From SASÒ ASCII Output to Word Document – A SASÒ Macro Approach

Jianlin 'Jay' Zhou, Quintiles, Inc., Kansas City, MO

ABSTRACT

Generating SAS ASCII output directly with PROC REPORT or a DATA _NULL_ step is the most common method used by SAS programmers in the pharmaceutical industry. However, because of the many limitations of SAS ASCII files for presentation and handling, more and more medical writers, biostatisticians, publishers, and expert reviewers now request SAS reports in Word document format. This paper introduces a macro that provides a process to produce aesthetically improved Word documents with very limited programming effort. The macro was designed to expand the presentation capability of ASCII files, to accommodate various fonts, font sizes, and margins, to add one of eighteen pagination styles to the final output, and to provide users with an option to compile individual files into one file for easy handling. The macro also automates the process of previewing, saving in either Word document or RTF format, and closing by interfacing with MS Word.

INTRODUCTION

The major task of SAS programmers in the pharmaceutical industry is to generate files, such as statistical outputs, summary tables, listings, and Case Report Form tabulations, directly generated from SAS procedures or a DATA _NULL_ step, for database audits, regulatory submissions, and ad-hoc requests. Sometimes programmers might get frustrated trying to keep the same appearance, especially pagination, of SAS statistical outputs directly generated with a SAS/STAT procedure, as other tables, listings, or data tabulations for the same clinical study report. Because SAS does not provide any customizing pagination features such as styles and the total number of pages, it is not an easy task to have the same pagination style (e.g. Page 1 of 200) for all outputs of an entire study to make the report more acceptable and complete.

With SAS version 6.12, all of the outputs created with PROC PRINTTO are ASCII text files. Because SAS ASCII output is not a Word file, it limits the presentation of special symbols and sub- and super-scripts with desired fonts, font sizes, and margins. The page orientation, text alignment, and readability are lost if the output is directly opened in Word. Therefore, manual post-processing or manipulation of SAS ASCII outputs is required in order to maintain its character alignment and readability in Word, which can be time-consuming.

For that reason, SAS programmers have been trying to come up with innovative ways of creating Word documents. Wehr (1996) introduced his Print Driver (%print) that can generate text, RTF, PostScript, or HTML files. Because the %print macro can only perform a single format task with each invocation, many macro calls within a DATA _NULL_ step involving substantial programming effort are needed even to generate a simple table, which might result in time-consuming and difficult program maintenance. Cunningham (1998) developed a SAS macro to insert delimiters between rows and columns and to convert the delimited text table to an MS Word table by interfacing with MS Word. This method works well for small and simple tables, but may not be applicable for large and complex tables, as sometimes rows or columns cannot easily be separated, and some manual editing cannot be omitted. Peszek, et al. (1999), presented two SAS macros, %rtf and %wrapup, to generate Word tables in RTF format. The first macro, used within a DATA _NULL_ step, provides full control over the table appearance with multiple calls, but good experience in the DATA _NULL_ step with PUT statements, as well as a substantial amount of programming effort, are required. The second macro can generate an RTF table with a single call, but it may not be flexible enough to create a complex table with multiple pages. Yam (2000) described a method used to generate Word tables by transferring SAS data to Excel with Dynamic Data Exchange (DDE), which serves as an intermediary for repackaging SAS data with a Visual Basic for Application (VBA) macro and interfacing with Word via Object Linking and Embedding (OLE) to get the Excel table into Word. With this method, 24 SAS variables must be derived in order to instruct Excel to format a table appropriately. Therefore, the process may be time-consuming and difficult to maintain. Within Quintiles, some of my colleagues insert a text file or copy the SAS output from the SAS output window and paste it into Word, then run a VBA macro to format the output and save it as Word document. The method is very simple and easy, but only one file is processed at a time and some minor manual effort is required. Additionally, it will not change the pagination provided by SAS.

The SAS macro discussed in this paper was developed to automate the conversion of any ASCII file to a Word document by employing the Rich Text Format (RTF) language as a bridge and taking the advantage of interfacing with Word via DDE. The macro functions as an RTF 'writer' to embed the RTF control words and symbols in the text file. The macro also invokes Word to insert the RTF file into Word and saves it as Word document (if requested). The macro also gives users an opportunity to customize pagination, including the total number of pages, page numbering styles, and positions, which is not provided by any SAS procedure and otherwise requires additional programming effort. In general, the macro provides a fast, reliable, and simple way to create Word documents automatically from computer-generated text files. It presents a significant time saving over a manual process and any existing method known.

THE MACRO %SAS2WORD

The macro processes ASCII files with .lst, .txt, .tbl, .lis, .apn, and .app extensions in the Windows, Unix, and VMS platforms. It can be called in batch or non-batch mode to perform the following functions:

- Process one or all files under a specific directory with a single call.
- Control the appearance of the Word document by specifying style, page orientation, margins, font type, and font size.
- Provide pagination that includes customizing features (18 styles) and total number of pages.
- Compile multiple text files into one file for easy handling.
- Invoke MS Word to preview the file in Windows.
- Save the file as text (.txt, .lst, or any other extension), RTF (.rtf) or Word document (.doc)

The call to the macro takes the following form (defaults shown):

```
%sas2word(in      =,
          out     =,
          ext     =,
          o       =l,
          pstyle  =plain text,
          style   =,
          location =t,
          margin  =1 1 1 1,
          font    =Courier New,
          fs      =8,
          append  =n,
          show    =y )
```

The macro has 12 not case sensitive keyword parameters as described below:

in

Determines the directory path where the text files reside. All files are converted if only the path is defined, whereas a single file is processed if the name and extension of a file is specified with the path.

out

Defines the directory path where the processed files will reside. By default, the processed files will be stored under the same directory as the original text files.

ext

Specifies the extension to use for the files processed by the macro. This parameter activates the processes of embedding RTF control words and symbols and of interfacing with MS Word. For instance, the macro saves the RTF file as a Word document in MS Word if **ext**=doc. By default, the original extension is used, and the macro neither embeds RTF code nor interfaces with MS Word. This is useful when a pagination macro is desired to add customized page numbering to the files.

o

Defines the page orientation, with l for landscape (default) or p for portrait.

pstyle

Specifies Normal or Plain Text for the Word style, which provides a collection of format settings for the document, dictating the arrangement of text on pages, the shape of the paragraphs, and the characteristics of the letters. This parameter is designed to set the style in order to maintain consistency with the style used elsewhere in the clinical study report.

style

Specifies the pagination style. Eighteen styles are available (Table 1). For instance, the style Page x of y is applied if **style**=pagexofy. By default, the pagination of the input file will not be changed.

Table 1. Pagination Style and Example

Style	Example
pagexofy	Page 1 of 10
pagexofyp	(Page 1 of 10)
xofy	1 of 10
xofyp	(1 of 10)
pxofy	p. 1 of 10
pxofyp	(p. 1 of 10)
px	p. 1
pxp	(p. 1)
pagex	Page 1
pagexp	(Page 1)
pagen	1
pagenp	(1)
pagexpy	Page 1/10
pagexpyp	(Page 1/10)
pxpy	p. 1/10
pxpyp	(p. 1/10)
xpy	1/10
xpyp	(1/10)

location

Determines where the page number is placed on a page. By default, **location**=t, which directs the page number to print on the right top corner. The other option is **location**=b for the page number to print on the right bottom corner. However, this parameter loses its function if a token PAGEXOFY (described subsequently in the section on Pagination) is programmed to print on the output, which is designed to reserve space for the page number. This allows flexibility to present the pagination in any location.

margin

Specifies page margins in inches with four variants, separated from each other by a space, and representing top, bottom, left, and right margins, respectively. The default is one inch for each side, which meets FDA requirements. Any number, including decimals, can be used to define the parameter.

font

Determines the specific font type used in the RTF or Word file. Loss of text alignment and readability occurs if a proportional font, such as Times New Roman or Arial, is used. To maintain character alignment and readability, the monospace fonts, such as Courier, Courier New, SAS Monospace, or SAS Monospace Bold, should be used.

fs

Defines the font size. By default, the font size is 8.

append

Indicates whether or not to append all text files in one output file. The order in which the files are appended is based on the alphabetic order of the file names. Options are yes (Y) or no (N), with no being the default.

show

Controls the interface between SAS and MS Word. Options are yes (Y), close (C), or no (N). When the parameter is specified as Y or C, and **ext**=rtf or **ext**=doc, the SAS system opens MS Word if it is not running. The processed file is inserted into Word and saved. The difference between Y and C is that the file stays open to view for the former and is closed for the latter. SAS will not talk to MS Word if **show**=no. In this case, a Word document will not be generated even if **ext**=doc; rather, the output will be an ASCII file with a doc extension.

TECHNICAL DETAILS

THE RTF LANGUAGE

The key to making the macro work is the RTF code that is employed to format a SAS ASCII file as an RTF file, which consists of unformatted text, control words, control symbols, and groups. A control word, not longer than 32 characters, comprises a backslash '\' followed by a formatted command that RTF uses to mark printer control codes. A control symbol consists of a backslash followed by a single, non-alphabetic character. The backslash acts as a trigger to identify a control word or symbol. A group consists of control words or symbols with text enclosed in braces '{}', which define the territory of a group (Microsoft Press, 1997).

In order for any RTF Reader to interpret the code appropriately, the RTF file header must consist of the character set, font table, and style sheet added to the beginning of a text file. The following RTF code, generated by the %sas2word macro and used as a file header, allows users to define the properties of the document, including style, page size and orientation, fonts, and font sizes. The bolded text indicates the places where macro variables are used.

{\rtf1\ansi\ansicpg1252\uc1 \deff0\deflang1033\deflangfe1033
{\fonttbl{\f0\fmodern\fcharset0\fpr1 **Courier New**;}
{\f1\froman\fcharset2\fprq2 Symbol;}}
{\stylesheet{\widctlpar\adjustright \f0\fs20\cgrid \snext0
Normal;}{*\cs10 \additive Default Paragraph font;}
{\s15\widctlpar\adjustright \f0\fs20\cgrid \sbasedon0 \snext15 **Plain
Text**;}}
\paperw**15840**\paperh**12240**\lndscpsxn
\margt**1728**\margb**864**\margl**1728**\margr**864**
\s15\sl192\slmult1\widctlpar\adjustright
{\f0 \fs**18**

In addition to the header, the macro also embeds two additional RTF control words, **\line** and **\page,** to control the text in the text body. **\line** is added to each text line to indicate the end of the line. The ASCII page break symbol is replaced with **\page** indicating a request for a page break. The whole RTF file is enclosed in braces '{}', which must be in balance. The RTF Reader will not open the file if the braces are unbalanced. This will occur when text fields contain an unbalanced brace.

As mentioned previously, the backslash '\' and braces '{}' have special meaning in RTF. However, these characters may be used as text. For example, c:\myfiles\test.sas, the SAS program name with its path, is in the footnote (Figure 1). In order for the backslashes to be visible in the RTF file, the %sas2word macro precedes them with a backslash, as in \\ . This holds the same for braces, as in \{ and \}.

To handle requests for presenting super- and sub-scripts and symbols in the document, RTF control words can be pre-embedded within the text files. Table 2 lists the most common and useful control words. These tags can be embedded in the outputs through SAS title and footnote statements and user-defined formats. For example, as shown in Figure 2, a superscripted **a** is used in the title to refer to footnote **a**. This request can be easily handled using the title and footnote statements with an RTF control group {\fs24 \super a}, where \fs24 indicates font size 12 and \super indicates superscript.

Table 2. Useful RTF Control Words

Control Word	Description
\f1 *	Turns on symbol, e.g. \f1 ¥ for symbol ∞
\b (\b0)	Turns on (off) bold
\i (\i0)	Turns on (off) italic
\u (\u0)	Turns on (off) underline
\super	Turns on superscript
\sub	Turns on subscript
\fs	Specifies font size, e.g. \fs24 for font size 12

* Only true if f1 is designed in the RTF header for symbol.

INTERFACING WITH WORD
Dynamic Data Exchange (DDE) establishes a link between SAS and other applications in the Windows environment, and provides a way to exchange information dynamically between applications. SAS passes WordBasic commands to Word with this method, to have it insert the processed RTF file, save the file as a Word document, and close the file, if requested.

The following code illustrates the interfacing process:

```
%*** create a filename for DDE linkage ***;
filename word DDE 'Winword|system';

%*** make sure Word is active ***;
%let fid=%sysfunc(fopen(word,S));

%*** invoke Word if it is not open ***;
%if &fid eq 0 %then %do;
   options noxwait noxsync;
   x ' "C:\Program Files\Microsoft Office\Office\winword.exe" ';
   data _null_;
      rc=sleep(6);
   run;
%end;

%*** WordBasic Commands ***;
data _null_;
   file word;
   put '[FileOpen.Name:= "' "path\filename.ext" '"]';
   %if &ext=doc %then %str(put
   "[FileSaveAs" """"path\filename.ext"", wdFormatDocument" "]";);
```

```
   %if &close gt 0 %then %str(put "[FileClose]";);
run;

%let rc=%sysfunc(fclose(&fid));
```

PAGINATION
The %sas2word macro performs pagination if the **style** parameter is specified with any style listed in Table 1. The macro searches for the carriage control character ('0C'x), erases the page number provided by SAS, and computes the total number of pages based on the total number of carriage control characters in the text file. It creates a temporary SAS data set that contains the entire output, with a field for page number with the desired style retained for every record (line). The temporary file will be deleted by the macro at the end of macro execution. The macro performs the pagination even if the **ext** parameter is not specified. This is useful for getting the desired pagination, but not changing the ASCII format of the file.

In most cases, the desired location of the page number is on the right top or bottom corner of a page. In order to have the flexibility to print the page number in any location requested, a token PAGEXOFY (or pagexofy) must be used to reserve the space. By using the SAS **tranwrd** function, the page number replaces the token with the desired style defined in the macro call. For instance, the token PAGEXOFY in the footnote (Figure 1) is replaced with the pagination specified as PAGEXOFYP style (Figure 2).

Table 3. Maximum Line Size and Page Size for Creating Word Documents with One-inch Margins on Each Side

Monospace Font	Orientation	Font Size	Line Size	Page Size
Courier (New)	Landscape	7	152	58
		8	134	51
		9	119	45
		10	107	41
		11	98	37
		12	89	34
	Portrait	7	111	81
		8	97	71
		9	86	63
		10	77	56
		11	70	52
		12	64	47
SAS Monospace (Bold)	Landscape	7	152	47
		8	134	41
		9	119	37
		10	107	33
		11	98	30
		12	89	27
	Portrait	7	111	66
		8	97	57
		9	86	51
		10	77	46
		11	70	42
		12	64	38

Because non-proportional fonts are required to maintain text alignment and readability, the selection of font size and margins is dependent on the line size and page size, which are defined in the SAS option statement. Table 3 provides the maximum line size and page size for different monospace fonts and font sizes to maintain one-inch margins on each side using a Normal style. The line size is not affected by different fonts of the same font size, but the page size is affected. More rows can be fit on a page with Courier or Courier New, which offers an option for presenting more data on a page than with SAS Monospace. Therefore, to achieve the goal of obtaining the desired font, font size, and margins for a Word document, but without changing the appearance and pagination of the original text file, it is important to select the right line size and page size when generating the ASCII file. For example, to create a Word document in a landscape orientation with Courier font, font

size 10, and one-inch margins on each side, the following SAS option statement should be used to obtain the maximum line size and page size for the text file:

 options ls=107 ps= 41;

EXAMPLE

Figure 1 presents the SAS ASCII output (for file test.lst residing in the c:\myfiles directory) generated using PROC REPORT with the following title and footnote statements. An RTF group {\fs24 \super a} is embedded in the statements in order to have the superscripted **a** with font size 12 in the Word file.

title 'Table 1.4. DRUG STUDY ANALYSIS{\fs24 \super a}';
title2 ' ';

… (the title and footnote statements for the solid lines are omitted)

footnote2 '{\fs24 \super a} Subject 5 was excluded from the analysis.';
footnote4 'Program: c:\myfiles\test.sas   PAGEXOFY';

```
Table 1.4. DRUG STUDY ANALYSIS{\fs24 \super a}              1


Days           Slope
Included       Estimate       p-value
3,4,5,6        -0.0236        0.1249
  4,5,6        0.0256         0.4798
   5,6         0.0103         0.6434




{\fs24 \super a} Subject 5 was excluded from the analysis.

Program: c:\myfiles\test.sas                      PAGEXOFY
```
Figure 1. SAS ASCII output from PROC REPORT

Figure 2 shows the Word document converted from the ASCII file with the following call statement:

%sas2word(in   = c:\myfiles\test.lst,
          ext   = doc,
          fs   = 10,
          style = pagexofyp,
)

```
Table 1.4. DRUG STUDY ANALYSIS^a

Days           Slope
Included       Estimate       p-value
3,4,5,6        -0.0236        0.1249
  4,5,6        0.0256         0.4798
   5,6         0.0103         0.6434



a Subject 5 was excluded from the analysis.

Program: c:\myfiles\test.sas              (Page 1 of 1)
```
Figure 2. Word table generated from %sas2word macro

By default, the Word file test.doc has one-inch margins on each side of the page, with the font Courier New, font size 10, and resides under the same directory as the text file. One can easily convert this Word document further to a PDF file using PDF Distiller.

CONCLUSION

The macro presented in this paper integrates SAS, RTF, and WordBasic code to provide a simple and fast way involving very limited programming effort to convert to a Word document any ASCII output generated with SAS procedures or DATA _NULL_ steps. It controls the appearance of the document with page orientation, paragraph style, margins, font, and font size, and provides programmers with 18 pagination style options. With this macro, it becomes easy to create statistical outputs with the same appearance, including pagination, as other outputs in a clinical study report, such as tables, tabulations, or listings. The shortcoming of the macro is that it uses non-proportional fonts to convert ASCII files to Word documents.

REFERENCES

Cunningham, G. (1998). An Easy-to-Use SAS Macro for Use with Microsoft Windows That Converts Existing Text Tables to Microsoft Word Tables. Proceedings of the PharmaSUG 98 Conference, 304-308.

Microsoft Press (1997), Microsoft MS-DOS, Windows, Windows NT, and Apple Macintosh Applications: Rich Test Format (RTF) Specification and Sample RTF Reader Program, Microsoft Technical Support Application Note, Version 1.5. 4/97-GC0165.

Peszek, I., Song, C., and Kuznetsova, O. (1999), Producing Tabular Reports in SAS® System in the Form of MS Word® Tables. Proceedings of the PharmaSUG 99 Conference, 298-293.

Wehr, P. (1996), %Print Drivers: Teaching SAS to Speak the Many Languages of Document Publication. Proceedings of the 21st Annual SAS® User's Group International Conference.

Yam, A. L. (2000), SAS® Software and Microsoft Office Visual Basic for Applications Make Beautiful Reports Together. Proceedings of the 25th Annual SAS® User's Group International Conference.

ACKNOWLEDGMENTS

SAS is a registered trademark or trademark of SAS Institute Inc. in the USA and other countries. ® indicates USA registration. Other brand and product names are registered trademarks or trademarks of their respective companies.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:
        Jay Zhou
        Quintiles, Inc.
        10245 Hickman Mills Drive
        Kansas City, MO 64134
        Work Phone: 816-767-6768
        Fax: 816-767-7347
        Email: jay.zhou@quintiles.com
               zhou_jay@yahoo.com