	-.08%
Terminal 10	36	-1,183	-.27%	.06%
Terminal 11	183	-5,322	-.3%	.08%
Terminal 12	123	-4,176	-.46%	.01%

Figure 8

CAN TWO TABLES BE DISPLAYED ON ONE TAB? YES!

To allow additional tables to be displayed within the existing worksheet you must first modify the ODS EXCEL OPTIONS statement to specify SHEET_INTERVAL = 'none'. This option controls when new worksheets are generated within the workbook. By default the option is set to SHEET_INTERVAL = 'table' so that a new worksheet will be created for each table in the report. Suppressing this default behavior by specifying SHEET_INTERVAL = 'none' will cause all subsequent output to be displayed within the existing worksheet, until the option is reset again in a later ODS EXCEL OPTIONS statement.

Once the SHEET_INTERVAL = 'none' option has been specified additional tables (or graphics) can be written to the existing worksheet simply by calling the procedures that produce them. The example code below illustrates adding a PRINT procedure after the REPORT procedure in our earlier code so that two tables will now be displayed sequentially in this first worksheet of the workbook. An additional title statement is also added between the two procedures, to generate a line of text between the two tables in the final output:

```
ods excel file="c:\Terminal Report.xlsx";
```

```
ods excel options(autofilter="1-5" sheet_name = "Terminal Comparison"
embedded_titles='yes' sheet_interval = 'none');
run;
```

```
title j = 1 "Terminal Comparisons";
title2 j = 1 "January 1 - February 1, 2017";
run;
```

```
proc report . . .
run;
```

```
title j = 1 "Overall Shortage";
```

```
proc print data = DemoSummary split = "*" style(header)={just=c} label
noobs
var  totnetdel totdiffnet;
var  totper / style = {width = 100 tagattr="format:###.##\%"};
label totper = "Overall % Short";
run;
```

```
ods excel close;
```

The worksheet generated from this code is shown in Figure 9. Notice that, just as was seen with PROC REPORT, PROC PRINT utilizes any existing formats when displaying the output. Similarly, any existing labels will be used for column headings, as long as the LABEL option is specified in the PROC PRINT statement. Additional labels and formats, as well as column width specifications, can be added within the PRINT procedure. For variables assigned percent formats with decimal allocations, such as the variable TOTPER in the DEMOSUMMARY dataset, tagattr="format:###.##%" must again be specified to ensure the data values are displayed correctly, if you are using SAS 9.4 Maintenance Release 3.

	A	B	C	D	E
1	Terminal Comparisons				
2	January 1 - February 1, 2017				
3					
4	Terminal	# Deliveries in Analysis	Shortage Across All Deliveries (Gallons)	Overall % Short This Period	Change Since Last Reported
5	Terminal 1	39	-2,087	-.53%	-.18%
6	Terminal 2	42	-939	-.31%	-.03%
7	Terminal 3	457	-10,524	-.29%	.05%
8	Terminal 4	68	-2,191	-.47%	.03%
9	Terminal 5	605	-6,109	-.13%	.03%
10	Terminal 6	40	-1,498	-.52%	-.06%
11	Terminal 7	374	-17,352	-.49%	.03%
12	Terminal 8	347	-12,253	-.37%	-.06%
13	Terminal 9	73	-132	-.04%	-.08%
14	Terminal 10	36	-1,183	-.27%	.06%
15	Terminal 11	183	-5,322	-.3%	.08%
16	Terminal 12	123	-4,176	-.46%	.01%
17					
18	Overall Shortage				
19					
20	Total Net Deliveries in Analysis (Gallons)	Total Net Shortage in Analysis (Gallons)	Overall % Short		
21	12,404,913	-39,258	-.32%		
22					
23					

Figure 9

ADDING A NEW WORKSHEET AND INCLUDING PLOTS

When using the ODS EXCEL destination to create an Excel workbook you are not limited to creating only a single worksheet. Nor are you limited to producing only data tables. Statistical graphics (SG) procedures such as SGPLOT and SG PANEL allow high quality statistical plots to be included in the report as easily as tables were added with the REPORT and PRINT procedures.

ADDING A NEW WORKSHEET

New worksheets can be added to a workbook by including a new ODS EXCEL OPTIONS statement followed by the procedure(s) whose output should appear on the new tab. The syntax for this is illustrated in the example code below:

```
ods excel file="c:\Terminal Report.xlsx";
ods excel options(autofilter="1-5" sheet_name = "Terminal Comparison"
embedded_titles='yes' sheet_interval = 'none');
run;

title ... ;

proc report . . .
run;

title ... ;

proc print ... ;
run;
```

```
ods excel options(sheet_name = "Terminal Plots" sheet_interval = 'proc');
```

```
proc of your choice ...;  
run;
```

The output from this procedure will appear on the second tab of the workbook.

```
ods excel close;  
run;
```

In this example it was necessary to specify two options in the ODS EXCEL OPTIONS statement in order for the new worksheet to be correctly created. First, a name for the new worksheet was specified using the SHEET_NAME option. Then the value of SHEET_INTERVAL was set to 'proc'. This instructs SAS to start a new worksheet whenever a new procedure is called. It was necessary to add this here because SHEET_INTERVAL had been set to 'none' in the earlier ODS EXCEL OPTIONS statement of the example code, to allow for the printing of multiple tables on the first tab of the report.

CREATING A GRAPH WITH SGPLOT

First introduced in SAS 9.2, the SGPLOT procedure is a powerful tool capable of generating a variety of statistical graphics including bar charts, histograms, boxplots, scatterplots, and series plots. For additional information and examples of graphics that can be created via PROC SGPLOT (and SGPANEL) two useful references are "Graphs are Easy with SAS 9.4" (Mantage, 2015) and "Graphing Made Easy with SGPLOT and SGPANEL Procedures" (Slaughter and Delwiche, 2015). Once you've created the graphic of your choice using these procedures, all that is needed to send the graphic to an Excel worksheet is to add that code inside the ODS wrapper for the ODS Excel destination.

In the example code below, an SGPLOT procedure is used to generate a vertical bar chart on the second worksheet of the workbook. An axis table, available as part of the SGPLOT procedure starting in SAS 9.4, is created along the bottom of the plot to show the total shortage for each terminal, in thousands of gallons. If you would like to prevent the title from the previous worksheet from appearing on this tab, you can reset the title with a blank TITLE statement, as show in the line marked ⓐ, below. If you would like the graph to appear larger (or smaller) than it does by default, that can be accomplished by specifying a desired graphic width, as shown in the line marked ⓑ:

```
ods excel file="c:\Terminal Report.xlsx" ;  
ods excel options(autofilter="1-5" sheet_name = "Terminal Comparison"  
embedded_titles='yes' sheet_interval = 'none');  
run;  
  
title ... ;  
  
proc report . . .  
run;  
  
title ... ;  
  
proc print ... ;  
run;  
  
ods excel options(sheet_interval = 'proc' sheet_name = "Terminal Plots");  
ods graphics on / width = 8in; ⓐ  
title; ⓑ  
run;  
  
proc sgplot data = demoplot noautolegend;  
vbarparm category = terminal_id response = pershort /x2axis fillattrs =  
(color = lightred) ;
```

```

xaxistable totshort / position = bottom separator title = "Analysis Volumes
(Thousands of Gallons)";
x2axis label = "Terminal Shortages" grid;
yaxis label = "% Short" grid;
run;

ods excel close;
run;

```

The worksheet generated from this code is shown in Figure 10.

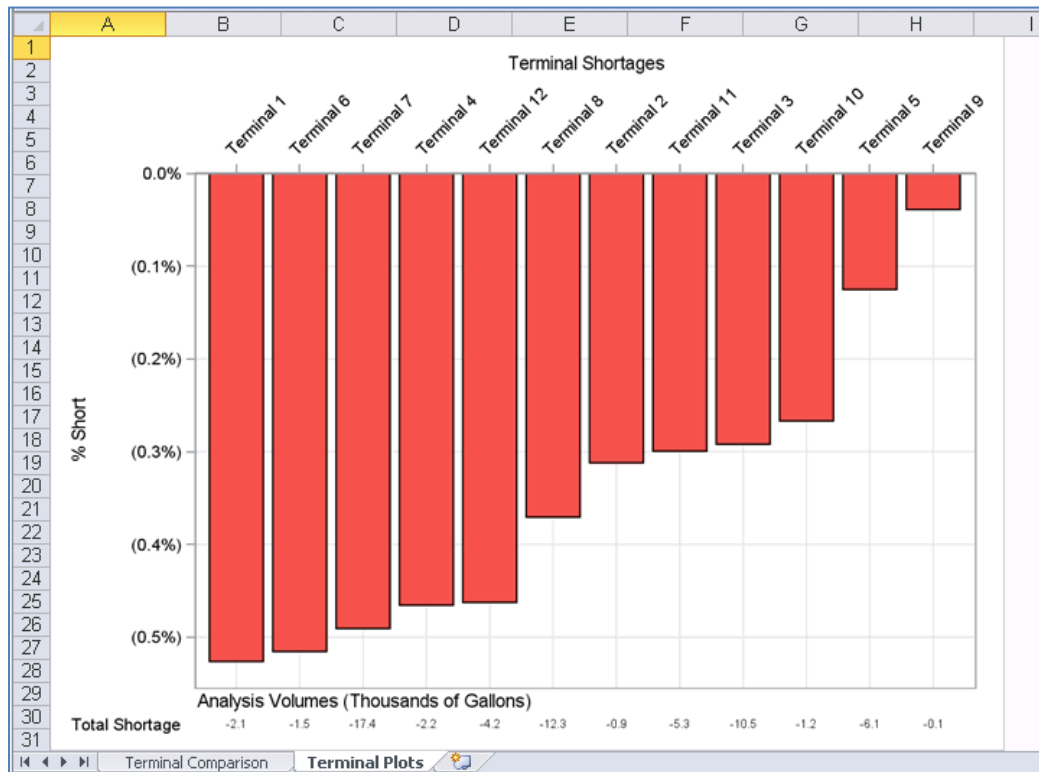


Figure 10

GENERATING A SERIES OF PLOTS WITH SGPANEL

The ODS EXCEL destination makes it so easy to add SAS graphics to Excel worksheets, why stop with a single plot? The SGPANEL procedure can be used to produce a series of graphs displayed as panels in a single output graphic, and that graphic can be included in the Excel worksheet as easily as a table or single plot. The example code below illustrates use of the SGPANEL procedure to create a series of 12 plots displayed on a third worksheet of the existing sample workbook. These plots show the shortage trends over time for the twelve terminals in the sample data set. Notice that a new line (marked Ⓐ) has been added at the start of the example code, this is necessary so that the graph size can be increased to 12 inches, as shown in the line marked Ⓑ:

```

options papersize=("16in", "14in"); run; Ⓐ
ods excel file="c:\Terminal Report.xlsx" ;
ods excel options(autofilter="1-5" sheet_name = "Terminal Comparison"
embedded_titles='yes' sheet_interval = 'none');
run;

title ... ;

```

```

proc report . . .
run;

title ... ;

proc print ... ;
run;

ods excel options(sheet_interval = 'proc' sheet_name = "Terminal Plots");
ods graphics on / width = 8in;
title;
run;

proc sgplot ... ;
run;

ods excel options(sheet_name = "Terminal Timelines" );
ods graphics on / width = 12in; ⑩

proc spanel data = demotimeline noautolegend ;
panelby terminal_id /rows = 3 columns = 4 novarname spacing = 10 nowall
skipemptycells;
series x = order y = percentdiff / markers markerattrs = (symbol =
circlefilled color = black size = 7) lineattrs = (thickness = 2 color =
red);* datalabel = terminal_id datalabelpos = right;
colaxis label = "Date" max = -1 thresholdmax = 0 grid;
rowaxis label = "% Short";
refline 0 /axis = y;
run;

ods excel close;
run;
quit;

```

The worksheet generated from this code is shown in Figure 11.

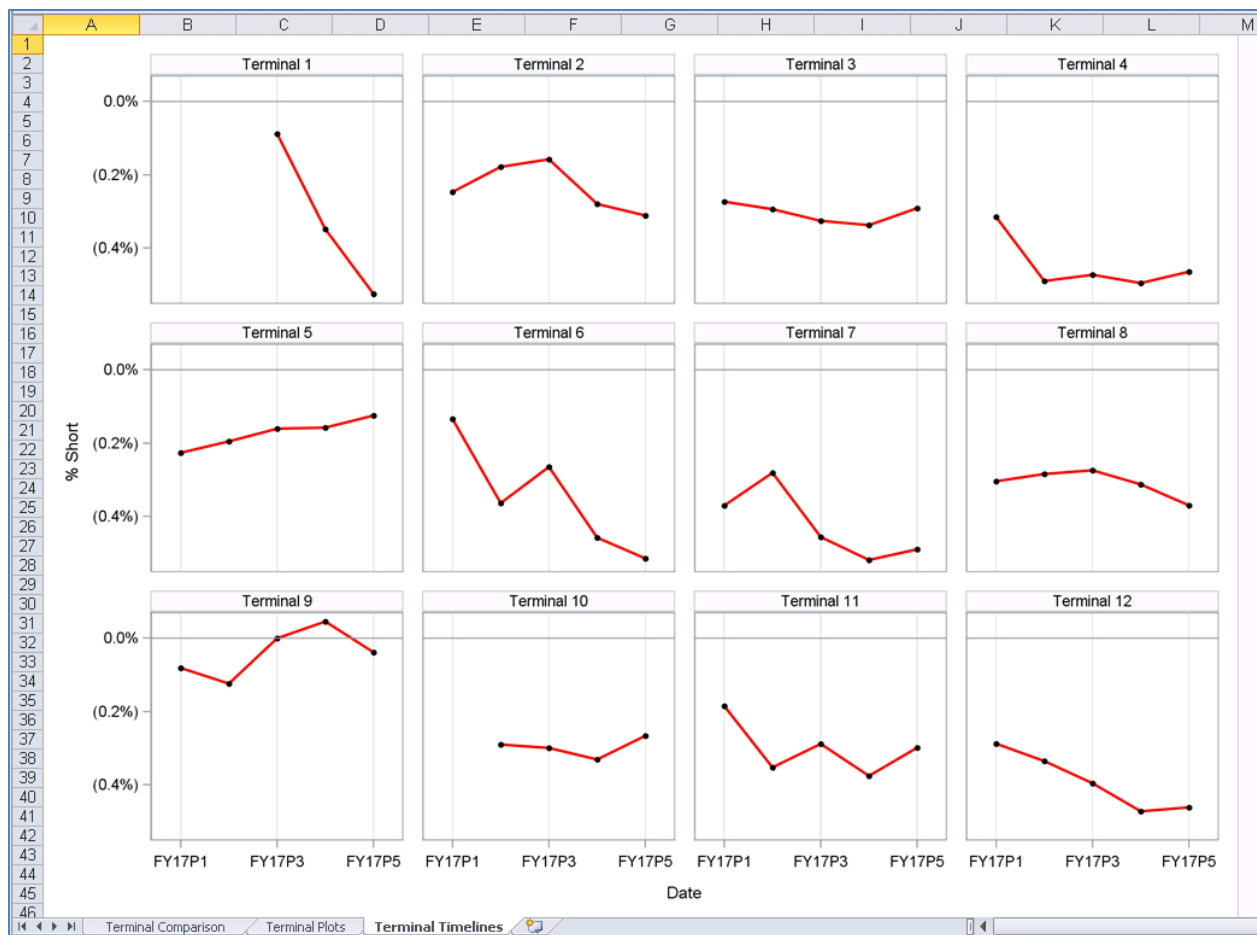


Figure 11

CONCLUSION

The new ODS destination for Excel offers users the opportunity to quickly and easily create professional quality, multi-tabbed Excel workbooks. This paper has illustrated just a few of the many ways that output from procedures such as REPORT, PRINT, SGPLOT, and SGPANEL can now be readily incorporated into Excel reports.

REFERENCES AND RECOMMENDED READING

- Huff, Gina. 2016. "An 'Excel'ent Journey: Exploring the New ODS EXCEL Statement." *Proceedings of the SAS Global Forum 2016 Conference*. Las Vegas, NV: SAS Institute Inc. Available at: <http://support.sas.com/resources/papers/proceedings16/2780-2016.pdf>
- Mantage, Sanjay. 2015. "Graphs are Easy with SAS 9.4" *Proceedings of the SAS Global Forum 2015 Conference*. Dallas, TX: SAS Institute Inc. Available at: <https://support.sas.com/resources/papers/proceedings15/SAS1780-2015.pdf>
- Parker, Chevell. 2016. "A Ringside Seat: The ODS Excel Destination versus the ODS ExcelXP Tagset." *Proceedings of the SAS Global Forum 2016 Conference*. Las Vegas, NV: SAS Institute Inc. Available at: <https://support.sas.com/resources/papers/proceedings16/SAS5642-2016.pdf>
- SAS Institute Inc. (2016). *SAS 9.4 Output Delivery System: User's Guide, Fifth Edition*

Slaughter, Susan and Lora Delwiche. 2015. "Graphing Made Easy with SGPLOT and SGPANEL Procedures" *Proceedings of the SAS Global Forum 2015 Conference*. Dallas, TX: SAS Institute Inc. Available at: <https://support.sas.com/resources/papers/proceedings15/2441-2015.pdf>

Zender, Cynthia and Allison Booth. 2013. "Turn Your Plain Report into a Painted Report Using ODS Styles." *Proceedings of the SAS Global Forum 2013 Conference*. San Francisco, CA: SAS Institute Inc. Available at: <http://support.sas.com/resources/papers/proceedings13/366-2013.pdf>

CONTACT INFORMATION

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

Caroline Walker
Warren Rogers LLC
cwalker@warrenrogers.com