Paper PO04-2011
Pagination in Clinical Trial PROC REPORT ODS

Chao Su, Advanced Clinical, Deerfield, IL, USA
William Conover, Advanced Clinical, Deerfield, IL, USA

Abstract

PROC REPORT is used extensively in clinical trial SAS programming. However, the procedure itself does not have a direct option to provide correct pagination for customized reports within the output delivery system (ODS). In response to this need, this paper describes a series of macros combined with a simple DATA STEP that provides accurate, manually controlled pagination of ODS output in rich text format (RTF) or portable document format (PDF) files.

Introduction

With the advantages of more flexibility compared to PROC TABULATE and less complexity compared to DATA _NULL_, PROC REPORT is widely used for creating a large variety of charts, reports, and data summaries. PROC REPORT is not a straightforward procedure, however. When tables or listings are created in RTF or PDF for a clinical trial via the ODS, the number of lines per page can be determined in advance, but there is no direct option to calculate the number of lines needed per observation when text wrapping occurs within a cell. Text wrapping can cause elements that should be displayed together on the same page to split over 2 or more pages. Manual pagination control is needed, to some extent, to determine where the page splitting occurs.

This paper describes a series of macros that provide accurate manual control of the output pagination by giving PROC REPORT the information it needs for correct page breaks. Explanation is broken down into these steps: First, counting the number of rows per observation that each column variable requires given the respective cell widths. Next, storing the maximum rows needed for each observation. Lastly, calculating the number of rows needed and applying it to a variable used by PROC REPORT to determine the desired page breaks. In our application, the main macro is %LNCNT which calls macros %WRAPVAR and %CUTHALF. Then a simple DATA STEP is used to create a pagination variable for PROC REPORT.

Macros

1. %WRAPVAR(INDSN, OUTDSN, WRAPVAR, WIDTH)

This macro is a modified version of D.W. Peterson's code that is used to generate user-specified text flow within PROC REPORT (Peterson 1999). For our purposes, it calculates the row number of each observation for a given column and consists of 4 parameters:

INDSN	the name of the input data set
OUTDSN	the name of the output data set
WRAPVAR	the name of column variable of interest, maximum value of 300 characters (can be increased if needed)
WIDTH	the width of column specified in PROC REPORT by number of characters

Although the SAS code for %WRAPVAR can be found at the end of this paper, the basic operation is described here.

- The output dataset will use the same name as the input dataset name if the macro parameter of OUTDSN is missing.
- A data step is used to split the text string of the column variable into rows according to the specified width and calculate the occupied row number, which is stored as INUM.
- For each observation, a DO loop determines the correct position to split the character string.

2. %CUTHALF

So far, the macro %WRAPVAR assumes that words in the string are shorter than the given width. In other words, the appropriate position can always be found. But if the length of a word is longer than the specified width, %WRAPVAR is unable to split the string. In these cases, the macro %CUTHALF is called within %WRAPVAR and splits the string by placing a hyphen "-" where the split occurs. Note that the hyphen placement is not necessarily grammatically correct.

Examples

A SAS dataset named TEST_DAT is created to test the macros % WRAPVAR and % CUTHALF as:

```
proc sql;
          create table test_dat
             (var1 char(30),
              var2 char(30));
          insert into test_dat
          values('ab cdef',  'abcd efghigklmnopqr')
          values('abcdefghi jklmnopqrstuv wxy za', 'abc defg ') ;
quit;
```

When VAR1 is given WIDTH=5 and VAR2 given WIDTH=10 are analyzed separately, the outputs are:

Table 1 (WIDTH=5**)**

var1	lnum
abc defg	2
abcd efgh-igkl-mnop-qr	5

Table 2 (WIDTH=10**)**

var2	lnum
abcdefghi jklmnopqr-stuv wxy za	4
ab cdef	1

In Table 1, the lines needed (LNUM) is 2 for the first observation and 5 for the second observation. The second observation of VAR1 has the value "abcd efghigklmnopqr". This text string length is larger than WIDTH=5. Therefore, the split position is located after the 4th character and a hyphen "-" is inserted. Table 2 shows what happens when the width is increased to 10 for a different set of data.

3. %LNCNT(INDSN = , OUTDSN = , WRAPVAR=, WIDTH =)

The limitation of the macro %WRAPVAR is that it can only deal with one column variable per each call. To manipulate multiple column variables, the macro %LNCNT is utilized to determine the maximum number of rows needed among multiple column variables for each observation. The following example illustrates the use of the %LNCNT macro.

```
%lncnt(indsn  = test_dat,
       Outdsn = out_dat,
       Wrapvar= var1 var2,
       width  = 5    10);
```

The output from %LNCNT is shown below.

Table 3 (Multiple columns with varying widths)

var1	var2	Lnum_var1	Lnum_var2	lnum
abc defg	abcdefghi jklmnopqr-stuv wxy za	2	4	4
abcd efgh-igkl-mnop-qr	ab cdef	5	1	5

Notice that the variable LNUM chooses the maximum required rows between VAR1 and VAR2 for each observation.

The Simple DATA STEP

A simple DATA STEP is constructed to assign page breaks to the variable PGCNT every 29 rows of output, while keeping GROUP1 and GROUP2 information on the same page. The assignment of 29 rows per page can be easily changed below.

```
data out_dat;
  set out_dat;
      by group1 group2;
      retain pgcnt 1 lncnt 0;
      lncnt=lncnt + first.group1*2 + lnum; *← blank row;
      if lncnt >=29  then do;  *← number of rows per page;
          pgcnt=pgcnt+1;
          lncnt = lncnt + lnum;
      end;
run;
```

The above code assumes that a blank row is inserted at the beginning of GROUP1.
Within the PROC REPORT procedure, PGCNT is used as page break ORDER or GROUP variable. This variable follows a BREAK statement as shown below:

```
break after pgcnt / page;
```

Conclusion

Correct pagination in PROC REPORT can be time consuming and frustrating, but the system of macros described here has proven very useful in practice .In the Clinical trials industry time is crucial. Any tool that can make it easier for a SAS programmer to enhance table or listing display should be seriously considered, and this is definitely one of them.

Macro Codes

%WRAPVAR Macro

```
%macro wrapvar(indsn, outdsn, wrapvar, width );
       %if &outdsn. eq %then %let outdsn=&indsn.;
       data &outdsn. &indsn.;
            length &wrapvar. $300.;
            retain z_ptr lnum&wrapvar;
            set &indsn.;
                lnum&wrapvar = 1;            *** The occupied display row number;
                ptr = &width. + 1;
                z_ptr = &width.;
                do while(substr(&wrapvar., ptr,1) ne "");
                   ptr = ptr - 1;
                   if ptr eq 0 then do;
                      %cuthalf;
                   end;
                end;
                substr(&wrapvar., ptr) =  ''||substr(&wrapvar., ptr+1);
                do while(ptr le length(&wrapvar.));
                   ptr = ptr + (&width.) + 1;
                   z_ptr = ptr - 1;
                   do while (substr(&wrapvar., ptr,1) ne "");
                      ptr = ptr - 1;
                      if z_ptr - ptr eq &width. then do;
                         %cuthalf;
                      end;
                   end;
                   substr(&wrapvar., ptr) = ''||substr(&wrapvar., ptr+1);
                   lnum&wrapvar = lnum&wrapvar + 1;
                end;
                if lnum < lnum&wrapvar then lnum = lnum&wrapvar;  **** Choose the
maximum actual occupied row number;
            drop ptr z_ptr;
       run;
%mend wrapvar;
```

%CUTHALF Macro

```
%macro cuthalf;
    z_var = substr(&wrapvar.,1,z_ptr-1) || "-" || substr(&wrapvar., z_ptr);
    &wrapvar. = z_var;
    drop z_var;
    ptr = z_ptr + 1;
%mend cuthalf;
```

%LNCNT Macro

```
%macro lncnt(indsn   = ,
             Outdsn  = ,
             Wrapvar =,
             width   =
                  );

    data &indsn ;
       set &indsn;
           lnum = 1;
    run;
    %let i= 1;
    %let lntvar = %scan(&wrapvar, &i);
    %let widvar = %scan(&width, &i);
```

```
      %do %while (%qscan(&wrapvar, &i) ne );
          %wrapvar(&indsn, &outdsn, &lntvar, &widvar);
          %let i = %eval(&i + 1);
          %let lntvar = %scan(&wrapvar, &i);
          %let widvar = %scan(&width, &i);
      %end;
%mend;
```

References

Donald W. Peterson and John R. Gerlach. 1999. User-Specified Text Flow Inside the Report Procedure. Miami Beach, FL: Proceedings of the 24th Annual SAS Users Group International; Paper 79.

Michele M. Burlew. 2006. SAS MACRO Programming Made Easy, 2nd edition. Cary, NC: SAS Institute Inc.

Acknowledgements

We would like to acknowledge the statistical team, especially the programming group, at Advanced Clinical for providing the opportunity to develop and test these macros. Thanks especially to Angela Roster for her review and editorial assistance for the paper.

Contact Information

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

Chao Su
Advanced Clinical
10 Parkway North, Suite 350
Deerfield, IL 60015
Phone: 847-267-1776
Fax: 847-267-1432
E-mail: csu@acrxs.com
Web: http://www.advancedclinical.com