PhUSE 2015

Paper CC01

ODS DOCUMENT, a practical example

Ruurd Bennink, OCS Consulting B.V., ‘s-Hertogenbosch, the Netherlands

ABSTRACT

The ODS DOCUMENT destination (in short ODS DOCUMENT) is perhaps the most underutilized feature of ODS. Most likely this is due to the fact that ODS DOCUMENT does not create output that you can view directly as with ODS PDF and ODS HTML. This makes ODS DOCUMENT more abstract and may therefore give some threshold fear to SAS programmers.

INTRODUCTION

However, ODS DOCUMENT is a handy tool for the creation of a single PDF package of Tables/Listings/Figures (TLF), thereby offering attractive output e.g. with a Table of Contents (ToC) with links to the corresponding TLF package (bookmarks). Although ODS DOCUMENT requires sophisticated programming, once the facility for the creation of a single PDF package of TLF has been implemented correctly in each TLF program separately and the program to create the TLF set (to be called the replay program) is final, it can be used over and over again in other clinical studies, simply by copying the replay program to the new location and, of course, after updating this program in accordance with the new environment. Also make sure then that all TLF programs that produce output to be included in the set of TLF have the piece of SAS code included that creates the corresponding TLF as a binary output. This paper will show that a major part of this piece of SAS code does not need any updates when copied from one TLF program to the other. Of course, this is a great advantage

The purpose of this paper is first to explain the use of ODS DOCUMENT for the creation of a set of TLF in PDF format in more general terms (it would be far beyond the scope of this paper to explain all technical details in depth). This is achieved in three steps:

1) Explain the basics of ODS DOCUMENT.
2) Explain the necessary update of each separate TLF program that is required to be included in the PDF package.
3) Explain the structure of the replay program.

This is achieved as much as possible by showing (parts of) SAS programs.

1. ODS DOCUMENT BASICS

Using ODS DOCUMENT is not very different from the use of other ODS features as ODS PDF or ODS RTF, it uses the same programming technique of opening the DOCUMENT destination with the ODS statement and closing it with the ODS CLOSE statement. All of the DATA steps and procedures between those two statements will be stored in an ODS DOCUMENT. Below there is a simple example:

```
ODS PDF File='path\filename1.pdf';
ODS DOCUMENT Name=libname.mydocument(Write);

PROC MEANS Data=mydata;
   CLASS treatgrp;
   VAR gender;
RUN;
```

```
ODS DOCUMENT CLOSE;
ODS PDF CLOSE;
```

This example shows that the ODS DOCUMENT statement is very much like the ODS PDF statement. However, rather than giving a filename to the output, the name of an ODS DOCUMENT is given. This is because the ODS DOCUMENT output is in a binary format (a file with extension SAS7BITM) that is readable only for ODS DOCUMENT and PROC DOCUMENT. The created binary output can either be stored as a temporary file or as a permanent file. In this example the file is saved in the folder with library name LIBNAME.

By default, if the name of the output document already exists, the new output will be appended to the already existing output. Because this is often not what is intended by the programmer, the Write mode should be used to guarantee that existing output will be overwritten once ODS DOCUMENT runs again. The Write mode is 'switched on' by placing 'write' between brackets after the document name. As in the example above

```
ODS DOCUMENT Name=libname.mydocument(Write);
```

As mentioned earlier, the thus created output LIBNAME.MYDOCUMENT is not directly viewable. It is stored as a binary output. To visualize the output the DOCUMENT procedure is necessary:

```
ODS PDF File='path\replaymydocument.pdf' Style=journal;

PROC DOCUMENT Name=libname.mydocument;
   Replay;
RUN;

ODS PDF CLOSE;
```

The thus created PDF output will be identical to the directly created PDF output. So we may wonder what the advantage of ODS DOCUMENT is, as it seems now like a step too much. Of course, this example is very simple, but what if the output creation is a very time consuming process, like the creation of a set of TLF is? In that case, ODS DOCUMENT delivers the output much faster than the corresponding direct PDF output is created. Suppose that a stakeholder requests the same output, but with a different lay-out. Because all binary output is stored as permanent files, you simply have to re-run the replay program with a different style option for the ODS PDF statement:

```
ODS PDF File='path\replaymydocument.pdf' Style=analysis;

PROC DOCUMENT Name=libname.mydocument;
   Replay;
RUN;

ODS PDF CLOSE;
```

Although the original PDF output may be stored with a different style option, the newly created output has the requested 'Style=analysis'.

2. INDIVIDUAL TLF PROGRAMS

As mentioned in the previous section the first step is the creation of a binary file by each TLF program to be stored as a permanent file.

The next example shows the SAS program table_1.sas (in abbreviated form, because it just serves as an example of the ODS DOCUMENT destination) to achieve this. The table to create is Table 1 with demographic characteristics subject's gender and age by treatment group. The part in italic font is to be replaced as a next step. Text between <> is not SAS code, but indicates something more in general e.g. <font settings> indicates that several font options can be entered.

```
%INCLUDE 'path\overhead.sas'; /* Program with all libnames, options, formats,
macro runs etc. */

DATA dm;
   SET adam.dm; /* Permanent analysis dataset with demographics */

   KEEP gender age treatgrp;
RUN;

%tablemacr; /* Standard table macro to create the requested summary statistics
and frequencies */

ODS PDF File='path\table_1.pdf';
ODS DOCUMENT Name=libname.tab1(Write);

TITLE justify=l <font settings> "Table 1: Subject's age and gender";
FOOTNOTE1 justify=l <font settings> "Total = Sum of Test and Control.";
FOOTNOTE2 justify=l <font settings> "[N]: Number of subjects of the analysis
population.";
FOOTNOTE3 justify=l <font settings> "Denominator in percentage calculation is sum
of non-missing cases.";

FOOTNOTE9  Justify=l <font settings> "table_1.sas" justify=r "&sysdate.";
FOOTNOTE10 Justify=l <font settings> "&study." Justify=r "&pageno";

PROC REPORT …;
   COLUMN …;

   DEFINE …;
   DEFINE …;
   <etc.>
RUN;

ODS PDF CLOSE;
ODS DOCUMENT CLOSE;

/* EOF */
```

This program will work fine and will create binary output file TAB1.SAS7BITM as a permanent file in the folder with library name LIBNAME. The program also creates and stores table 1 as PDF output, to be viewed directly. However the disadvantage of this program is that the program name has to be typed in manually at several places in the program.
As a solution for this macro variables have to be created. To achieve this, the part in italic font should be replaced by the following piece of SAS code:

```
%LET prog_name = table_1;
%LET tname     = Table 1: Subject`s age and gender;
```

```
%LET procdoc   = tab1;
%LET footn     = %str(
FOOTNOTE1 justify=l <font settings> "* Total = Sum of Test and Control";
FOOTNOTE2 justify=l <font settings> "[N]: Number of subjects of the analysis
population.";
FOOTNOTE3 justify=l <font settings> "Denominator in percentage calculation is sum
of non-missing cases.";
);

ODS PDF File="path\&prog_name..pdf";
ODS DOCUMENT Name=libname.&procdoc.(write);

TITLE;
FOOTNOTE;

TITLE1 justify=l "&tname.";
&footn.; /* Optional FOOTNOTEs */

FOOTNOTE9 justify=l <font settings> "&prog_name..sas" justify=r "&sysdate.";
FOOTNOTE10 justify=l <font settings> "&study." justify=r &pageno.;
```

Because this paper will demonstrate that also the TITLE and name of the binary output file need to be used multiple times, it is better to use them as macro variables too.

If we are to create a second table ‘Table 2’, the above piece of SAS code needs to be included in the program ‘table_2.sas’ with the necessary updates to the macro variables &prog_name, &tname, &procdoc and &footn.

However, to make sure that the replay program uses TITLEs and FOOTNOTEs with identical font types, font heights, justifications etc. all program specific TITLEs and FOOTNOTEs should be included in the replay program. The challenge is how that should be managed. The following piece of SAS code (to be appended to the piece of SAS code immediately above this text) shows how that is solved:

```
PROC SQL;
   Create table &procdoc as
      Select *
           From dictionary.TITLEs;
QUIT;

DATA &procdoc;
   LENGTH progtext font $32;
   SET &procdoc;

   IF type = 'T' THEN font = <TITLE font settings>;
   IF type = 'F' THEN font = <FOOTNOTE font settings>;
   if type = 'F' and number = 9 then progtext = Scan(text,1);

RUN;

DATA _null_;
   SET &procdoc;

   File "%sysfunc(pathname(libname))\&procdoc..txt" Lrecl=300;

   IF type = 'T' THEN Put '%LET proclabel = ' text ';';
   IF type = 'F' AND number = 9 THEN Put '%LET prog_name = ' progtext ';';
   IF type = 'T' THEN Put 'TITLE' @;
   IF type = 'F' THEN Put 'FOOTNOTE' @;
```

```
                    Put number font ' "' text '";';

run;
```

Please note that this piece of SAS code does not need any updates when used in a different TLF program.

The PROC SQL statement creates a temporary SAS dataset named identical to the binary output file created by ODS DOCUMENT named &procdoc. This dataset has the following structure and contents:

TYPE	NUMBER	TEXT
T	1	Table 1: Subject`s age and gender
F	1	* Total = Sum of Test and Control
F	2	[N]: Number of subjects of the analysis population.
F	3	Denominator in percentage calculation is sum of non-missing cases.
F	4	
F	5	
F	6	
F	7	
F	8	
F	9	Table_1.sas<Date of output creation>
F	10	<Study name>Page ^{thispage}/^{lastpage}

Please note that FOOTNOTEs 9 and 10 are incorrect, but these FOOTNOTEs are not relevant, because they will be replaced by different FOOTNOTEs 9 and 10 in the replay program, because the set of TLF requires different standard FOOTNOTEs, as will be presented in section 3.

As a next step this SAS file is being used for the creation of a text file with name &procdoc.txt and is stored in the same folder as the binary ODS DOCUMENT output with identical name &procdoc. This text file is nothing more than SAS code with TITLE and FOOTNOTE statements and two %LET statements. One is to create the bookmark variable &proclabel, to be used by the Table of Contents (ToC) section of the replay program. Note that &proclabel is identical to &tname as it should be. The second %LET statement creates macro variable &prog_name to be used in the standard FOOTNOTE9 for the set of TLF.

3. THE REPLAY PROGRAM

Below an entire replay program is presented as an example that includes ten tables in the PDF set of TLF, named Table_1 to Table_10 with corresponding binary ODS DOCUMENT outputs tab1 to tab10. Explanations are given between /* … */ as SAS program comment and general text (to be filled in by the programmer) is placed between <>. Not every detail is explained. It would go far beyond the scope of this paper to explain all technical aspects in detail.

```
TITLE;
FOOTNOTE;

%INCLUDE 'path\Overhead.sas'; /* Program with all libnames, options, formats,
macro runs etc. */

%LET docsvar=
Tab1
Tab2
Tab3
Tab4
Tab5
```

```
Tab6
Tab7
Tab8
Tab9
Tab10
;

OPTIONS nodate nonumber;
/* Necessary options, because date and page number are to be included in a
standard way, defined by FOOTNOTEs 9 and 10 */

ODS PDF File="path\<milestone>_<study name>_&sysdate..pdf" style=journal;

%macro run_overall_op;

/* TABLE OF CONTENTS PART */

 ODS escapechar='^';
/* Text as TITLE for the ToC */
 ODS PDF startpage=now text = '^{style [color=black just=l font_face=Arial
font_size=11pt font_weight=bold ] <Milestone and Study name>}';

 ODS PDF text = ' ';
 ODS PDF text = ' ';

 ODS PDF text = '^{style [color=black just=l font_face=Arial font_size=10pt
font_weight=bold ] TABLE OF CONTENTS}';
 ODS PDF text = ' ';


 ODS DOCUMENT Name=work.contents(write);

 %LET i=1; /* Start with the first binary file */
 %LET doc=%scan(&docsvar,&i);

 %DO %while (&doc NE);

/* Define filename to include TITLEs, FOOTNOTEs, program name and label */
/* In the step below the text file with all TITLEs and FOOTNOTEs and two %LET */
/* statements is included. As explained in section 2 */
 FILENAME tfoot "%sysfunc(pathname(libname))\&doc..txt";
 %INCLUDE tfoot;

/* To include the correct FOOTNOTE9 and FOOTNOTE10 statements */
 TITLE;
 FOOTNOTE;

 FOOTNOTE9  <font settings> j=l "<Milestone – Study name>" j=r "Table of
contents";
 FOOTNOTE10 <font settings> j=l "<Study name>" j=c 'Page ^{thispage}' j=r
"&sysdate";

/* Select cut-off for TOC */
```

```
/* Suppose that two pages of ToC are required, the second page starting with
table 6 */
 %IF &doc=tab6 %THEN
  %DO;
   ODS PDF startpage=now text = ' ';
   ODS PDF text = "^{style [url='#%lowcase(&doc)' linkcolor=white color=blue
just=l   font_face=Arial cellheight=0.5cm font_size=8pt] &proclabel   }"; /* Set
bookmark */
  %END;
  %ELSE %DO;
   ODS PDF text = ' ';
   ODS PDF text = "^{style [url='#%lowcase(&doc)' linkcolor=white color=blue
just=l font_face=Arial cellheight=0.5cm font_size=8pt] &proclabel   }";

 %IF &doc=tab5 %THEN
 %DO;
  ODS PDF text = ' ';
  ODS PDF text = '^{style [color=black just=l font_face=Arial font_size=8pt
font_weight=bold ] Continued...}';
 %END;

 %END;

 %LET i=%eval(&i+1);
 %LET doc=%scan(&docsvar,&i); /* Continue with the next binary output file */

 %END;

 ODS DOCUMENT CLOSE;

 ODS PDF startpage=now;

 PROC DOCUMENT Name=work.contents;
   Replay / activefootn activeTITLE;
 QUIT;
/* Options ACTIVEFOOTN and ACTIVETITLE are necessary to make sure that the TITLEs
and FOOTNOTEs are used with correct settings as defined in this program */


/* TABLES AND LISTINGS PART */
 %LET i=1;
 %LET doc=%scan(&docsvar,&i);
 %DO %while (&doc NE);

 ODS PDF startpage=now; /* Start each table on a new page */
 ODS PDF anchor="%lowcase(&doc)"; /* To ensure a correct link to the specified
URL= in the TABLE OF CONTENTS PART */

 TITLE;
 FOOTNOTE;

/* Define filename to include TITLEs, FOOTNOTEs, program name and label */
/* In the step below the text file with all TITLEs and FOOTNOTEs and two %LET */
```

```
/* statements is included. As explained in section 2 */
 FILENAME tfoot "%sysfunc(pathname(libname))\&doc..txt";
 %INCLUDE tfoot;

 FOOTNOTE9 <font settings> j=l "<Milestone>" j=r "&prog_name..sas";
 FOOTNOTE10 <font settings> j=l "<Study name>" j=c 'Page ^{thispage}' j=r
"&sysdate";
/* These footnotes 9 and 10 will overwrite the incorrect footnotes in the
&doc..txt file */

 PROC DOCUMENT Name=libname.&doc;
    Replay / activefootn activeTITLE;
 QUIT;

 %LET i=%eval(&i+1);
 %LET doc=%scan(&docsvar,&i); /* Start with the next binary output file */

%END;

%mend run_overall_op;

%run_overall_op

ODS PDF CLOSE;

/* EOF */
```

CONCLUSION

There are quite some advantages to the use of ODS DOCUMENT as a tool to create a PDF set of TLF. Benefits are summarized below:

- Replaying requires minimal computer resources hence output is created very fast.
- The size measured in MB of the TLF package is much less compared to the sum of the sizes of each separate TLF output. This makes it attractive to be sent to stakeholders by e-mail.
- No other PDF software than SAS is necessary to create such a set of TLF.
- Attractive output with a ToC with links to the corresponding TLF in the set.
- Stakeholders (Study managers, Researchers, Medical Monitors etc.) are always pleased to receive one coherent set of TLF instead of all separate documents.

However, there are challenges:

- Lack of flexibility: a set of TLF can be in one PROC REPORT style only. Sometimes this can be inconvenient e.g. tables may have a preferred style that may conflict with the preferred style for graphs, so does ‘style=journal’ deliver nice tables, but colored graphs are then not possible. Therefore ‘Style=analysis’ is a better option. To avoid this, a separate set of graphs can be created, but this is a sub-optimal solution.
- The set can be created in one orientation only, landscape or portrait, while some tables may ask for a different orientation.
- To create a nice set of TLF with ODS DOCUMENT in RTF format is also possible. The set of tables is fine, but somehow the ToC is not satisfactorily included e.g. wrong TITLE, wrong page cut-off, no FOOTNOTEs.

Making a balance, as a final conclusion, I think the benefits of ODS DOCUMENT outweigh the doubts clearly.

CONTACT DETAILS

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

Ruurd Bennink
OCS Consulting B.V.
P.O. Box 3434
5203 DK ‘s-Hertogenbosch
The Netherlands
+31 (0)73 523 6000
sasquestion@ocs-consulting.com
http://www.ocs-consulting.com/nl