Paper RF-09-2013

An Efficient Approach to Automatically Convert Multiple Text Files (.TXT) to Rich Text Format Files (.RTF) Using SAS

Xingxing Wu, inVentiv Health Clinical, New York City, NY
Jyoti N. Rayamajhi, Eli Lilly and Company, Indianapolis, IN

ABSTRACT

A large number of legacy and Third-Party-Organization (TPO) provided clinical trial statistical analysis output files stored in SAS Drug Development (SDD) are in .txt (text) format. Because of many advantages of rich text format (RTF) files compared with text files, it's a common task to convert these files from text format to RTF to meet the submissions, regulatory responses, or other requirements. In addition, clinical reviewers often desire to have the RTF outputs since they can write comments on them. This paper provides an efficient and easy-to-use approach to automatically convert multiple text files to RTF files. This approach can directly run in SDD without relying on other tools, such as Microsoft Office VBA, and SDD desktop connection tool. Furthermore, it can also be directly used in other SAS development environment, such as PC SAS. In this paper, an innovative approach is also proposed to resolve the issue that the underline character "_" in the text files cannot be displayed after converted to RTF files in some situations. Compared with other approaches, the proposed one has the advantage of robustness. It can be directly applied to any situation without requiring the users to adjust the code to fit into their own situations.

This paper is intended for the audiences with some general knowledge about Base SAS and RTF.

INTRODUCTION

SDD is a product developed by SAS Institute Inc. to serve as both the clinical data repository and the SAS program development environment. It has been used by some pharmaceutical companies for their clinical researches, such as Eli Lilly Company. In SDD, many legacy SAS outputs and the analysis tables provided by TPO are in text format. The text files, however, are usually less expressive, and thus RTF are becoming a popular format of clinical trial analysis files. Therefore, it becomes a common task to convert text files to RTF files to meet different kinds of clinical research requirements. It would be very painful to manually convert each text file to RTF file with required styles. It is necessary to provide an approach to automatically convert large number of text files to RTF files efficiently. For this purpose, the first step is to automatically collect the list of text files needing conversion. The approaches commonly used today are based on system commands, and cannot be directly used in SDD unless the SDD desktop connection tool is used to map SDD folder to a local driver. In order to reduce the dependence and to easy the use, this paper provides two approaches to directly collect the file list based on the folder information. The second step is to convert each text file in the list to related RTF file. Several approaches have been provided in the past, such as the approach based on Microsoft Office VBA, and the approach based on adding RTF codes to the files in the data step. In SDD, the approach that can be directly used for the conversion without the aid of other tools should be based on the RTF manipulation. However, there is a known issue existing in this kind of approach. The underline character "_" in the text files sometime cannot be displayed after converted to RTF files even though it is there in the RTF files. Some paper suggested a solution to this issue. This solution, however, requires the users to update the code based on their own word processor. To overcome this shortage, an innovative approach to solve this issue is also proposed in this paper. It requires no adjustment of the code and can be directly used in all situations. Although the approaches proposed in this paper are originally for the programs running in SDD, it can also be directly applied to other SAS development environment.

GET THE FILE LIST

In order to get all the text file names stored in a folder, the commonly used approaches are based on the system commands, such as using

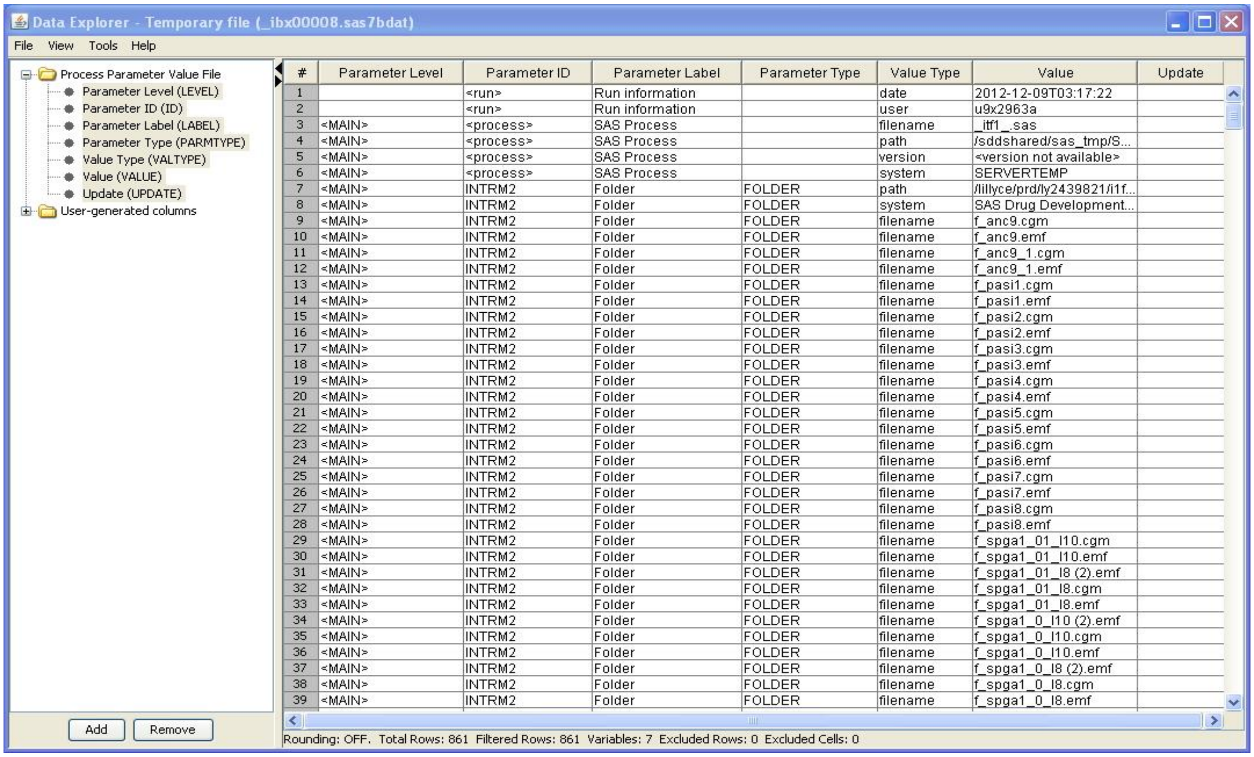
- `x "dir mvs\*.  /b > filelist.txt ";`
- `call system( "dir c:\logs\*.*  /b > c:\logs\filelist.txt" );`

- `filename tt pipe 'dir/b c:\temp\*.rtf';`

These approaches, however, cannot be directly used in SDD, unless the SDD folder is mapped to a local driver using SDD desktop connection tool. This paper provides two methods which can be directly applied to SDD without the support of any other tools.

METHOD ONE:

This method is specific to SDD and cannot be used in other SAS development environment. In SDD, there is a file called SDDPARMS which holds all the process parameter values, such as the process name, folder names, and the files names, as shown in the following graph:



#	Parameter Level	Parameter ID	Parameter Label	Parameter Type	Value Type	Value	Update
1		<run>	Run information		date	2012-12-09T03:17:22	
2		<run>	Run information		user	u9x2963a	
3	<MAIN>	<process>	SAS Process		filename	_itf1_.sas	
4	<MAIN>	<process>	SAS Process		path	/sddshared/sas_tmp/S...	
5	<MAIN>	<process>	SAS Process		version	<version not available>	
6	<MAIN>	<process>	SAS Process		system	SERVERTEMP	
7	<MAIN>	INTRM2	Folder	FOLDER	path	/lillyce/prd/ly2439821/i1f...	
8	<MAIN>	INTRM2	Folder	FOLDER	system	SAS Drug Development...	
9	<MAIN>	INTRM2	Folder	FOLDER	filename	f_anc9.cgm	
10	<MAIN>	INTRM2	Folder	FOLDER	filename	f_anc9.emf	
11	<MAIN>	INTRM2	Folder	FOLDER	filename	f_anc9_1.cgm	
12	<MAIN>	INTRM2	Folder	FOLDER	filename	f_anc9_1.emf	
13	<MAIN>	INTRM2	Folder	FOLDER	filename	f_pasi1.cgm	
14	<MAIN>	INTRM2	Folder	FOLDER	filename	f_pasi1.emf	
15	<MAIN>	INTRM2	Folder	FOLDER	filename	f_pasi2.cgm	
16	<MAIN>	INTRM2	Folder	FOLDER	filename	f_pasi2.emf	
17	<MAIN>	INTRM2	Folder	FOLDER	filename	f_pasi3.cgm	
18	<MAIN>	INTRM2	Folder	FOLDER	filename	f_pasi3.emf	
19	<MAIN>	INTRM2	Folder	FOLDER	filename	f_pasi4.cgm	
20	<MAIN>	INTRM2	Folder	FOLDER	filename	f_pasi4.emf	
21	<MAIN>	INTRM2	Folder	FOLDER	filename	f_pasi5.cgm	
22	<MAIN>	INTRM2	Folder	FOLDER	filename	f_pasi5.emf	
23	<MAIN>	INTRM2	Folder	FOLDER	filename	f_pasi6.cgm	
24	<MAIN>	INTRM2	Folder	FOLDER	filename	f_pasi6.emf	
25	<MAIN>	INTRM2	Folder	FOLDER	filename	f_pasi7.cgm	
26	<MAIN>	INTRM2	Folder	FOLDER	filename	f_pasi7.emf	
27	<MAIN>	INTRM2	Folder	FOLDER	filename	f_pasi8.cgm	
28	<MAIN>	INTRM2	Folder	FOLDER	filename	f_pasi8.emf	
29	<MAIN>	INTRM2	Folder	FOLDER	filename	f_spga1_01_l10.cgm	
30	<MAIN>	INTRM2	Folder	FOLDER	filename	f_spga1_01_l10.emf	
31	<MAIN>	INTRM2	Folder	FOLDER	filename	f_spga1_01_l8 (2).emf	
32	<MAIN>	INTRM2	Folder	FOLDER	filename	f_spga1_01_l8.cgm	
33	<MAIN>	INTRM2	Folder	FOLDER	filename	f_spga1_01_l8.emf	
34	<MAIN>	INTRM2	Folder	FOLDER	filename	f_spga1_0_l10 (2).emf	
35	<MAIN>	INTRM2	Folder	FOLDER	filename	f_spga1_0_l10.cgm	
36	<MAIN>	INTRM2	Folder	FOLDER	filename	f_spga1_0_l10.emf	
37	<MAIN>	INTRM2	Folder	FOLDER	filename	f_spga1_0_l8 (2).emf	
38	<MAIN>	INTRM2	Folder	FOLDER	filename	f_spga1_0_l8.cgm	
39	<MAIN>	INTRM2	Folder	FOLDER	filename	f_spga1_0_l8.emf	

Rounding: OFF. Total Rows: 861 Filtered Rows: 861 Variables: 7 Excluded Rows: 0 Excluded Cells: 0

Figure 1. SDDPARMS File

From this file, we can use the following code to get all the text file names in a folder, and then put the file names separated by the character “*” into one macro variable.

```
data filelist (keep=path valtype filename);
  length filename path $100;
  retain path;
  set &sddparms;
  where id="&dir" and valtype in ('path' 'filename');
  if valtype='path' then path=value;
  filename=value;
  if index(filename, ‘.txt’)>0
run;

proc sql noprint;
  select filename into :filelist separated by '*'
  from filelist;
Quit;
```

METHOD TWO:

The above method is specifically proposed for the SDD users. Now, we introduce another approach which can be used in both SDD and other SAS development environment.

```
filename exdir "&inlib";

%global allfile;
%macro dir( );
  %let dirid = %sysfunc(DOPEN(exdir));
  %let dircnt = %sysfunc(DNUM(&dirid));
  %put dircnt &dircnt;
  %let filelist=;
  %do i = 1 %to &dircnt;
    %let dirread&i = %sysfunc(DREAD(&dirid,&i));
    %if %scan(&&dirread&i, 2, .)=txt %then %do;

       %let filelist= &&dirread&i*&filelist;
    %end;
  %end;
  %let rc =%sysfunc(DCLOSE(&dirid));
%mend dir;
```

By method one or method two, we can get a macro variable which holds all the text file names like filelist=file1.txt*file2.txt*file3.txt*file4.txt*file5.txt;

CONVERT EACH TEXT FILE TO RELATED RTF FILE

After getting the text file name list, we can then start to convert each text file in the list into related RTF file automatically. Basically, there are three kinds of approaches for the conversion. The first approach is based on Microsoft Office VBA [1]. The second approach is through a SAS macro that uses ODS RTF to first read into the text file and then convert it to an RTF file. For this approach, the style can be changed using the template. For the third approach, the RTF contents that control the style are directly inserted into the text file in the SAS data step [2]. The first approach cannot be used in SDD without the additional support of other tools. For the reason of convenience and effectiveness, the paper implements the conversion by the RTF manipulation.

```
%macro text2rtf(in=,
               out=,
               ps=);

    data _null_;
      infile &in lrecl=1000 end=last;
      file "&out" lrecl=1000 ;
      input;
      if _n_=1 then do;
        put
        '{\rtf1\ansi\deff4\deflang1033' /
        '{\fonttbl {\f4\froman\fcharset0\fprq2 Arial;}' /
        '          {\f5\fswiss\fcharset0\fprq2 Arial;}'           /
        '          {\f11\fmodern\fcharset0\fprq1 Courier New;}'   /
        '          {\f14\fmodern\fcharset255\fprq2 Modern;}}'     /
        '{\stylesheet{\sb14\sa144\sl-300\slmult0\nowidctlpar \f4 \snext0 Normal;}'
                    '{\s27\fi-1944\li1944\sb240\sa120\sl259\slmult0\keep\keepn\nowidctlpar'
                     ' \b\f5\fs22 \sbasedon43\snext0 Tbl Title Cont;}'
                    '{\s34\sl-179\slmult0\nowidctlpar \b\f11\fs16 '
                     '\sbasedon41\snext34 md_SAS Tbl Entry;}'
                    '{\s41\sl259\slmult0 \keep\keepn\nowidctlpar \f4\fs20 '
                     '\sbasedon0\snext41 md_Tbl Entry;}'
                    '{\s43\fi-1944\li1944\sb240\sa120\sl259\slmult0\keep\keepn\nowidctlpar '
                     '\b\f5\fs22 \sbasedon0\snext0 Tbl Title;}} '
            '\paperw15840\paperh12240\margl1440\margr1440\margt2160\margb1440\gutter0 '
            '\widowctrl\ftnbj \sectd\lndscpsxn\headery1800\footery1080\linex0 \fs0 ' @;
      end;
      else do;
         if mod(_n_,&ps)=1 then put '\par \pard\plain \s34\sl-179\slmult0\nowidctlpar
\b\f11\fs16\page' _infile_;
         else put '\par \pard\plain \s34\sl-179\slmult0\nowidctlpar \b\f11\fs16 ' _infile_;
      end;
      if last then put '}';
run;
%mend;

%macro rpt();
%let ii=1;
%do %while(%scan(%scan(&filelist,&ii,*),2, .) = txt);
    %let filein=%scan(%scan(&filelist,&ii,*), 1, .);
    %text2rtf(in=exdir(&filein..txt), out=&rtfout/&filein..rtf, PS=46);
     %let ii=%eval(&ii+1);
%end;
%mend rpt;
%rpt;
```

The following is one example of the conversion from text file to RTF file:

TEXT FILE:

```
Listing of all Adverse Events leading to deaths that occurred during the study
Safety Analysis Set
XXX-XX-XXXX Study Parts A (Weeks 0-12) and B (Weeks 12-24)

_________________________________________________________________________________________

Treatment           Center/    MedDRA System Organ Class/  Start Date/  Stop Date/
Part A /            Patient    MedDRA Preferred Term/      [Study Day]/ [Study Day]/
Treatment           Number     Investigator's Term         Visit        Visit
Part B

_________________________________________________________________________________________
No cases to report
```

CONVERTED RTF FILE

```
Listing of all Adverse Events leading to deaths that occurred during the study
Safety Analysis Set
XXX-XX-XXXX Study Parts A (Weeks 0-12) and B (Weeks 12-24)



Treatment           Center/    MedDRA System Organ Class/  Start Date/  Stop Date/
Part A /            Patient    MedDRA Preferred Term/      [Study Day]/ [Study Day]/
Treatment           Number     Investigator's Term         Visit        Visit
Part B



No cases to report
```

SOLUTION TO THE UNSEEN UNDERLINE CHARACTER IN RTF FILES

From above example, we can notice that the converted RTF file does not display the underline which exists in the text file, when opened with MS Word. As discussed in other papers, the approach through RTF manipulation has one issue that the underline character "_" cannot be seen in the converted RTF file, even though these underline characters do exist in the RTF file. One approach has been suggested to solve this issue by adjusting the page size and line size. For this approach, however, the users need to change the code based on their own word process until the underline character can be displayed. This approach is not very convenient, and might not be able to be applied in the case when the page size and line size of the RTF file cannot be adjusted arbitrary. Now, an innovative and convenient approach is proposed in this paper. It does not need the adjustment and can be directly applied in all situations.

In order to display the underline character, the RTF command “\ul” is added to the front of the underline character “_”, and the RTF command “\ul0 “ is inserted after the underline character at the same time. After this, the underline character “_” is now becoming “\ul_\ul0”. Through this update, the underline character becomes visible when the RTF file is opened with MS Word. The following code is created for this purpose:

```
data _null_;
   infile tofile3 lrecl=16383  ;
   file "&out"  lrecl=32767 ;
   input ;
   _infile_ = tranwrd(_infile_,"_", "\ul_\ul0 ");
   put _infile_;
run;
```

The above code is simple. However, it looks awkward when applied to the case when a straight line is formed by several underline character “_” connected together. In this case, the “____” will become “\ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0 \ul_\ul0”.

To overcome this shortcoming, now we improve this approach to make it more intelligent by comparing a character with its previous and next characters. The updated code is shown as follows:

```
data _null_;
      length b b2 b3 b4 $1 b1 $3;
      infile tofile2 lrecl=16383  recfm = F length=len;
      file "&out" lrecl=32767 recfm = N ;
      input @1 b $char1. @;
if len>=3 then do;
      a1="\ul_\ul0 ";
      a2="\ul_";
      a3="_\ul0 ";
      a4="\ul0 ";
      a5="\ul_";
      if b="_" then put a5 $char4. @;
      else put b $char1. @;

      do i = 1 to len-2;
        input @i b1 $char3. @;
        b2=substr(b1, 1, 1);
        b3=substr(b1, 2, 1);
        b4=substr(b1, 3, 1);
        if (b3="_" and b2 ne '_' and b4 ne '_') then put a1 $char9. @;
        else if (b3="_" and b2 ne '_') then put a2 $char4. @;
        else if (b3="_" and b4 ne '_') then put a3 $char6. @;
        else if (b3 ne "_" and b2="_" and i=1) then put a4 $char5. @;
        else put b3 $char1. @;
      end;
      if (b4="_" and b3 ne '_') then put a1 $char9. @;
      else if b4="_" then put a3 $char6. @;
      else put b4 $char1. @;
end;
else do;
   input ;
   _infile_ = tranwrd(_infile_,"_", "\ul_\ul0 ");
   put _infile_;
end;
```

Through the above code, the “____” is now converted to “\ul____\ul0”.

The following is the converted RTF file based on the above approach. We can notice that the underline can now be displayed when opened with MS Word.

```
Listing of all Adverse Events leading to deaths that occurred during the study
Safety Analysis Set
XXX-XX-XXXX Study Parts A (Weeks 0-12) and B (Weeks 12-24)

_________________________________________________________________________________________

Treatment          Center/    MedDRA System Organ Class/  Start Date/  Stop Date/
Part A /           Patient    MedDRA Preferred Term/      [Study Day]/ [Study Day]/
Treatment          Number     Investigator's Term         Visit        Visit
Part B

_________________________________________________________________________________________
No cases to report
```

CONCLUSION

In this paper, an efficient approach is proposed to convert multiple text files into RTF files automatically. The approach is originally developed to run in SDD. It, however, can also be directly applied to other SAS development environment. In addition, an innovative and convenient method is proposed to solve the issue of unseen underline character in the converted RTF file when opened with MS Word. The solution can be directly applied to all the situations without adjustment.

REFERENCES

[1] SUGI 31. Converting Multiple SAS Output Files to Rich Text Format Automatically without Using ODS, Paper 133-31. Ling Y. Chen.

[2] PharmaSUG 2010. Making an RTF file Out of a Text File, With SAS, Paper CC13. David Franklin.

CONTACT INFORMATION

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

Name: Xingxing Wu
Enterprise: inVentiv Health Clinical
Address: 6405 70th Street
City, State ZIP: Middle Village, NY 11379
Work Phone: 212-561-5205
E-mail: Xingxing.wu@inventivhealth.com

Name: Jyoti N. Rayamajhi
Enterprise: Eli Lilly and Company
Address: Drop Code 1534 MC-88-03
City, State ZIP: Indianapolis, IN 46285
Work Phone: 317-433-3699
E-mail: rayamajhi_jyoti@lilly.com