**Paper CC03**

# SAS to R to SAS

Philip R Holland, Holland Numerics Limited, Royston, Herts, UK


## ABSTRACT

The aim of this paper is to describe one method of passing SAS data from SAS to R, using R to produce a graph, then passing that graph back to SAS for inclusion in an ODS document. The R programming language provides a wide range of graphical functionality, some of which are unavailable, or time-consuming to achieve, in SAS/GRAPH. Using the method described this functionality can be made available to SAS applications. The basic principles demonstrated here could also be adapted to create character-based reports using R for inclusion in SAS reports.


## SOFTWARE ENVIRONMENT

- The examples described in this paper use Windows XP, but any platform compatible with SAS and R could have been used.
- The methods can be used in any version of Base SAS from version 7 onwards. No other licensed SAS components are required.
- R requires 2 non-standard add-on libraries to be installed to support the techniques used in this paper. The "**Hmisc**" library adds R functions to import "foreign" data into R, e.g. SAS data, comma-separated value (CSV) data, etc. The library requires an additional SAS macro, **%exportlib**, which can be used to export a folder of SAS datasets into a collection of CSV files to be read into R using the **sasxport.get** function. The "**lattice**" library adds R functions to create "trellis" graphics. The "**grDevices**" library is supplied as part of the R system and includes functions to create a variety of image file formats, including JPEG, GIF and PNG.

## PROGRAM FLOW

**SAS activity**

**R activity**

Select SAS dataset to transfer and save the dataset to folder.:

```
LIBNAME new 'c:\temp\new';

PROC datasets LIB=new KILL;
RUN;
QUIT;

DATA sasuser.v_prdsale / VIEW = sasuser.v_prdsale;
  SET sashelp.prdsale;
  LENGTH yyq $6;
  yyqtr = year + (quarter - 1)/4;
  mon = MONTH(month);
  yyq = PUT(month, YYQ6.);
  yq = INTCK('QTR', '31dec1992'd, month);
  SELECT (country);
    WHEN ('U.S.A.') cntry = 'USA';
    WHEN ('GERMANY') cntry = 'DE';
    WHEN ('CANADA') cntry = 'CA';
    OTHERWISE;
  END;
RUN;

PROC SUMMARY DATA=sasuser.v_prdsale MISSING NWAY;
  CLASS cntry yq product;
  VAR actual;
  OUTPUT OUT=new.prdsale SUM=;
RUN;
```

**SAS activity**

**R activity**

Export folder to CSV files (using **%exportlib**), including the contents of the folder and any SAS user formats. Note that the folders must be written with '**/**' separators, even if you are running the program in Windows. The macro Exports all SAS datasets in a data library to CSV files. One of the datasets is assumed to be the result of PROC FORMAT CNTLOUT= if any user formats are referenced. Numeric variables are formatted in BEST16 format so that date/time variables will be exported with their internal numeric values. A special file _contents_.csv is created to hold, for all datasets combined, the dataset name, dataset label, variable names, labels, formats, types, and lengths.

```
/* Macro exportlib
Usage:
  %exportlib(lib, outdir, tempdir);
Arguments:
  lib     - SAS libname for input datasets
  outdir  - directory in which to write .csv files (default ".")
  tempdir - temporary directory to hold generated SAS code
            (default C:/WINDOWS/TEMP)
*/

%MACRO exportlib(lib, outdir, tempdir);
  %IF %QUOTE(&outdir)= %THEN %LET outdir=.;
  %IF %QUOTE(&tempdir)= %THEN %LET tempdir=C:/WINDOWS/TEMP;
  OPTIONS NOFMTERR;
  PROC COPY IN=&lib OUT=work;
  RUN;
  PROC CONTENTS DATA=work._ALL_ NOPRINT
    OUT=_contents_(KEEP=memname memlabel name type label format length
                        nobs);
  RUN;
  PROC EXPORT DATA=_contents_ OUTFILE="&outdir/_contents_.csv" REPLACE;
  RUN;
  DATA _NULL_;
    SET _contents_;
    BY memname;
    FILE "&tempdir/_export_.sas";
    RETAIN bk -1;
    IF FIRST.memname & (nobs > 0) THEN DO;
      PUT 'DATA ' memname ';';
      PUT '  SET ' memname ';';
      PUT '  FORMAT _NUMERIC_ BEST14.;';
      PUT 'RUN;';
      PUT 'PROC EXPORT DATA=' memname;
      PUT '            OUTFILE="' "&outdir/" memname +bk '.csv"';
      PUT '            REPLACE;';
      PUT 'RUN;';
    END;
  RUN;
  %INCLUDE "&tempdir/_export_.sas";
%MEND exportlib;

PROC FORMAT CNTLOUT=_cntlout;
RUN;

%exportlib(new, c:/temp/r, c:/windows/temp);
```

**SAS activity**

**R activity**

Generate R code (including **sasxport.get**) to read CSV files and write the generated graph to a JPEG file of 480x480 pixels:

```
DATA _NULL_;
  FILE 'c:\temp\r\program.r' LRECL=1024;
  PUT 'library(Hmisc)';
  PUT 'library(lattice)';
  PUT 'library(grDevices)';
  PUT "sasdata <- sasxport.get('c:/temp/r', method=('csv'))";
  PUT "trellis.device(jpeg, file='c:/temp/r/program.jpg',";
  PUT '               width=480, height=480)';
  PUT 'trellis.par.set(theme=col.whitebg())';
  PUT "trellis.par.set('background',list(col='white'))";
  PUT "trellis.par.set('plot.symbol',list(col='blue'))";
  PUT "trellis.par.set('dot.symbol',list(col='blue'))";
  PUT "trellis.par.set('axis.line',list(col='red'))";
  PUT "trellis.par.set('box.rectangle',list(col='red'))";
  PUT "trellis.par.set('par.xlab.text',list(col='green'))";
  PUT "trellis.par.set('par.ylab.text',list(col='green'))";
  PUT "trellis.par.set('par.zlab.text',list(col='green'))";
  PUT "trellis.par.set('axis.text',list(col='green'))";
  PUT 'xyplot(actual ~ yq | product*cntry';
  PUT '      ,data=sasdata$prdsale';
  PUT "      ,xlab = 'Quarter'";
  PUT "      ,ylab = 'Actual Sales'";
  PUT '      ,panel = function(x, y) {';
  PUT '                               panel.grid(h=-1, v=-1)';
  PUT '                               panel.xyplot(x, y)';
  PUT '                               panel.loess(x, y';
  PUT '                                          ,span=1';
  PUT '                                          ,degree=2';
  PUT '                                          )';
  PUT '                              }';
  PUT "      ,main = 'Plotted using R'";
  PUT '      )';
  PUT 'dev.off()';
  PUT 'q()';
RUN;
```

Execute R command line, including R code file as the input program. In this case the R program can be found in the Windows default program path:

```
OPTIONS XWAIT XSYNC;
X "r.exe --no-save --quiet <""c:\temp\r\program.r"" >""c:\temp\r\program.log""";
```

**SAS activity**

**R activity**

Execute R code, outputting R log to a text file and the graph to a JPEG file. Note that the white background is required for most ODS Styles to allow the resulting graphs to coordinate with their colour schemes. The default background for R graphs is a light grey

```
library(Hmisc)
library(lattice)
library(grDevices)
sasdata <- sasxport.get('c:/temp/r', method=('csv'))
trellis.device(jpeg, file='c:/temp/r/program.jpg',
               width=480, height=480)
trellis.par.set(theme=col.whitebg())
trellis.par.set('background',list(col='white'))
trellis.par.set('plot.symbol',list(col='blue'))
trellis.par.set('dot.symbol',list(col='blue'))
trellis.par.set('axis.line',list(col='red'))
trellis.par.set('box.rectangle',list(col='red'))
trellis.par.set('par.xlab.text',list(col='green'))
trellis.par.set('par.ylab.text',list(col='green'))
trellis.par.set('axis.text',list(col='green'))
xyplot(actual ~ yq | product*cntry
      ,data=sasdata$prdsale
      ,xlab = 'Quarter'
      ,ylab = 'Actual Sales'
      ,panel = function(x, y) {
                                panel.grid(h=-1, v=-1)
                                panel.xyplot(x, y)
                                panel.loess(x, y
                                           ,span=1
                                           ,degree=2
                                           )
                               }
      ,main = 'Plotted using R'
      )
dev.off()
```

Close R session:

```
q()
```

Copy the R log file into the SAS log:

```
DATA _NULL_;
  INFILE 'c:\temp\r\program.log';
  FILE LOG;
  INPUT;
  PUT '**R: ' _INFILE_;
RUN;
```

Open the ODS destination, e.g. HTML:

```
ODS ESCAPECHAR='^';
ODS HTML FILE='c:\temp\r\report.html' STYLE=minimal
         GPATH='c:\temp\r' GTITLE GFOOTNOTE;
```

or RTF:

```
ODS ESCAPECHAR='^';
ODS RTF FILE='c:\temp\r\report.rtf' STYLE=minimal
        GTITLE GFOOTNOTE;
```

| SAS activity<br>R activity |
| --- |
| Incorporate the JPEG file in the SAS report in HTML:<br>`DATA _NULL_;`<br>`  FILE PRINT;`<br>`  PUT "<IMG SRC='c:\temp\r\program.jpg' BORDER='0'>";`<br>`RUN;`<br>or RTF:<br>`DATA _NULL_;`<br>`  FILE PRINT;`<br>`  PUT "^S={PREIMAGE='c:\temp\r\program.jpg'}";`<br>`RUN;` |
| Close ODS destination:<br>`ODS _ALL_ CLOSE;` |

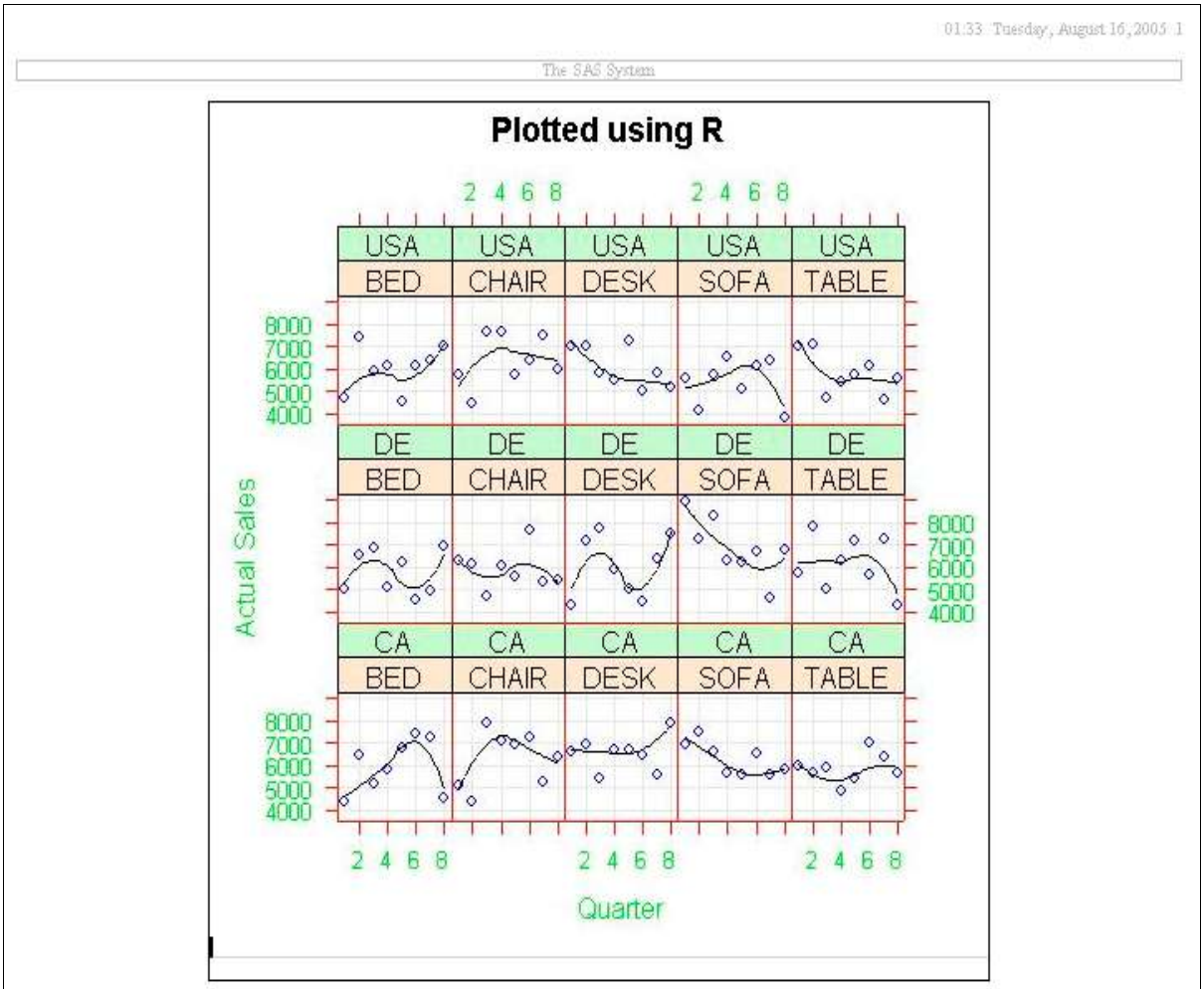


*Illustration 1 RTF Output*

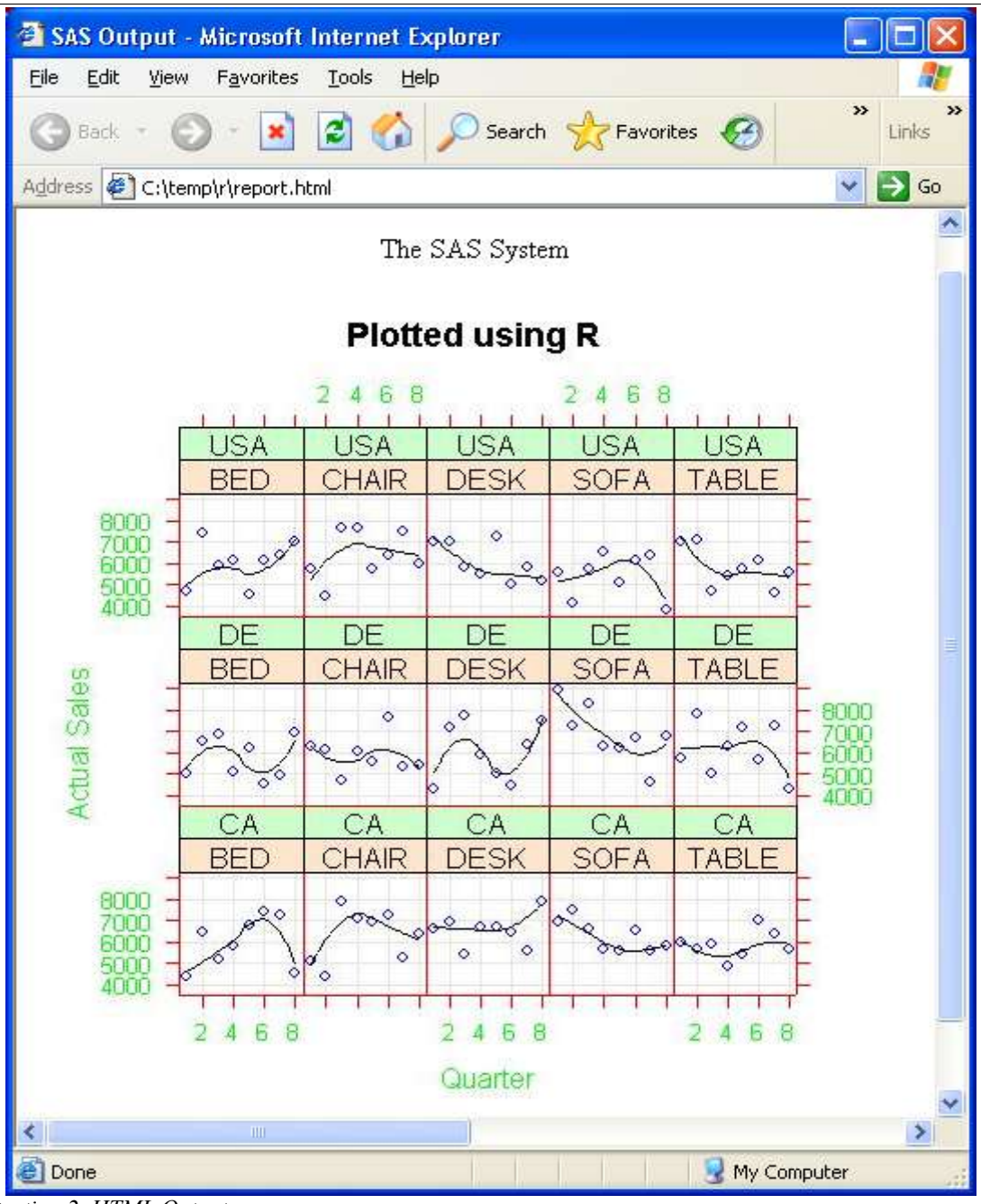

*Illustration 2 HTML Output*

**CODING ISSUES**

As an experienced SAS programmer, but an inexperienced R programmer, I had to resolve the following issues while developing this reporting application:

- HTML reports require a different syntax for displaying external image files to that used for non-HTML reports, e.g. RTF, PDF, etc., so the code must include separate code sections for use with HTML and non-HTML destinations.
- As different output destinations have different acceptable image formats, try to select a compatible image format for all the expected output destinations.
- As the export processing creates CSV files for every SAS dataset in the specified folder, limiting the number of SAS datasets in that folder will reduce the run time required for the R code to import the data.
- The R code is executed by calling the R system in line-command mode using the SAS X statement. The XSYNC and XWAIT SAS System options must be set before calling R.
- R programs may fail with minimal error information in the R log file.

## CHARACTER-BASED REPORTS

Most R statistical functions can direct their output to text files, instead of the screen, in the same way that the graphical functions can write directly to image files, e.g.:

```
library(Hmisc)
sasdata <- sasxport.get('c:/temp/r', method=('csv'))
attach(sasdata$prdsale)
sink('program.txt')
summary(sasdata$prdsale)
```

SAS code like the following could then be used to include the text from the report generated in R into an ODS report in SAS, remembering to select an ODS Style where the text can be read against the report background:

```
DATA _NULL_;
  INFILE 'program.txt';
  FILE PRINT;
  INPUT;
  PUT _INFILE_;
RUN;
```

## SOFTWARE RESOURCES AND FURTHER READING

- R Project for Statistical Computing: **www.r-project.org**
- **%exportlib**: **biostat.mc.vanderbilt.edu/twiki/pub/Main/Hmisc/exportlib.sas**
- The Complete Guide to the SAS Output Delivery System.

## CONTACT INFORMATION

The author is a consultant for Holland Numerics Ltd and can be contacted at the following address:


| | |
|---|---|
| | Philip R Holland |
| address: | Holland Numerics Ltd |
| | 94 Green Drift |
| | Royston |
| | Herts. SG8 5BT |
| | UK |
| e-mail: | phil@hollandnumerics.com |
| web: | www.hollandnumerics.com |
| tel. (mobile): | +44-(0)7714-279085 |


This paper and associated sample SAS code can be downloaded from the Holland Numerics Ltd web site at:
**www.hollandnumerics.com/SASPAPER.HTM**