Paper TS03

Graph Automation Using Integrated SAS/S-PLUS Solution

Qian Wang, MSD (Europe), Inc., Brussels, Belgium
Carl Herremans, MSD (Europe), Inc., Brussels, Belgium

ABSTRACT

Graphical display of clinical data and statistical analysis results is often required for clinical study reports, journal publications, and presentations to the regulatory agencies.

Graphs can be created using "point-and-click" software packages, without programming efforts from the end-users. However, each time the graphic requirement changes, the plot has to be re-generated or revised manually. Besides, such process is too labor-intensive and time-consuming to handle large scale graphical needs for some studies.

Alternatively, the plot generation can be automated through programming. But the traditional approaches demand substantial efforts and require considerable technical knowledge from the end-users.

Ideally, a graphical tool should combine the ease-of-use from the "point-and-click" graphical software packages with the automated scalability from the programming approaches. This paper discusses one possible implementation of such a tool, the SPGRAPH SAS® macro. It passes the plot data and annotation requirements from SAS to S-PLUS®, generates the graphic using S-PLUS and returns the graphic back to SAS for inclusion into an RTF document.

INTRODUCTION

Graphics are indispensable to the presentation and visualization of clinical data and statistical analysis results. They are used to support the clinical study reports, journal publications, and meetings with the regulatory agencies.

The end-users of the graphics generally refer to those individuals within clinical, regulatory and statistical groups. They are usually familiar with common graphical software packages, but might not necessarily have programming experience. The possibility of using "point and click" WYSIWYG interface for layout modification is therefore greatly appreciated.

In a clinical trial, similar graphics are sometimes needed at multiple stages in the statistical analysis, and possibly for different endpoints and time points. To be able to reuse prior efforts by automating the graphic creation process is especially important.

This paper describes the SPGRAPH macro, a SAS/S-PLUS solution on a MS-Windows® platform, which combines the advantages of both "point and click" software and programming automation.

AUTOMATED SAS GRAPHING APPROACHES

SAS/GRAPH is widely used for graph automation. The main drawback of this approach is the considerable programming efforts needed to create and maintain the SAS code. In addition, graphics from SAS/GRAPH are not editable, and the layout is not always preferred by those end-users who are used to the look from specialized graphical packages.

Alternatively, SAS programs can be implemented in a way which automatically interacts with any chosen graphical packages [1]. The end-users only need to provide a graphical layout template created from the "point-and-click" interface of a graphical software package, such as S-PLUS.

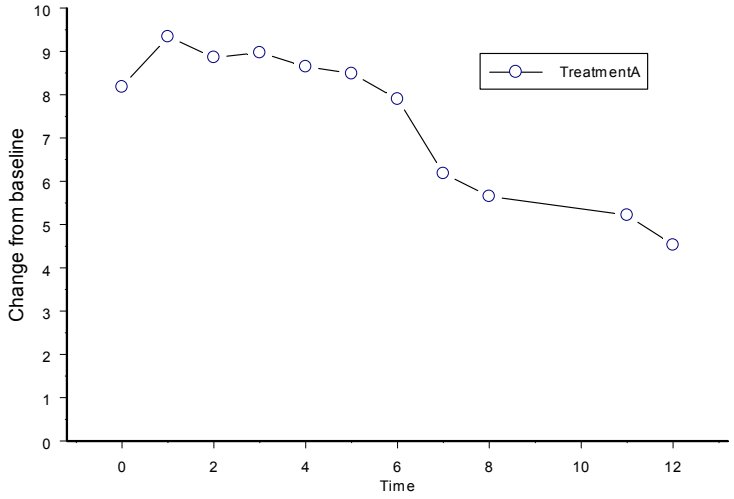
The SPGRAPH macro presented in this paper supports the generation of flexible, maintainable and automated graphics, by combining the desired features from both SAS and S-PLUS. All data manipulation and interfacing to S-PLUS are handled within SAS, while the creation of graphical layout itself is managed in S-PLUS and triggered by the SAS code automatically.

S-PLUS GUI SCRIPT

In S-PLUS, each graph is composed of individual Graphical User Interface (GUI) objects. The GUI objects can be created and modified by using the regular "point-and-click" functionality provided by the software. When dragging the GUI objects from the graph sheet into a script window, the corresponding S-PLUS GUI scripts are automatically generated. If graphical layout changes are required, the scripts may be modified and executed to reproduce the plot.

Figure 1 shows a line plot with isolated points representing the change from baseline values over time for a treatment group. The plot is created using the S-PLUS point-and-click GUI interface.

Figure1: Example line plot



The table below shows part of the corresponding S-PLUS GUI script, which can be transformed into a SAS dataset for manipulation and then restored into graphic. This feature allows automated update of S-PLUS GUI scripts via SAS programming.

Sample S-PLUS GUI script

```
guiCreate( "Graph2D", Name = "Test$1",
    HorizontalShift = "0.2",  …)
guiCreate( "LinePlot", Name = "Test$1$1",
    PlotNumber = "1",
    DataSet = "test",
    xColumn = "TIME",
    yColumn = "TreatmentA",   …)
guiCreate( "Axis2DLabelY", Name = "Test$1$Axis2dY1$Axis2DLabelY",
    LabelType = "Auto",        …)

guiCreate( "YAxisTitle", Name = "Test$1$Axis2dY1$YAxisTitle",
    UseDate = F,
    UseTime = F,
    Title = "Change from baseline",
    xPosition = "Auto",
    yPosition = "Auto",
    XJustification = "0",
    YJustification = "0",
    Angle = "90",
    Hide = F,
    UseAxesUnits = F,
    RelativeAxisX = "1",
    RelativeAxisY = "1",
    PanelToDraw = "All",
    Font = "Arial",
    FontSize = "16",
    FontColor = "Black",
    Bold = F,
    Italics = F,
    Underline = F,
    FillColor = "Transparent",
    FillPattern = "Empty",
```

Sample S-PLUS GUI script

```
    PatternColor = "Yellow",
    BorderStyle = "None",
    BorderColor = "Black",
    BorderWeight = "1",
    VerticalMargin = "0.162713",
    HorizontalMargin = "0.138716")

guiCreate( "Axis2dY", Name = "Test$1$Axis2dY1",
    LineStyle = "Solid",   …)
guiCreate( "XAxisTitle", Name = "Test$1$Axis2dX1$XAxisTitle",
    UseDate = F,
    UseTime = F,
    Title = "@Auto",          …)
guiCreate( "Axis2DLabelX", Name = "Test$1$Axis2dX1$Axis2DLabelX",
    LabelType = "Auto",       …)
guiCreate( "Axis2dX", Name = "Test$1$Axis2dX1",
    LineStyle = "Solid",   …)
guiCreate( "Axis2dY", Name = "Test$1$Axis2dY1",
    LineStyle = "Solid",   …)
guiCreate( "XAxisTitle", Name = "Test$1$Axis2dX1$XAxisTitle",
    UseDate = F,
    UseTime = F,
    Title = "@Auto",          …)
guiCreate( "Axis2DLabelX", Name = "Test$1$Axis2dX1$Axis2DLabelX",
    LabelType = "Auto",       …)
guiCreate( "Axis2dX", Name = "Test$1$Axis2dX1",
    LineStyle = "Solid",   …)
```

THE MACRO DESIGN

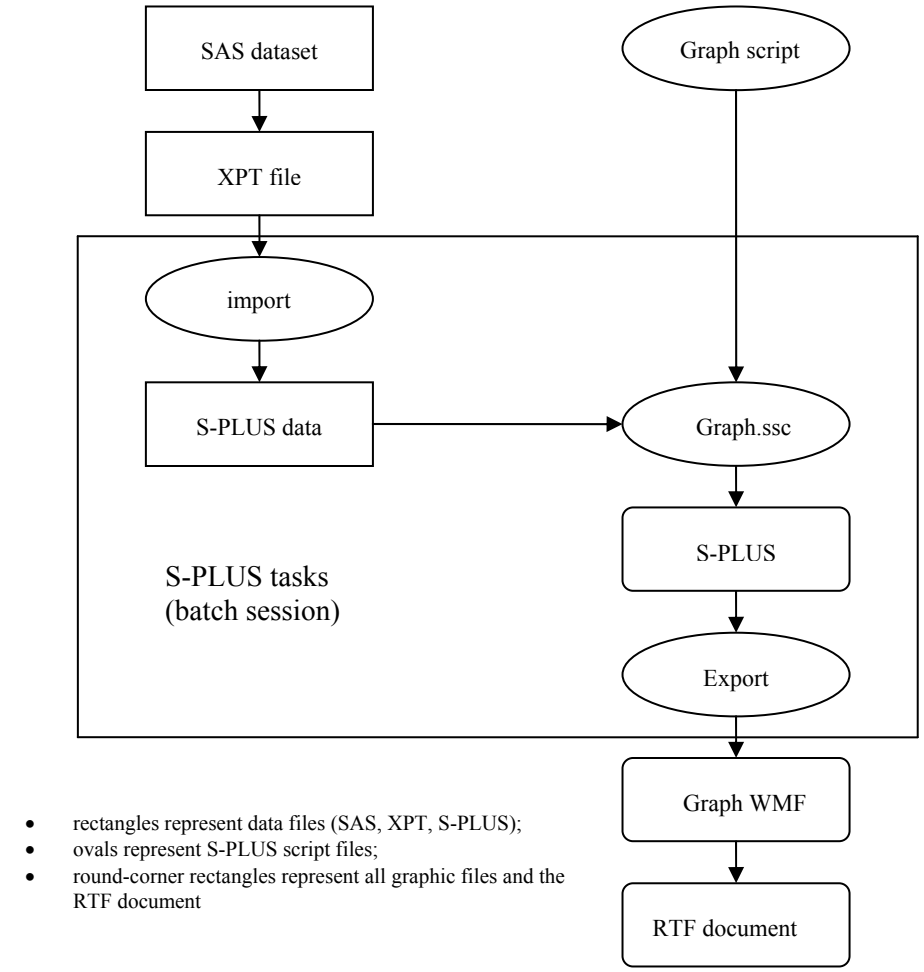
The SPGRAPH SAS macro requires as inputs, a SAS dataset with plot data and an S-PLUS GUI script as graphical layout template. Similar plots on different data can be generated easily by passing in different input SAS datasets. Any layout updates to the plot can be achieved by updating the GUI script through interactive GUI object modifications in S-PLUS.

Figure 2 shows the process flow of the macro design. The macro firstly transforms the input SAS dataset to XPT format, generates all necessary supporting S-PLUS command scripts (import and export), launches S-PLUS batch session and finally inserts the output graphic into an RTF document together with the graph titles and footnotes.

In detail, the S-PLUS batch session (shown in the box in the flow chart) performs the following tasks:

- imports the SAS-transport file (XPT);
- executes the S-PLUS script and creates the graph;
- exports the S-PLUS graphic to a Windows Metafile.

Figure2: SPGRAPH flow chart



IMPLEMENTATION DETAILS

GENERATE S-PLUS COMMAND SCRIPT FROM SAS

The SAS "PUT" statement can be used to write the S-PLUS commands into a flat file. The following example creates an S-PLUS command script file (spgraph_import.ssc) to import SAS XPT file (data.xpt) to S-PLUS.

```
Data _null_;
   File "C:\spgraph_import.ssc";
   put "import.data(DataFrame=""TEST"", " @;
   put "FileName=""C:\\data.xpt"", FileType=""SAS_TPT"") ";
run;
```

LAUNCH S-PLUS BATCH SESSION FROM SAS

To facilitate communication between SAS and S-PLUS, several operating system commands need to be defined and executed in order. A .bat file can be created using the "PUT" statement in SAS.

```
data _null_;
   file "C:\spgraph.bat";
   put "cd C:\Program Files\Insightful\S-PLUS61localclient\cmd ";
   %* change to drive where S-PLUS is installed*;
   %* the DOS prompt starts in the SAS default directory *;
   put "C:";
```

```
    put "S-PLUS /BATCH " "C:\spgraph_import.ssc " @;
    put " /BATCH " "graphscript" @;
    put " /BATCH " "C:\spgraph_export.ssc" ;
run;
```

The global statement SYSTASK COMMAND is then used to execute the operating system commands defined in the batch file, as shown in the example below. The function WAIT suspends the execution of the current SAS session until the task has completed.

```
systask command C:\spgraph.bat " wait;
```

ADD ANNOTATIONS TO THE GRAPH

Graphics sometimes need to include data-dependent texts or layout definition (e.g., legend with the count of the number of patients within a population). The SPGRAPH macro provides annotation capabilities for such customization. The process is listed below:

- An annotation dataset is provided as additional input to the SPGRAPH macro. It contains S-PLUS object identifiers (in the form of OBJECT/PATH/PROPERTY triplets) and their corresponding expected values. The dataset will be used to update the object definition in the original S-PLUS GUI script.
- The macro reads the original S-PLUS GUI Script into a SAS dataset, extracts the objects listed in the annotation and replaces their values with the expected ones.

The table below shows an annotation dataset as macro input. It contains three S-PLUS objects and their respective expected values.

Object	Path	Property	Value
CommentDate	*Spdat$1$1*	*Title*	*(N= 175)*
CommentDate	Spdat$1$2	Title	(N= 176)
CommentDate	Spdat$1$3	Title	(N= 187)

The next table shows the SAS dataset with the line-by-line parse results of the original S-PLUS GUI script. The object in Italic (defined by the OBJECT/PATH/PROPERTY triplet) is expected to be updated with the value from the annotation dataset.

Original	Object	Path	Property
GuiCreate("CommentDate", Name = "spdat$1$1",	CommentDate	Spdat$1$1	"CommentDate ", Name
UseDate = F,	CommentDate	Spdat$1$1	UseDate
UseTime = F,	CommentDate	Spdat$1$1	UseTime
Title="(N= XXX)",	*CommentDate*	*Spdat$1$1*	*Title*
xPosition = "2.48571",	CommentDate	Spdat$1$1	Xposition
yPosition = "0.426374",	CommentDate	Spdat$1$1	Yposition
…	…	…	…

By merging the above two datasets, the original title text (N= XXX) is replaced by the annotation (N= 175). The new GUI script with the updated title is shown below.

Updated GUI script
GuiCreate("CommentDate", Name = "spdat$1$1",
UseDate = F,
UseTime = F,
Title="(N= 175)",
xPosition = "2.48571",
yPosition = "0.426374",
…

INSERT GRAPHIC INTO AN RTF DOCUMENT

The generated S-PLUS graphic can be inserted automatically into an RTF document. The SAS ODS statement provides a solution by linking the graph to the document. However, it assumes the same drive mappings, directory structure and access from all users, which is not always the case.

Another alternative will be to embed the graphic into an RTF file. It is implemented in the SPGRAPH macro using a DDE connection with MS-WORD® [2].

The following code opens an MS-WORD document using the DDE system channel. The 'S' option used in fopen indicates sequential access method which allows SAS to determine when MS-WORD is open. The DO WHILE loop is implemented to give WINDOWS some additional time to launch the application. The loop is broken when the RTF document is open which triggers fopen to return a positive fsysid.

```
%** Open the DDE System Channel to word document *;
filename __ddd__ dde 'winword|system' lrecl=5000;
data _null_;
   call system("C:\rtfdocument.rtf ");
   fsysid=fopen('__ddd__','s');
   %** Sleep until word time to come up **;
   do while (fsysid le 0);
       x=sleep(1);
       fsysid=fopen('__ddd__','s');
   end;
   rc=fclose(fsysid);
run;
```

Once the RTF document is opened, the graphic can be inserted by executing the basic Word commands through the DDE channel.

```
%* Insert the graph in the opened RTF document containing the title *;
data _null_;
   file __ddd__;
   %* Move the insertion point to the end of the document *;
   put "[EndOfDocument]";
   %*Insert an INCLUDEPICTURE at the insertion point and save
     graphic data in the doc  **;
   put "[InsertPicture .Name=""C:\spgraphimage.WMF"", .LinkToFile=1]";
   %*Save a copy of the linked object and remove the specified link *;
   put "[EditLinks .SavePictureInDoc=1, .KillLink, .Link=""1"",
         .Application=""spgraphimage.WMF""]";
   %*Save and Close File ;
   put "[FileClose 1]";
run;
```

CONCLUSION

This paper describes the SPGRAPH SAS macro which combines SAS and S-PLUS in the graphic creation process. It illustrates how the plot data and annotation are passed from SAS to S-PLUS, how the graphic is generated using S-PLUS scripts then sent back to SAS to be embedded in an RTF document. By doing so, the advantages from both "point and click" software (S-PLUS) and programming automation (SAS) are integrated into one single solution.

REFERENCES

[1] Philip, R. H. "SAS to R to SAS". the Pharmaceutical Users Software Exchange Conference, Heidelberg, Germany, 2005.

[2] William, W. V. and Koen, V. "Fancy MS Word Reports Made Easy: Harnessing the Power of Dynamic Data Exchange — Against All ODS, Part II — ". Proceedings of the Twenty-Eighth Annual SAS Users Group International Conference, USA, 2003.

ACKNOWLEDGMENTS

The authors would like to express appreciation to Frederic Coppin for his input in the preparation of this paper.

CONTACT INFORMATION

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

Qian Wang
Senior Statistical Programmer
Merck Sharp & Dohme (Europe), Inc.
Clos du Lynx 5
B-1200 Brussels [Belgium]
Tel: ++32 27766303
E-mail: qian_wang@merck.com
Web: http://www.merck.com

Carl Herremans
Statistical Programming Analyst
Merck Sharp & Dohme (Europe), Inc.
Clos du Lynx 5
B-1200 Brussels [Belgium]
Tel: ++32 27766305
E-mail: carl_herremans@merck.com
Web: http://www.merck.com

Brand and product names are trademarks of their respective companies.