Paper 94-26

%ToC: A Macro for Generating Table of Contents from SAS Output

X. Hong Chen, McDougall Scientific Ltd., Toronto, Canada

ABSTRACT

In studies where a detailed display of data is required, such as in pharmaceutical research or clinical trial studies, reports will usually contain many statistical summary tables and data listings. Naturally, a table of contents is expected to accompany these tables and listings for easy reference. This was traditionally done either by typing/copying the titles and manually assigning the page number, or by embedding table of contents codes in a word processing package and then executing a built-in procedure within the package. However, both these methods are time consuming and error prone. To avoid this labor-intensive process a SAS macro, utilizing the SAS DATA STEP and REPORT procedure, was developed to automate the generation of a table of contents from SAS output and executed totally within SAS. The macro was designed to read in the title portion of each SAS output page, compute the page number, determine the first occurrence of a particular title, create “dot leaders” for the titles to simulate the flush right tab function found in most standard word processing packages, and finally output the complete table of contents for publication purpose.

INTRODUCTION

In pharmaceutical research and clinical trial studies, many study reports will contain a large number of statistical summary tables and data listings. Naturally, a table of contents is required to accompany these tables and listings for easy reference. Traditionally, this was done either by typing/copying the titles and manually assigning the page number or by embedding, in the report itself, table of contents codes within a word processing package. However, both these methods are time consuming and error prone. To make the matter worse, this labor-intensive process has to be repeated every time there was a change to the text of the title, its numbering, or its pagination, as a result of record addition or deletion. An alternative that could automate this process would avoid these problems and hence improve work efficiency and quality.

METHODS

In the production of study reports, statistical summary tables and data listings are programmed and executed individually to produce multiple tables and listings output files. The outputs are then appended to one another in some pre-determined logical (usually numerical) order, either by placing all programs sequentially within a batch file to create a single output file containing all tables or listings, or by appending each output into a single file using an operating system command (such as the UNIX *cat* command). The title information is displayed in the top portion of each SAS output page. To generate a table of contents, a logical approach would include the following steps: 1) read in the title portion from each page of the SAS output file and compute the page number; 2) determine and select the first occurrence of a particular title; 3) create “dot leaders” (a series of dots) for the titles to simulate the flush right tab function found in most word processing packages; and 4) output the complete table of contents.

1. READ IN THE TITLES AND COMPUTE PAGE NUMBER

Assume the file containing SAS output for all tables is called “table.lst” and the titles are created using the following SAS statements:

```
title1 “TABLE 1.1”;
title3 “Patient Demographics”;
```

To read in only the title portion of the output page, the logic would be to check if the first word of a line (observation) matches the word “TABLE” (using @ in the input statement to hold the data line). If the answer is yes, then input the table number, skip to the next line and input title3. Since a title appears on each page only once, each time a title is read in, page number will increase by 1. The following SAS code can be used to accomplish these tasks:

```
data in (keep=title1 title3 page);
  infile “table.lst”;
  length title1a number $10. title1 $20.;
  input +1 title1a @; * +1 to skip the page
break symbol at the first column;
  if title1a="TABLE" then do;
    input number / title3 $200.;  * skip to
next line to read title3;
    title1=compbl(title1a||number); *
combined for programming simplicity;
    page+1; * page number increased by 1;
    output;
  end;
```

2. DETERMINE THE FIRST OCCURRENCE OF A TITLE

Now that all titles from the output file have been read into a SAS dataset, the next task would be to identify and output the first occurrence of a title. The first occurrence of a title is selected by simply outputting the observations meeting the first.title1 criteria, after sorting the data by title1 and page.

```
proc sort data=in;
  by title1 page;

data in;
  set in;
  by title1 page;
  if first.title1;
```

3. ADD THE “DOT LEADERS” TO TITLES

The dot leaders should now be added to the end of each title, with the number of dots to be added dependant on the selected line size option for the resulting table of contents, the length of title3, the space needed for displaying the page number, as well as the space taken by the title1. Space taken by the title1 is the maximum length of title1 for all observations. This value is then assigned to a macro variable for later use.

```
data _null_;
  set in end=last;
  retain max;
  leng1=length(title1);
  if leng1>max then max=leng1;
  if last then call symput(“max”, put(max,
best.));
run;
```

The value for line size can be obtained from the GETOPTION system function %sysfunc(getoption(ls)), and the space available for title3 is calculated by subtracting the line size from the maximum space for title1 and allowing for 7 extra columns (2 for spacing and 5 for page number). If the length of title3 is greater than the space available for title3, then title3 will be split into two or more lines. The SAS code for adding the dot leaders would then be:

```
%let ls=%sysfunc(getoption(ls));
%let space=%eval(&ls-&max-7);

data in;
  set in;
  length title3n $200;
  leng3=length(title3);
  page_o=page; * for ordering purpose;
  if leng3>=&space then do;
    split=&space; * starting position to find
the splitting point(blank space);
    done=0;
    do while (done=0);
      if substr(title3,split,1)^='' then do;
* find the splitting point;
        split=split-1;
      end;
      else done=1; * splitting point found;
    end;
    line=1;
    title3n=substr(title3,1,split); * first
half of title3;
    page=.; * suppress page number for the
first line of the title;
    output;
    line=2; * next line for the remaining
title;
    page=page_o; * assign page number to the
second line;
    leng3r=length(substr(title3,split+1)); *
length of the remaining title3;
    dots=&space-leng3r; * number of dots
needed to be filled in;

title3n=compbl(substr(title3,split+1))||repea
t('.', dots);
  output;
  end;
  else do;
    dots=&space-leng3;
    title3n=compbl(title3)||repeat('.',
dots);
    line=1;
    output;
  end;
```

4. OUTPUT THE RESULTING TABLE OF CONTENTS

Outputting the resulting “Table of Contents” using the REPORT procedure is now quite straightforward, since all calculations and manipulations have already been completed. To distinguish the resulting table of contents from the normal SAS output, the PRINTTO procedure can be utilized to send the output to a pre-determined file (see Appendix B for a output sample).

```
filename outfile "table.ToC";
proc printto print=outfile new;
run;

options missing=’’ nodate nonumber; *
suppress system date and page numbering;

proc report data=in spacing=0 nowindows;
  col page_o title1 line title3n page;
  define page_o / order noprint;
  define title1 / order width=&max ' ';
  define line / order noprint;
  define title3n /display width=&space
spacing=2 ' ';
  define page / display width=5 f=5. ' ';
```

```
title1 “Table of Contents”;
title3 “ “;
run;
```

CONCLUSION

The above example presents the logical steps and corresponding code necessary for generating a table of contents for TABLE. In practice, the program is slightly altered into a macro (see Appendix A) so that it is capable of creating a table of contents for other output type as such LISTING, APPENDICE, etc. To reduce the number of macro parameters the user must supply to only one, it is necessary that the SAS output file for which table of contents is to be created be named “table.lst”, “listing.lst”, or “appendix.lst” as is required (i.e. the file name should be the same as the first word appearing on title1), and placed in the same directory where this program is executed from. Additional macro parameters defining the input and output files as well as the title keyword can be added if flexibility in file naming convention and location is preferred over programming simplicity.

In summary, the SAS macro “%ToC” can be used to automate the process of generating a Table of Contents from SAS output, hence, eliminating a once very time consuming and manual process. The macro is simple, easy to use, and creates a table of contents quickly and accurately. It can simply be re-run should any program output need to be added, deleted, or reorganized. Although only one title format was discussed in this paper (i.e. table type and numbering is on the first line, followed by a blank line and then by the main title), the program can be easily modified to handle other title formats commonly used (for example, no blank line between title1 and title3 or all titles on one line).

REFERENCES

SAS Institute Inc. (1990), *SAS Language: Reference, Version 6, First Edition*, Cary, NC: SAS Institute Inc.
SAS Institute Inc. (1997), *SAS Macro Language: Reference, First Edition*, Cary, NC: SAS Institute Inc.

TRADEMARKS

CONTACT INFORMATION

Hong Chen
McDougall Scientific Ltd.
90 Thorncliffe Park Drive
Toronto, ON
Canada M4H 1M5
Tel: (416)424-2092
Fax: (416)424-2095
Email: hchen@mcd-sci.on.ca

APPENDIX A: SAS CODE FOR %TOC.

```
/*********************************************
* SAS Program for generating Table of Contents
* from SAS output
* by Hong Chen, McDougall Scientific Ltd.,
* Toronto, Canada
* email: hchen@mcd-sci.on.ca
*
* The source (SAS output) file for which
* the ToC is to be generated MUST be named
* as table.lst, appendix.lst, or
* listing.lst and placed in the same
* directory where this program is to be
* run.
* The macro parameter “type” is the output
* type (e.g. table, listing, appendix …)
* Example:  %toc(table)
******************************************/

%macro toc(type);

options ls=85 ps=500 nonumber nodate missing='';
filename in "&type..lst";
filename out "&type..ToC";

data in (keep=title1 title3 page);
  infile in;
  length title1a number $10. title1 $20.;
  input +1 title1a $ @; * +1 to skip the page
break symbol at the first column;
  if upcase(title1a)=upcase("&type") then do;
    input number / title3 $200.;
    title1=compbl(title1a||number); * combined
for programming simplicity;
    page+1; * page number increased by 1;
    output;
  end;

proc sort data=in;
  by title1 page;

data in;
  set in;
  by title1 page;
  if first.title1;

data _null_;
  set in end=last;
  retain max;
  leng1=length(title1);
  if leng1>max then max=leng1;
  if last then call symput("max", put(max,
best.));
run;

%let ls=%sysfunc(getoption(ls));
%let space=%eval(&ls-&max-7);

data in;
  set in;
  length title3n $200;
  leng3=length(title3);
  page_o=page; * for ordering purpose;
  if leng3>=&space then do;
    split=&space; * starting position to find
the splitting point(blank space);
    done=0;
    do while (done=0);
      if substr(title3,split,1)^='' then do; *
find the splitting point;
        split=split-1;
      end;
      else done=1; * splitting point found;
    end;
    line=1;
    title3n=substr(title3,1,split); * first half
of title3;
    page=.; * supress page number for the first
line of the title;
    output;
    line=2; * next line for the remaining title;
    page=page_o; * assign page number to the
second line;
    leng3r=length(substr(title3,split+1)); *
length of the remaining title3;
    dots=&space-leng3r; * number of dots needed
to be filled in;

title3n=compbl(substr(title3,split+1))||repeat('
.', dots);
  output;
  end;
  else do;
    dots=&space-leng3;
    title3n=compbl(title3)||repeat('.', dots);
    line=1;
    output;
  end;

proc printto print=out new; * re-route the
result table of contents;
run;

proc report data=in spacing=0 nowindows;
  col page_o title1 line title3n page;
  define page_o / order noprint;
  define title1 / order width=&max ' ';
  define line / order noprint;
  define title3n /display width=&space spacing=2
' ';
  define page / display width=5 f=5. ' ';

title1 "Table of Contents";
title3 " ";
run
%mend toc;
```

APPENDIX B: TABLE OF CONTENTS OUTPUT SAMPLE

```
                                      Table of Contents

Table 1.1  Patient Demographics  .....................................    1
Table 1.2  Patient Disease History, Status at Each Visit, and Change
           from Baseline .............................................   11
Table 2.1  Study Completion ..........................................   25
Table 3.1  Adverse Events ............................................   26
```