Paper 219-2009

SAS® Graphs in Small Multiples

Andrea Wainwright-Zimmerman, Capital One, Richmond, VA

ABSTRACT

Edward Tufte has championed the idea of using "small multiples" as an effective way to present large amounts of data in limited space. This paper will discuss the issues of formatting SAS Graphs for scaling down, and the challenges of putting many graphs together in the same medium. I will also discuss options for drawing attention to specific graphs in this paper.

INTRODUCTION

Edward Tufte is a strong advocate for viewing data graphs in small multiples. By using the same design structure for all the graphs, the human eye can focus on the changes in the data and not be distracted by changes in the data frames. Small multiples allow us to see patterns between graphs more easily by putting them together on one page in a logical order, than if the mind had to remember details from one page to the next. Small multiples allow us to increase our “data-ink ratio”.

WIDGET PRODUCERS INC.

This paper will focus on a fictional widget-producing factory. They are using Statistical Process Quality Control and SAS/QC® software to monitor four metrics (diameter, opening, thickness, and weight) on their widgets. They are monitoring production on 10 different production lines but need a process flexible enough to allow them to add or remove lines in the future. They have HISTORY and LIMITS data sets that are inputs to PROC SHEWHART but are unhappy looking at all the separate graphs that PROC SHEWHART produces, each on a separate page.

SETTING UP THE FOLDERS

First, a macro for the home directory is established. A SAS library is created where the four HISTORY and four LIMITS data sets already exist. The DCREATE function is used to have SAS create a subfolder.

```
/*directory where all the data and output files will be stored*/
%let dir=D:\My Documents\widget_QC;
/*location of proc shewhart HISTORY and LIMITS data sets*/
libname qc "&dir\data";
/*hex value for the shade of red*/
%let _red="#CC2B1B";
/*generate folder to store the HTML and GIFs*/
data _null_;
newdirectory=dcreate("Website","&dir\");
run;
```

TRANSPOSE THE HISTORY DATA SET AND COMPARE TO THE LIMITS

The following macro will transpose each of the four HISTORY data sets and compare them to the matching LIMITS data set, resulting in a SAS data set that records which production lines have issues and on which metrics.

First, a PROC SQL is used to get a list of all dates in the HISTORY data set. Then a counter called NEWVAR is created. This is merged into the HISTORY data set. The resulting data set is transposed using NEWVAR as an ID, creating variables named _1, _2, etc. This results in the most recent data point being called _1, the second most recent is called _2, etc. Then, only the rows that represent means are kept, and the ‘X’ is removed from the end of the field so only the production line name remains. This allows it to be merged with the LIMITS data set.

```
%macro data_prep_and_test(file);
/* create a list of all the dates in the history*/
proc sql noprint;
create table temp_days as
   select distinct sasdate from qc.&file._history
   order by sasdate descending;
quit;

data temp_days;
set temp_days;
retain newvar 0;
newvar=newvar+1;
run;

proc sort data=qc.&file._history;
by descending sasdate;
run;

/*add the counter variable to the HISTORY data set*/
data temp;
merge qc.&file._history temp_days;
by descending sasdate;
run;

/*make the counter field the new name of the data columns*/
proc transpose data=temp out=x_hist_trans;
id newvar;
idlabel sasdate;
run;

data ready;
set x_hist_trans;
stat_name= substr(_name_,length(_name_),1);
if stat_name = 'X';/*eliminate N's and R's*/
prodname=substr(_name_,1,length(_name_)-1);
keep prodname _1;/*only keep most recent data point*/
run;

proc sort data=ready;
by prodname;
run;
```

Next, the results of transformation of the HISTORY data set and the LIMITS data set are merged, keeping just the name of the production lines, the most recent data point (_1), the Upper Control Limits (UCL), and the Lower Control Limits (LCL). The current data point is compared to the limits and a failure indicator is set to either 1 or 0.

```
/* test 1 is a simple test to see if the current */
/* data point is above the UCL or below the LCL  */
proc sort data=qc.&file._limits out=temp_limits;
by _var_;
run;

data test1_&file (keep=prodname fail_&file);
merge ready (in=n)
      temp_limits (drop=_subgrp_ _mean_ rename=(_var_=prodname));
by prodname;
if n;
```

```
fail_&file=0;
if _LCLX_>_1 or _1>_UCLX_ then fail_&file=1;

run;
%mend data_prep_and_test;
```

Then, the macro is run on the four sets of HISTORY and LIMITS data sets.

```
%data_prep_and_test(diameter)
%data_prep_and_test(weight)
%data_prep_and_test(thickness)
%data_prep_and_test(opening)
```

Lastly, the four results files are merged into one data set.

```
/*produce a list of the qc results*/
data current;
merge test1_diameter
      test1_weight
      test1_thickness
      test1_opening;
by prodname;
run;
```

PREPARING THE MACRO VARIABLES

Next, a list of all the variables in one of the HISTORY data sets is put into a SAS data set. Then, a data set is formed that has the names of the production lines, and the names of the fields that have the means in the HISTORY data sets. Then, a PROC SQL is used to establish a series of macro variables with the field name of each mean from the history data set (X1-X10), the name of the production line (V1-V10), and the failure indicators for the four metrics (D1-D10, W1-W10, T1-T10, and O1-O10). A macro called NVARS is also used to capture the SAS generated value of &SQLOBS. &NVARS will be used in an upcoming do loop.

```
/*create a list of all the vars in the HISTORY data set*/
proc sql noprint;
create table temp_var_list as
select distinct name
  from sashelp.vcolumn
  where libname="QC"
  and memname=upcase("DIAMETER_HISTORY");
quit;

/*keep only specific versions of the var list*/
data temp2_var_list;
set temp_var_list;
var=substr(name,1,length(name)-1);
stat=substr(name,length(name),1);
if name ne "sasdate";
if name ne "run_date";
if stat="X";
nameX=name;
drop stat name;
run;
```

```
/*set up all macro vars*/
proc sql noprint;
   select a.nameX, a.var, b.fail_diameter, b.fail_weight,
          b.fail_thickness, b.fail_opening
   into :X1 thru :X&SYSMAXLONG,
        :V1 thru :V&SYSMAXLONG,
        :D1 thru :D&SYSMAXLONG,
        :W1 thru :W&SYSMAXLONG,
        :T1 thru :T&SYSMAXLONG,
        :O1 thru :O&SYSMAXLONG
   from temp2_var_list a,
          current b
       where trim(b.prodname)=trim(a.var);
   %let nvars=&sqlobs;
quit;
```

TRANSFORMING SAS GRAPHS INTO SMALL MULTIPLES

Several SAS options and graph options are needed to set up appropriate small multiples graphs.

Symbol statements set up all the lines that will be graphed. These need to be consistent for all the graphs.

```
/*first three are QC lines*/
symbol1 interpol=join color=black;
symbol2 interpol=join color=black;
symbol3 interpol=join color=black;
/*this is the data line*/
symbol4 interpol=join color=blue width= 2;
```

Dates and page number should be turned off. Titles and footnotes should be cleared, and the axes, labels, and tick marks are turned off.

```
options nodate nonumber;
title;title2;footnote;
axis1 label=none value=none major=none minor=none;
axis2 label=none value=none major=none minor=none;
```

This will produce clean, consistent graphs without extraneous ink. The focus is on the data and the patterns it reveals.

COMBINING SMALL MULTIPLES ONTO ONE WEBPAGE

The widget plant would like to make the QC results available on-line so that each production line manager can see the results. ODS can be used to set up the needed rows and columns for an HTML document that can be made available on the widget company's intranet.

The ESCAPECHAR= statement establishes the '^' as the indicator that in-line formatting will be used. The LISTING CLOSE statement closes the traditional SAS Output window and prevents output from being written there. The RESULTS=OFF statement prevents SAS from opening the graphs in a viewer. Next, HTML is defined to be the desired output, and the BODY= option provides the path and file name for the HTML document. GPATH= defines the path for the graphs, and the URL=NONE prevents SAS from embedding the path in the HTML. This makes the HTML have relative paths so it will still work when the HTML file and the graph files are moved to a server to make them available to others. Lastly, the LAYOUT START statement begins the definition of a layout with five columns.

```
ods escapechar='^';
ods listing close;
ods results=off;
```

```
   ods html body="&dir\Website\qc_results.html"
            gpath="&dir\Website\" (URL=NONE)
            nogtitle;

   ods layout start columns=5;
```

PUTTING HEADERS AT THE TOP OF THE WEBPAGE

Before creating the graphs, a title and column headers are needed. There will be five columns--one for the name of the production line, and one for each of the four metrics. Each will be in a separate ODS REGION. The COLUMN_SPAN=five forces SAS to spread the title out across all five columns.

```
   /*title region*/
   ods region column_span=5;
   ods html text="^{style [just=center]}Widget Production Lines";

   /*5 header regions*/
   ods region;
   ods html text='^{style [just=center]}PRODUCTION LINE';

   ods region;
   ods html text='^{style [just=center]}DIAMETER';

   ods region;
   ods html text='^{style [just=center]}OPENING';

   ods region;
   ods html text='^{style [just=center]}THICKNESS';

   ods region;
   ods html text='^{style [just=center]}WEIGHT';
```

PUTTING A ROW OF GRAPHS ON THE WEBPAGE

The following macro will run once for each production line in the HISTORY and LIMITS data sets. It will make use of the NVARS macro established in an earlier PROC SQL to set the end of the do loop. This prevents having to hard code the limit at 10 and makes this flexible enough to handle changes in the number of production lines. The production line name will be written in the first region, and then a macro (one_graph) will be called four times. It will put each graph into separate regions as it goes across the webpage.

```
   %macro graph_all_rows;
   %do i=1 %to &nvars;
      /*put prodname*/
      ods region;

      ods html text="^{style [just=center]}&&v&i";

      /*put the 4 graphs in the row*/
      %one_graph(diameter,d)
      %one_graph(opening,o)
      %one_graph(thickness,t)
      %one_graph(weight,w)
   %end; /*do i=1 to &nvars*/
   %mend graph_all_rows;
```

The macro one_graph will be called once per production line. It uses PROC SQL to store the UCL, MEAN, and LCL for the current production line into macros. Then a temporary data set is created from the HISTORY data set. It only has the sasdate field, the mean for that particular production line, and the LCL, MEAN, and UCL. The GOPTIONS statement sets the output to be GIFs and sets the size to be 1.5 by 1 inches. The macro parameter N is used to pass in which of the four macro arrays are needed for that metric (D1-D10, O1-O10, T1-T10, or W1-W10). If there is a failure indicated by a 1, then the background color is set to red. Otherwise, it is made white. The plot statement plots the two limits and the overall mean first, then the data line last. The OVERLAY option causes all four lines to be on one graph, the HAXIS= and VAXIS= options use the previously defined axes that are blank. Lastly, the CFRAME= option uses the &COLOR macro that is based on if there is a failure or not.

```
   %macro one_graph(file,n);
   /*get limits*/
   proc sql noprint;
   select _LCLX_, _MEAN_, _UCLX_
      into :lclx, :x, :uclx
      from qc.&file._limits
      where trim(_var_)=trim("&&v&i");
   quit;

   /*set up data set with history data and limits on one variable*/
   data temp;
   set qc.&file._history (keep=sasdate &&x&i);
   lclx=&lclx;
   x=&x;
   uclx=&uclx;
   run;

   /*produce graph*/
   ods region;

   goptions device=gif gsfmode=replace hsize=1.5 in vsize=1 in;

   %if &&&n&i=1 %then %let color=&_red;
   %else              %let color=white;
   proc gplot data=temp;
        plot    lclx*sasdate
                uclx*sasdate
                   x*sasdate
               &&x&i*sasdate/overlay
                              haxis=axis1
                              vaxis=axis2
                              cframe=&color;
   quit;
   %mend one_graph;
```

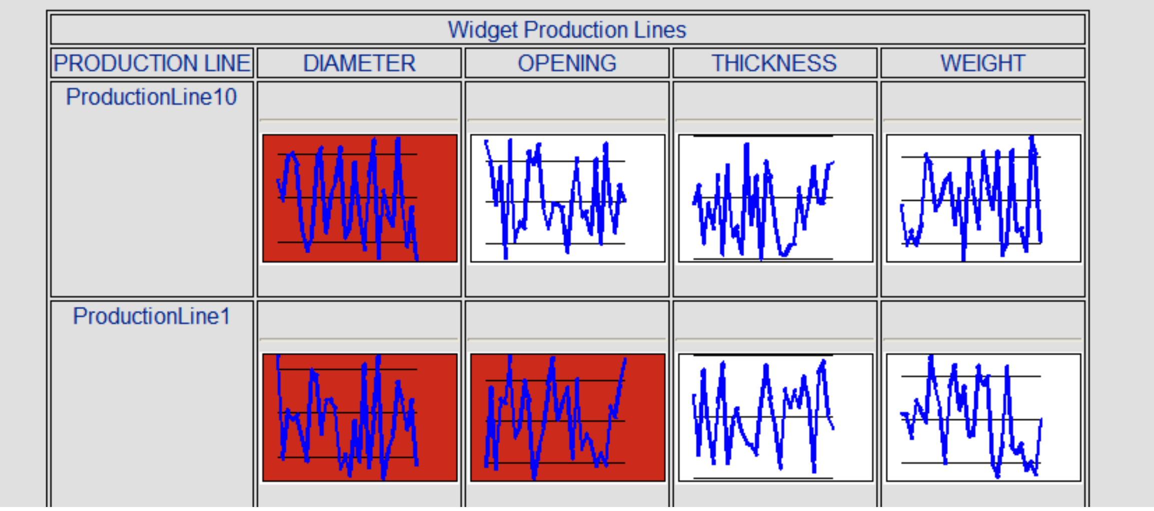
Lastly, the GRAPH_ALL_ROWS macro is called, the HTML doc that has been created is closed, and the normal SAS output window is reopened.

```
   %graph_all_rows
   ods html close;
   ods listing;
```

CONCLUSION

The resulting HTML doc has many advantages over the standard PROC SHEWHART output. Patterns across a production line can be seen easily by scanning across the row. Patterns affecting all production lines can be seen easily by scanning down the page. The red background color causes the eye to go quickly to those graphs that indicate a metric beyond its control limits. However, the eye will also go to odd-looking graphs that might indicate a pattern that is of concern, even if it has not exceeded the control limits yet.

RESULTING WEBPAGE:



Widget Production Lines

PRODUCTION LINE	DIAMETER	OPENING	THICKNESS	WEIGHT
ProductionLine10				
ProductionLine1				

APPENDIX

EXAMPLE DATA

For those not familiar with PROC SHEWHART's HISTORY and LIMITS data sets, the following examples are provided.

The LIMITS data set requires fields called _VAR_, _MEAN_, _LCLX_, _UCLX_, and _SUBGRP_. The values of _VAR_ must be found as field names in the HISTORY data set as shown in next example. The value of _SUBGRP_ must also be a field in the HISTORY data set.

QC.DIAMETER_LIMITS:

var	_mean_	_lclx_	_uclx_	_subgrp_
ProductionLine1	1	0.9	1.1	sasdate
ProductionLine2	1	0.9	1.1	sasdate
ProductionLine3	1	0.9	1.1	sasdate
ProductionLine4	1	0.9	1.1	sasdate
ProductionLine5	1	0.9	1.1	sasdate
ProductionLine6	1	0.9	1.1	sasdate
ProductionLine7	1	0.9	1.1	sasdate
ProductionLine8	1	0.9	1.1	sasdate
ProductionLine9	1	0.9	1.1	sasdate
ProductionLine10	1	0.9	1.1	sasdate

The HISTORY data set must have a field that matches the value of _SUBGRP_ from the LIMITS data set. The mean for each process must be stored in a variable that is named after the process (matching _VAR_ from LIMITS) and ending in 'X'. Similarly, the sample size must be in a field ending in 'N' and either the range or the standard deviation in a field ending in 'R' or 'S' respectively.

QC.DIAMETER_HISTORY:

SASDATE	ProductionLine1X	ProductionLine1N	ProductionLine1R	ProductionLine2X	…
17652	0.88306	5	0.09282	1.0778	…
17651	1.02907	5	0.00664	1.14455	…
17650	0.96184	5	0.14406	0.87469	…
17649	1.02984	5	0.07876	0.8783	…
17648	1.07082	5	0.1166	1.03729	…
17647	0.94826	5	0.08918	1.09102	…
17646	0.91745	5	0.13773	0.9089	…
17645	0.85349	5	0.09751	1.08288	…
17644	1.12869	5	0.20267	1.04026	…
17643	0.99678	5	0.00831	1.03238	…
17642	0.87846	5	0.0447	1.07218	…
17641	1.11018	5	0.0555	0.90818	…
17640	0.89029	5	0.06755	1.09986	…
17639	0.98285	5	0.0553	1.14551	…

COMPLETE CODE

```
/*directory where all the data and output files will be stored*/
%let dir=D:\My Documents\CRM\VLM\VLM code\SESUG;

/*location of proc shewhart HISTORY and LIMITS data sets*/
libname qc "&dir\data";

/*hex value for the shade of red*/
%let _red="#CC2B1B";


/*generate folder to store the HTML and GIFs*/
data _null_;
newdirectory=dcreate("Website","&dir\");
run;

%macro data_prep_and_test(file);
/***************************/
/*preps the history data set*/
/***************************/
proc sql noprint;
create table temp_days as
select distinct sasdate from qc.&file._history
order by sasdate descending;
quit;

data temp_days;
set temp_days;
retain newvar 0;
newvar=newvar+1;
run;

proc sort data=qc.&file._history;
by descending sasdate;
run;
```

```
/*add the counter variable to the HISTORY data set*/
data temp;
merge qc.&file._history temp_days;
by descending sasdate;
run;

/*make the counter field the new name of the data columns*/
proc transpose data=temp out=x_hist_trans;
id newvar;
idlabel sasdate;
run;

data ready;
set x_hist_trans;
stat_name= substr(_name_,length(_name_),1);
if stat_name = 'X';/*eliminate N's and R's*/
prodname=substr(_name_,1,length(_name_)-1);
keep prodname _1;
run;

proc sort data=ready;
by prodname;
run;

/***************/
/**begin test 1*/
/***************/

/* test 1 is a simple test to see if the current */
/* data point is above the UCL or below the LCL  */
proc sort data=qc.&file._limits out=temp_limits;
by _var_;
run;

data test1_&file (keep=prodname fail_&file);
merge ready (in=n)
      temp_limits (drop=_subgrp_ _mean_ rename=(_var_=prodname));
by prodname;
if n;

fail_&file=0;
if _LCLX_>_1 or _1>_UCLX_ then fail_&file=1;
run;
%mend data_prep_and_test;

%data_prep_and_test(diameter)
%data_prep_and_test(weight)
%data_prep_and_test(thickness)
%data_prep_and_test(opening)

/*produce a list of the qc results*/
data current;
merge test1_diameter
      test1_weight
      test1_thickness
      test1_opening;
by prodname;
run;
```

```
/*create a list of all the vars in the HISTORY data set*/
proc sql noprint;
create table temp_var_list as
select distinct name
  from sashelp.vcolumn
  where libname="QC"
  and memname=upcase("DIAMETER_HISTORY");
quit;

/*keep only specific versions of the var list*/
data temp2_var_list;
set temp_var_list;
var=substr(name,1,length(name)-1);
stat=substr(name,length(name),1);
if name ne "sasdate";
if name ne "run_date";
if stat="X";
nameX=name;
drop stat name;
run;

/*set up all macro vars*/
proc sql noprint;
   select a.nameX, a.var, b.fail_diameter, b.fail_weight,
          b.fail_thickness, b.fail_opening
   into :X1 thru :X&SYSMAXLONG,
        :V1 thru :V&SYSMAXLONG,
        :D1 thru :D&SYSMAXLONG,
        :W1 thru :W&SYSMAXLONG,
        :T1 thru :T&SYSMAXLONG,
        :O1 thru :O&SYSMAXLONG
   from temp2_var_list a,
           current b
      where trim(b.prodname)=trim(a.var);
   %let nvars=&sqlobs;
quit;
/***********************/
/*set up graph settings*/
/***********************/
/*first three are QC lines*/
symbol1 interpol=join color=black;
symbol2 interpol=join color=black;
symbol3 interpol=join color=black;
/*this is the data line*/
symbol4 interpol=join color=blue width= 2;

options nodate nonumber;
/*remove all titles and footnotes*/
title;title2;footnote;
/*blank out both axis*/
axis1 label=none value=none major=none minor=none;
axis2 label=none value=none major=none minor=none;

/*begin ODS*/
ods escapechar='^';
ods listing close;
ods results=off;
ods html body="&dir\Website\qc_results.html"
         gpath="&dir\Website\" (URL=NONE)
         nogtitle;

ods layout start columns=5;

/*title region*/
ods region column_span=5;
ods html text="^{style [just=center]}Widget Production Lines";

/*5 header regions*/
ods region;
ods html text='^{style [just=center]}PRODUCTION LINE';

ods region;
ods html text='^{style [just=center]}DIAMETER';

ods region;
ods html text='^{style [just=center]}OPENING';

ods region;
ods html text='^{style [just=center]}THICKNESS';

ods region;
ods html text='^{style [just=center]}WEIGHT';

%macro graph_all_rows;
%do i=1 %to &nvars;
   /*put prodname*/
   ods region;
   ods html text="^{style [just=center]}&&v&i";

   /*put the 4 graphs in the row*/
   %one_graph(diameter,d)
   %one_graph(opening,o)
   %one_graph(thickness,t)
   %one_graph(weight,w)
%end; /*do i=1 to &nvars*/
%mend graph_all_rows;

%macro one_graph(file,n);
/*get limits*/
proc sql noprint;
select _LCLX_, _MEAN_, _UCLX_
   into :lclx, :x, :uclx
   from qc.&file._limits
   where trim(_var_)=trim("&&v&i");
quit;

/*set up data set with history data and limits on one variable*/
data temp;
set qc.&file._history (keep=sasdate &&x&i);
lclx=&lclx;
x=&x;
uclx=&uclx;
run;

/*produce graph*/
ods region;
```

```
goptions device=gif gsfmode=replace hsize=1.5 in vsize=1 in;

%if &&&n&i=1 %then %let color=&_red;
%else              %let color=white;
proc gplot data=temp;
     plot    lclx*sasdate
             uclx*sasdate
                x*sasdate
          &&x&i..*sasdate/overlay
                          haxis=axis1
                          vaxis=axis2
                          cframe=&color;
run;
quit;
%mend one_graph;

%graph_all_rows
ods html close;
ods listing;
```

REFERENCES

Tufte, Edward R. 2001. The Visual Display of Quantitative Information. Cheshire, CT: Graphics Press LLC.

Tufte, Edward R. 1990. Envisioning Information. Cheshire, CT: Graphics Press LLC.

Haworth, Lauren E. 2001. *Output Delivery System: The Basics*. Cary, NC: SAS Institute Inc.

ACKNOWLEDGMENTS

I would like to thank all members of SAS-L who have answered my questions and shared their wealth of knowledge through the years.

Thanks to VASUG members and officers for their support and to the Capital One statistician/SAS programmer community from which I have learned much.

CONTACT INFORMATION

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

Andrea Wainwright-Zimmerman
Capital One
15000 Capital One Drive
Richmond, VA 23238
Work Phone: 804-284-7681
E-mail: andrea.zimmerman@capitalone.com