PharmaSUG China 2024 - Paper PO-10032

A Method to Compare the RTF Files in Batch Mode and Generate a Checklist and Tracked RTF Files as Comparison Outputs

Sachiel Li, Xinran Tang, InnoCare Pharma Limited

ABSTRACT

The automated tool for comparing RTF-format TFL outputs in batch mode using VBScript codes in SAS macro in Window system aims to detect discrepancies among TFL outputs generated from different batch of statistical analysis and placed in two separate file folders. The main objective is to generate a comparison checklist and tracked TFLs to easily find out the changes and updates in TFLs which will facilitate the TFL reviewing by statisticians and report drafting by writers. The checklist in spreadsheet is the summary of comparison results which includes the list of TFL types (Table, Figure or Listing), TFL name and title, change type (updated, removed, or newly added), and flags indicating discrepancies placed in the body texts or in title and footnote. Simultaneously, any updated TFLs will undergo a comparison process using Microsoft Office Word's functionality, resulting in the creation of folder with tracked RTF files.

For comparison checklist, in cases any disparity exists between TFL outputs with the same file name in the two folders, the Title column will be visually highlighted using color markers. Additionally, hyperlinks will be provided to facilitate direct access to the corresponding RTF-format tracked version of the TFL output.

This automated tool can streamline the comparison process, eliminate the manual effort and visual comparison with naked eyes. By utilizing VBScript codes in SAS, the macro ensures efficiency and accuracy when reviewing the TFL outputs. The comparison checklist and RTF-format tracked files are efficient tools for statisticians, programmers, medical staff, and writer who need review TFLs, identify the updates in statistical outputs and assess the impact of changes in the outputs.

INTRODUCTION

For a long time, statisticians and SAS programmers have been facing a challenging problem—how to identify if there are any changes in the TFL (Tables, Figures, and Listings) outputs when the same raw data undergoes two different runs. It becomes crucial to identify whether differences exist and pinpoint the specific areas where these differences occur. Conducting a complete re-audit can be time-consuming, while not re-auditing raises concerns about potential disparities between the two sets of TFL outputs. This issue is particularly prominent for pharmaceutical companies, especially when providing feedback to Contract Research Organizations (CROs) that subsequently re-run the programs to update TFL outputs. Having a tool that can directly detect differences between two runs of TFL results and track the exact locations of these differences would be incredibly practical. Such a tool would enable efficient and intuitive assessment of whether disparities exist between two sets of TFL results.

Moreover, this tool would be beneficial for statistical and medical oversight. For example, it would facilitate the examination of cumulative differences in TFL outputs based on different raw data, specifically in the context of Adverse Events (AE) tables. By reviewing the tracked version of RTF reports, it becomes easier to observe the additions of adverse events.

The general idea of this tool is as follows: firstly, it reads and stores the filenames of all RTF documents in two specified folders into a SAS dataset. Secondly, using VBS (Visual Basic Scripting) language, it iteratively converts these RTF documents into plain text files (TXT) by separating them into cells and rows. Then, SAS reads these TXT files and utilizes the PROC COMPARE statement to compare and output the differences. Lastly, it selects the tables with differences and employs Microsoft Office Word's “COMPARE” function to conduct a detailed comparison, resulting in the creation of RTF documents with tracked changes, documenting the disparities between the two sets of TFL outputs. This entire process can be automated by implementing a SAS macro.

READING AND OUTPUTTING ALL RTF FILE NAMES FROM TWO FOLDERS INTO SAS DATASET

The below SAS macro enables the reading of all RTF files within a specified folder and generates a SAS dataset containing the names of all the RTF files within that folder. The macro includes three parameters: “doc_path” for specifying the folder path, “doc_type” for specifying the file type (with a default value of RTF), and “out” for specifying the name of the output SAS dataset.

```
%macro list_files(doc_path=, doc_type=RTF, out=);
  %local filrf rc did name i;
  %let rc=%sysfunc(filename(filrf,&doc_path));
  %let did=%sysfunc(dopen(&filrf));
  data &out.;
    length Tflname $200 type $10.;
    Tflname='';
    ord=.;
    %do i = 1 %to %sysfunc(dnum(&did));
      %let name=%qsysfunc(dread(&did,&i));
      %if %qupcase(%qscan(&name,-1,.)) = %upcase(&doc_type.) %then %do;
        Tflname="&name.";
        if prxmatch('#^(t)(_|-)(.*)#i',Tflname) then do;
          ord=1; type='Table'; end;
        else if prxmatch('#^(f)(_|-)(.*)#i',Tflname) then do;
          ord=2; type='Figure'; end;
        else if prxmatch('#^(l)(_|-)(.*)#i',Tflname) then do;
          ord=3; type='Listing'; end;
        output;
      %end;
    %end;
    keep ord type Tflname;
    proc sort;
      by ord Tflname;
  run;
  %global &out._nobs;
  %let &out._nobs=&sysnobs.;
  %let rc=%sysfunc(dclose(&did));
  %let rc=%sysfunc(filename(filrf));
%mend list_files;

%list_files(doc_path=&file_old., doc_type=&doc_type., out=file_old);
%list_files(doc_path=&file_new., doc_type=&doc_type., out=file_new);
```

Upon executing the aforementioned program, two SAS datasets named “file_old.sas7data” and “file_new.sas7data” are generated. These datasets list the names of RTF files in the respective compared folders. To facilitate subsequent utilization of VBScript with customizable full path names, an additional preprocessing step can be performed. In the resulting dataset “file_2”, five new variables are incorporated, each encompassing the complete path name information. The variables “File_old” and “File_new” denote the absolute paths of the RTF files to be compared. The variables “File_old_txt” and “File_new_txt” indicate the locations of the corresponding TXT files after converting the RTF files into TXT format. The variable “Compare_output” defines the pathway for producing the tracker version of the RTF file. It is important to note that I have designated a temporary SAS path for storing the generated TXT files. This measure is implemented to mitigate potential conflicts arising from concurrent access to intermediate files by other server users.

```
%let workpath = %sysfunc(getoption(work));
proc sql;
  create table file_1 as select coalesce(a.ord, b.ord) as ord,
    coalescec(a.type, b.type) as type, a.Tflname as Tflname_old,
    b.Tflname as Tflname_new
  from file_old a full join file_new b on a.ord=b.ord and a.type=b.type
```

```
    and a.Tflname=b.Tflname order by ord;
quit;

data file_2;
  set file_1;
  id=_n_;
  Tflname=coalescec(Tflname_new,Tflname_old);
  if Tflname_old>'' then do;
    File_old=strip("&file_old.")||'\'||Tflname_old;
    File_old_txt=strip("&workpath.")||'\'||
      prxchange("s#(.+)(\.&doc_type)#$1_old.txt#i",-1,Tflname_old);
  end;
  if Tflname_new>'' then do;
    File_new=strip("&file_new.")||'\'||Tflname_new;
    File_new_txt=strip("&workpath.")||'\'||
      prxchange("s#(.+)(\.&doc_type)#$1_new.txt#i",-1,Tflname_new);
  end;
  if Tflname_old>'' and Tflname_new>'' then
    Compare_output=strip("&Compare_output.")||'\'||
    prxchange("s#(.+)(\.&doc_type)#$1_tracker.&doc_type#i",-1,Tflname_new);
run;
```

COMPARING TWO SETS OF RTF DOCUMENTS AND GENERATING A CHECKLIST

The general procedure entails converting RTF files to plain text (TXT) using VBS, followed by reading the TXT files into SAS as datasets. At this stage, SAS functions and Perl regular expressions are employed to identify titles, footnotes, and table content within the RTF files. Additionally, the relevant titles are extracted. Finally, the PROC COMPARE procedure is utilized to compare their corresponding SAS datasets.

CONVERTING RTF TO PLAIN TXT USING VBS, IDENTIFYING TITLES AND FOOTNOTES, AND EXTRACTING CORRESPONDING TITLES

The following macro explains the process of converting RTF files to TXT format using VBS and subsequently reading them into SAS as datasets. It is important to note that different companies may have varying TFL shell templates, and certain modifications might be necessary to accommodate specific logic within the macro. However, the underlying principles remain similar, focusing on using keywords to identify titles and footnotes.

For example, our company has established specific guidelines for the output template of statistical tables. The title is presented in two lines, with the first line starting with “Table 14.x.x” followed by the title itself. The second line contains the analysis population enclosed in parentheses. The table's main body follows these lines, and footnotes are placed at the bottom of each page, beginning with “Note:”. By leveraging these keywords and considering the layout, we can utilize Perl regular expressions to differentiate and identify the relevant components.

It is widely known that the SAS program's execution time and the raw data between two batch runs may vary. To avoid comparing these textual information, I have incorporated the macro variable “&remove_wording” during the data step, allowing for the exclusion of non-comparable information when generating the “txt_&name.&i” dataset. Additionally, the macro variable “title&name.&i” contains the extracted title wordings from each table.

```
%macro read_rtf(name=);
  data _null_;
    file "&workpath\rtf_to_txt.vbs";
    put 'Set objWord = CreateObject("Word.Application")';
    put 'objWord.Visible = False';

    put 'Set objDoc = objWord.Documents.Open("' "&&&name."  '")';
```

```
  put 'objDoc.SaveAs "' "&&File_&name._txt."  '", 2, False, "", True, "
    ", False, False, False, False, False, 65001, True, wdCRLF, 0';
  put 'objDoc.Close';
  put 'objWord.Quit';
  put 'Set objDoc = Nothing';
  put 'Set objWord = Nothing';
run;

filename vbs pipe "cscript ""&workpath\rtf_to_txt.vbs""" console = min;

data _null_;
  infile vbs;
  input;
  stop;
run;

data txt_&name._&i.;
  infile "&&File_&name._txt." length=length filename=filename
  encoding="utf-8";
  input;
  id=_n_;
  string=strip(_infile_);
  if string='' then line_break=1;
  retain title_flag1 title_flag2;
  if prxmatch('#^(Table ?\d+\.*|Figure ?\d+\.*|Listing ?\d+\.*)#i',
  string) then do;
    title_flag1+1;title_flag2+1;end;
  if line_break=1 then title_flag1=.;
  if title_flag1=1 then title_flag=title_flag2;
  retain footnote_flag1 footnote_flag2;
  if prxmatch('#^(Note:).*#i',string) then do;
    footnote_flag1+1; footnote_flag2+1;end;
  if line_break=1 then footnote_flag1=.;
  if footnote_flag1=1 then footnote_flag=footnote_flag2;
  if prxmatch("#^(&remove_wording.).*#i",string) then delete;
  keep id string title_flag footnote_flag;
run;

data title_&name._&i.;
  set txt_&name._&i.;
  where title_flag>.;
  by title_flag id;
  retain del;
  if prxmatch('#^(\(|（|<).*(\)|）|>)\s*$#i',string) then del=id;
  if first.title_flag then del=.;
  if id>del>. then delete;
run;

data title_&name._&i.;
  set title_&name._&i.;
  by title_flag id;
  retain title;
  length title $1000;
  if first.title_flag then title=string;
  else title = cats(title, string);
  if last.title_flag;
run;
```

```
  %let title_&name.&i.=;
  proc sql noprint;
    select title into: title_&name.&i. from title_&name._&i.;
  quit;
%mend read_rtf;
```

COMPARING DIFFERENCES BETWEEN TWO RTF FILES USING PROC COMPARE

The following code describes the process of sequentially numbering the records in file_2 and creating a new dataset called file_3. Next, the system macro variable "&sysnobs" represents the number of observations in file_3. Then, using a do loop in SAS, each record in the dataset file_3 is passed into the macro "%read_rtf", which allows for the conversion of each RTF file into a TXT format. The SAS PROC COMPARE function is then used to compare the title/footnote and table body of each document separately. The differences found during the comparison are outputted to the macro variable "&sysinfo". Since executing the do loop repeatedly can result in the creation of multiple SAS datasets and consume a significant amount of computer memory, the SAS datasets generated in the previous loop iteration are deleted after each iteration.

```
data file_3;
  set file_2;
  i=_n_;
run;

%let n_obs=&sysnobs.;
%put &n_obs.;

%do i=1 %to &n_obs.;
  %if &i le &n_obs. %then %do;
    data _null_;
      set file_3;
      where i=&i.;
      call symput('old', strip(file_old));
      call symput('new', strip(file_new));
      call symput('File_old_txt', strip(File_old_txt));
      call symput('File_new_txt', strip(File_new_txt));
    run;

    %global title_old&i. title_new&i. compare_body&i. compare_headfoot&i.;

    %if "&old." ne "" %then %do; %read_rtf(name=old); %end; %else %do; %let
title_old&i.=%nrstr();; %end;
    %if "&new." ne""  %then %do; %read_rtf(name=new); %end; %else %do; %let
title_new&i.=%nrstr();; %end;

    %if ("&old." ne  "") and ("&new." ne  "") %then %do;
      proc compare base=txt_old_&i.(keep=string title_flag footnote_flag
where=(string>'' and title_flag=. and footnote_flag=.))
        compare=txt_new_&i.(keep=string title_flag footnote_flag
where=(string>'' and title_flag=. and footnote_flag=.)) criteria=0.00000001
noprint;
      run;
      %let compare_body&i.=&sysinfo;

      proc compare base=txt_old_&i.(keep=string title_flag footnote_flag
where=(string>'' and (title_flag>. or footnote_flag>.)))
```

```
        compare=txt_new_&i.(keep=string title_flag footnote_flag
where=(string>'' and (title_flag>. or footnote_flag>.)))
criteria=0.00000001 noprint;
      run;
      %let compare_headfoot&i.=&sysinfo;
    %end;
    %else %do;
      %let compare_body&i.=.;
      %let compare_headfoot&i.=.;
    %end;
    %if &i ge 2 %then %do;
      %if %sysfunc(exist(work.txt_old_%eval(&i.-1))) %then %do;
        proc delete data=work.txt_old_%eval(&i.-1); run; %end;
      %if %sysfunc(exist(work.txt_new_%eval(&i.-1))) %then %do;
        proc delete data=work.txt_new_%eval(&i.-1); run; %end;
      %if %sysfunc(exist(work.title_old_%eval(&i.-1))) %then %do;
        proc delete data=work.title_old_%eval(&i.-1); run; %end;
      %if %sysfunc(exist(work.title_new_%eval(&i.-1))) %then %do;
        proc delete data=work.title_new_%eval(&i.-1); run; %end;
    %end;
  %end;
%end;
```

GENERATING EXCEL SPREADSHEET OF CHECKLIST

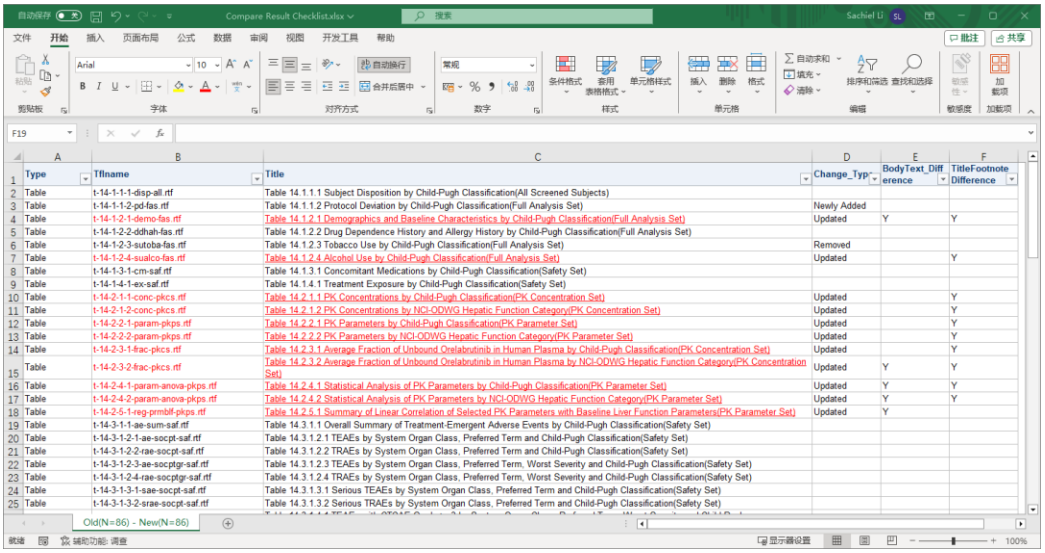
Assigning the extracted title text information (macro variable “&&title_old&i” representing the title of the 1st batch run RTF, and macro variable “&&title_new&i” representing the title of the 2nd batch run RTF) and the comparison results (macro variable “&&compare_body&i” representing the differences in the Table Body, and macro variable “&&compare_headfoot&i” representing the differences in the Title/Footnote) to the dataset file_4. Then, creating and retaining the necessary variables and information for the output checklist in the dataset toc_output. The code '~S={color=red}' will mark the “Tflname” variable with red font color. The “Change_Type” variable indicates the differences between the RTF files in the two folders. “Update” indicates that changes exist in the new RTF file of 2nd batch run comparing to the corresponding output in 1st batch run, “Removed” indicates that the RTF file is included in the 1st batch run but not in the 2nd batch run, and “Newly Added” indicates the opposite, i.e., the RTF file is included in the 2nd batch run but not in the 1st batch run. Additionally, when the value of variable “BodyText_Difference” is Y, it indicates changes exist in the table body, and when the value of variable “TitleFootnote_Difference” is Y, it indicates changes exist in the title or footnote. Then, using ODS TAGSETS.EXCELXP, exporting the toc_output dataset as an XML file with hyperlinks. At this point, we prefer to display the data in Excel format. To make the Title column more noticeable, we mark the fonts in red with the underline. To achieve this, we utilize VBScript to directly convert the XML file to an XLSX file, thus achieving this functionality.

```
data file_4;
  set file_3;
  length Title_old Title_new $1000.;
  %do i=1 %to &n_obs.;
    if i=&i. then do;
      Title_old="&&title_old&i.";
      Title_new="&&title_new&i.";
      BodyText_difference_N=&&compare_body&i.;
      TitleFootnote_difference_N=&&compare_headfoot&i.;
    end;
  %end;

  if BodyText_difference_N>0 then BodyText_difference='Y';
  if TitleFootnote_difference_N>0 then TitleFootnote_difference='Y';
run;
```

```
data toc_output;
  retain Type Tflname Title Change_Type BodyText_Difference
TitleFootnote_Difference;
  set file_4;
  Tflname_ord=Tflname;
  length Title $1000.;
  Title=coalescec(Title_new, Title_old);
  if Compare_output>'' and (BodyText_difference_N>0 or
TitleFootnote_difference_N>0) then do;

Title='=HYPERLINK("'||strip(prxchange("s#(.+)(\.&doc_type)#$1_tracker.&doc_
type#i",-1,Tflname))||'","'||strip(Title)||'")';
    Tflname='~S={color=red}'||Tflname;
  end;

  length Change_Type $20;
  if Tflname_old>"" and Tflname_new="" then Change_Type="Removed";
  else if Tflname_old="" and Tflname_new>"" then Change_Type="Newly Added";
  else if Compare_output>'' and (BodyText_difference_N>0 or
TitleFootnote_difference_N>0) then Change_Type="Updated";
  else Change_Type='';


  keep ord Type Tflname_ord Tflname Title Change_Type BodyText_Difference
TitleFootnote_Difference;
  proc sort;
    by ord Tflname_ord;
run;

ods results off;
ods listing close;
ods tagsets.excelxp file="&workpath.\Compare Result Checklist.xml"
style=excel
          options(
              autofilter='all'
              row_repeat='header'
              fittopage='yes'
              pages_fitwidth='1'
              pages_fitheight='100'
              sheet_name="Old(N=&file_old_nobs.) - New(N=&file_new_nobs.)"
              absolute_column_width="10,25,80,10,10,10"
              autofit_height='yes'
              sheet_interval='none'
              gridlines='yes'
              auto_subtotals='off'
               );

proc print data=toc_output noobs;
  var Type Tflname Title Change_Type BodyText_Difference
TitleFootnote_Difference;
run;

ods tagsets.excelxp close;
ods listing;
ods results on;
```

```
%**Add red color and underline on title;
data _null_;
  file "&workpath\save_as_xlsx.vbs";
  put 'Set objExcel = CreateObject("Excel.Application")';
  put 'objExcel.Visible = True';

  put 'Set objWorkbook = objExcel.Workbooks.Open("'    "&workpath.\Compare
Result Checklist.xml"      '")';
  put 'Set objWorksheet = objWorkbook.Worksheets(1)';

  put 'intLastRow = objWorksheet.Cells(objWorksheet.Rows.Count, 4).End(-
4162).Row ';

  put 'For i = 1 To intLastRow';
  put '    If objWorksheet.Cells(i, 4).Value = "Updated" Then';
  put '        objWorksheet.Cells(i, 3).Font.Color = RGB(255, 0, 0) ';
  put '        objWorksheet.Cells(i, 3).Font.Underline = True ';
  put '    End If';
  put 'Next';
  put 'newFilePath = "' "&Compare_output.\Compare Result Checklist.xlsx"
'"';
  put 'objWorkbook.SaveAs newFilePath, 51';
  put 'objWorkbook.Close';
  put 'objExcel.Quit';
run;

filename vbs pipe "cscript ""&workpath\save_as_xlsx.vbs""" console = min;

data _null_;
  infile vbs;
  input;
  stop;
run;
```

Display 1 is an example to display the checklist. We can see that the Tflname and Title columns of updated TFLs are marked in red with the underline. It is easy for people to directly click the hyperlink on the Title column to bring up the corresponding TFL with tracked changes in **Display 2**.



	Type	Tflname	Title	Change_Type	BodyText_Difference	TitleFootnote Difference
2	Table	t-14-1-1-1-disp-all.rtf	Table 14.1.1.1 Subject Disposition by Child-Pugh Classification(All Screened Subjects)			
3	Table	t-14-1-1-2-pd-fas.rtf	Table 14.1.1.2 Protocol Deviation by Child-Pugh Classification(Full Analysis Set)	Newly Added		
4	Table	t-14-1-2-1-demo-fas.rtf	Table 14.1.2.1 Demographics and Baseline Characteristics by Child-Pugh Classification(Full Analysis Set)	Updated	Y	Y
5	Table	t-14-1-2-2-ddhah-fas.rtf	Table 14.1.2.2 Drug Dependence History and Allergy History by Child-Pugh Classification(Full Analysis Set)			
6	Table	t-14-1-2-3-sutoba-fas.rtf	Table 14.1.2.3 Tobacco Use by Child-Pugh Classification(Full Analysis Set)	Removed		
7	Table	t-14-1-2-4-sualco-fas.rtf	Table 14.1.2.4 Alcohol Use by Child-Pugh Classification(Full Analysis Set)	Updated		Y
8	Table	t-14-1-3-1-cm-saf.rtf	Table 14.1.3.1 Concomitant Medications by Child-Pugh Classification(Safety Set)			
9	Table	t-14-1-4-1-ex-saf.rtf	Table 14.1.4.1 Treatment Exposure by Child-Pugh Classification(Safety Set)			
10	Table	t-14-2-1-1-conc-pkcs.rtf	Table 14.2.1.1 PK Concentrations by Child-Pugh Classification(PK Concentration Set)	Updated		Y
11	Table	t-14-2-1-2-conc-pkcs.rtf	Table 14.2.1.2 PK Concentrations by NCI-ODWG Hepatic Function Category(PK Concentration Set)	Updated		Y
12	Table	t-14-2-2-1-param-pkps.rtf	Table 14.2.2.1 PK Parameters by Child-Pugh Classification(PK Parameter Set)	Updated		Y
13	Table	t-14-2-2-2-param-pkps.rtf	Table 14.2.2.2 PK Parameters by NCI-ODWG Hepatic Function Category(PK Parameter Set)	Updated		Y
14	Table	t-14-2-3-1-frac-pkcs.rtf	Table 14.2.3.1 Average Fraction of Unbound Orelabrutinib in Human Plasma by Child-Pugh Classification(PK Concentration Set)	Updated		Y
15	Table	t-14-2-3-2-frac-pkcs.rtf	Table 14.2.3.2 Average Fraction of Unbound Orelabrutinib in Human Plasma by NCI-ODWG Hepatic Function Category(PK Concentration Set)	Updated	Y	Y
16	Table	t-14-2-4-1-param-anova-pkps.rtf	Table 14.2.4.1 Statistical Analysis of PK Parameters by Child-Pugh Classification(PK Parameter Set)	Updated	Y	Y
17	Table	t-14-2-4-2-param-anova-pkps.rtf	Table 14.2.4.2 Statistical Analysis of PK Parameters by NCI-ODWG Hepatic Function Category(PK Parameter Set)	Updated	Y	Y
18	Table	t-14-2-5-1-reg-prmbif-pkps.rtf	Table 14.2.5.1 Summary of Linear Correlation of Selected PK Parameters with Baseline Liver Function Parameters(PK Parameter Set)	Updated	Y	
19	Table	t-14-3-1-1-ae-sum-saf.rtf	Table 14.3.1.1 Overall Summary of Treatment-Emergent Adverse Events by Child-Pugh Classification(Safety Set)			
20	Table	t-14-3-1-2-1-ae-socpt-saf.rtf	Table 14.3.1.2.1 TEAEs by System Organ Class, Preferred Term and Child-Pugh Classification(Safety Set)			
21	Table	t-14-3-1-2-2-rae-socpt-saf.rtf	Table 14.3.1.2.2 TRAEs by System Organ Class, Preferred Term and Child-Pugh Classification(Safety Set)			
22	Table	t-14-3-1-2-3-ae-socptgr-saf.rtf	Table 14.3.1.2.3 TEAEs by System Organ Class, Preferred Term, Worst Severity and Child-Pugh Classification(Safety Set)			
23	Table	t-14-3-1-2-4-rae-socptgr-saf.rtf	Table 14.3.1.2.4 TRAEs by System Organ Class, Preferred Term, Worst Severity and Child-Pugh Classification(Safety Set)			
24	Table	t-14-3-1-3-1-sae-socpt-saf.rtf	Table 14.3.1.3.1 Serious TEAEs by System Organ Class, Preferred Term and Child-Pugh Classification(Safety Set)			
25	Table	t-14-3-1-3-2-srae-socpt-saf.rtf	Table 14.3.1.3.2 Serious TRAEs by System Organ Class, Preferred Term and Child-Pugh Classification(Safety Set)			

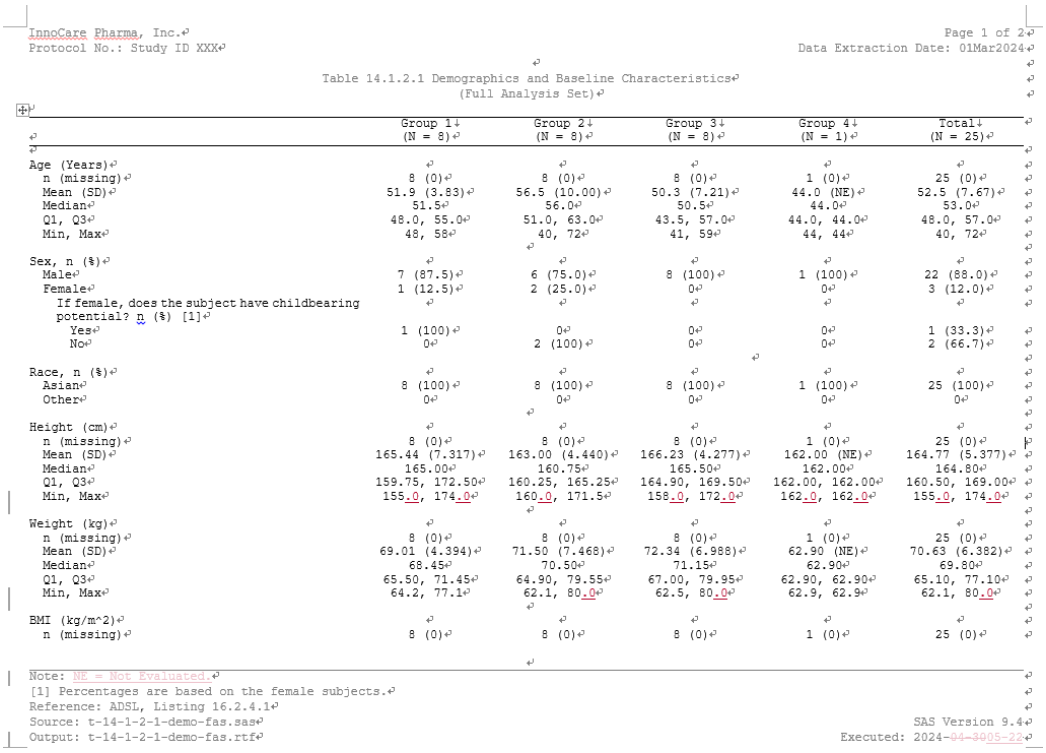
Display 1.An Example of Checklist Styles

COMPARING RTF FILES WITH WORD'S COMPARE FUNCTION FOR DIFFERENCES

We can observe that from the dataset file_4, we select the RTF files with differences in BodyText or TitleFootnote to generate a new dataset called file_5. Finally, we utilize VBScript to instruct Word's Compare function to perform individual comparisons and output a tracked version of the RTF files.

```
data file_5;
    set file_4;
    where Compare_output>'' and (BodyText_difference_N>0 or
TitleFootnote_difference_N>0);
    j=_n_;
  run;

  %let compare_n=&sysnobs.;
  %put &compare_n.;

%do j=1 %to &compare_n.;
  %if &j le &compare_n. %then %do;
    data _null_;
      set file_5;
      where j=&j.;
      call symput('old', strip(file_old));
      call symput('new', strip(file_new));
      call symput('Compare_Result', strip(Compare_output));
    run;

    data _null_;
      file "&workpath\compare_word.vbs";
      put 'Dim objWord, objDoc1, objDoc2, objResultDoc';
      put 'Set objWord = CreateObject("Word.Application")';
      put 'objWord.Visible = False';
      put 'Set objDoc1 = objWord.Documents.Open("'   "&old."   '")';
      put 'Set objDoc2 = objWord.Documents.Open("'   "&new."   '")';
      put 'objWord.CompareDocuments objDoc1, objDoc2, 0, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1';
      put 'Set objResultDoc = objWord.ActiveDocument';
      put 'objResultDoc.SaveAs2 "'    "&Compare_Result."    '", '
"&doc_type_param.";
      put 'objDoc1.Close False';
      put 'objDoc2.Close False';
      put 'objWord.Quit';
      put 'Set objDoc1 = Nothing';
      put 'Set objDoc2 = Nothing';
      put 'Set objResultDoc = Nothing';
      put 'Set objWord = Nothing';
    run;

    filename vbs pipe "cscript ""&workpath\compare_word.vbs""" console =
min;

    data _null_;
      infile vbs;
      input;
      stop;
    run;
  %end;
%end;
```

Display 2 presents a Tracked version of an RTF file. We can clearly observe that the decimal precision in the "Min, Max" row of the "Height (kg)" statistic has been modified. An additional zero has been added after the decimal point, preserving one decimal place. Additionally, we can see that a new footnote has been added, stating "NE = Not Evaluated". Please note that the data shown in the image is a dummy result and does not represent the actual clinical data of any specific project.



InnoCare Pharma, Inc. — Page 1 of 2
Protocol No.: Study ID XXX — Data Extraction Date: 01Mar2024

Table 14.1.2.1 Demographics and Baseline Characteristics
(Full Analysis Set)

	Group 1 (N = 8)	Group 2 (N = 8)	Group 3 (N = 8)	Group 4 (N = 1)	Total (N = 25)
Age (Years)					
n (missing)	8 (0)	8 (0)	8 (0)	1 (0)	25 (0)
Mean (SD)	51.9 (3.83)	56.5 (10.00)	50.3 (7.21)	44.0 (NE)	52.5 (7.67)
Median	51.5	56.0	50.5	44.0	53.0
Q1, Q3	48.0, 55.0	51.0, 63.0	43.5, 57.0	44.0, 44.0	48.0, 57.0
Min, Max	48, 58	40, 72	41, 59	44, 44	40, 72
Sex, n (%)					
Male	7 (87.5)	6 (75.0)	8 (100)	1 (100)	22 (88.0)
Female	1 (12.5)	2 (25.0)	0	0	3 (12.0)
If female, does the subject have childbearing potential? n (%) [1]					
Yes	1 (100)	0	0	0	1 (33.3)
No	0	2 (100)	0	0	2 (66.7)
Race, n (%)					
Asian	8 (100)	8 (100)	8 (100)	1 (100)	25 (100)
Other	0	0	0	0	0
Height (cm)					
n (missing)	8 (0)	8 (0)	8 (0)	1 (0)	25 (0)
Mean (SD)	165.44 (7.317)	163.00 (4.440)	166.23 (4.277)	162.00 (NE)	164.77 (5.377)
Median	165.00	160.75	165.50	162.00	164.80
Q1, Q3	159.75, 172.50	160.25, 165.25	164.90, 169.50	162.00, 162.00	160.50, 169.00
Min, Max	155.0, 174.0	160.0, 171.5	158.0, 172.0	162.0, 162.0	155.0, 174.0
Weight (kg)					
n (missing)	8 (0)	8 (0)	8 (0)	1 (0)	25 (0)
Mean (SD)	69.01 (4.394)	71.50 (7.468)	72.34 (6.988)	62.90 (NE)	70.63 (6.382)
Median	68.45	70.50	71.15	62.90	69.80
Q1, Q3	65.50, 71.45	64.90, 79.55	67.00, 79.95	62.90, 62.90	65.10, 77.10
Min, Max	64.2, 77.1	62.1, 80.0	62.5, 80.0	62.9, 62.9	62.1, 80.0
BMI (kg/m^2)					
n (missing)	8 (0)	8 (0)	8 (0)	1 (0)	25 (0)

Note: NE = Not Evaluated.
[1] Percentages are based on the female subjects.
Reference: ADSL, Listing 16.2.4.1
Source: t-14-1-2-1-demo-fas.sas — SAS Version 9.4
Output: t-14-1-2-1-demo-fas.rtf — Executed: 2024-04-30 05:22

Display 2. An Example of Tracked RTF

MACRO PARAMETER CONFIGURATION

The following code is an example illustrating how the entire macro is used. The whole of codes above is wrapped within the SAS macro compare_rtf, which includes five macro parameters. The parameter “doc_type” has a default value of "rtf". The parameter “file_old” assigns the path of the TFL output from the 1st batch run, while the parameter “file_new” assigns the path of the TFL output from the 2nd batch run. The parameter “Compare_output” assigns the output path for the checklist and tracked RTF files.

It is important to emphasize the macro parameter “Remove_wording”, which lists the cell contents in the table that should not be compared. The prefixes of these contents are separated by '|' as a delimiter. For example, the SAS program execution time (Executed:) and the raw data extraction date (Data Extraction Date:) between the two batch runs will not be included in the comparison. The changes in “program executed date” and “data extraction date” will not be identified and tracked.

```
%compare_rtf(
  doc_type=rtf,
  file_old=mypath\dryrun1,
  file_new=mypath\dryrun2,
  Compare_output=mypath\out,
  Remove_wording=%str(Executed:|Data Extraction Date:)
    );
```

CONCLUSION

Overall, the automated tool presented in this paper can significantly improve the efficiency and accuracy of comparing TFL outputs in batch mode. It can eliminate the laborious manual comparison and reduce the risk of oversight or human error. The comparison checklist and RTF-format tracked file are valuable efficient tools for stakeholders involved in reviewing TFL outputs, and can improve the reliability and quality of statistical analysis and report writing in clinical trials. However, this tool is not able to identify and compare the graphics in Figure outputs. Manual review is still required to find out the changes in graphics.

CONTACT INFORMATION

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

<Name: Sachiel Li>
<Enterprise: InnoCare Pharma>
<E-mail: Sachiel.li@innocarepharma.com>