Paper 221-28

Using SAS® Software and Visual Basic for Applications to Produce Microsoft Graph Charts

Lori S. Parsons, Ovation Research Group, Seattle, WA

ABSTRACT

While SAS/GRAPH software produces high quality graphs for presentations, it is often desirable to provide Microsoft Graph charts. An analyst often produces statistics and graphs for an investigator. The investigator writes up the results and presents a talk accompanied by Microsoft PowerPoint slides or a manuscript prepared in Microsoft Word. A SAS® graph can easily be inserted into PowerPoint or Word. However, if the investigator wants changes in the appearance of the graph, the graph has to go back to the analyst to re-write SAS code. By providing a Microsoft Graph chart, the investigator can modify the appearance.

It is more efficient, and less prone to error, to automatically populate the data cells of a chart. By using SAS Output Delivery System (ODS) and Visual Basic for Applications (VBA), data can be generated in a SAS procedure and data cells of a Microsoft Graph chart automatically populated.

This paper describes a technique to produce Microsoft Graph charts directly from SAS. This method can be used for any of the chart types available in Microsoft Graph and use data from any SAS procedure. SAS 7.0 or higher, Microsoft Office 97 or higher, and basic knowledge of ODS and VBA are required.

BACKGROUND

Microsoft Graph

Microsoft Graph is a supplementary application included with Microsoft Word that can be used to create, import and edit charts and graphs. These charts and graphs are embedded objects; they can be inserted into any application that supports Object Linking and Embedding. Microsoft Graph allows control over all aspects of the chart appearance such as color, size, legend, labels, headings, data markers, etc.

The following can be created with Microsoft Graph:

- Area charts
- Bar charts
- Column charts
- Line charts
- Pie charts
- X-Y charts
- Surface charts
- Bubble charts
- Stock charts
- Cylinder charts
- Cone charts

Automating Tasks in Microsoft Word

Beginning with Microsoft Office 97, the programming environment was standardized such that all Office applications used Visual Basic for Applications (VBA). Stetz (SUGI 25 Proceedings, 2000) described how to take full advantage of VBA to automate tasks in Microsoft Word, directly from SAS®.

SAS Output Delivery System

Beginning with SAS 7.0, the Output Delivery System (ODS) provides a method to select desired data from all of the SAS procedures. Data files can easily be made for the purpose of importing into a Microsoft Office product.

Using SAS and VBA to Produce Charts

It is much more efficient, and far less prone to error, to automatically populate the data cells of a chart, as opposed to doing data entry or cutting and pasting data. By using SAS ODS and VBA, data can be generated in a SAS procedure and data cells of a Microsoft Graph chart can be automatically populated.

This paper describes a technique to produce Microsoft Graph charts directly from SAS. As examples, three types of charts will be created from two SAS procedures. A column chart will be created with FREQ procedure data. A X-Y chart (Receiver Operating Characteristic Curve) and a stock chart (Odds Ratios and 95% Confidence Intervals) will be created with PROC Logistic data. This method can be used to create any of the chart types available in Microsoft Graph and use data from any SAS procedure.

Initial Set-up

1. Make sure Microsoft Graph is installed

A Typical installation of Word includes Graph; a Minimum installation does not include Graph, and a Custom installation allows the user to choose whether to install Graph. If Graph is not installed, run the Setup program from the Office CD and then choose the Add/Remove option. When the list of Office components appears, choose Office Tools and then click on the Change Options button. In the box that appears, choose Programs → Microsoft Word → Setup from the Start menu in Windows. If Microsoft Office is running, the options will be Programs → Microsoft Office → Setup form the Start menu. When setup appears, choose the Custom option and follow the instructions for installing Graph.

2. Stetz Method

Stetz provided two macros (***RUN_VBA.SAS*** and ***RUN_VBA.BAS***) to automate Word tasks. To set-up

Word, the *RUN_VBA.BAS* macro should be created in Word as described by Stetz. The following, as presented in his paper, describes his method:

- The DATA step generates a Word VBA macro written to a text file
- The RUN_VBA SAS macro generates and executes a windows batch file that:
  - Sets Windows environment variables based on the macro parameter values specified
  - Invokes Work using the /mRun_VBA startup switch
- The Run-VBA Word Macro:
  - Reads the values of the environment variables set in the batch file
  - Minimized word (minimize=Y)
  - Creates a new module in the Normal template
  - Reads in the text file containing the VBA macro generated in the SAS DATA step
  - Executes the VBA macro
  - Deletes the new module from the Normal template
  - Deletes the text file containing the VBA macro
  - Notifies the user vial a dialog box that the macro has completed (notify=Y)
  - Closes Word after the user acknowledges the dialog box (exitword=Y)

METHODS

STEPS TO PRODUCING THE GRAPH IN SAS

There are four steps to produce a Microsoft Graph chart from SAS. These steps are the same for all chart types and for data from all SAS procedures.

STEP 1. Create a Microsoft Graph chart template

First, create and save a Word document containing the desired chart type. Leave the data sheet empty; the data cells will be populated by SAS. For this paper, this file will be called the chart template. The general steps for creating a Microsoft Graph chart are:

a) Open a Word document and place the insertion point where you want the new graph to appear.
b) Choose Insert → Object, and be sure the Create New tab is foremost; double-click on Microsoft Graph 2000 Chart (or Microsoft Graph 97 Chart, if using Word 97) in the Object Type list.
c) Graph will open with its own toolbar, displaying a datasheet window that looks like a small spreadsheet, together with a chart window.
d) Select the right chart type. To change the type of the chart in "chart edit mode" use the Chart → Chart Type command to select the desired chart type.
e) Leaving the first row blank, define the rows. The variable value labels will be used for the column headings and will go into the first row. See Figure 1 in Example 1.
f) Use Microsoft Graph's commands to set up the desired appearance of the chart. Resize the graph by dragging the size box in the lower right corner of its window.
g) Click anywhere outside to graph to return to the Word document.
h) Save as a Word document.

STEP 2. Create data file to import into the chart

In this step, SAS procedure code is used to create the output data file that will be imported into the chart. This step will vary depending on the procedures that are run and the type of chart chosen. In general, the file will contain fixed-width columns with the first line left blank.

STEP 3. Create VBA code to produce chart

In this step, a DATA _null_ step is used to create an output file containing VBA code. SAS macros have been developed, and are presented in this paper, that use PUT statements to insert VBA code into the output file. The macro **VBA_LIB**, found in **Appendix 1**, contains these individual macros. The following describes what the individual macros do:

%MACRO TblPage Sets up a new document in Landscape orientation.

%MACRO TblPPage Sets up a new document in Portrait orientation.

%MACRO InsFile Inserts a Word Document File.

%MACRO InsPict Inserts a Picture.

%MACRO InsWDObj Inserts a Word Document Object.

%MACRO RunGraph
- Inserts the chart template file created in Step 1
- Imports the data created in Step 2
- Saves the document as a Word Documents with the name specified in the call statement

%MACRO saveas Saves the document as a Word Document with the name specified in the call statement.

%MACRO fileclos Closes Word.

STEP 4. Run the VBA code

The VBA code file created in Step 3 is run in Step 4. The macro **RUN_VBA.SAS** as written and described by Stetz is used here. This macro can be found in **Appendix 2** of this paper.

RESULTS

For Examples 1, 2 and 3, define the study locations.

```
/* Define Format Library */
LIBNAME LIBRARY
'D:\Admin\SUGI\SUGI28\SASWork\DataSetX\Formats';

/* Define Study Library */
LIBNAME STUDY
    'D:\Admin\SUGI\SUGI28\SASWork\DataSetX';
```

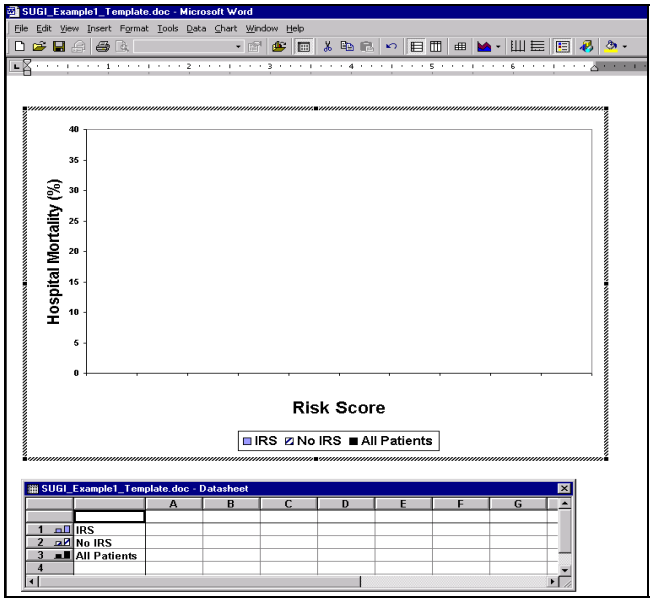
```
/* Define subdirectories for VBA Code file */
%let path1 =
     D:\Admin\SUGI\SUGI28\SASWork\SasPrograms;

/* Define subdirectory for Figures */
%let path2 =
     D:\Admin\SUGI\SUGI28\SASWork\Figures;

/* Include VBA Macro Library code */
%include "&path1.\VBA_Macro_Library.txt";
%VBA_LIB;
```

EXAMPLE 1:
Column Chart with PROC Freq data

STEP 1. Create a Microsoft Graph chart template

Figure 1 shows a Microsoft Graph that was created for a column chart. This will be the template for producing the final chart containing SAS procedure data. This column chart will display the percent mortality by risk score. Each risk score will be broken down into three groups (IRS Patients, NO IRS Patients, and All Patients combined). This file has been saved as **SUGI_Example1_Template.doc**. The first row has been left blank; these cells will be filled with the variable value labels. The remaining rows will be filled with the group data.

Figure 1: Template file of Column Chart



STEP 2. Create data file to import into the chart

The following code is used to create the data file that will be imported into the chart. In this example, PROC Freq is used to generate the data; a 3-level table is requested. The data file that is created by PROC Freq is named **graph**.

```
PROC FREQ data=study.TIMITBLS;
  table POPLGrp * deceased * TIMICD
  /out=graph outpct;
run;
```

Keep the data fields and the records that will be inserted into the chart. In this case, the data fields are the grouping variable (poplgrp), the row variable (timicd), and the column percent (pct_col). The records are for the desired outcome of mortality (deceased=1).

```
DATA word (keep = poplgrp timicd pct);
   set graph;
   where deceased = 1;
   pct = round(pct_col,.1);
run;
```

Using a DATA _null_ step, create the data file that will be imported into the graph. Name the data file **WordFigure.dat**. The contents of **WordFigure.dat** are in **Appendix 3**. The row variable (timicd) is coded 0 – 9. This row variable is used to define the column of the data file [(timicd+1 * 10].

```
DATA _null_  ;
  file "&path2.\WordFigure.dat" print ls = 200
  ps=66 n=pagesize notitles ;
  set word;
  put #1 @((timicd+1) * 10) timicd;
  if POPLGrp=1 then
        Put #2 @((timicd+1) *10 ) pct;
        else if poplgrp = 2 then
           put #3 @((timicd+1) * 10) pct;
         else if poplgrp = 3 then
               put #4 @((timicd+1) * 10) pct;
RUN;
```

STEP 3. Create VBA code to produce chart

The following code is used to create the file containing the VBA code that will be run to produce the chart. The file that is created is named **SUGI1.bas** and can be seen in **Appendix 3**.

Define the macro variables ***program***, ***dfile*** and ***template*** for the current chart. The macro variable ***path2*** was defined in the initial setup of study locations.

```
%let program=SUGI1;
%let dfile = WordFigure.dat;
%let template=SUGI_Example1_Template.doc;
```

Run the DATA _null_ step to create the SUGI1.bas file. Use the macro **TblPPage** to set up the new Word document in Portrait orientation. Use the macro **RunGraph** to Insert the chart template file, import the data created in Step 2 and save the new chart file.

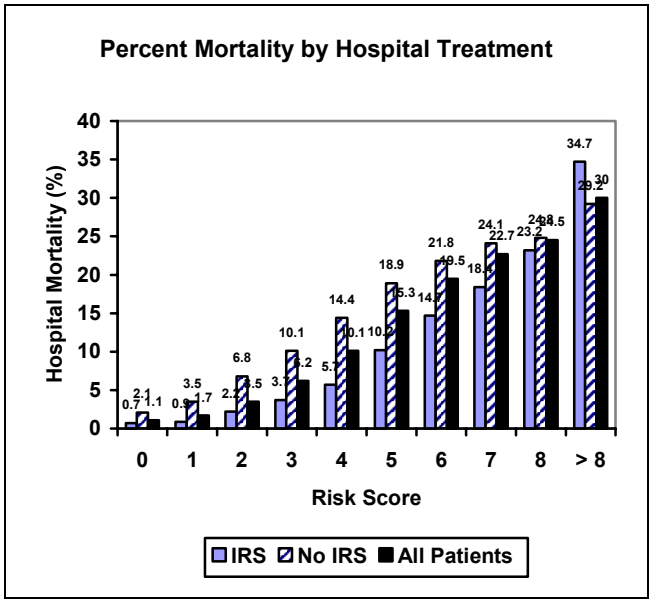
```
DATA _null_ ;
   file "&path2\&program..bas" ;

/* Start the procedure */
   put "Sub &program()" ;

/* Use the macro TblPPage for Portrait */
   %TblPPage;

/* Call the Macro RunGraph */
  %RunGraph(&path2\&dfile,"&template.");
/* End the prodedure */
   put 'End Sub' ;
run ;
```

STEP 4. Run the VBA code

The following code runs the file SUGI1.bas (&program.) created in Step 3. The result will be the Word document SUGI1.doc which contains the column chart, populated with the SAS data, saved in the defined location (&path2.). **Figure 2** shows the results of Example 1, a column chart populated with PROC Freq data.

```
%let wordpath=d:\progra~1\micros~1\office ;
%RUN_VBA(wordpath=&wordpath,
         vba_path=&path2, vba_pgm=&program,
          visible=Y, minimize=N, notify=n,
         exitword=y) ;
```

Figure 2: Column Chart with PROC Freq data

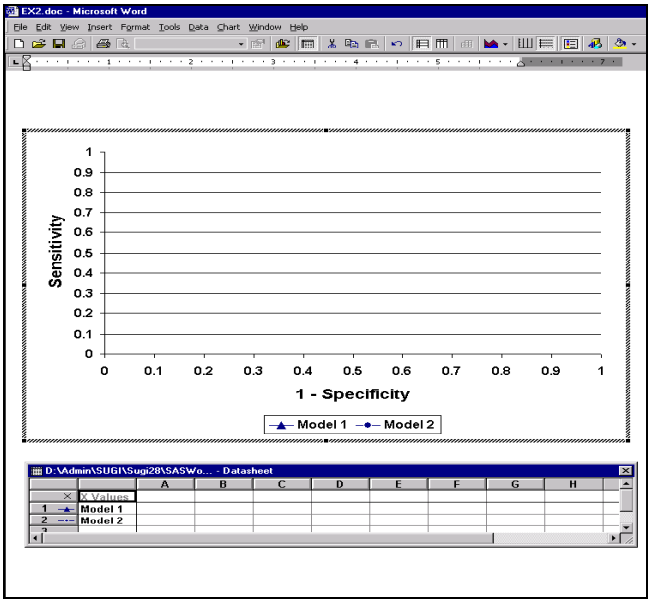


EXAMPLE 2:
X-Y Chart with PROC Logistic data

STEP 1. Create a Microsoft Graph chart template

Figure 3 shows a Microsoft Graph that was created for a X-Y chart. This file has been saved as **SUGI_Example2_Template.doc**. This X-Y chart will display two Receiver Operating Characteristic (ROC) Curves with data generated from PROC Logistic. The first row has been left blank; the cells will be filled with the variable value labels. The second row will be filled with results from the first PROC Logistic analysis (Model 1). The third row will be filled with results from the second PROC Logistic analysis (Model 2).

Figure 3: Template file of a X-Y Chart



STEP 2. Create data file to import into the chart

The following code is used to create the data file that will be imported into the chart. This example is more complex than Example 1. In this example, the results of the *final* step of *two* independent models (two runs of PROC Logistic) will be included in the same graph.

PROC Logistic, with a SELECTION = Stepwise option, is used to generate a stepwise logistic regression analysis. The OUTROC option is used to create an output SAS data set that contains the data necessary to produce the ROC curve. This output data set will contain data for each step of the stepwise model. Since only data from the *final* step is to be included in the graph, a DATA _null_ and *symput* call is used to determine the final step. Another DATA step is used to select the final step data.

Since PROC Logistic and two DATA steps will each be run twice, once for Model 1 and once for Model 2, the macro MLR was written and called twice. The remaining code can easily be modified to include more than two models per graph, if desired. The final data file that is created is named **WordFigures.dat** and its contents are in **Appendix 4**.

```
%MACRO MLR
(Dataset,endpoint,independ,where,outroc);

/* ****************************************  */
/* Macro call variables:                     */
/* Dataset = the name of the SAS data set    */
/* endpoint = the dependent variable         */
/* independ = the independent variables      */
/* where = the criteria for the WHERE option */
/* outroc = the dame of the output data set  */
/* ****************************************  */

/* Run PROC Logistic with the OUTROC option */
PROC Logistic DATA=&Dataset. Descend NameLen=40;
MODEL &endpoint. = &independ.
```

```
  /SELECTION = STEPWISE RISKLIMITS LACKFIT
RSQUARE PARMLABEL outroc=&outroc.;
WHERE &where.;
RUN;

/* Determine the FINAL step of the stepwise model */
DATA _null_;
   set &outroc.;
   call symput('finalstep',_step_);
run;

/* Save data of the FINAL Step */
DATA F&outroc.;
   set &outroc.;
   where _step_ = &finalstep.;
run;
%MEND MLR;
```

Task 1: Call the Macro MLR for Model 1. Use the macro variable **MLR1** to define the list of independent variables. Name the output data set Model1 (this is done with the macro call variable **outroc**).

Define the independent variables

```
%LET MLR1 = GScore2 GScore3 GScore4 GScore5
GScore6 GScore7 GScore8 GScore9 GScore10
GScore11 GScore12 GScore13 GScore14;
```

Call the Macro MLR for Model 1

```
%MLR(study.anal,ceshock,&MLR1,(denom=1),Model1);
```

Task 2: Call the Macro MLR for Model 2. Use the macro variable **MLR2** to define the list of independent variables. Name the output data set **Model2** (this is done with the macro call variable **outroc**).

Define the independent variables

```
%LET MLR2 =  nrmiSx5_2 nrmiSx5_3 nrmiSx5_4
nrmiSx5_5 nrmiSx5_6 nrmiSx5_7 nrmiSx5_8
nrmiSx5_9 nrmiSx5_10;
```

Call the Macro MLR for Model 2

```
%MLR(study.anal,ceshock,&MLR2,(denom=1),Model2);
```

Task 3: Keep the data fields that will be inserted into the chart; in this case, the Y-value and the X-value. Create a variable named MODEL to define which model the data came from (Model 1 or Model 2).

```
DATA TModel1 (keep= xvalue yvalue model);
   set FModel1;
   Yvalue = round(_sensit_,.01);
   Xvalue = round(_1mspec_,.01);
   MODEL = 1;
run;
DATA TModel2 (keep= xvalue yvalue model);
   set FModel2;
   Yvalue = round(_sensit_,.01);
   Xvalue = round(_1mspec_,.01);
   MODEL = 2;
run;
```

Task 4: Using a DATA _null_ step, create the data file that will be imported into the graph. Name the data file WordFigure.dat.

```
DATA _null_   ;
   file "&path2.\ROC.dat" print ls = 200  ps=66
        n=pagesize notitles ;
   set TModel1 TModel2;
   colno + 5;

   /* Model 1 Data*/
   if MODEL = 1 then do;
      put #1 @colno xvalue;
      Put #2 @colno yvalue;
   end;

   /* Model 2 Data*/
   if MODEL = 2 then do;
      put #1 @colno xvalue;
      Put #3 @colno yvalue;
   end;
RUN;
```

STEP 3. Create VBA code to produce chart

The following code is used to create the file containing VBA code that will be run to produce the chart. The file that is created is named **ROCFigure.bas**. The data file that is used is called **ROC.dat** and the template is called **SUGI_Example2_Template.doc**.

Define the macro variables ***program***, ***dfile*** and ***template*** for the current chart. The macro variable ***path2*** was defined in the initial setup of study locations.

```
%let program=ROCFigure;
%let dfile = ROC.dat;
%let template=SUGI_Example2_Template.doc;
```

Run the DATA _null_ step to create the ROCFigure.bas file. Use the macro **TblPPage** to set up the new Word document in Portrait orientation. Use the macro **RunGraph** to Insert the chart template file, import the data created in Step 2 and save the new chart file.

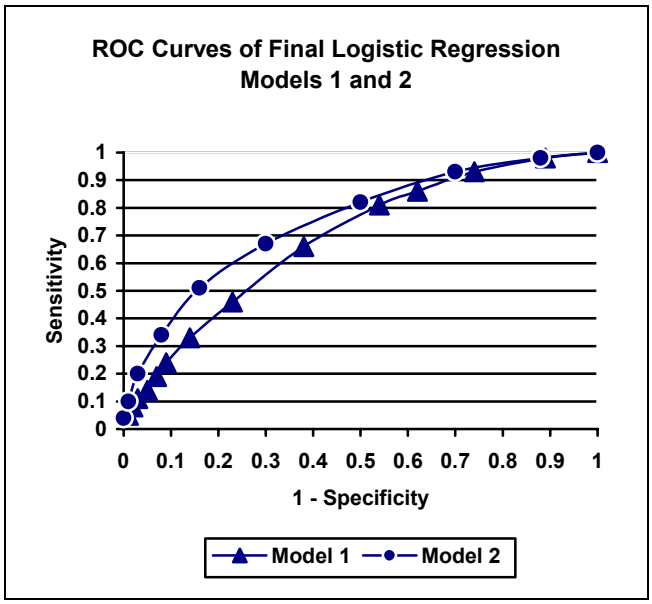
```
DATA _null_ ;
   file "&path2\&program..bas" ;

/* Start the procedure */
   put "Sub &program()" ;

/* Use the macro TblPPage for Portrait */
   %TblPPage;

/* Call the Macro RunGraph */
  %RunGraph(&path2\&dfile,"&template.");

/* End the prodedure */
   put 'End Sub' ;
run ;
```

STEP 4. Run the VBA code

The following code will run the file ROCFigure.bas (&program.) created in Step 3. The result will be the Word document ROCFigure.doc which contains the X-Y chart, populated with SAS data, and saved in the defined location (&path2.). **Figure 4** shows the results of Example 2, a X-Y chart populated with PROC Logistic data.

```
%let wordpath=d:\progra~1\micros~1\office ;
%RUN_VBA(wordpath=&wordpath,
         vba_path=&path2, vba_pgm=&program,
          visible=Y, minimize=N, notify=n,
         exitword=y) ;
```

Figure 4: X-Y Chart with PROC Logistic data

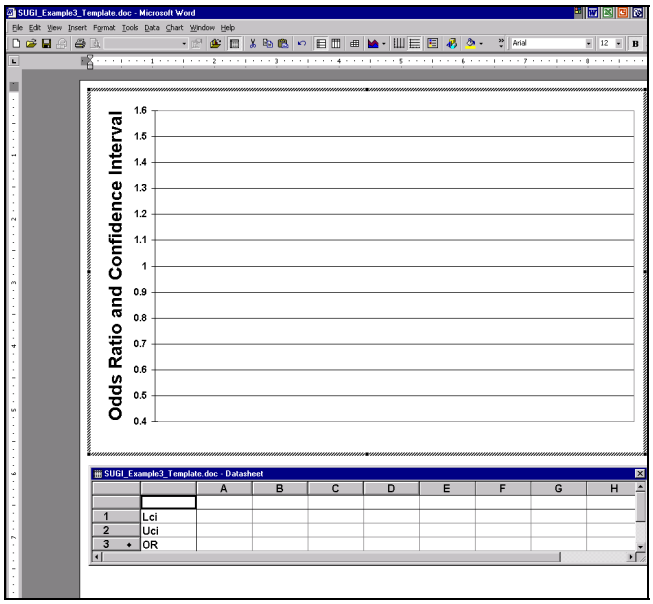


EXAMPLE 3:
Stock Chart with PROC Logistic data

STEP 1. Create a Microsoft Graph chart template

Figure 5 shows a Microsoft Graph that was created for a stock chart. This file has been saved as **SUGI_Example3_Template.doc**. This stock chart will display the odds ratio and confidence interval generated from PROC Logistic. The first row has been left blank; the cells will be filled with the variable names. The second row will be filled with the lower confidence interval, the third row will be filled with the upper confidence interval, and the fourth row will be filled with the odds ratio.

Figure 5: Template of Stock Chart



STEP 2. Create data file to import into the chart

The following code is used to create the data file that will be imported into the chart. In this example, PROC Logistic is used to generate the data. **ODS select** and **output** code is used to create the data file with the odds ratio and confidence interval data.

```
/* Select LR Information to output            */
ODS select OddsRatios;
ODS output OddsRatios=fOddsRatios;

/* Run logistic regression */
PROC LOGISTIC DATA=study.anal Descend
NameLen=40;
MODEL deceased = ptage male white mhcabg mhdiab
mhcad mhptca mhbleed
  /SELECTION = STEPWISE RISKLIMITS LACKFIT
RSQUARE PARMLABEL;
RUN;

/* Turn ODS output "Off" */
ODS output close;
```

Using a DATA _null_ step, create the data file that will be imported into the graph. Name the file **WordFigure.dat**. The contents of **WordFigure.dat** are in Appendix 5. The incremented counter variable **colno** is used to define the column number. **Effect** is the variable name.

```
DATA _null_   ;
   file "&path2.\WordFigure.dat" print ls = 200
        ps=66 n=pagesize notitles ;
   set fOddsRatios;
   colno + 10;
   Put #1 @colno Effect;
   Put #2 @colno LowerCL;
   Put #3 @colno UpperCL;
   Put #4 @colno OddsRatioEst;
RUN;
```

STEP 3. Create VBA code to produce chart

The following code is used to create the file containing VBA code that will be run to produce the chart. The file that is created is named **OddsRatios.bas**. The data file that is used is called **WordFigure.dat** and the template is called **SUGI_Example3_Template.doc**.

Define the macro variables ***program***, ***dfile*** and ***template*** for the current chart. The macro variable ***path2*** was defined in the initial setup of study locations.

```
%let program=OddsRatios;
%let dfile = WordFigure.dat;
%let template=SUGI_Example3_Template.doc;
```

Run the DATA _null_ step to create the OddsRatios.bas file. Use the macro **TblPPage** to set up the new Word document in Portrait orientation. Use the macro **RunGraph** to Insert the chart template file, import the data created in Step 2 and save the new chart file.

```
DATA _null_ ;
   file "&path2\&program..bas" ;

/* Start the procedure */
   put "Sub &program()" ;
```

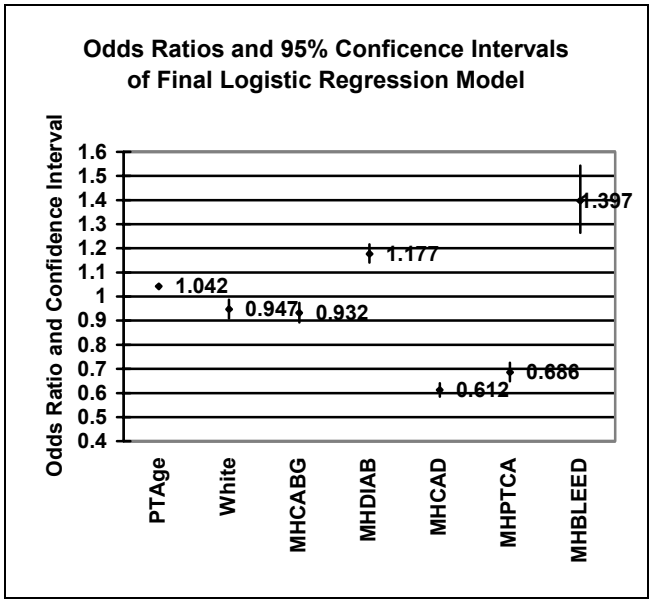
```
/* Use the macro TblPPage for Portrait */
   %TblPPage;

/* Call the Macro RunGraph */
  %RunGraph(&path2\&dfile,"&template.");

/* End the prodedure */
   put 'End Sub' ;
run ;
```

STEP 4. Run the VBA code

The following code will run the file OddsRatios.bas (&program.) created in Step 3. The result will be the Word document OddsRatios.doc which contains the stock chart, populated with SAS data, and saved in the defined location (&path2.). **Figure 6** shows the results of Example 3, a stock chart populated with PROC Logistic data.

```
%let wordpath=d:\progra~1\micros~1\office ;
%RUN_VBA(wordpath=&wordpath,
         vba_path=&path2, vba_pgm=&program,
          visible=Y, minimize=N, notify=n,
         exitword=y) ;
```

Figure 6: Stock Chart with PROC Logistic data



CONCLUSIONS

By using SAS Output Delivery System and Visual Basic for Applications, data can be generated in a SAS procedure and data cells of a Microsoft Graph chart automatically populated. There are four steps involved in creating a Microsoft Graft in this manner. This method can be used for any of the chart types available in Microsoft Graph and use data from any SAS procedure.

The four steps are as follows:
STEP 1. Create a Microsoft Graph chart template
STEP 2. Create data file to import into the chart
STEP 3. Create VBA code to produce chart
STEP 4. Run the VBA code

Examples have been given in this paper to produce three different types of graphs using two SAS procedures. SAS macros have been presented that use PUT statements to inset VBA code into the output file. The macro **VBA_LIB**, found in **Appendix 1**, contains these individual macros. The Stetz method (SUGI 25 Proceedings, 2000) has been described which uses VBA to automate tasks in Microsoft Word, directly from SAS.

REFERENCES

Allison, Paul D. Logistic Regression Using the SAS® System: Theory and Application, Cary, NC: SAS Institute Inc., 1999. 304 pp.

Mansfield, Ron (1997). Mastering Word 97, SYBEX Inc., 684-676.

McFedries, Paul (1999). VBA for Microsoft Office 2000 Unleashed, Sams Publishing.

SAS Institute Inc., Logistic Regression Examples Using the SAS® System, Version 6, First Edition, Cary, NC: SAS Institute Inc., 1995. 163 pp.

SAS Institute Inc., SAS/STAT® User's Guide, Version 8, Cary, NC: SAS Institute Inc., 1999. 3884 pp.

Stetz, Mark (2000). "Using SAS® Software and Visual Basic for Applications to Automate Tasks in Microsoft Word: An Alternative to Dynamic Data Exchange", Proceedings of the Twenty-Fifth Annual SAS Users Group International Conference, 21, 136-141.

TRADEMARKS

CONTACT INFORMATION

Contact the author at:

Lori S. Parsons
Ovation Research Group
305 Fir Place
Edmonds, WA 98020
Work Phone: (425) 672-8782
Fax: (425) 672-7282
Email: lparsons@ovation.org

APPENDIX 1

```
/* ************************************** */
/* ************************************** */
/* Macro VBA_LIB                          */
/* VBA code Macro Library                 */
/* Lori Parsons 2002                      */
/* ************************************** */
/* ************************************** */
%MACRO VBA_LIB ;

/* ************************************** */
/* Put Text                               */
/* ************************************** */
   %MACRO puttext(text) ;
      put 'Selection.TypeText '
          "Text:=""&text""" ;
   %MEND puttext ;

/* ************************************** */
/* Page setup for Landscape Tables        */
/* ************************************** */
%MACRO TblPage;
    put 'Selection.MoveLeft Unit:=wdCharacter,
        Count:=6';
    put 'Selection.MoveUp Unit:=wdLine,
        Count:=2';
    put 'With ActiveDocument.PageSetup';
    put '     .LineNumbering.Active = False';
    put '     .Orientation = wdOrientLandscape';
    put '     .TopMargin = InchesToPoints(0.5)';
    put '     .BottomMargin =
        InchesToPoints(0.17)';
    put '     .LeftMargin =
        InchesToPoints(0.17)';
    put '     .RightMargin =
        InchesToPoints(0.17)';
    put '     .Gutter = InchesToPoints(0)';
    put '     .HeaderDistance =
        InchesToPoints(0.5)';
    put '     .FooterDistance =
        InchesToPoints(0.5)';
    put '     .PageWidth = InchesToPoints(11)';
    put '     .PageHeight = InchesToPoints(8.5)';
    put '     .FirstPageTray =
        wdPrinterDefaultBin';
    put '     .OtherPagesTray =
        wdPrinterDefaultBin';
    put '     .SectionStart = wdSectionNewPage';
    put '     .OddAndEvenPagesHeaderFooter =
        False';
    put '     .DifferentFirstPageHeaderFooter =
        False';
    put '     .VerticalAlignment =
        wdAlignVerticalTop';
    put '     .SuppressEndnotes = True';
    put '     .MirrorMargins = False';
    put 'End With';
%MEND TblPage;

/* ************************************** */
/* Page setup for Portrait Tables         */
/* ************************************** */
   %MACRO TblPPage;
    put 'Selection.MoveLeft Unit:=wdCharacter,
        Count:=6';
    put 'Selection.MoveUp Unit:=wdLine,
        Count:=2';
    put 'With ActiveDocument.PageSetup';
    put '     .LineNumbering.Active = False';
    put '     .Orientation = wdOrientPortrait';
    put '     .TopMargin = InchesToPoints(0.5)';
    put '     .BottomMargin =
        InchesToPoints(0.5)';
    put '     .LeftMargin = InchesToPoints(0.5)';
    put '     .RightMargin =
        InchesToPoints(0.17)';
    put '     .Gutter = InchesToPoints(0)';
    put '     .HeaderDistance =
        InchesToPoints(0.5)';
    put '     .FooterDistance =
        InchesToPoints(0.5)';
    put '     .PageWidth = InchesToPoints(8.5)';
    put '     .PageHeight = InchesToPoints(11)';
    put '     .FirstPageTray =
        wdPrinterDefaultBin';
    put '     .OtherPagesTray =
        wdPrinterDefaultBin';
    put '     .SectionStart = wdSectionNewPage';
    put '     .OddAndEvenPagesHeaderFooter =
        False';
    put '     .DifferentFirstPageHeaderFooter =
        False';
    put '     .VerticalAlignment =
        wdAlignVerticalTop';
    put '     .SuppressEndnotes = True';
    put '     .MirrorMargins = False';
    put 'End With';
   %MEND TblPPage;

/* ************************************** */
/* Insert a File                          */
/* ************************************** */
   %MACRO InsFile(Pth,InFile);
    put 'ChangeFileOpenDirectory "' "&Pth."
        '\"';
    put 'Selection.InsertFile FileName:="'
        &InFile. '", Range:="",
        ConfirmConversions:=False, Link:=False,
        Attachment:=False'
   %MEND InsFile;

/* ************************************** */
/* Insert an Object -- Picture            */
/* ************************************** */
   %MACRO InsPict(Pth,InFile);
    put 'ChangeFileOpenDirectory "' "&Pth."
        '\"';
    put 'Selection.InlineShapes.AddPicture
        FileName:="' &InFile. '",
        LinkToFile:=False,
        SaveWithDocument:=True'
   %MEND InsPict;

/* ************************************** */
/* Insert an Object -- Word Document File */
/* ************************************** */
   %MACRO InsWDObj(Pth,InFile);
    put 'ChangeFileOpenDirectory "' "&Pth."
        '\"';
    put 'Selection.InlineShapes.AddOLEObject
        ClassType:="Word.Document.8",
        FileName:="' &InFile. '" ,
        LinkToFile:=False, DisplayAsIcon:=False';
   %MEND InsWDObj;

/* ************************************** */
/* Update data in a WORD GRAPH file       */
/* ************************************** */
   %MACRO RunGraph(Dfname,FigTFile);

    /* Insert the Chart Template file */
      put ''' ';
      put '''Insert the Chart Template file';
      %InsFile(&path2.,&FigTFile.);

    /* Import the Chart Data */
      put '''  ';
      put '''Import the Chart Data';
```

```
      put 'Dim mychart As Object';
      put 'Dim r As Range';
      put
        'ActiveDocument.InlineShapes(1).OLEFormat
        .Edit';
      put 'Set mychart =
        ActiveDocument.InlineShapes(1).OLEFormat.
        Object';
      put
        'mychart.Application.DataSheet.Range("A0:
        Z25").Clear';
      put 'mychart.Application.FileImport
        FileName:="' "&Dfname." '"';

      put 'mychart.Application.Update';
      put 'Set mychart = Nothing';

   /* Save the new chart and Close Word */
         put ''' ';
      put '''Save the new chart and Close Word';
      %saveas(&path2\&program);
      %fileclos;
   %MEND RunGraph;

/* ************************************* */
/* Save file                             */
/* ************************************* */
   %MACRO saveas(fname,
               format=wdFormatDocument) ;
      put 'ActiveDocument.SaveAs '
           "FileName:=""&fname"", "
           "FileFormat:=&format" ;
   %MEND saveas ;

/* ************************************* */
/* Close Word                            */
/* ************************************* */
   %MACRO fileclos ;
      put 'ActiveDocument.Close '
      'SaveChanges:=wdDoNotSaveChanges' ;
   %MEND fileclos ;

%MEND VBA_LIB ;
```

APPENDIX 2

```
/* ************************************* */
/* ************************************* */
/*
Macro RUN_VBA.SAS
Written by Mark Stetz (2000)

Macro to Generate a batch file to set
environment variables and invoke Word using the
/m option to run the RUN_VBA Word macro. The
RUN_VBA Word macro uses the environment
variables' values to read in and execute the VBA
macro source code file(s) generated by the
calling SAS program.
*/
/* ************************************* */
/* ************************************* */

%MACRO RUN_VBA(wordpath=,
               vba_path=,
                vba_pgm=,
                visible=Y,
               minimize=N,
                 notify=N,
               exitword=N) ;
   options noxwait ;
      /* Create the Batch File */
      DATA _null_ ;
         file "&vba_path\&vba_pgm..bat" ;
         put "set VBA_PATH=&vba_path" ;
         put "set VBA_PGM=&vba_pgm" ;
         put "set VISIBLE=&visible" ;
         put "set MINIMIZE=&minimize" ;
         put "set NOTIFY=&notify" ;
         put "set EXITWORD=&exitword" ;
         put "start &wordpath\winword.exe
            /mRUN_VBA" ;
         put "del ""&vba_path\&vba_pgm..bat"" " ;
      run ;

     /* Run the batch file created in the previous
          DATA step */
      DATA _null_ ;
        call system("""&vba_path\&vba_pgm..bat""");
      run ;

%MEND RUN_VBA ;
```

APPENDIX 3

Data file created in Example 1 (WordFigures.dat)

```
0       1       2       3       4       5       6       7       8       > 8
0.7     0.9     2.2     3.7     5.7     10.2    14.7    18.4    23.2    34.7
2.1     3.5     6.8     10.1    14.4    18.9    21.8    24.1    24.8    29.2
1.1     1.7     3.5     6.2     10.1    15.3    19.5    22.7    24.5    30
```

VBA code file created in Example 1 (SUGI1.bas)

```
Sub SUGI1()
Selection.MoveLeft Unit:=wdCharacter, Count:=6
Selection.MoveUp Unit:=wdLine, Count:=2
With ActiveDocument.PageSetup
    .LineNumbering.Active = False
    .Orientation = wdOrientPortrait
    .TopMargin = InchesToPoints(0.5)
    .BottomMargin = InchesToPoints(0.5)
    .LeftMargin = InchesToPoints(0.5)
    .RightMargin = InchesToPoints(0.17)
    .Gutter = InchesToPoints(0)
    .HeaderDistance = InchesToPoints(0.5)
    .FooterDistance = InchesToPoints(0.5)
    .PageWidth = InchesToPoints(8.5)
    .PageHeight = InchesToPoints(11)
    .FirstPageTray = wdPrinterDefaultBin
    .OtherPagesTray = wdPrinterDefaultBin
    .SectionStart = wdSectionNewPage
    .OddAndEvenPagesHeaderFooter = False
    .DifferentFirstPageHeaderFooter = False
    .VerticalAlignment = wdAlignVerticalTop
    .SuppressEndnotes = True
    .MirrorMargins = False
End With
'
'Insert the Chart Template file
ChangeFileOpenDirectory "D:\Admin\SUGI\SUGI28\SASWork\Figures\"
Selection.InsertFile FileName:="SUGI_Example1_Template.doc", Range:="", ConfirmConversions:=False,
Link:=False, Attachment:=False
'
'Import the Chart Data
Dim mychart As Object
Dim r As Range
ActiveDocument.InlineShapes(1).OLEFormat.Edit
Set mychart = ActiveDocument.InlineShapes(1).OLEFormat.Object
mychart.Application.DataSheet.Range("A0:Z25").Clear
mychart.Application.FileImport FileName:="D:\Admin\SUGI\SUGI28\SASWork\Figures\WordFigure.dat"
mychart.Application.Update
Set mychart = Nothing
'
'Save the new chart and Close Word
ActiveDocument.SaveAs FileName:="D:\Admin\SUGI\SUGI28\SASWork\Figures\SUGI1",
FileFormat:=wdFormatDocument
ActiveDocument.Close SaveChanges:=wdDoNotSaveChanges
End Sub
```

APPENDIX 4

Data file created in Example 2 (WordFigures.dat)

```
0.01 0.02 0.03 0.05 0.07 0.09 0.14 0.23 0.38 0.54 0.62 0.74 0.89 1    0    0.01 0.03 0.08 0.16 0.3  0.5  0.7  0.88 1
0.05 0.08 0.11 0.14 0.19 0.24 0.33 0.46 0.66 0.81 0.86 0.93 0.98 1
                                                                      0.04 0.1  0.2  0.34 0.51 0.67 0.82 0.93 0.98 1
```

APPENDIX 5

Data file created in Example 3 (WordFigures.dat)

```
 PTAge     White     MHCABG    MHDIAB    MHCAD     MHPTCA    MHBLEED   CHF
1.041     0.908     0.892     1.140     0.585     0.649     1.264     1.626
1.044     0.987     0.973     1.216     0.641     0.725     1.543     1.733
1.042     0.947     0.932     1.177     0.612     0.686     1.397     1.678
```