The Tabulate Procedure: One Step Beyond the Final Chapter

Stephen M. Noga, Rho Inc.
Jeffery M. Abolafia, UNC-CH, Department Of Biostatistics

Introduction

PROC TABULATE® is one of the most powerful and flexible procedures for producing customized reports. In a series of papers, Dan Bruns (1991, 1996, 1997) has demonstrated the power of PROC TABULATE, outlining both the basic and more advanced features of PROC TABULATE. However, when a customized report is needed, PROC TABULATE is often passed over in favor of more time consuming and tedious methods such as PROC REPORT, FILE PRINT, or a series of data steps and PROC PRINT. We feel that this is partly due to the fact that the full potential of PROC TABULATE has not been utilized. This paper demonstrates how PROC TABULATE can be used in conjunction with other basic SAS® techniques to produce customized reports. Through a series of examples, we will show how many customized reports produced with PROC REPORT or FILE PRINT can be more easily constructed with the TABULATE Procedure. This paper assumes a basic understanding of the TABULATE Procedure. The series of papers by Dan Bruns cited above provide an excellent tutorial on PROC TABULATE.

Sample Data Set

The data set used in the examples presented in this paper contains 9,784 observations and the following variables:

SEX	Sex (1=male 2=female)
TRT	Treatment Group (0=placebo 1=drug)
AGE	Age in Years
CHOL2	Total Cholesterol (Mg/dl)
LDL	Low Density Cholesterol (Mg/dl)
HDL	High Density Cholesterol (Mg/dl)
FUTIME	Follow-up Time
MORT	All Cause Mortality (0=no 1=yes)
CVDDEATH	Cardiovascular Death (0=no 1=yes)
CHDDEATH	Coronary Heart Disease Death(0=no 1=yes)
DUMMY	Constant (=1 for every observation)

Producing Your Basic Table

A Basic premise of this paper is that PROC TABULATE is more efficient for table generation than is the data _null_/put method, and it can be quite flexible if certain techniques are utilized. This paper uses a sequence of tables to demonstrate this.

The first table displays the means of several continuous variables by gender and treatment group. However basic this
table is, it does let us begin to look at some of the differences between the two methods of table generation.

TABLE 1. Mean Lipids By Gender And Treatment Group

Gender	Lipid	Treatment Group	
		PLACEBO	DRUG
MALE	TOTAL CHOLESTEROL	223.44	222.35
	CHOLESTEROL IN LDL	147.31	146.68
	CHOLESTEROL IN HDL	44.74	44.68
FEMALE	TOTAL CHOLESTEROL	224.75	225.15
	CHOLESTEROL IN LDL	145.31	146.95
	CHOLESTEROL IN HDL	58.14	58.64

```
PROC FORMAT ;
  VALUE TFT
    0="PLACEBO"
    1="DRUG"   ;
  VALUE SFT
    1="MALE"
    2="FEMALE" ;

PROC TABULATE DATA=&in FORMAT=8.2
  NOSEPS;
  CLASS TRT SEX;
  VAR CHOL2 LDL HDL ;
  TABLE SEX=''*(CHOL2 LDL HDL)*MEAN=
     ' ',TRT='Treatment Group'
     / box = 'Gender        Lipid'
       row=float rts=40;
  FORMAT TRT TFT. SEX SFT. ;
RUN;
```

Two of the major benefits of generating the table with PROC TABULATE are:

1) No data manipulation was necessary

No additional PROCs or data steps were needed to get the data into a format that could be used in the table. PROC TABULATE grouped the data by gender, variable, and treatment group and then calculated the means.

2) A table layout was not needed

```
--------------------------------------------------
.                                  Treatment Group
Gender       Lipid                 ---------------
                                   PLACEBO   DRUG
--------------------------------------------------
MALE         TOTAL CHOLESTEROL      223.44  222.35
             .  LDL CHOLESTEROL     147.31  146.68
                HDL CHOLESTEROL      44.74   44.68
FEMALE       TOTAL CHOLESTEROL      224.75  225.15
             .  LDL CHOLESTEROL     145.31  146.95
             .  HDL CHOLESTEROL      58.14   58.64
--------------------------------------------------
```

Through the use of PROC TABULATE, you avoid the chore of designing the table layout: what specific columns should the variables go in, which lines and what columns should the header use, underlining, and a data _null_ step was not needed to figure out how to put all this information in a table.

Overall, we think it is evident that PROC TABULATE is the preferred option in this example. Now if you want to argue that the data _null_ option would give you the ultimate in flexibility, well (a deep subject), that point must be conceded, but we would ask at what cost (time, monotony, frustration)? The next two examples demonstrate how PROC TABULATE can work in conjunction with some other SAS techniques/options to increase it's own flexibility.

Improvement In Flexibility - Part I

Table 1 is an example of what PROC TABULATE will generate by default. Note that the gender and lipid columns are left aligned and the placebo and drug columns are right aligned. Also, the first row of the table is up against the underline from the header. PROC TABULATE has this hangup against leading blanks, so some alternate strategy must be employed if you want to incorporate these in your table. Table 2 shows an easy way to indent variables, although as you will see in part II, not the method we would recommend.

Notice that the box string now contains a '.' as the first character. By having this period in the string, PROC TABULATE is forced to recognize all the leading blanks before the word 'Gender' as part of the actual string. Since all the blanks are kept in the string, the length of the string exceeds the RTS (Row Title Spaces) length, and thus the words 'Gender' and 'Lipid' are wrapped onto the second header line. This same technique is used in the label assignments for the LDL and HDL variables causing them to be indented.

Of course, now you have periods(.) displayed in your table. If you do not want the periods to be displayed, then your options for removing them are using white out, or using an editor to change them to blanks. In either case, that's an additional step that can be avoided by using a better technique.

TABLE 2. Mean Lipids By Gender And Treatment Group

```
box = '.
Gender     Lipid';
Label   LDL='.  LDL CHOLESTEROL'
        HDL='.  HDL CHOLESTEROL'  ;
```

Improvement In Flexibility - Part II

In order to generate Table 3, additional steps were used. These steps provide the user with more control over the displayed output from PROC TABULATE.

TABLE 3. Mean Lipids By Gender And Treatment Group

```
*** Create Unprintable Hex Character ***;
%let ffx = FF ;
DATA _null_;
 call sympt('blnk',
   trim(left(input("&ffx",$hex2.))));
RUN;

**********************************
 Modify Formats To Take Advantage
 Of The Unprintable Character And
 add new picture formats to center
 columns and display an '*' when
 necessary.
**********************************;
PROC FORMAT;
  VALUE sft
    1="&blnk                &blnk
        MALE"
    2="&blnk                &blnk
```

```
--------------------------------------------------
                                   Treatment Group
                                   ---------------
 Gender    Lipid                   PLACEBO    DRUG
--------------------------------------------------

  MALE     TOTAL CHOLESTEROL        223.44  222.35
             LDL CHOLESTEROL        147.31  146.68
             HDL CHOLESTEROL         44.74   44.68

  FEMALE   TOTAL CHOLESTEROL        224.75  225.15*
             LDL CHOLESTEROL        145.31  146.95
             HDL CHOLESTEROL         58.14   58.64
--------------------------------------------------

* Actual level is higher than the expected level.
```

```
        FEMALE" ;
  PICTURE valf (round)
    low-high = '   009.99    ' ;
  PICTURE toohif (round)
    low-<225 = '   009.99    '
    225-high = '   009.99*   ' ;
RUN;

*** Create Table 3 ***;
PROC TABULATE DATA=&in NOSEPS ;
  CLASS TRT SEX;
  VAR CHOL2 LDL HDL ;
  TABLE SEX=''*CHOL2*f=toohif.
  LDL*f=valf. HDL*f=valf.)*MEAN='',
  TRT='Treatment Group'/ row=float rts=40
  box = "&blnk                  &blnk
         &blnk                  &blnk
         &blnk     Gender    Lipid";
  FORMAT TRT TFT. SEX SFT. ;
  LABEL CHOL2 = "&blnk
              TOTAL CHOLESTEROL"
        LDL  = "&blnk  LDL CHOLESTEROL"
        HDL  = "&blnk  HDL CHOLESTEROL"; RUN;
```

The first step creates the macro variable blnk, which contains an unprintable character. This macro variable now acts as a holding character, like the '.' in example 2, and thus allows what appears like leading blanks to be used in formats, labels, and the box assignment.

In the SFT format and again in the box assignment, at least two &blnk variables are used in each case. If you go back and look at the box assignment in Table 2, the 'Gender Lipid' header line was only able to be dropped one line even after using the leading '.' . Once 'Gender Lipid' was wrapped to the second line, all the remaining leading blanks were deleted. If those are to be kept, then another 'holding' character must be inserted. By this multiple usage of &blnk in the format SFT, we were able to: put the 'Gender Lipid' on the third line of the header and still indent 'Gender' instead of it being left aligned; and insert a blank line before the first row for MALES; and create a double space between the male and female lipid values.

The use of picture formats VALF and TOOHIF allow the means to be centered instead of the default right alignment PROC TABULATE uses. The picture format toohif also displays an asterisk(*) if the value is greater than or equal to 225.

Including Sample Sizes in Variable Formats

It is often necessay to include the sample size for each level of a class variable in a variable format. Note that in Table 4 below, the sample sizes for the drug and placebo groups are displayed. One obvious strategy to obtain the sample sizes would be to first run a program to get the sample sizes. Then in a second program type the sample sizes into a variable format. A more dynamic approach consisting of three steps is demonstrated below. First, the sample sizes are obtained from PROC SUMMARY. Second, the sample sizes are stored in macro variables. Third, the macro variables are included as part of the format for treatment group.

TABLE 4. Mean Lipids by Treatment Group

	Treatment Group	
	PLACEBO (N=4337)	DRUG (N=4447)
TOTAL CHOLESTEROL	224.04	223.67
CHOLESTROL IN LDL	146.38	146.81
CHOLESTROL IN HDL	50.95	51.24

```
/* Step 1: Get Sample Size */
PROC SUMMARY DATA=&in NWAY ;
  CLASS TRT ;
  VAR AGE ;
  OUTPUT OUT=OUT1 N=NS ;
RUN;

/* Step 2: Store Sample Sizes
           in Macro Variables */
DATA _NULL_ ;
  SET OUT1 ;
  IF TRT=0 THEN
  CALL SYMPUT('NP',PUT(NS,4.));
  ELSE IF TRT=1 THEN
  CALL SYMPUT('ND',PUT(NS,4.));
RUN;

/* Step 3: Create Format for
           Treatment Group */
PROC FORMAT ;
  VALUE TFT  0="PLACEBO(N=&NP)"
             1="DRUG (N=&ND)" ;
RUN;

/* Create Table 4 */
PROC TABULATE DATA= &in
  FORMAT=8.2 NOSEPS ;
  CLASS TRT;
  VAR CHOL2 LDL HDL ;
  TABLE (CHOL2 LDL HDL)*MEAN=' ',
  TRT='Treatment Group'/ROW=FLOAT;
  FORMAT TRT TFT. ;
RUN;
```

Using Statistics to Get Other Statistics

Table 5 is an example of a very common table type in a clinical trial. Table 5 shows the number and percent of subjects with selected events by treatment group. Where percent is defined as:

$$\frac{\text{\# of events in a group}}{\text{\# of subjects in a group}} \times 100$$

This table can easily be produced in PROC TABULATE by using the SUM statistic to calculate the number of events and the PCTSUM statistic to calculate the percent of subjects with a selected event. To use this method, all events should be coded 0/1, with 1 indicating the event occured. Second, a variable should be created that equals 1 for every observation.

For any event, if we sum all values for a given cell, we will get the number of events for that cell. The PCTSUM will give us the percent of subjects with an event for a given cell because the numerator is the sum of all values for the event variable in a given cell, or just the number of

events for that cell. For our denomenator definition we use <dummy>, where dummy has a value of one for every observation. The denomenator is the sum of all values of dummy for a given cell, or just the number of observations in that cell. Finally when using the PCTSUM statistic, the result is converted from a proportion to a percentage.

TABLE 5. Number and Percent of Selected Events by Treatment Group

	Treatment Group			
	PLACEBO (N=4337)		DRUG (N=4447)	
	No.	%	No.	%
All Cause Mortality	503	11.59	525	11.80
Cardiovascular Death	237	5.46	258	5.80
Coronary Heart Death	107	2.46	118	2.65

```
PROC TABULATE DATA= &in
  FORMAT=PFT. NOSEPS;
  CLASS TRT;
  VAR MORT CVDDEATH CHDDEATH
      DUMMY ;
  TABLE MORT CVDDEATH CHDDEATH,
  TRT='Treatment Group'*
  (SUM='No.'*F=NFT.
  PCTSUM<DUMMY>='%') / rts=22 ;
  FORMAT TRT TFT. ;
RUN;
```

Table 6 is an example of another common table type in a clinical trial. Table 6 shows the number of events and the event rate of selected events by treatment group. Where event rate is defined as:

$$\frac{\text{\# of events in a group}}{\text{\# person years in a group}} \times 1000$$

PROC TABULATE along with the use of picture formats can be used very efficiently to both calculate and display the statistics for this table. As in our previous example, all events should be coded 0/1. The sum statistic is used as shown above to calculate the number of events in a group or cell. The PCTSUM statistic will be used to calculate the number of events per 1000 person years.

The numerator is again the sum of all values for the event variable in a given cell, or simply the number of events for that cell. For our denomemator definition we use <futime> where futime equals the number of years of follow-up for every observation. The denomenator is the sum of all values of futime for a given cell. Finally we have to convert the event rate from rate per one year of follow-up to rate per

1000 person years of follow-up. This is easily accomplished using a picture format with the MULT option.

TABLE 6. Number and Rate of Selected Events by Treatment Group

	Treatment Group			
	PLACEBO (N=4337)		DRUG (N=4447)	
	No.	RATE	No.	RATE
All Cause Mortality	503	9.65	525	9.83
Cardiovascular Death	237	4.54	258	4.83
Coronary Heart Death	107	2.05	118	2.20

```
PROC FORMAT ;
  VALUE TFT 0="PLACEBO  (N=&NP)"
            1="DRUG   (N=&ND)" ;
  PICTURE NFT 0-HIGH = ' 009 ' ;
  PICTURE PFT
    0-HIGH='009.99 '(MULT=1000);
RUN;

PROC TABULATE DATA= &in
  FORMAT=NFT. NOSEPS ;
  CLASS TRT;
  VAR MORT CVDDEATH CHDDEATH
      FUTIME ;
  TABLE MORT CVDDEATH CHDDEATH,
   TRT='Treatment Group'*
   (SUM='No.' PCTSUM<FUTIME>=
   'RATE'*F=PFT.) / RTS=22 ;
  FORMAT TRT TFT. ;
RUN;
```

Producing Tables with Statistics not Calculated by PROC Tabulate

It is often necessary to produce tables with statistics not calculated by PROC TABULATE. Table 7 is a common table type. It displays the sample size, minimum, maximum, mean, standard deviation, the 25th percentile, the median, and the 75th percentile for a set of continuous variable.

While PROC TABULATE does not calculate percentiles, it does provide a flexible and efficent method for displaying these statistics. In the example below, we demonstrate a three step approach for producing Table 7. Macro variables are also used to make the application more dynamic. First, we use PROC UNIVARIATE to both calculate and output all necessary statistics. Second, we use a data step to

restructure the data from one record to one record per variable. Finally, we use tabulate to display the statistics.

TABLE 7. Descriptive Statistics

	LDL	HDL
Number	747.00	764.00
Minimum	20.00	5.00
Maximum	425.00	195.00
Mean	146.60	51.09
Std. Deviation	42.46	16.72
25th percentile	117.00	39.00
50th percentile	144.00	49.00
75th percentile	173.00	60.00

```
%let in=sc.se9700 ;
%let nvars=2 ;
%let vars=LDL HDL ;

PROC UNIVARIATE data=&in noprint;
  var &vars ;
  output out=out1 n=_n1-_n&nvars
  min=_min1-_min&nvars
  max=_max1-_max&nvars
  mean=_mean1-_mean&nvars
  std=_std1-_std&nvars
  q1=_q11-_q1&nvars
  median=_q21-_q2&nvars
  q3=_q31-_q3&nvars ;
RUN;

* restructure output data set *
* for PROC TABULATE           * ;
DATA tab(keep= score n--q3 ) ;
    SET out1 ;

    array _n{&nvars} ;
    array _min{&nvars} ;
    array _max{&nvars} ;
    array _mean{&nvars} ;
    array _std{&nvars} ;
    array _q1{&nvars} ;
    array _q2{&nvars} ;
    array _q3{&nvars} ;

    do i=1 to &nvars ;
       score=i ;
       n = _n{i} ;
       min=_min{i} ;
       max=_max{i} ;
       mean=_mean{i} ;
       std=_std{i} ;
       q1=_q1{i} ;
       q2=_q2{i} ;
       q3=_q3{i} ;
       output ;
    end;
RUN;

PROC FORMAT  ;
  VALUE    sftf     1    = 'LDL'
                    2    = 'HDL';
  PICTURE cft (round)
      low - high  =  ' 009.00  '
       other        =  '    0    ' ;
RUN;
```

```
PROC TABULATE  DATA=tab  FORMAT=cft.
  NOSEPS;
  CLASS score  ;
  VAR n min max mean std q1 q2
      q3;
  TABLE (n='Number'
         min='Minimum'
         max='Maximum'
        mean='Mean'
        std='Std. Deviation'
        q1='25th percentile'
        q2='50th percentile'
        q3='75th percentile')
        *SUM=' ',
        SCORE=' '/ ROW=FLOAT
        RTS=19 ;
   FORMAT  score sftf.  ;
RUN;
```

Converting Tabulate Output To A HTML Formatted Table

Our final example demonstrates how PROC TABULATE output could easily be presented as an HTML formatted table. SAS Institute has recently developed a set of Web Tools, one of which is the HTML Tablulate Formatter. The HTML Tablulate Formatter can be invoked by calling the macro TAB2HTM. This MACRO is supplied by SAS Institute. The macro, as well, as complete documentation can be found at: http://www.sas.com/rnd/web.

The SAS code below presents a simple example of using the HTML Tablulate Formatter in batch mode (it also runs in interactive mode). To use the TAB2HTM macro:

1. Set line size to 195 or less in the options statement.

2. Formchar must be set to:

‘82838485868788898a8b8c’x

This can be done by using the FORMCHAR system option or the FORMCHAR option in PROC TABULATE.

3. Your program can only generate output from PROC TABULATE. If your program generates output from any other procedure the macro will fail. To capture other types of output, use the HTML Output Formatter.

In batch mode, output is captured as it is being produced. To turn capture mode on type the following:

```
%tab2htm(capture=on,runmode=b)
```

This tells SAS that you want to capture everything until capture mode is turned off. After all PROC TABULATE steps, capture mode must be turned off as follows:

```
%tab2htm(htmlfile=xxx.html, capture=off,
```

```
opendmode=replace,runmode=b)
```

Where:

- HTMLFILE Specifies the name of the HTML file where formatted output will be written.
- CAPTURE Indicates the capture mode. Valid values are “on” and “off”.
- OPENMODE Indicates whether the new HTML file overwrites the information currently in the specified file or if the new ouput is appended to the end of the existing file. Valid values are “append” and “replace”.
- RUNMODE Specifies whether you are running the macro in batch or interactive mode. Valid values are “B” and “I”.

The TAB2HTM macro contains a myriad of other parameters. These parameters can be used to customize the display of the HTML formatted table.

Below is the SAS code needed to display Table 4 as an HTML formatted table.

```
OPTIONS LS=173;

%tab2htm(capture=on,runmode=b)

PROC TABULATE DATA=in
  FORMAT=8.2 NOSEPS
  FORMCHAR='828384858688788898a8b8c'x   ;
  CLASS TRT;
  VAR CHOL2 LDL HDL ;
  TABLE (CHOL2 LDL HDL)*MEAN=' ',
  TRT='Treatment Group' ;
  FORMAT TRT TFT. ;
RUN;

%tab2htm(htmlfile=SE9704.htm,
  capture=off,openmode=replace,
  runmode=b)
```

Conclusion

Our goal in writing this paper was to make the SAS user take another look at utilizing PROC TABULATE for custom report generation, and hopefully, it has been achieved. After reading this paper, your imagination should be stimulated into seeing new possibilities for PROC TABULATE, not just the ones presented here. Nothing you have read in this paper is new, it's just been presented in a format which says "Look how easy it is to get a very presentable table out of PROC TABULATE when used in conjunction with what SAS has to offer".

Acknowledgments

The authors would like to thank Kiduk Yang and Ravi Mathew for providing input and reviewing drafts of this paper.

References

Bruns, Dan (1991), “The Utter Simplicity of the TABULATE Procedure”, Proceedings of the Sixteenth Annual SAS Users Group International Conference.

Bruns, Dan (1996), “The Utter Simplicity of the TABULATE Procedure - The Sequel”, Proceedings of the Twenty-First Annual SAS Users Group International Conference.

Bruns, Dan (1997), “The Utter Simplicity of the TABULATE Procedure - The Final Chapter”, Proceedings of the Twenty-Second Annual SAS Users Group International Conference.

SAS Institute Inc. (1990), SAS Guide to TABULATE Processing, Second Edition. Cary, NC: SAS Institute Inc.

SAS Institute Inc., SAS Institute Web Tools. http://www.sas.com/rnd/web.

Author Contact Information

Jeffrey Abolafia
University of North Carolina
Department of Biostatistics CB #803
Chapel Hill, NC 27514
uccjma.cscc@mhs.unc.edu

Steve Noga
Rho Inc.
121 S. Estes Dr., Suite 100
Chapel Hill, NC 27514
snoga@rhoworld.com