

Customizing and Automating your Graphs using the SAS SG Procedures

Jesse A. Canchola, Roche Molecular Systems, Pleasanton, CA USA

Shiva Narra, Roche Molecular Systems, Pleasanton, CA USA

Alen Dzidic, University of Zagreb, Croatia

ABSTRACT

The SAS SG procedures are arguably the best tools in SAS for creating and customizing your graphs. In the past, automating and customizing the SG procedures may have been complex and tedious. However, many new options have been implemented in the SG procedures beginning with SAS Version 9.4 that promise to make customization and automation easier. The SAS user is taken through examples that show the improvements and is provided with a road map for successfully leveraging the power of the SAS SG procedures.

INTRODUCTION

Whether you are generating one graph or similar but repeated figures (e.g., one for each subject), the SAS Institute (Cary, NC) provides for a rich and flexible toolbox with their SAS SG procedures.

The SG procedures are:

SGPLOT: Produces a single graph

SGPANEL: Produces multi-panel (classification) graphs with common axes

SGSCATTER: Produces multi-panel graphs with common or different axes

Delwiche and Slaughter (2012) give a comprehensive overview of the SG procedures while Matange (2016) gives a deeper dive. The current paper augments the SG procedures capabilities with the advanced topics of automation and customization using the SAS Macro language, SAS IML (Interactive Matrix Language) and, to a limited extent, the SAS SQL and SAS Annotate facility.

The authors start with a short review of the current capabilities and then detail how to automate a graph or repetitive figures.

CREATING ONE OR A FEW GRAPHS

In the past, SG stood for “Statistical Graphics”. However, advancements and enhancements over the years have made the SAS SG procedures much more than that so that the authors feel that the SG acronym may now be more correctly represented as “Splendid Graphics” or simply, “SAS Graphics”.

Let us start with one example with all patients/subjects together. For this we use the simulated patient Hepatitis C Therapy Response data (HepCTR; Appendix A). Briefly, the HepCTR study examines the patient Hepatitis C viral load response with treatment over a period of 30 weeks. Two sets of data are simulated, the “Responders” (i.e., patients responding to therapy) whose viral load reduces to undetectable levels over time and the “Relapsers” (i.e., patients not responding to therapy) whose viral load comes back up at a point in time during the treatment regimen. The reader should proceed using the following steps for generating the Responder data set and plotting the results.

Step 1: Run the Appendix A simulated data code to use the HepCTR Responder data example.

Step 2: **All Patients One Graph**. Run the following code to produce a standard SGPLOT graph:

```
ods noproctitle ;
ods RTF file = "C:\WUSS2018_Responders_Overall_&sysdate..RTF"
         style=MonoChromePrinter ;
* above: sysdate appends current data to your file name and style gives white background ;
```

```

title2 "" ;
footnote1 "Responder Patients: Overall (Subjects combined in aggregate)" ;
* Overall ;
proc sort data=Responders; by TP ; run ;
proc sgplot data=Responders ;
    scatter y=log10Observed x=TP / group=Test ;
    yaxis label = "Overall Assay Test (log10 units)" ;
    xaxis label = "Weeks" ;
run ;
ods rtf close ;
title2 ; footnote1 ; * resets title2 and footnote1 ;

```

Figure 1 shows the results of running the *standard SAS code* above for the Hepatitis C Therapy Response (HepCTR) Simulated Aggregate Data for Responders across 30 weeks on treatment.

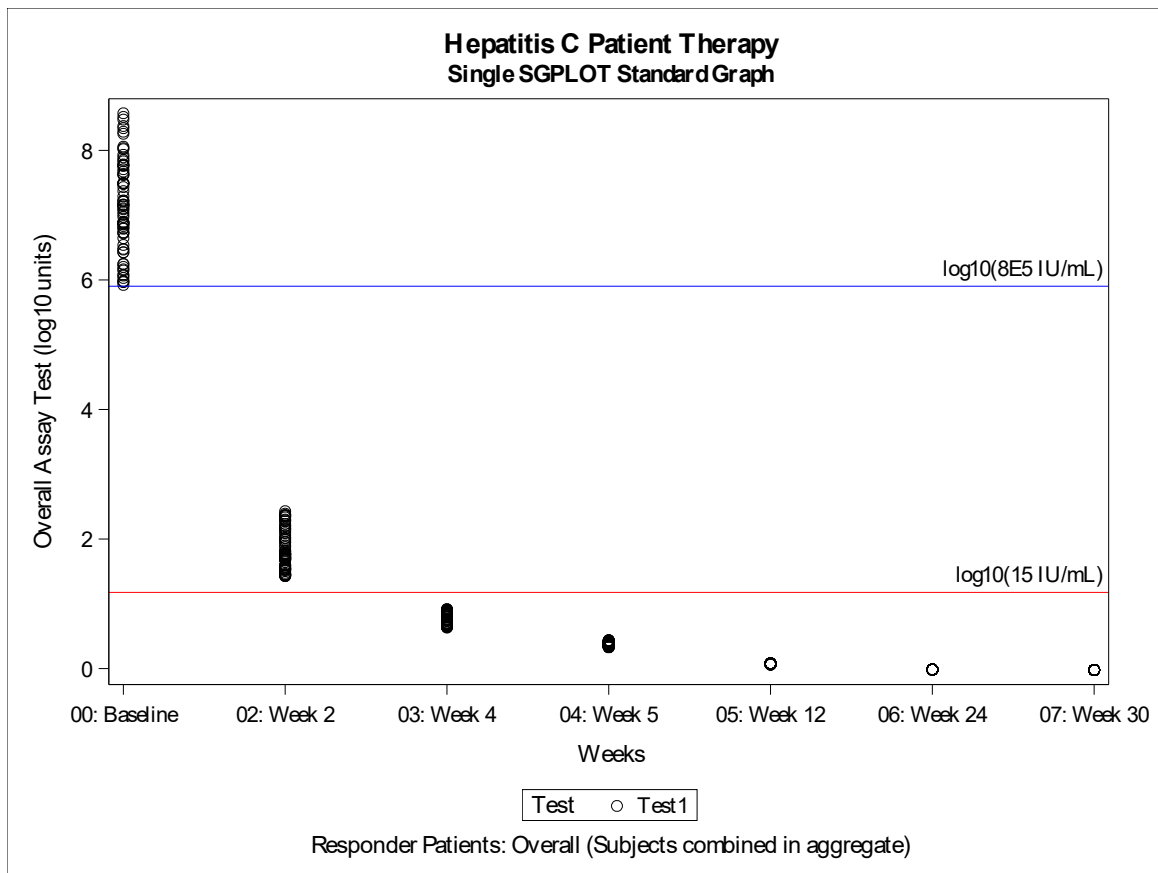


Figure 1. Hepatitis C Therapy Response (HepCTR) Simulated Aggregate Data for Responders

Now, suppose pre-SAS v9.4, if the user wished to replicate an SGPLOT across many, many subjects (not just a few where quick repeated code would save the day), for example, he/she would need to perform some fancy programming.

AUTOMATION

Producing one to a few graphs normally is quite easy to do with a quick cut and paste or macroizing your repeating code to use with a few macro calls. However, at times, one may need to produce the same graph for more than just a few times (say, 50, 100, 200, etc.) For example, **one may need to expand (or**

separate) an aggregate graph with a great many patients (or subjects) into individual patient graphs. For this, the copy/paste and macroizing of the simple code does not work efficiently at all! This necessitates an automated solution. We give an example using the simulated patient Hepatitis C Therapy Response data (HepCTR; Appendix A).

Step 3: **Many Patients in Separate Graphs with pre-SAS v9.4.** Run the following *fancy* code to produce separate SGPLOT graphs using SAS Macro code.

```
* Create macro variables with values and a total count of distinct values
to iterate the later macro by SubjectID ;
proc sql noprint ;
  select distinct SampleID into :varVal1- from Responders ;
  %let varCount = &SQLOBS. ;
quit ;
* SAS-provided style macro for ODS should run in any SAS installation ;
* cycles through the indicated types (in example below, up to 4 groups)
in your SG graphs when "style=markers" is indicated on the ODS line ;
%modstyle(
  name = markers ,
  parent = listing ,
  type = CLM ,
  linestyles = solid dash shortdash dot ,
  colors = green blue purple red ,
  markers = circle triangle square diamond ) ;

* this will re-direct to a non-server path - useful if working on a
restricted server that if path not set will produce write errors;
ods listing style=markers gpath="c:\\" ;
options orientation = landscape ; * setting page orientation to landscape;
ods RTF file="c:\WUSS2018_Sacramento_InThePast_&sysdate..RTF"
  gtitle style=markers ; * exporting to an RTF document ;
%macro bySampleID ;
  %do index = 1 %to &varCount ;
    title1 "Hepatitis C Patient Therapy Example" ;
    title2 "Multiple SGPLOT Graphs by Patient using Proc SQL and
      SAS Macro" ;
    proc sgplot data = Responders
      (where=(SampleID=&&varVal&index.)) ;
      scatter y=log10Observed x=TP / MARKERATTRS=(SIZE=8px)
        group=TEST name="TEST" ;
      yaxis values = (0 TO 8 BY 0.5)
        labelattrs=(color=Grey family=Imago)
        valueattrs=(color=Grey family=Imago) ;
      refline 5.903089987 / axis=y label="log10(8E5 IU/mL)"
        lineattrs=(color="blue")
        labelloc=inside labelpos=max ;
      refline 1.17609126 / axis=y label="log10(15 IU/mL)"
        lineattrs=(color="red")
        labelloc=inside labelpos=max ;
      inset "Sample ID = &&varVal&index." / position = topright ;
      label TP = "Time Point"
        log10Observed = "Result (log10 IU/mL)" ;
    run ;
  %end ;
%mend bySampleID ;
%bySampleID ;
```

```
ods rtf close ;
options orientation = portrait ; * resetting page orientation to portrait;
```

Figure 2 shows the three graphs of the 10000 results of running the SAS code above for the Hepatitis C Therapy Response (HepCTR) Simulated by-Subject Data using PROC SQL and SAS/MACRO code for Responders across 30 weeks on treatment.

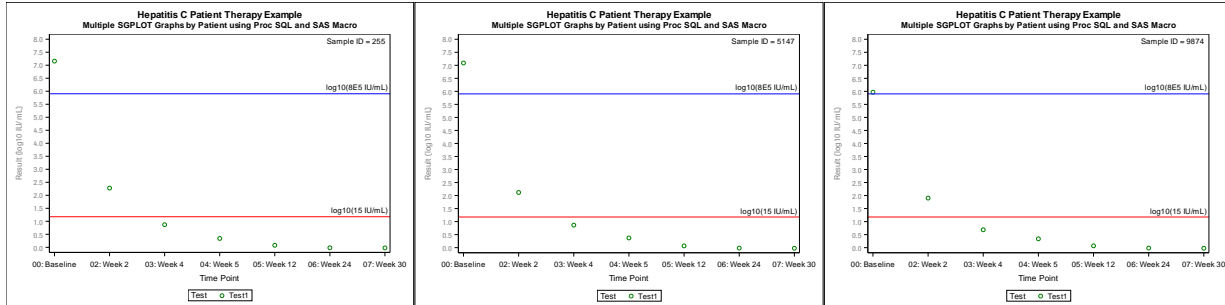


Figure 2. Hepatitis C Therapy Response (HepCTR) Simulated by-Subject Data using PROC SQL and SAS/MACRO code for Responders (3 of 10000 graphs shown)

Recent advancements in the SAS SG procedures have added the “by” statement to make it much easier than the macro coding found in **Step 3**, above.

Step 4: Patients in Separate Graphs with SAS v9.4. Run the following code to produce separate SGPLOT graphs. Notice the “by” in the code is red, indicating it may not be allowed in the SGPLOT code. However, this is acceptable and will run in the meantime that SAS Institute adds this to the procedure-allowed code and turn it blue (current SAS implementation used is 9.4 M5):

```
ods noproctitle ;
ods RTF file = "C:\WUSS2018_Responders_bySubject_&sysdate..RTF" style=MonoChromePrinter ;
* above: sysdate appends current data to your file name and style gives white background ;
title2 "" ;
footnote1 "Responder Patients: Individual Subjects (separately)" ;
proc sort data=Responders ; by SampleID TP ; run ;
proc sgplot data=Responders ;
  by SampleID ;
  scatter y=log10Observed x=TP / group=Test ;
  YAXIS LABEL = "Overall Assay Test (log10 units)" ;
  XAXIS LABEL = "Weeks" ;
run ;
ods rtf close ;
title2 ; footnote1 ;
```

Figure 3 shows the three graphs of the 10000 results of running the SAS code above for the Hepatitis C Therapy Response (HepCTR) Simulated by-Subject Data for Responders across 30 weeks on treatment.

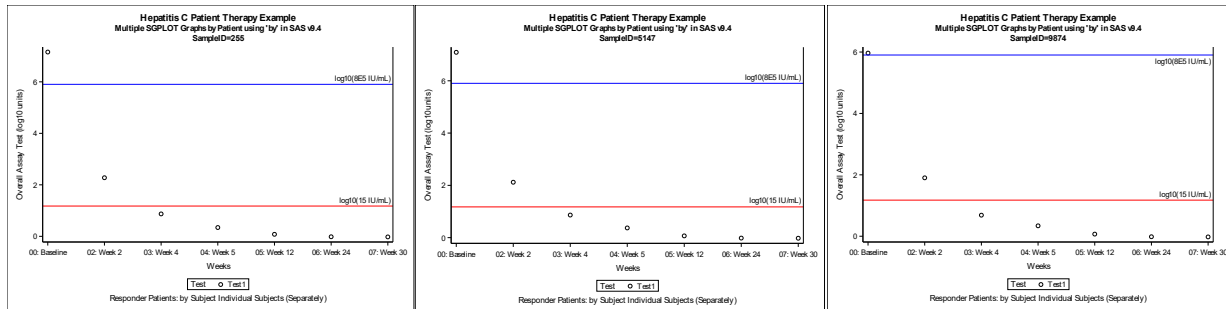


Figure 3. Hepatitis C Therapy Response (HepCTR) Simulated by-Subject Data for Responders (3 of 10000 graphs shown)

CUSTOMIZATION

Sometimes, we are forced to use customization because the SG procedures may not have a user-requested option available as of yet. For example, the user cannot currently add a “targeted” regression equation in an SGPLOT graph. Here is an example of this using simulated log-log data from two tests (Log-Log; Appendix B).

Step 5: Adding a Regression Equation and Other Items to Your Graph. The following code steps first create a regression equation using PROC REG (**Step 5a**), saving the results using ODS OUTPUT OUT, then using PROC IML (**Step 5b**) to extract bits and pieces of what you need, declaring them macro variables then adding them to your SGPLOT graph (**Step 5c**). This is an example that can be generalized to add any type of procedure output to any SG graph that you may desire to customize.

Step 5a: Creating a Regression Equation and Saving Results. After running the SAS code in **Appendix B** to create the Log-Log data, run the following code to produce a regression equation using PROC REG whilst saving the result using the ODS OUTPUT OUT option [NOTE: You will need to use the “ods trace on ;” at the beginning of your code that you want to extract parameter estimates from and “ods trace off ;” at the end of that code in order to find the correct output for the “ods output ...” (See, for example, Oltsik, 2008 and **APPENDIX D**)].

```
* bits and pieces ;
ods trace on ;
ods output ParameterEstimates=Parms1 FitStatistics=FitStats1 ;
proc reg data= TwoTests_OneRep ;
  model Test2 = Test1 ;
  output out=OLSresids1 predicted=pred1 residual=resids1 press=press1 ;
run ; quit ;
ods trace off ;
```

Step 5b: Extract Information from ODS OUTPUT for IML Processing. Next, run the next SAS code that uses the SAS IML procedure to extract the parameter estimates from the saved results in **Step 5a** (see also temporary data sets in **APPENDIX D**).

```
proc iml ;
  use Parms1 ; * use the Parms1 data set from above ;
  read all VAR{estimate lowercl uppercl} into X ; * read only vars we need;
  close Parms1 ;

  use FitStats1 ; * use the FitStats1 data set from above ;
  read all VAR{NVALUE2} into Y ; * read only var we need;
  close Fitstats1 ;

  use OLSresids1 ; * use the OLSresids1 data set from above ;
  read all VAR{SubjID} into Z ; * extract SubjID to count how many IDs ;
```

```

close OLSresids1 ;

N_Tot_1 = nrow(Z[,1]) ; * count number of observations in SubjID ;

ols1_b0 = X[1,1] ; * extract intercept in Y = b0 + b1*X equation ;
ols1_b1 = X[2,1] ; * extract slope ;

CI95_LL1_b0 = X[1,2] ; * extract lower confidence limit for intercept ;
CI95_UL1_b0 = X[1,3] ; * extract upper confidence limit for intercept ;

CI95_LL1_b1 = X[2,2] ; * extract lower confidence limit for slope ;
CI95_UL1_b1 = X[2,3] ; * extract upper confidence limit for slope ;

Rsquare1 = Y[1,1] ; * extract R-square value ;

* assign a value to a macro variable for use in SG procedures ;
call symputx("N_Tot_1",N_Tot_1) ;

call symputx("ols1_b0",round(ols1_b0,0.001)) ;
call symputx("ols1_b1",round(ols1_b1,0.001)) ;

call symputx("CI95_LL1_b0",round(CI95_LL1_b0,0.001)) ;
call symputx("CI95_UL1_b0",round(CI95_UL1_b0,0.001)) ;

call symputx("CI95_LL1_b1",round(CI95_LL1_b1,0.001)) ;
call symputx("CI95_UL1_b1",round(CI95_UL1_b1,0.001)) ;

call symputx("Rsquare1",round(Rsquare1,0.01)) ;
run ;

```

Step 5c: Adding the Macro Variable Results to SGPLOT. Finally, run the next SAS code that uses the macro variable results in **Step 5b** to populate the SGPLOT graph as shown in Figure 4, below.

```

ods rtf file = "C:\Results_MethodComparison_FullMonty_&sysdate..rtf"
          gtitle style=markers ; * gtitle will add your titles to the
top of your graph - NOTE: if your title still does not show up, perform an
"ODS_ALL_close" command right before your code as illustrated in
APPENDIX B and re-run your code ;
proc sgplot data = TwoTests_OneRep noautolegend ;
  scatter x=Test1 y=Test2 / markerattrs=(size=5px) ;
  reg      x=Test1 y=Test2 / markerattrs=(size=5px)
          LINEATTRS = (THICKNESS=0.9 COLOR=Blue PATTERN=1) name="OLS" ;

  LINEPARM x=0 y=0 slope=1 / LEGENDLABEL = "Unity: Y=X"
          LINEATTRS = (THICKNESS=0.2 COLOR=Black PATTERN=2) name="Unity" ;

  XAXIS LABEL = "Test 1 (log10 cp/mL)" VALUES = (1 TO 9 BY 0.5) ;
  YAXIS LABEL = "Test 2 (log10 cp/mL)" VALUES = (1 TO 9 BY 0.5) ;

  KEYLEGEND "Unity" "OLS" / LOCATION=outside POSITION=bottom ;
  TITLE1 "Method Comparison Study" ;
  TITLE2 "Test 2 (log10 cp/mL) vs. Test 1 (log10 cp/mL)" ;

  FOOTNOTE1 ; FOOTNOTE2 ;
  INSET "OLS Regression (N= &&N_Tot_1) "
        "Y = &&ols1_b0 + &&ols1_b1 X"

```

```

"R-square= &&Rsquare1"
"95% CI Intercept (&&CI95_LL1_b0, &&CI95_UL1_b0)"
"95% CI for Slope: (&&CI95_LL1_b1, &&CI95_UL1_b1)"
/ POSITION = BOTTOMRIGHT BORDER;

run ;
ods rtf close ;

```

Figure 4, below, shows customized SGPLOT graph using the Log-Log method comparison/correlation data from Appendix B following the above **Steps 5a to 5c**.

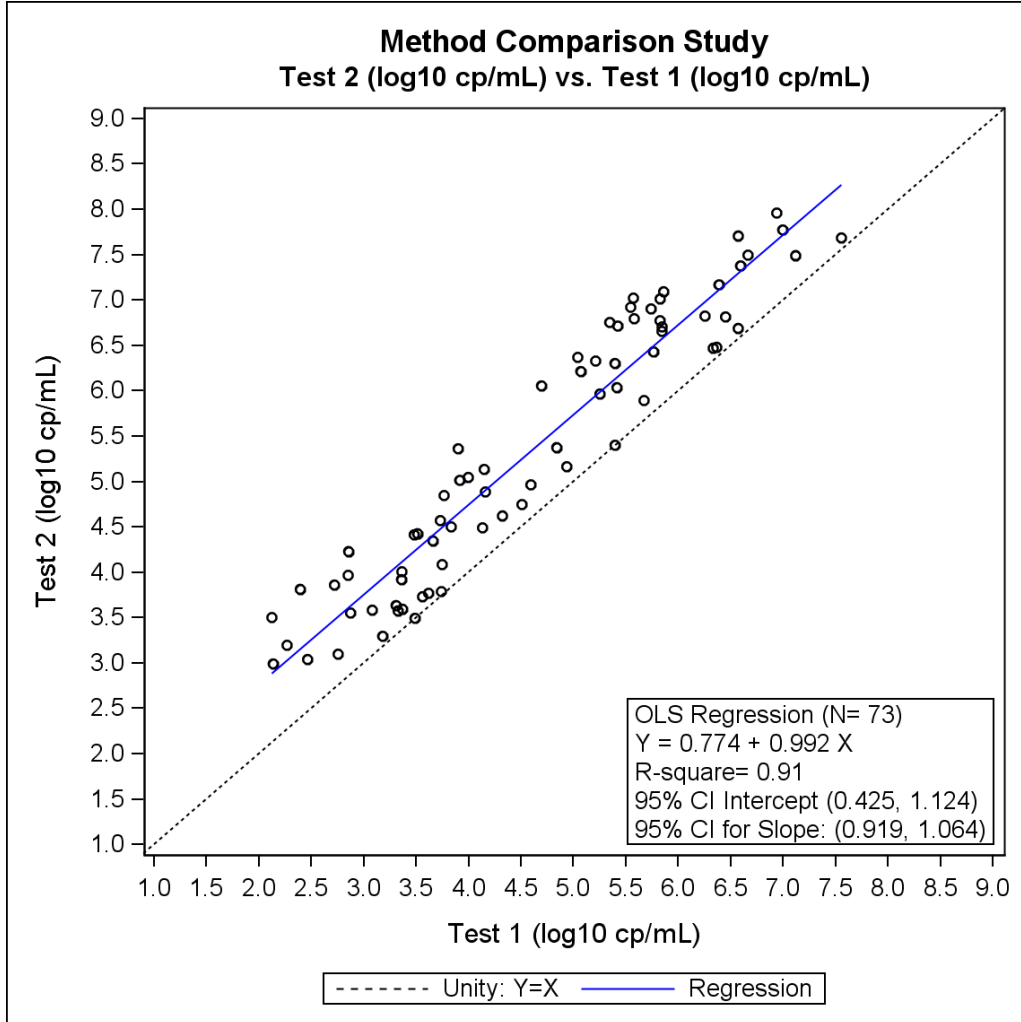


Figure 4. Method Comparison/Correlation Study Comparing Test 2 to Test 1 using (Log-Log) Simulated Log-Log Data and both SAS IML and SGPLOT procedures.

MACROTIZING YOUR CUSTOMIZATION

The next natural step is to macrotize your customized code to use again and again. **Appendix C** shows the `odsplot` SAS MACRO that macrotizes the customization in the previous section using the Log-Log method comparison data. The macro is shown in its entirety, including the macro call at the end of the appendix.

CONCLUSION

The authors have laid a path for automation and customization using the SG procedures. Along the way, more than a few useful code snippets were shown that can be quite useful in other SAS automation and customization endeavors.

REFERENCES

Heath D (2009). "Paper 324-2009: Secrets of the SG Procedures". *Proceedings of the SAS Global Forum 2009 Conference*. Washington, DC: The SAS Institute, Inc., Cary, NC. Available at <https://support.sas.com/resources/papers/proceedings09/324-2009.pdf>

Matange S (2011). "Paper 281-2011: Tips and tricks for clinical graphs using ODS graphics". *Proceedings of the SAS Global Forum 2011 Conference*. Las Vegas, NV: The SAS Institute, Inc., Cary, NC. Available at <https://support.sas.com/resources/papers/proceedings11/281-2011.pdf>

Matange S (2016). "PharmaSUG 2016 – Paper DG02: Clinical graphs using SAS". *Proceedings of the PharmaSUG 2016 Conference*. Denver, CO: The SAS Institute, Inc., Cary, NC. Available at <https://support.sas.com/resources/papers/proceedings16/SAS4321-2016.pdf>

Oltsik M (2008). "ODS and Output Data Sets: What you need to know". *Proceedings of the SAS Global Forum 2008 Conference*. San Antonio, TX: The SAS Institute, Inc., Cary, NC. Available at <http://www2.sas.com/proceedings/forum2008/086-2008.pdf>

Slaughter SJ, Delwiche LD (2012). "Paper 259-2012: Graphing made easy with SG Procedures". *Proceedings of the SAS Global Forum 2012 Conference*. Orlando, FL: The SAS Institute, Inc., Cary, NC. Available at <http://support.sas.com/resources/papers/proceedings12/259-2012.pdf>

Slaughter SJ, Delwiche LD (2015). "Paper 2441-2015: Graphing made easy with SG PLOT and SG PANEL Procedures". *Proceedings of the SAS Global Forum 2012 Conference*. Washington, DC: The SAS Institute, Inc., Cary, NC. Available at <https://support.sas.com/resources/papers/proceedings15/2441-2015.pdf>

ACKNOWLEDGMENTS

The authors wish to acknowledge the editors and reviewers for their corrections and suggestions. Specifically, we wish to thank our Roche colleagues and Alison Canchola for their attention to detail from which this paper and resulting presentation are greatly enhanced in quality. However, any remaining errors belong to the authors alone.

RECOMMENDED READING

- *Statistical Graphics Procedures by Example*
- *A Handbook of Statistical Graphics using SAS®*
- *Clinical Graphics Using SAS®*

CONTACT INFORMATION

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

Jesse A. Canchola, MS, PStat®
Roche Molecular Systems, Inc.
Phone: +1.925.730.8125
eMail: Jesse.Canchola@Roche.Com
Web: <https://molecular.roche.com/>

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

APPENDICES

Appendix A. Generated Example Data Set. Longitudinal Patient Hepatitis C Therapy Response.

```
* RESPONDERS ;
* generate 100 patients with 13 time points each ;
%LET SampleID = 100 ; * generate 100 subjects ;
%LET NumSamples = 13 ; * with 13 time points each ;
data Xvalues (rename = (i = SampleID j = Time)) ;
  seed = 81638 ;
  do i = 1 to &SampleID ;
    do j = 1 to &NumSamples ;
      X = 35 * ranuni(seed) ; * generate numbers from 0 to 35 as time
                             variable in weeks ;
      output ;
    end ;
  end ;
run ;

* simulating responder patients ;
data Responders_TMP0 (drop=Seed S B0 B1 B2 B3) ;
  set Xvalues ;
  seed = 183609 ;

  b0 = 6.6 ; b1 = -4.5E-01 ; b2 = 1.2 ; b3 = 2.01E-01 ;
  * rational model ;
  Y = ( b0 + b1 * X ) / ( 1 + b2 * X + b3 * X**2 ) ;

  * jittering X values for each subject ;
  s = 1 ; * scale factor ;
  jY = Y + s * ranuni(seed) - 0.001 ;
  if jY < 0 then jY = 0 ;

  Test1 = Y ;
  Test2 = jY ;
run ;
data Responders_Test1 (rename=(Test1=log10Observed)) ;
  set Responders_TMP0 (drop=Y jY Test2) ;
  Test = "Test1" ;
run ;
data Responders_Test2 (rename=(Test2=log10Observed)) ;
  set Responders_TMP0 (drop=Y jY Test1) ;
  Test = "Test2" ;
run ;
data Responders_TMP ;
set Responders_Test1 Responders_Test2 ;
run ;

data Responders_TMP1 ;
  set Responders_TMP ( rename=(X=Weeks) ) ;

  Responder = 1 ;

  * Select time points at Weeks (+-3 days) ;
  if 0.0 < Weeks < 1.0 then TP = "00: Baseline" ;
```

```

else if 1.0 <= Weeks < 1.4 then TP = "01: Week 1" ;
else if 1.6 <= Weeks < 2.4 then TP = "02: Week 2" ;
else if 3.6 <= Weeks < 4.4 then TP = "03: Week 4" ;
else if 5.6 <= Weeks < 6.4 then TP = "04: Week 5" ;
else if 11.6 <= Weeks < 12.4 then TP = "05: Week 12" ;
else if 23.6 <= Weeks < 25.4 then TP = "06: Week 24" ;
else if 29.6 <= Weeks < 30.4 then TP = "07: Week 30" ;

* keep only the data for specific Time periods ;
if substr(TP,1,2) in ("00","01","02","03","04","05","06","07") ;

* indicator variable of just 1s to sum up in Proc SQL below ;
indicator = 1 ;

TimeStem = substr(TP,1,2) ;
run ;

* if multiple dates within subject generated by simulation then take the
first date ;
proc sort data=Responders_TMP1 ; by SampleID TP TimeStem ; run ;
data Responders_TMP2 ;
  set Responders_TMP1 ;
  by SampleID TP TimeStem ;
  if first.TimeStem ;
run ;

* keep if number of time points is 4 ;
proc sql ;
  create table Responders_GE4WKS as
  select
    SampleID
      ,sum(indicator) as Total
  from Responders_TMP2
  group by SampleID
  order by SampleID ;
quit ;

proc sort data=Responders_TMP2 ; by SampleID ; run ;
proc sort data=Responders_GE4WKS ; by SampleID ; run ;
data Responders (drop=indicator timestem total) ;
  merge Responders_TMP2 Responders_GE4WKS ;
  by SampleID ;
run ;
* end data set creation for Responders ;
* END OF SAS CODE ;

```

Appendix B. Generated Example Data Set. Log-Log Data Generation and SGPLOT Customization

```

* Log-Log Data using SGPLOT: Customization ;
%LET SampleID = 150 ; * generate 150 subjects ;
%LET NumSamples = 2 ; * with 2 replicate each but later only take first
replicate ;
data Xvalues (rename = (i = SubjID j = Replicate)) ;
  seed = 603287 ;
  do i = 1 to &SampleID ;
    do j = 1 to &NumSamples ;
      X = ranuni(seed) ;
      output ;
    end ;
  end ;
run ;

* simulating responder patients ;
data TwoTests_OneRep (drop=Seed S B0 B1) ;
  set Xvalues ;
  seed = 284631 ;

  b0 = 1.24 ; b1 = 9.98 ;
  Y = b0 + b1 * X ; * linear model ;

  * jittering X values for each subject ;
  s = 1.5 ; * scale factor for spread ;
  jY = Y + s * ranuni(seed) - 0.01 ; * random uniform distribution to
                                     generate random observations ;
  if jY < 0 then jY = 0 ;

  * let Y be Test1 and the jittered Y be Test2 ;
  Test1 = Y ;
  Test2 = jY ;

  * restrict observations from 2 to 8 using Test1 and Test2 ;
  if ( 2 <= Test1 <= 8 ) AND ( 2 <= Test2 <= 8 ) ;

  * select first of two replicates for pedagogical reasons ;
  if replicate = 1 ;
run ;

* END Log-Log data creation ;
***** ;
***** ;

ods _ALL_ close ;
* Use the MODSTYLE macro supplied by SAS to change the default colors and
markers ;
%modstyle(name = markers ,
          parent = listing,
          type = CLM ,
          markers = circle triangle square) ; *circle triangle square
                                             circlefilled trianglefilled diamondfilled) ;

* The investigators want a square plot (no proc template necessary) ;

```

```

ods graphics / width = 550px height = 550px ;

ods graphics /
imagename="\rpbmssasp1\biometrics\Global\Presentations\WUSS\2018_Sacramento\
Results\PNG" imagefmt=png ;

options mprint mlogic symbolgen ; * prints ;

* bits and pieces ;
ods output ParameterEstimates=Parms1 FitStatistics=FitStats1 ;
proc reg data= TwoTests_OneRep ;
  model Test2 = Test1 / clm cli clb ;
  output out=OLSresids1 predicted=pred1 residual=resids1 press=press1 ;
run ; quit ;

proc iml ;
  use Parms1 ;
  read all VAR{estimate lowercl uppercl} into X ;
  close Parms1 ;

  use FitStats1 ;
  read all VAR{NVALUE2} into Y ;
  close Fitstats1 ;

  use OLSresids1 ;
  read all VAR{SubjID} into Z ;
  close OLSresids1 ;

  N_Tot_1 = nrow(Z[,1]) ;

  ols1_b0 = X[1,1] ;
  ols1_b1 = X[2,1] ;

  CI95_LL1_b0 = X[1,2] ;
  CI95_UL1_b0 = X[1,3] ;

  CI95_LL1_b1 = X[2,2] ;
  CI95_UL1_b1 = X[2,3] ;

  Rsquare1 = Y[1,1] ;

  call symputx("N_Tot_1",N_Tot_1) ;

  call symputx("ols1_b0",round(ols1_b0,0.001)) ;
  call symputx("ols1_b1",round(ols1_b1,0.001)) ;

  call symputx("CI95_LL1_b0",round(CI95_LL1_b0,0.001)) ;
  call symputx("CI95_UL1_b0",round(CI95_UL1_b0,0.001)) ;

  call symputx("CI95_LL1_b1",round(CI95_LL1_b1,0.001)) ;
  call symputx("CI95_UL1_b1",round(CI95_UL1_b1,0.001)) ;

  call symputx("Rsquare1",round(Rsquare1,0.01)) ;
run ;

* Customize the SGPLOT using the result from above ;

```

```

ods rtf file = "C:\Results_MethodComparison_FullMonty_&sysdate..rtf" gtitle
style=markers ;
proc sgplot data = TwoTests_OneRep noautolegend ;
  scatter x=Test1 y=Test2 / markerattrs=(size=5px) ;
  reg      x=Test1 y=Test2 / markerattrs=(size=5px)
          LINEATTRS = (THICKNESS=0.9 COLOR=Blue PATTERN=1) name="OLS" ;

          LINEPARM x=0 y=0 slope=1 / LEGENDLABEL = "Unity: Y=X"
          LINEATTRS = (THICKNESS=0.2 COLOR=Black PATTERN=2)
                    name="Unity" ;

XAXIS LABEL = "Test 1 (log10 cp/mL)" VALUES = (1 TO 9 BY 0.5) ;
YAXIS LABEL = "Test 2 (log10 cp/mL)" VALUES = (1 TO 9 BY 0.5) ;

KEYLEGEND "Unity" "OLS" / LOCATION=outside POSITION=bottom ;
TITLE1 "Method Comparison Study" ;
TITLE2 "Test 2 (log10 cp/mL) vs. Test 1 (log10 cp/mL)" ;

FOOTNOTE1 ; FOOTNOTE2 ;
INSET "OLS Regression (N= &&N_Tot_1) "
      "Y = &&ols1_b0 + &&ols1_b1 X"
      "R-square= &&Rsquare1"
      "95% CI Intercept (&&CI95_LL1_b0, &&CI95_UL1_b0)"
      "95% CI for Slope: (&&CI95_LL1_b1, &&CI95_UL1_b1)" / POSITION =
BOTTOMRIGHT BORDER;
run ;
ods rtf close ;
* END OF SAS CODE ;

```

Appendix C. SGPLOT Customization and Macrotization of Code in Appendix B: The OLSplot SAS Macro

```

/* *****
Program:      OLSplot.sas
Revision:     --

Purpose:      Method comparison scatter plot using ordinary least square (OLS)
regression

Macro Inputs: (req=required)
  dsin       = example.mf_cornell : req
  analyte    = CMV      : req
  unit       = cp/mL    : req
  visit      = ,        : visit variable, can be blank
  subject    = ,        : subject variable
  xvar       = Test1    : req x-axis variable
  xlab       = "Test 1 (log10 cp/mL)" : x-axis label, can be blank
  yvar       = Test2    : y-axis variable, req
  ylab       = "Test 2 (log10 cp/mL)" : y-axis label, can be blank
  grid       = "NO"    : graph grid requested, req: Yes or No
  parmno     = 1       : req: analysis number: >= 1
  graphmin   = 2       : req: minimum graph value (both axes)
  graphmax   = 6       : req: maximum graph value (both axes)
  graphincrement = 0.5 : req: graph increment (both axes)
  title1     = "CMV-326: Platform Comparison Study for CMV Viral Load"
              : first title, can be blank
  title2     = "Artus ASR CMV (log10 cp/mL) vs. TaqMan® CMV (log10 cp/mL)"
              : second title, can be blank

Created by:   Jesse A. Canchola
Creation Date: 13Mar2017

Modified by:  Jesse A. Canchola
Modify Date:  07-Jul-2017
Modify Reason: Adjusted code to be more flexible plus added confidence
              Limits to the PROC REG output for future use

Key:         N/A=Not Applicable
***** */
%macro OLSplot(
  dsin = ,
  where = ,
  analyte = ,
  unit = ,
  visit = ,
  subject = ,
  xvar = ,
  xlab = ,
  yvar = ,
  ylab = ,
  grid = ,
  parmno = ,
  graphmin = ,
  graphmax = ,
  graphincrement = ,

```

```

        title1 = ,
        title2 = ) ;

ods output ParameterEstimates=Parms&parmno. FitStatistics=FitStats&parmno. ;
proc reg data= &dsin ;
    model &yvar = &xvar / clm cli clb ;
    output out=OLSresids&parmno. predicted=pred&parmno. residual=resids&parmno.
    press=press&parmno. ;
    %if %upcase(&visit) >= 0 %then %do ;
        %str(where visitn = &visit ; ) ;
    %end ;
run ; quit ;

proc iml ;
    use Parms&parmno. ;
    read all VAR{estimate lowercl uppercl} into X ;
    close Parms&parmno. ;

    use FitStats&parmno. ;
    read all VAR{NVALUE2} into Y ;
    close Fitstats&parmno. ;

    use OLSresids&parmno. ;
    read all VAR{&subject.} into Z ;
    close OLSresids&parmno. ;

    N_Tot_&parmno. = nrow(Z[,1]) ;

    ols&parmno._b0 = X[1,1] ;
    ols&parmno._b1 = X[2,1] ;

    CI95_LL&parmno._b0 = X[1,2] ;
    CI95_UL&parmno._b0 = X[1,3] ;

    CI95_LL&parmno._b1 = X[2,2] ;
    CI95_UL&parmno._b1 = X[2,3] ;

    Rsquare&parmno. = Y[1,1] ;

    call symputx("N_Tot_&parmno.",N_Tot_&parmno.) ;

    call symputx("ols&parmno._b0",round(ols&parmno._b0,0.001)) ;
    call symputx("ols&parmno._b1",round(ols&parmno._b1,0.001)) ;

    call symputx("CI95_LL&parmno._b0",round(CI95_LL&parmno._b0,0.001)) ;
    call symputx("CI95_UL&parmno._b0",round(CI95_UL&parmno._b0,0.001)) ;

    call symputx("CI95_LL&parmno._b1",round(CI95_LL&parmno._b1,0.001)) ;
    call symputx("CI95_UL&parmno._b1",round(CI95_UL&parmno._b1,0.001)) ;

    call symputx("Rsquare&parmno.",round(Rsquare&parmno.,0.01)) ;
run ;

proc sgplot data = &dsin noautolegend ;
    scatter x=&xvar y=&yvar / markerattrs=(size=5px) ;
    reg      x=&xvar y=&yvar / markerattrs=(size=5px)
            lineattrs = (thickness=0.9 color=blue pattern=1) name="OLS" ;

```



```

lineparm x=0 y=0 slope=1 / legendlabel = "Unity: Y=X"
      lineattrs = (thickness=0.2 COLOR=Black pattern=2) name="Unity" ;

      %if %upcase(&grid) = "YES" %then %do ;
          %str( XAXIS LABEL = &xlab VALUES = (&GraphMin TO &GraphMax BY
&GraphIncrement) GRID ; ) ;
          %end ;
          %if %upcase(&grid) ^= "YES" %then %do ;
          %str( XAXIS LABEL = &xlab VALUES = (&GraphMin TO &GraphMax BY
&GraphIncrement) ; ) ;
          %end ;

          %if %upcase(&grid) = "YES" %then %do ;
          %str( YAXIS LABEL = &ylob VALUES = (&GraphMin TO &GraphMax BY
&GraphIncrement) GRID ; ) ;
          %end ;
          %if %upcase(&grid) ^= "YES" %then %do ;
          %str( YAXIS LABEL = &ylob VALUES = (&GraphMin TO &GraphMax BY
&GraphIncrement) ; ) ;
          %end ;

KEYLEGEND "Unity" "OLS" / LOCATION=outside POSITION=bottom ;
TITLE1 &title1 ;
TITLE2 &title2 ;

FOOTNOTE1 ; FOOTNOTE2 ;
INSET "OLS Regression (N= &&N_Tot_&parmno.) "
      "Y = &&ols&parmno._b0 + &&ols&parmno._b1 X"
      "R-square= &&Rsquare&parmno."
      "95% CI Intercept (&&CI95_LL&parmno._b0, &&CI95_UL&parmno._b0)"
      "95% CI for Slope: (&&CI95_LL&parmno._b1, &&CI95_UL&parmno._b1)" /
POSITION = BOTTOMRIGHT BORDER;

      %if %upcase(&visit) >= 0 %then %do ;
          %str(where visitn = &visit ; ) ;
          %end ;
run ;
%mend OLSplot ;

* ***** ;
* MACRO CALL for OLSplot.sas ;
* ***** ;
%include "C:\OLSplot.sas" ;

ods _ALL_ close ;
* Use the MODSTYLE macro supplied by SAS to change the default colors and
  markers ;
%modstyle(name = markers ,
          parent = listing,
          type = CLM ,
          markers = circle triangle square) ;

* The investigators want a square plot (no proc template necessary) ;
ods graphics / width = 550px height = 550px ;

```

```

ods graphics / imagename="C:\PNG" imagefmt=png ;

* For any macro troubleshooting ;
options mprint mlogic symbolgen ;

ods rtf file = "C:\Results_MethodComparison_&sysdate..rtf" gtitle
style=markers ;
%OLSplot(
  dsin      = TwoTests_OneRep , /* required */
  where     = ,
  analyte   = CMV      , /* required */
  unit      = cp/mL    , /* required */
  visit     = ,        /* visit variable, can be blank */
  subject   = SubjID ,
  xvar      = Test1    , /* required */
  xlab      = "Test 1 (log10 cp/mL)" , /* can be blank */
  yvar      = Test2    , /* required */
  ylab      = "Test 2 (log10 cp/mL)" , /* can be blank */
  grid      = "NO"     , /* required: Yes or No */
  parmno    = 1        , /* required: analysis number: >= 1 */
  graphmin  = 1        , /* required */
  graphmax  = 9        , /* required */
  graphincrement = 0.5 , /* required */
  title1    = "Method Comparison Study", /* can be blank */
  title2    = "Test 2 (log10 cp/mL) vs. Test 1 (log10 cp/mL)" ; /* can be
                                                                    blank */

ods rtf close ;

* END OF SAS CODE ;

```

Appendix D. Extraction of variables from a PROC using ODS TRACE

Step 5a in the **CUSTOMIZATION** section requires the **ODS TRACE** to discover the correct naming of the **ODS OUTPUT** parameters. The example given shows the following:

First pass. To discover the correct parameters for the **ODS OUTPUT** naming, use **ODS TRACE ON** and **ODS TRACE OFF** to bracket your PROC code – whatever PROC that may be. In our case it is PROC REG:

```
ods trace on ;  
proc reg data= TwoTests_OneRep ;  
  model Test2 = Test1 ;  
run ; quit ;  
ods trace off ;
```

Bracket your PROC (whatever that may be) with these.

Note that these results are found in the SAS LOG window/tab **after** running your code.

Upon running this code, you will see the output in the **SAS LOG** window/tab as shown in the **Output 5a** SAS log output box below.

Output 5a. SAS Log Output using ODS TRACE for correct ODS OUTPUT parameter discovery and naming.

Output Added:

```
-----  
Name:      NObs  
Label:     Number of Observations  
  
Template:  Stat.Reg.NObs  
Path:     Reg.MODEL1.Fit.Test2.NObs  
-----
```

Output Added:

```
-----  
Name:      ANOVA  
Label:     Analysis of Variance  
Template:  Stat.REG.ANOVA  
Path:     Reg.MODEL1.Fit.Test2.ANOVA  
-----
```

Output Added:

```
-----  
Name:      FitStatistics  
Label:     Fit Statistics  
Template:  Stat.REG.FitStatistics  
Path:     Reg.MODEL1.Fit.Test2.FitStatistics  
-----
```

Output Added:

```
-----  
Name:      ParameterEstimates  
Label:     Parameter Estimates  
Template:  Stat.REG.ParameterEstimates  
Path:     Reg.MODEL1.Fit.Test2.ParameterEstimates  
-----
```

Output Added:

```
-----  
Name:      OutputStatistics  
Label:     Output Statistics  
Template:  Stat.Reg.OutputStatistics  
Path:     Reg.MODEL1.ObswiseStats.Test2.OutputStatistics  
-----
```

Output Added:

```
-----  
Name:      ResidualStatistics  
Label:     Residual Statistics  
Template:  Stat.Reg.ResidualStatistics  
Path:     Reg.MODEL1.ObswiseStats.Test2.ResidualStatistics  
-----
```

Selected PROC REG ODS output data set names (among 6 possible data set outputs: **Nobs**, **ANOVA**, **FitStatistics**, **ParameterEstimates**, **OutputStatistics**, **ResidualStatistics**) using the ODS TRACE ON and ODS TRACE OFF option bracketing your PROC code.

Note that these results are found in the SAS LOG window/tab **after** running your code (as shown in "**First Pass**" above).

Second pass. The actual naming of discovered parameters for **ODS OUTPUT** happens here:

```
ods output ParameterEstimates=Parms1 FitStatistics=FitStats1 ;  
proc reg data= TwoTests_OneRep ;  
  model Test2 = Test1 ;  
run ; quit ;
```

Name the chosen parameters with whatever name you choose and run your code to get **the SAS data sets output below with your given names (PARMS1, FITSTATS1)**.

Temporary SAS Data Sets Created Using ODS OUTPUT code above.

The screenshot shows two SAS ODS OUTPUT tables. The first table, titled 'PARMS1', has columns: Model, Dependent, Variable, DF, Estimate, StdErr, tValue, and ProbT. It contains two rows of parameter estimates. The second table, titled 'FITSTATS1', has columns: Model, Dependent, Label1, cValue1, nValue1, Label2, cValue2, and nValue2. It contains three rows of fit statistics. Red callout boxes with arrows point to the table titles 'PARMS1' and 'FITSTATS1' in the screenshot. A larger red box at the bottom left contains the text 'SAS temporary data sets created using ODS OUTPUT.'

Model	Dependent	Variable	DF	Estimate	StdErr	tValue	ProbT
1 MODEL1	Test2	Intercept	1	0.77433	0.17518	4.42	<.0001
2 MODEL1	Test2	Test1	1	0.99153	0.03619	27.40	<.0001

Model	Dependent	Label1	cValue1	nValue1	Label2	cValue2	nValue2
1 MODEL1	Test2	Root MSE	0.43422	0.434215	R-Square	0.9136	0.913595
2 MODEL1	Test2	Dependent Mean	5.36763	5.367629	Adj R-Sq	0.9124	0.912378
3 MODEL1	Test2	Coeff Var	8.08951	8.089512			0

PARMS1 **FITSTATS1**

SAS temporary data sets created using ODS OUTPUT.