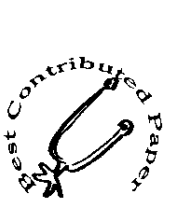

A Set of SAS® Macros for Producing Customized Reports

Greg Grandits, Division of Biostatistics, University of Minnesota, Minneapolis, Minnesota
Ken Svendsen, Division of Biostatistics, University of Minnesota, Minneapolis, Minnesota

ABSTRACT

Producing customized reports is a high priority in many research and business settings. SAS® has several procedures that generate reports, but only the PUT statement within a data step gives complete flexibility to placement of text and data values. However, data step processing and writing PUT statements is cumbersome to program. This paper describes a set of macros that produce customized reports that are easy to program and give complete flexibility to placement of text and data values.

The user first defines columns and column widths across the report page. Text or data values (often summary statistics) are then moved to these columns and specified lines using macros MOVE or NMOVE. Items can be moved to single columns or to combined columns with the items automatically centered. Text can also be underlined. Lines to where items are moved can be listed or a starting line can be given along with the number of lines to be printed with optional skip patterns. Summary statistics are accessible using macro BREAKDN which uses PROC SUMMARY to produce a one observation data set. Statistics are placed in array type names (e.g., M1-M20) which can be easily moved to the report page after a SET statement. An example of a call is %nmove (M1-M20,col=2 3,line=12L10,pat=1st/1).

INTRODUCTION

Producing customized reports using computer software is a high priority in many settings in research and business. Programs that produce these reports should be easy to write and modify. Besides making report tables aesthetically pleasing the ease of updating or modifying customized report programs conserves resources and reduces the chance for transcription or programming errors.

The reports desired can be varied, often containing summary statistics of variables (N's, MEAN's, SD's, etc.) but can also contain other information such as p-values and regression or correlation coefficients. SAS® has several procedures for use in generating reports including PROC TABULATE and PROC REPORT. Through use of these, often with some ingenuity, one can generate reports that are sufficient for many needs. However, these procedures are still limited in their flexibility in placing text or data values which often leads the programmer to use DATA STEP PROCESSING with PUT statements to generate the required report. However, data step processing is time consuming to program and is not always easy to use.

This paper describes a set of macros that produce customized reports that are easy to program and give complete flexibility to placement of text and data values to the report page. The macro BREAKDN produces summary statistics for given variables by levels of given class variables and puts them in a one observation data set that can be easily moved to the report page. Macro MOVE moves character strings onto the report page. Macro NMOVE moves numeric variables (usually means, sdev's, etc.) produced by BREAKDN (or some other way) onto the report page.

The macros MOVE and NMOVE generate one or more SAS PUT statements based on the parameters sent to the macros.

Use of Macros to generate a report

Report programs are made up of:

1) a REPORT statement that indicates a new report is starting.

2) a COLSET statement that defines columns and column widths across a report page to which text or data values are later moved.

3) text moves using MOVE. Features include centering, underlining and repeating text.

4) a SET statement that reads in a SAS data set that is usually a single record (observation) containing summary statistics. Usual summary statistics are obtained from Macro BREAKDN which places the statistics in array type names (eg, M1-Mn contains the means where n depends on the number of variables broken down in the BREAKDN statement, the number of class variables, and whether totals or subtotals are requested).

5) one or more NMOVE statements moving the summary statistics onto the report page. NMOVE allows the whole vector of values (means, sdev's, etc.) to be moved on the report page by simply referring to the beginning and ending values of the vector (eg, M1-M50 moves 50 means).

Detailed Use of Macros

1) REPORT is used simply as %REPORT which indicates the start of a new report.

2) COLSET is used as follows:

   %COLSET (col1size nospaces"x" col2size nospaces"x"...);

   Ex: %COLSET (25 10 2x 10 2x 10);

   This statement sets up 4 columns. The first column is 25 positions long and the last 3 columns are each 10 positions long. Two spaces are placed between the last 3 columns. Nothing is written to these positions, and is used to set off text or data values from other columns. COLSET needs to be called before any move statements are made and is called once for each report.

3) MOVE is used as follows:

   %MOVE ('string1':'string2'...:'stringk', line=, col=, center=, under=, pat=, mfirst=);

strings - character strings enclosed in quotes separated by colons.

Ex: 'Men':'Women':'Total' - a total of 3 strings

line - this is the line numbers to which the strings will be moved. This can be given in 2 ways:

1) line = line1 line2 ... linek (Each line number is separated by a space.)

2) line = startline"L"nolines (A starting line followed by the number of lines to be printed to.)

Ex 1: line = 12 21 33 moves strings to lines 12, 21, and 33.

Ex 2: line = 12L3 moves strings to lines 12, 13, and 14.

This can be combined in one statement if desired.

Ex 3: line = 12L3 24L3 32 33 moves to lines 12 to 14, 24 to 26, and lines 32 and 33.

If the line statement is omitted %MOVE uses the last line statement given.

col - this lists the columns to be printed to that were set up in the COLSET statement. Columns can also be combined.

col = col1"del"col2"del"...colk. The delimiters can be:

(space) moves strings to each specified column
. moves strings to each column between and including the columns listed
- moves strings to combined columns

Ex 1: col = 3 4 8 moves to columns 3, 4, and 8.

Ex 2: col = 3 4-8 moves to columns 3 and a column formed by combining columns 4 through 8.

Ex 3: col = 2.8 moves to columns 2 through 8.

If the column statement is omitted %MOVE uses the last column statement. Also, the last column can be referenced as the "0" column.

center - set to "Y" if centering is desired; set to "N" if string is to be left justified. The default is center.

under - set to "Y" if string is to be underlined. The default is not underlined. The parameter "U" can be used instead of "under".

pat - used to skip lines. The line statement must be the second form, i.e., line=10L12 for example.

Ex: pat=5th/1 skips a line after every 5 lines.

mfirst - sets whether you want columns or lines advancing first when strings are moved. Set to "L" for lines to move first (default) or "C" for columns to move first.

Example of %MOVE statements:

```
%MOVE ('Special Intervention':'Usual Care',line=10,
        col=2-3 4-5,u=y);
```

This statement produces the following text in the lines and columns specified:

Special Intervention Usual Care

```
%MOVE('Total Cholesterol':'LDL-C':'HDL-C',
        col=1,line=10L6,center=n,pat=3rd/1)
```

This statement produces the following text. Note that the strings are repeated until the total lines are exhausted.

```
Total Cholesterol
LCL-C
HDL-C

Total Cholesterol
LCL-C
HDL-C
```

4) NMOVE is used as follows:

```
%NMOVE(var1"del"var2"del"...vark,line=,col=,fmt=,pat=,
        mfirst=,scaler=);
```

The line=, col=, mfirst=, and pat= statements are identical to the statements used in %MOVE. The possible delimiters for listing the "vars" are a space or a dash (-). A space indicates the variable is just a single variable. A dash is used when you want to print consecutive variables with a consecutive numeric suffix.

Ex 1: %NMOVE (age chol dbp,...) moves the value of the variables age, chol and dbp.

Ex 2: %NMOVE(M1-M50,...) moves the variables m1, m2, m3...m50.

fmt - num.num where the first value is the width the variable is to be printed and the second value is the number of decimals to be printed. All values are centered over the column(s) indicated in the column statement.

The default format is 5.1.

Ex: fmt=7.2 centers over 7 positions within the columns stated and prints 2 decimal places.

scaler - if this option is used each variable in the %NMOVE statement is multiplied by the value specified before printing. This is useful for converting fractions to percentages or vice versa.

Use of Macro BREAKDN

%BREAKDN (class=, var=, out=, data=, out=, sfirst=)

This macro reads the SAS data set specified in DATA using PROC SUMMARY and computes statistics for each variable specified in VAR by each level of each variable specified in CLASS, and outputs one observation containing these statistics to the data set specified in OUT. The statistics calculated are N, MEAN, SDEV, SUM, MIN, MAX, and SE. They are contained in the variables n1-n?, m1-m?, s1-s?, sum1-sum?, min1-min?, max1-max?, and se1-se?, respectively, where ? is explained below.

The parameters to the call are:

class - cvar1 nlev(T) cvar2 nlev(T)...

This lists each class variable for which statistics are to be calculated, followed by the number of levels in the class variable. The level can be suffixed by "T" if statistics for the total are desired. Statistics are calculated for each combination of class levels including totals if indicated. The class variables must be coded consecutively from one to the highest level.

Example: class = sex 2t group 6.

This indicates that there are two class variables, SEX and GROUP with 2 and 6 levels, respectively. Statistics for each level of GROUP for men and women combined will also be computed.

var - var1 var2 ... vark

This is the list of variables to compute statistics for.

Example: var = age dbp sbp chol cigs

data - is the SAS data set to be read

out - is the SAS data set the statistics computed will be outputed. This data set will contain only 1 observation.

sfirst - indicates the order in which the statistics are stored. SFIRST = VAR indicates that the statistics for the variables in VAR are stored first for a given class combination;

SFIRST = CLASS indicates that the statistics for the variables in CLASS are stored first for a given variable in VAR.

The way the statistics are stored into variables is best explained by examples. Suppose the call to BREAKDN is as follows:

%BREAKDN(class=sex 2t, var=age dbp sbp chol cigs, out=table1);

If SFIRST is set to VAR (default) then:

The n's for the 5 variables for sex=1 are stored in n1-n5.
The n's for the 5 variables for sex=2 are stored in n6-n10.
The n's for the 5 variables for both sexes are stored in n11-n15.

The variables are similarly stored for the mean (m1-m15), standard deviation (s1-s15), sum (sum1-sum15), min (min1-min15), max (max1-max15) and standard error of the mean (se1-se15).

If SFIRST is set to CLASS then:

n1-n3 contain the n's for the variable age for sex=1, sex=2 and sex=all, respectively, and n4-n6 contain the n's for the variable dbp for sex=1, sex=2 and sex=all, respectively. Similarly n7-n9, n10-n12, and n13-n15 contain the n's for the variables sbp, chol and cigs for each level of sex.

An example with two class variables is:

%BREAKDN(class=sex 2 race 3t, var=age dbp sbp chol cigs,
out=table1);

If SFIRST is set to VAR then:

The n's for the 5 variables for sex=1 and race=1 are stored in n1-n5.
The n's for the 5 variables for sex=1 and race=2 are stored in n6-n10.
The n's for the 5 variables for sex=1 and race=3 are stored in n11-n15.
The n's for the 5 variables for sex=1 and all races together are stored in n16-n20.

Similarly, for sex=2 and race=1,2,3 and total are stored in n21-n40.

If SFIRST is set to CLASS then:

n1-n8 contain the n's for age for sex=1, race=1; sex=2, race=2; sex=1, race=3; sex=1, race=all; sex=2, race=1; sex=2, race=2; sex=2, race=3; and sex=2, race=all, respectively. Similarly n9-n40 contains the n's for the other four variables.

***** EXAMPLE PROGRAM *****

```
%include options ;
%include move;
%include breakdn ;

data temp ;
 set tomhsdat.rand ;
 set tomhsdat.ql ;

%let list = qlghi  qlef  qlmhi  qlgfi  qlspa  qlsfr  qlsc  ;

%breakdn(class=sex 2t cgroup 2, var=&list, out=qlfop, sfirst= class);

%report ;
%colset(30 11 11 2x 11 11 2x 11 11);

%move('Average Quality of Life Indexes Over Followup by':
      'Sex and Study Group for TOMHS Participants',
      col=1-0,line=312) ;
%move('Men':'Women':'Total', col=2-3 4-5 6-7, line=7, u=y);
%move('Group 1':'Group 2', col=2.7, line=9, u=y);
%move('QL Index',col=1,center=n,u=y) ;
%move('General Health':
      'Energy/Fatigue':
      'Mental Health':
      'General Functioning':
      'Satis. Physical Abilities':
      'Social Functioning':
      'Social Contacts',
 col=1,center=n,line=1217,pat=1st/1);
%move('Number of Participants',col=1,center=n,line=27) ;

set qlfop;

%nmove(m1-m42, col=2.7, line=1217, pat=1st/1, mfirst=c, fmt=5.1) ;
%nmove(n1-n6, line=27, fmt=3.0);
run;
```

**** THIS PRODUCES THE FOLLOWING REPORT ****

Average Quality of Life Indexes Over Followup by
Sex and Study Group for TOMHS Participants

	Men		Women		Total	
QL Index	Group 1	Group 2	Group 1	Group 2	Group 1	Group 2
General Health	40.6	39.4	40.3	40.0	40.5	39.6
Energy/Fatigue	18.6	18.2	17.6	17.3	18.2	17.8
Mental Health	65.7	63.4	63.6	63.4	64.9	63.4
General Functioning	14.3	14.0	13.7	13.6	14.1	13.9
Satis. Physical Abilities	4.9	4.8	4.7	4.8	4.8	4.8
Social Functioning	3.8	3.8	3.7	3.6	3.8	3.7
Social Contacts	5.9	5.9	5.9	5.9	5.9	5.9
Number of Participants	402	143	252	90	654	233

Discussion

Much effort has been put into making SAS procedures that would handle user needs to generate reports. In fact, specific courses are given by SAS® on report writing. Although procedures such as PROC TABULATE and PROC REPORT can be used to generate often times reasonable looking reports, they have limitations. The report macros described in this paper have tremendous flexibility and are simple to use. The move macros described do nothing more than produce PUT statements. However, the macros do most or all the tedious work the user would otherwise have to do and keep track of. Features such as underlining, automatic centering, text repeating when extra lines or columns are given, skip patterns in lines, and moving array type variables by specifying first and last suffix values make these macros attractive to use.

The key to making the numeric moves easy is to get the information needed into a SAS data set containing a single observation. Then a simple SET statement makes available the data needed to be moved. The macro BREAKDN does this when usual summary statistics are desired by levels of one or more class variables. Since the summary statistics are stored in array like variables, these statistics can easily be moved onto the report page. Of course, there are times when the user wishes to move other types of information onto the report page (frequency distributions, correlation or regression coefficients, p-values, etc.). For these cases the user may need to write a macro that processes an output data set or listing file from a procedure to make a one observation data set. Then as described above this data can be moved easily onto the report page. Thus, there are no limitations as to the type of statistics or statistical summaries that can be moved onto the report page. This author has written several of these type of macros including one that creates a one observation data set containing the counts, percentages, and cumulative percentages of levels of a variable by levels of another variable.

A possible drawback to using these macros to generate reports is the added CPU time needed to run them compared to SAS procedures. However, the programming time saved to write and modify report programs will likely offset any additional costs associated with the slower running of the program.

Acknowledgments

The authors with to acknowledge the late Ronald Schultz, whose many ideas from a report generating program written by him over a decade ago were incorporated in the above macros.

Address correspondence to:

Greg Grandits
Suite 200
2221 University Avenue Southeast
Minneapolis MN 55414-3080

Phone: (612) 626-9033
INTERNET: greg-g@blueox.ccbr.umn.edu