Paper SA19-2011

A Lazy Programmer's Macro for Descriptive Statistics' Tables

Matthew C. Fenchel, M.S., Cincinnati Children's Hospital Medical Center, Cincinnati, OH
Gary L. McPhail, M.D., Cincinnati Children's Hospital Medical Center, Cincinnati, OH
Rhonda D. VanDyke, Ph.D., Cincinnati Children's Hospital Medical Center, Cincinnati, OH

Abstract

The purpose of this macro (%DSTMAC) is to provide a flexible tool that outputs a reader-friendly descriptive statistics table (DST) -- complete with special characters & formatting -- with only the minimal necessary input. The user is able to specify a "By" variable (separate DST's will be produced for each level of "By") and a "Group" variable (statistical comparisons can be made across different levels of "Group"). However, the user does not have to specify which variables are categorical or continuous, how many levels of the "By" there are, nor how many levels of the "Group" variable there are. Means ± standard deviations (n) are produced for continuous variables; n (%) are produced for categorical variables. If the user desires a comparison between "Groups", p-values for t-tests / ANOVA and Kruskal-Wallis nonparametric tests are produced for continuous variables. For categorical variables, p-values for the likelihood ratio chi-square test and Fisher's exact test are calculated.

Introduction

Almost every study or statistical analysis requires some type of descriptive statistics table (DST). Early in the study, such a table provides investigators with an initial "picture" of the data characteristics. In a final grant or manuscript, a DST provides important background information about the data that were used in the final analyses. While such tables are often described as providing "descriptive statistics", initial (and inferential) comparisons between study groups or time-points are very often included in the form of p-values.

These tables may sound simple, but they can be time-consuming to create. Variables often need to be individually specified, since it is rare that all variables within the data set are represented in a DST. Different SAS® procedures may be required for the calculation of descriptive statistics and tests, depending on the variable. The presentation of these results may also differ depending on the variable. Groups or time-points need their own columns. Then there is the whole question of how to format the results for both continuous (n, mean, median, SD, IQR, etc.) and categorical (n, %, etc.) variables. Finally, there is the combining of all results into a reader-friendly table which -- in the absence of rather complex SAS® programming -- is probably still often done "by hand".

The purpose of this macro (%DSTMAC) is to provide the statistician, programmer or analyst with a flexible tool that outputs a reader-friendly DST with only the minimal necessary input. Most of this input involves specifying paths, data set names and a title for the output. The user is able to specify a "By" variable (separate DST's will be produced for each level of the "By" variable) and a "Group" variable (comparisons can be made across different levels of the "Group" variable). However, the user does not have to specify which variables are categorical or continuous, how many levels of the "By" there are, nor how many levels of the "Group" variable there are. There is also no requirement to use "By" or "Group" variables.

The macro is designed to be used (via the %INCLUDE statement) in the user's main SAS® program, in which he is working with the data. It can be used by someone with intermediate knowledge of SAS®. Both a final data set and an ODS RTF file are generated.

Example

An example of the direct, unaltered output from the %DSTMAC macro is on page 2. Means ± standard deviations (n) are shown for continuous variables; n (%) are shown for categorical variables. P-values for t-tests / ANOVA (P_F) and Kruskal-Wallis nonparametric tests (P_KW) are produced for continuous variables. For categorical variables, p-values for the likelihood ratio chi-square (P_LR) and Fisher's exact test (P_Fish) are calculated.

The "Sub-Cat" column lists the various levels for each categorical variable. For such variables, the corresponding p-values are listed more than once -- i.e. for each level of the respective categorical variable. (We hope to change this in a future version of this macro.)

The "Group" variable in this case was "Gender", as noted in the fourth line of the title. There was no "By" variable in this case (third line of the title). The first line of the title is specified by the user. Otherwise, the other three title lines are generated automatically by the program.

The values for “n = “ in the column headings show the total number of patients in that level of “Group”. The values for n (shown in parenthesis after each mean and standard deviation) show how many patients in that level of “Group” had a measurement for that variable.

The output generated in the *.rtf document is in landscape orientation.

The data used for this and all examples in this paper come from the Cystic Fibrosis Center at Cincinnati Children’s Hospital Medical Center. An original purpose of this study was to determine whether levels of serum vitamin D affect lung function in patients with cystic fibrosis, as measured by forced expiratory volume over one second percent predicted (FEV1Pct). This table shows descriptive statistics and tests by Gender, since key health outcomes in cystic fibrosis can vary by sex.

Vitamin D Study - First Annual Obs
Mean ± SD (n) or Counts (%)
By Variable Level = None
Statistics & Comparisons by Gender

Variable	Sub-Cat	Female (n = 106)	Male (n = 107)	Compare1	P-F1	P-KW1	P-LR1	P-Fish1	VarNumber
Age		10.17 ± 3.79 (106)	10.86 ± 3.74 (107)	Female-Male	0.1831	0.1375	.	.	1
BMI		17.38 ± 4.98 (106)	17.72 ± 4.06 (107)	Female-Male	0.5866	0.1593	.	.	2
FEV1Pct		85.19 ± 24.63 (106)	87.70 ± 22.16 (107)	Female-Male	0.4360	0.4962	.	.	4
VitaminD		27.65 ± 12.82 (104)	25.79 ± 8.79 (107)	Female-Male	0.2184	0.6889	.	.	3
CHMRSA	No	102 (96.23%)	104 (97.20%)	Female-Male	.	.	0.6909	0.7213	5
CHMRSA	Yes	4 (3.77%)	3 (2.80%)	Female-Male	.	.	0.6909	0.7213	5
VitDCat	ge_30	34 (32.69%)	31 (28.97%)	Female-Male	.	.	0.5584	0.6548	8
VitDCat	lt_30	70 (67.31%)	76 (71.03%)	Female-Male	.	.	0.5584	0.6548	8
dF508	A0	11 (10.38%)	10 (9.35%)	Female-Male	.	.	0.2717	0.2839	7
dF508	A1	43 (40.57%)	33 (30.84%)	Female-Male	.	.	0.2717	0.2839	7
dF508	A2	52 (49.06%)	64 (59.81%)	Female-Male	.	.	0.2717	0.2839	7

The %DSTMAC macro is designed to list all continuous variables first, categorical variables second. Within those two groupings, variables are listed alphabetically (SAS® lists uppercase letters first, lowercase second). However, an additional sorting option is available, using the “VarNumber” column. The “VarNumber” column lists the order a variable had in the original data set. The above table could be easily sorted in Microsoft Word using this column. (Of course, the SAS® code could also be changed internally to sort the data set before the PRINT procedure is used.)

The concept behind the above table is that it is ready for viewing by statisticians and investigators for an initial “look” at the characteristics of the data. Additionally, with only a few alterations in the formatting (within Microsoft Word or other word-processing program), the table could be ready for a grant or manuscript.

Getting Started

As mentioned previously, a main motivation behind the %DSTMAC macro was to develop code that required minimal user input. In addition, it was intended that the %DSTMAC macro be “included” within a user’s main program (in which he may be conducting other analyses with the data). Below is the actual -- and only -- programming statements used to generate the above table.

```
data vitannual3;
  retain ID Age BMI VitaminD FEV1Pct CHMRSA Gender dF508 VitDCat;
  set cf.vitannual2;
  keep ID Age BMI VitaminD FEV1Pct CHMRSA Gender dF508 VitDCat;
run;

%let saspgm='B:\McPhail Gary\Vit D\SAS\MWSUG 2011 Descriptive Stat UPDATED.sas';
%include &saspgm;
```

```
%DSTMAC (vitannual3, ID, NONE, Gender, TST, B:\McPhail Gary\SAS\,
          Descriptive Gender FObsC, Vitamin D Study - First Annual Obs);
```

In the DATA step, only a few things are done. First, with the RETAIN statement, the variables are put in the order desired. The SET statement simply inserts a previously created data set (one that was used for all other analyses). In the KEEP statement, only those variables are kept that should appear in the table.

In the %LET statement, the full path and name of the descriptive statistics program is assigned to the macro variable "saspgm". The %INCLUDE statement then tells SAS® to include and execute all statements from that program -- which in this case is entirely composed of the %DTSMAC macro and supporting comments.

Finally, the %DSTMAC macro is executed, with the specific assignments to the macro variables that will be used. In the %DSTMAC macro itself, the first line of actual code is:

```
%macro DSTMAC (DSNAME, ID, BYVAR, GROUP, TEST, OUTPATH, OUTNAME, TITLE);
```

Definitions for each input (macro variable) follow. (The actual values used in the example can be seen at the top of the page in the %DSTMAC statement.)

DSNAME = Name of data set. If this is not in the work directory (recommended), user will need to specify the full path.

ID = Name of the subject or ID variable in the data set. No statistics will be generated for this variable.

BYVAR = The "By" variable. Separate DST's will be produced for each level of the "By" variable. If there is no "By" variable, then enter the word "NONE" (all caps).

GROUP = The "Group" variable. Levels of "Group" will be compared and p-values produced in the DST. If there is no "Group" variable, then enter the word "NONE" (all caps).

TEST = You have two choices to enter. For most cases, enter "TST" (all caps). If you have a "Group" variable with exactly three (3) levels, and you want overall (ANOVA) results instead of all two-way comparisons, then enter the word "OVERALL" (all caps).

OUTPATH = Specify the path/folder where the final *.rtf document and data set(s) will be sent to.

OUTNAME = Specify the name that you want for the final Word (*.rtf) document.

TITLE = Specify the first line of the title that will appear in the Word (*rtf) output. This title will appear in blue.

Requirements

a) Data set being used must be "stacked" -- i.e. one row per observation. Each variable is in one column only.
b) It is assumed each row is an independent observation for the purposes of descriptive and inferential statistics. In other words, there is no adjustment for repeated measures. (However, having repeated measures will not "crash" the macro.)
c) The columns in the data set must be correctly formatted -- text for categorical variables, numeric for continuous variables.
d) Ideally, the data set should only contain those variables that will be used. Otherwise, the output will contain summary information for variables that may not be needed or desired.
e) If the user wants the output to have a certain order of the variables, those variables should be in that order already. The program or output can then be adjusted to accommodate that. Default output for the variables is alphabetical within variable type -- all continuous variables are listed first, then categorical variables.
f) Using unusual characters (_, -, %, etc.) in levels of the "By" or "Group" program will crash the program.
g) If levels of the "By" or "Group" variables are numerals only (1, 2, 3, etc.) -- even if these variables have a character format -- this will crash the program. For example, if you have a "Group" variable with levels "10", "20", "30", etc., you will need to change these (or create a new variable for "Group") -- perhaps with levels "A10", "A20", "A30", or something similar.

Programming Overview

The %DSTMAC macro relies heavily on imbedded macros, macro variables and %TO-%DO loops. Extensive use of the DATA step, the SQL, FREQ, TRANSPOSE, CONTENTS, and NPAR1WAY procedures, and ODS OUTPUT are used. To attempt to go through the macro “line by line” would exceed the scope of this paper. Rather, the authors have chosen to give an overview of each section (which are so designated in the macro), along with selected SAS® code -- which will hopefully provide the users with a general picture of how the %DSTMAC macro functions. The authors try to point out some of the more “unique” SAS® statements that were particularly helpful in making this macro work.

Section A: Assignment of Macro Variables and Determining Variable Formats

In this section (using DATA steps), all levels of any “By” and “Group” variables are assigned to their own specific -- and numbered -- macro variables. In our example, &GRP1 might equal “Female” while &GRP2 equals “Male”. The purpose of making these assignments is for ease of calculating statistics later, and for use in labels and titles. For doing this, we use the CALL SYMPUTX routine, which assigns a value to a macro variable, but also eliminates all leading and trailing blanks in the process. (We discovered through much trial and error that this latter result was necessary when using values from these macro variables as labels.)

In addition, two macro variables are created that hold the “count” value for the number of levels in the “By” and “Group” variables. Again, in our example, since Gender was our “Group”, the &GRPCOUNT variable = 2.

Below is an example of the code that was used to accomplish what was described above.

```
proc sort data=all1 nodupkey out=temp3;
  by &group;
  where &group ne "";
run;

data temp4;
  set temp3;
  by &group;
  cnt = left(put(_n_,2.));
  y = resolve(&group);
  call symputx("grp"||cnt, y);
  if Last.&group = 1 then do; call symput("grpcount",_n_); end;
run;
```

Next, PROC TRANSPOSE is used to create a new data set (for continuous variables only) and to combine all variable names under one new variable, and all values from all variables under one new variable for each level of “By” (which for our example is “None”). A small section of this new data (de-identified) is shown here.

macroid	Gender	None	Variable
1	Male	8.870	Age
2	Female	8.020	Age
3	Male	9.040	Age
4	Male	10.970	Age
5	Male	6.130	Age
(omitted)			
1	Male	16.723	BMI
2	Female	17.336	BMI
3	Male	14.989	BMI
4	Male	16.922	BMI
5	Male	16.169	BMI

The reason for doing this is to simply the SAS® code that is necessary to calculate statistics and obtain the output. Using the above as an example, statistics will be calculated (only) for the variable “None”, for each level of “Gender” and “Variable.” This saves the headache of actually trying to list all of the variables for which statistics are needed in upcoming procedures.

The reader probably noticed the “macroid” variable in the above table. This is a generated variable that gives each row of data its own unique identification number. In this way, if a user has a data set with more than one observation per subject (i.e. repeated measures), this is “eliminated” through regarding each row as an independent observation.

Finally, using PROC CONTENTS, the number and type of each variable are noted (not including the &ID variable). In our example, “Age” was the first variable in the data set, so its variable number (“VARNUM”) = 1. Since it is numeric, its variable TYPE = 1. For categorical variables, TYPE = 2. The use of the variable number was explained earlier. TYPE is used to determine which variables are analyzed as continuous variables (using PROC SQL and PROC NPAR1WAY), and which variables will be analyzed as categorical variables (with PROC FREQ).

Section B: Statistics and Tests for Continuous Variables

The first part of Section B uses PROC SQL to generate means and standard deviations (for continuous variables only). There is a %TO-%DO loop that goes through each level of the “By” variable and saves separate data sets for each level. The main PROC SQL code is given below.

```
proc sql;
  create table means1_&&byv&i as
  select Variable, VarNum, &group as Group, avg(&&byv&i) as Mean_&&byv&i,
         std(&&byv&i) as STD_&&byv&i, n(&&byv&i) as N_&&byv&i,
           cats("(",put(n(&&byv&i),4.),")") as N2_&&byv&i
  from all2
  group by Variable, VarNum, &group;

  create table means2_&&byv&i as
  select Variable, VarNum, Group, catx(' ',put(Mean_&&byv&i, 6.2), '±',
         put(STD_&&byv&i, 6.2), N2_&&byv&i) as Final
  from means1_&&byv&i;
  quit;
```

The first CREATE TABLE statement is used primarily to generate means and standard deviations for each variable, by “Group.” However, the CATS function is also used to initially place the count for each variable within parentheses, and then save that combination as a new categorical variable. The CATS function concatenates a character string, while removing all leading and trailing blanks.

The second CREATE TABLE statement is used for creating the final combination of mean, standard deviation and n, complete with all special characters. This is saved as a new variable. Using PROC TRANSPOSE (not shown), these final statistics are placed in columns for each level of “Group”, within rows for each continuous variable being analyzed.

The second part of Section B calculates p-values for testing differences between “Groups” for each continuous variable (if a “Group” variable was named). Once again, there is a %TO-%DO loop through each level of the “By” variable. In addition -- by using the number of levels in “Group” (saved in the &GRPCOUNT variable) and by using the user-defined value for &TEST -- the %DSTMAC macro determines whether all pair-wise comparisons will be made between levels of “Group”, or only an overall test.

A couple of additional details should be noted here. There is really only one scenario available where the all pair-wise option is different from the overall test: when there are exactly three levels of “Group” and &TEST = “TST”. Obviously if there are only two levels of “Group”, the pair-wise and overall tests will produce the same results. If there are four or more levels of “Group”, the %DSTMAC macro will only conduct an overall test. This is due to the sheer size of the output that would be generated if all pair-wise comparisons were conducted between four or more levels. (If a user does have such a situation -- four or more levels and all pair-wise comparisons are needed -- the %DSTMAC macro could simply be run more than once, using data sets that only have two or three levels of “Group” represented. Or a “By” variable could be created for different comparisons among “Groups.”)

It is also important to mention at this point that no multiple comparison adjustments are made when all pair-wise comparisons are conducted.

PROC NPAR1WAY -- although being primarily a procedure for non-parametric statistics -- does offer the useful option of obtaining ANOVA statistics as well. Thus, we use this procedure to obtain p-values from both the Kruskal-Wallis test (which reduces to the Wilcoxon rank-sum statistic where there are only two levels of “Group”) and the

ANOVA test (which reduces to a t-test where there are only two levels of “Group”). The %IF-%THEN statement comes into play, only if there are three levels of “Group” and only if all pair-wise comparisons are requested.

```
  proc npar1way anova wilcoxon data=all2;
    by Variable notsorted;
    var &&byv&i;
    class &group;
    %if &n = 3 %then %do;
      where &group ne "&&grp&j";
      %end;
    output out=test(keep = Variable _VAR_ P_F P_KW) anova wilcoxon;
    run;
```

In the later part of this section, DATA steps (again using the CATS function) and PROC SQL are used to create the “Compare1”, “P-F1” and “P-KW1” columns shown in our original example (page 2). These are merged with the previous created columns for each level of “Group” -- in the example on page 2, the “Female” and “Male” columns.

Section C: Statistics and Tests for Categorical Variables

The obvious purpose of Section C is to duplicate for categorical variables what was done for continuous variables. In this case, frequencies and percentages will be calculated, along with p-values from tests for association (likelihood chi-square ratio test and Fisher’s exact test). PROC FREQ is used for the statistics and p-values.

First (using similar DATA steps as were previously described and used), macro variables are created for each categorical variable, a macro variable is created with the count of the number of categorical variables, and a new data set is created with the “macroid”, “Group”, and categorical variables only.

The next step is to use PROC FREQ to simply determine how many subjects are in each level of “Group”, for each level of the “By” variable.

```
  proc freq data=catdata;
    where &byvar = "&&byv&i";
    tables &group;
    ods output OneWayFreqs=F_&group;
    run;
```

These counts will be used at the top of each “Group” column (as shown in the table on page 2).

The main calculation of counts and percentages -- by categorical variable and each level of “Group” -- is next.

```
  proc freq data=catdata;
    where &byvar = "&&byv&i";
    tables &group*&&catvar&j / chisq crosslist;
    ods output CrossList=F_&&catvar&j;
    run;
```

The CROSSLIST option and the CROSSLIST ODS table was a helpful feature, which displays the cross-tabulation tables in a column format (ODS) instead of the usual cell format. An example of the output is given on the next page.

DATA steps and PROC TRANSPOSE are then used to combine the counts and percentages for each variable in the respective columns for the difference levels of “Group”. The code is slightly more complicated than what was used for the continuous variables because categorical variables usually have two or more levels.

The second half of this section involves calculating the likelihood ratio chi-square and Fisher p-values for test of association. Once again, the %IF-%THEN statement comes into play, only if there are three levels of “Group” and only if all pair-wise comparisons are requested.

```
  proc freq data=catdata;
    where &byvar = "&&byv&i";
    %if &n = 3 %then %do; where &byvar = "&&byv&i" and &group ne "&&grp&j"; %end;
    tables &group*&&catvar&k / chisq fisher;
    ods output ChiSq=astest&&catvar&k(where=(Statistic="Likelihood Ratio Chi-Square"))
               FishersExact = fisher&&catvar&k(where = (Name1 = "XP2_FISH"));
    run;
```

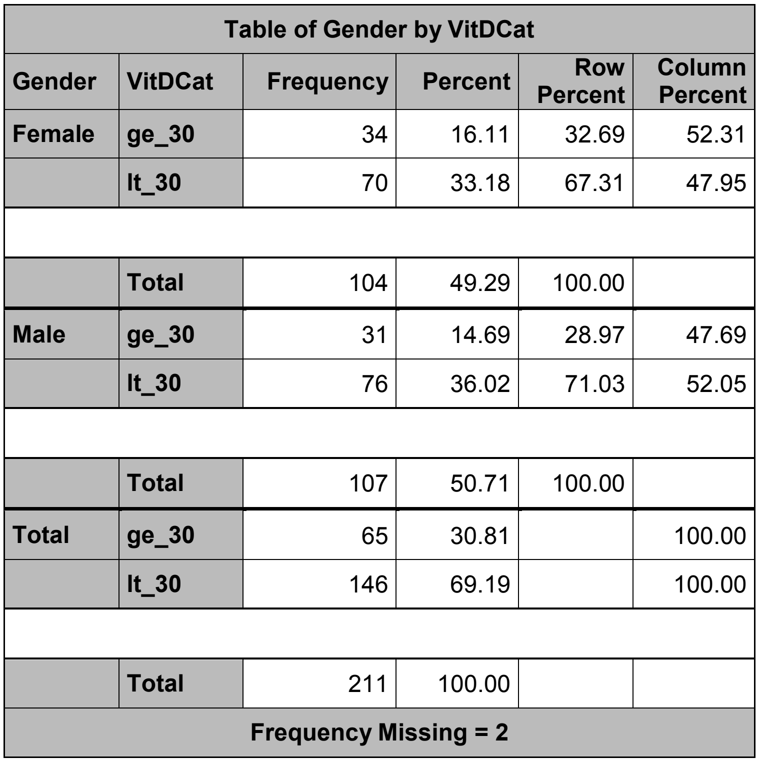

Table of Gender by VitDCat					
Gender	VitDCat	Frequency	Percent	Row Percent	Column Percent
Female	ge_30	34	16.11	32.69	52.31
	lt_30	70	33.18	67.31	47.95
	Total	104	49.29	100.00	
Male	ge_30	31	14.69	28.97	47.69
	lt_30	76	36.02	71.03	52.05
	Total	107	50.71	100.00	
Total	ge_30	65	30.81		100.00
	lt_30	146	69.19		100.00
	Total	211	100.00		
Frequency Missing = 2					

One issue that can arise -- especially with data sets that have low numbers of observations -- is that the ODS OUTPUT data set will not be created, because one or more levels of “Group” have zero observations for a given categorical variable. This will cause an error in the next DATA step, as it tries to read a non-existent data set. Fortunately, we can check for that using the following statements:

```
%if %sysfunc(exist(astest&&catvar&k)) %then %goto cont1;
  %else %goto exit1;
```

The %SYSFUNC function simply determines whether the data set exists or not. If it does exist, then the first %GOTO function directs SAS® to the next DATA step for further processing of the data set produced by PROC FREQ. If the data set fails to be created, then the second %GOTO function directs SAS® to the end of the %DO-%TO loop that is analyzing each categorical variable individually. At that point, the program moves on to the next categorical variable, or -- if there are no more to be analyzed -- this section comes to an end.

If the data set does exist, then the DATA step is used to combine results and create the “Compare1”, “P-LR1” and “P-Fish1” columns.

```
data Not_&&grp&j; retain &byvar Variable Compare1 Compare2 Compare3 P_LR&j P_Fish&j;
  set both&&catvar&k;
  Variable = "&&catvar&k";
  &byvar = "&&byv&i";
  P_LR&j = round(Prob, .0001); label P_LR&j = "P-LR&j";
  P_Fish&j = round(NValue1, .0001); label P_Fish&j = "P-Fish&j";
  %if &j = 1 %then %do;
     %if &n = 3 %then %do; Compare1 = cats("&grp2","-","&grp3"); %end;
     %if &n ne 3 %then %do; Compare1 = cats("&grp1","-","&grp2"); %end;
     %if &test = OVERALL %then %do; Compare1 = "OVERALL"; %end;
     keep &byvar Variable P_LR&j P_Fish&j Compare1 Compare2 Compare3;
     run; %end;
  %if &j = 2 %then %do; Compare2 = cats("&grp1","-","&grp3");
    keep &byvar Variable P_LR&j P_Fish&j Compare1 Compare2 Compare3; run; %end;
  %if &j = 3 %then %do; Compare3 = cats("&grp1","-","&grp2");
    keep &byvar Variable P_LR&j P_Fish&j Compare1 Compare2 Compare3; run; %end;
  run;
```

The code above may seem complicated (and it is to an extent), but there is an overall description. Most of the code is involved in preparing the “Compare” columns. As described before, there will be either only one “Compare” column in the final table -- either showing the two levels of “Group” being compared, or showing that an overall test was done for three or more groups -- or there will be three “Compare” columns, one for each of the three pair-wise comparisons that will be done when there are exactly three levels of “Group” and the user wants all pair-wise comparisons.

The interesting thing about this section of code is the ability -- within a macro -- to run imbedded &IF-&THEN-&DO loops that dictate what will be produced in the DATA step. This allows the user to take information that is external to the data set and use it to decide how to process the DATA step.

The final DATA steps in this section combine the frequency statistics with the test statistics for categorical variables.

Section D: Final Merging of Data Sets and Word Output

This section is probably the most “straightforward” of all: the final continuous data set (containing statistics and p-values) is stacked or set with the final categorical data set (containing statistics and p-values). Again using imbedded &IF-&THEN-&DO loops (depending on how many levels of “Group” there were), the variables are put in order that appears in the final table (see page 2). ODS RTF FILE statements and the PRINT procedure are used to output the final table(s) to an *rtf file. (Obviously this could be changed to other output types, if the user so desired.)

As mentioned earlier, there are three TITLE statements that are pre-programmed (the user specifies the first TITLE at the beginning of the program). Again, of course, the user could go into the main code and change any of these three titles, if he so desired.

Additional Examples

We thought showing a couple of different examples of the output might be helpful to readers. The first example shows the output when there is no “Group” variable. In other words, no comparisons are done. This might be useful if an investigator simply wants an overview of the entire data set, prior to looking at variables by some “Group”. The output given below is preceded by the macro statement used to produce the table.

```
  %DSTMAC (vitannual3, ID, NONE, NONE, TST, B:\McPhail Gary\Vit D\SAS\,
          Descriptive All FObsB, Vitamin D Study - First Annual Obs);
```

Vitamin D Study - First Annual Obs

Mean ± SD (n) or Counts (%)

By Variable Level = None

Statistics Only

No Tests (No Group Variable Defined)

Variable	Sub-Cat	Statistics (n = 213)	VarNumber
Age		10.52 ± 3.77 (213)	1
BMI		17.55 ± 4.53 (213)	2
FEV1Pct		86.45 ± 23.40 (213)	4
VitaminD		26.71 ± 10.97 (211)	3
CHMRSA	No	206 (96.71%)	5
CHMRSA	Yes	7 (3.29%)	5
Gender	Female	106 (49.77%)	6
Gender	Male	107 (50.23%)	6
VitDCat	ge_30	65 (30.81%)	8
VitDCat	lt_30	146 (69.19%)	8
dF508	A0	21 (9.86%)	7
dF508	A1	76 (35.68%)	7
dF508	A2	116 (54.46%)	7

Please notice in this table that “Gender” is included in the “Variable” column. In the table on page 2, Gender was obviously not listed in that column, since it was the “Group” variable.

The next example shows a “Group” variable -- dF508 -- with three levels. In this example, we only want overall tests performed, not all pair-wise comparisons. (To save space, the “VarNumber” column was deleted.)

Vitamin D Study - First Annual Obs
Mean ± SD (n) or Counts (%)
By Variable Level = None
Statistics & Comparisons by dF508

Variable	Sub-Cat	A0 (n = 21)	A1 (n = 76)	A2 (n = 116)	Compare1	P-F1	P-KW1	P-LR1	P-Fish1
Age		11.44 ± 3.69 (21)	10.79 ± 3.75 (76)	10.17 ± 3.79 (116)	OVERALL	0.2662	0.1550	.	.
BMI		17.49 ± 2.76 (21)	17.29 ± 2.45 (76)	17.73 ± 5.70 (116)	OVERALL	0.7998	0.6516	.	.
FEV1Pct		83.66 ± 29.31 (21)	85.77 ± 23.93 (76)	87.40 ± 21.99 (116)	OVERALL	0.7592	0.9647	.	.
VitaminD		26.80 ± 8.84 (21)	25.75 ± 10.24 (75)	27.31 ± 11.79 (115)	OVERALL	0.6362	0.5475	.	.
CHMRSA	No	21 (100.00%)	75 (98.68%)	110 (94.83%)	OVERALL	.	.	0.1566	0.3438
CHMRSA	Yes	0 (0.00%)	1 (1.32%)	6 (5.17%)	OVERALL	.	.	0.1566	0.3438
Gender	Female	11 (52.38%)	43 (56.58%)	52 (44.83%)	OVERALL	.	.	0.2717	0.2839
Gender	Male	10 (47.62%)	33 (43.42%)	64 (55.17%)	OVERALL	.	.	0.2717	0.2839
VitDCat	ge_30	9 (42.86%)	22 (29.33%)	34 (29.57%)	OVERALL	.	.	0.4698	0.4593
VitDCat	lt_30	12 (57.14%)	53 (70.67%)	81 (70.43%)	OVERALL	.	.	0.4698	0.4593

Finally, in the Appendix (placed there due to the length of the output, in which we needed to use landscape orientation), we shown an example of the same “Group” with three levels -- however, requesting all pairwise comparisons. The macro statement used to produce that output is also given in the Appendix.

Conclusion

Unfortunately, the program is too long to include in this paper -- but the primary author is glad to send it via e-mail upon request. Please see the contact information below.

As we mentioned in the beginning, one of our main goals was to create a user-friendly program that is easy to implement (with a few inputs into the macro statement), that can produce ready-to-read descriptive statistics tables. Besides our own use, a number of statisticians within the Division of Biostatistics and Epidemiology at Cincinnati Children’s Hospital Medical Center have successfully used this program with actual data that they are analyzing. It is our hope that it will be helpful to a wider audience as well.

Future Work

For future work, the authors would like to make a few improvements.

1) For categorical variables (which have the p-value repeated in the output table), create some type of replacement for the duplicated p-values. (Perhaps “see p-value above”, or something similar.)

2) Create an option to request medians and interquartile ranges, instead of means and standard deviations. The challenge with this option is that PROC SQL cannot be used to calculate medians for columns. In other words, the current PROC SQL code that is currently being used for means and standard deviations cannot simply be altered or modified. Most likely, separate code using the MEANS procedure will have to be written.

3) Create options where the user can choose *not* to print certain columns in the table. For example, perhaps the user does not need to see the “VarNumber” column -- or does not need both sets of p-values for the continuous variables or categorical variables. While those columns can be easily deleted from the current table, not printing them in the first place would save a little time and perhaps some much needed space in the printout.

4) When no “Group” variable is specified, calculate test statistics for a) whether means for continuous variables are significantly different from zero, and b) whether proportions differ between different levels of categorical variables.

Acknowledgements

We would like to thank the following people for beta-testing this program and for their insightful input into its final development. All of these researchers work in the Division of Biostatistics and Epidemiology at Cincinnati Children’s Hospital Medical Center.

Lynn Darbie, M.S., Biostatistician
Patricia Herbers, M.S., Biostatistician
Chunyan Liu, M.S., Biostatistician
Jesse Pratt, M.A., M.S., Biostatistician
Meredith Tabangin, M.P.H, Senior Epidemiologist

Contact Information

Comments, questions and requests for the %DSTMAC macro are welcome. The author may be contacted at:

Matthew Fenchel
Division of Biostatistics and Epidemiology
Cincinnati Children’s Hospital Medical Center
MLC 5041, 3333 Burnet Avenue
Cincinnati, OH 45229-3039

E-mail: Matthew.Fenchel@cchmc.org

Appendix

The table below is an example of the output when there are three levels of “Group” and all pair-wise comparisons are requested. For the sake of space, we deleted the “VarNumber” column, all “P_KW” columns and all “P_Fish” columns. The macro statement used to produce this output is shown immediately below. (By way of reminder, no multiple comparison adjustments are made for the p-values.)

```
%DSTMAC (vitannual3, ID, NONE, dF508, tst, B:\McPhail Gary\Vit D\SAS\, Descriptive dF508 Two-Way,
         Vitamin D Study - First Annual Obs);
```

Vitamin D Study - First Annual Obs
Mean ± SD (n) or Counts (%)
By Variable Level = None
Statistics & Comparisons by dF508

Variable	Sub-Cat	A0 (n = 21)	A1 (n = 76)	A2 (n = 116)	Compare1	P-F1	P-LR1	Compare2	P-F2	P-LR2	Compare3	P-F3	P-LR3
Age		11.44 ± 3.69 (21)	10.79 ± 3.75 (76)	10.17 ± 3.79 (116)	A1-A2	0.2643	.	A0-A2	0.1572	.	A0-A1	0.4829	.
BMI		17.49 ± 2.76 (21)	17.29 ± 2.45 (76)	17.73 ± 5.70 (116)	A1-A2	0.5200	.	A0-A2	0.8481	.	A0-A1	0.7454	.
FEV1Pct		83.66 ± 29.31 (21)	85.77 ± 23.93 (76)	87.40 ± 21.99 (116)	A1-A2	0.6270	.	A0-A2	0.4981	.	A0-A1	0.7351	.
VitaminD		26.80 ± 8.84 (21)	25.75 ± 10.24 (75)	27.31 ± 11.79 (115)	A1-A2	0.3516	.	A0-A2	0.8514	.	A0-A1	0.6717	.
CHMRSA	No	21 (100.00%)	75 (98.68%)	110 (94.83%)	A1-A2	.	0.1354	A0-A2	.	0.1526	A0-A1	.	0.4836
CHMRSA	Yes	0 (0.00%)	1 (1.32%)	6 (5.17%)	A1-A2	.	0.1354	A0-A2	.	0.1526	A0-A1	.	0.4836
Gender	Female	11 (52.38%)	43 (56.58%)	52 (44.83%)	A1-A2	.	0.1108	A0-A2	.	0.5235	A0-A1	.	0.7322
Gender	Male	10 (47.62%)	33 (43.42%)	64 (55.17%)	A1-A2	.	0.1108	A0-A2	.	0.5235	A0-A1	.	0.7322
VitDCat	ge_30	9 (42.86%)	22 (29.33%)	34 (29.57%)	A1-A2	.	0.9727	A0-A2	.	0.2380	A0-A1	.	0.2490
VitDCat	lt_30	12 (57.14%)	53 (70.67%)	81 (70.43%)	A1-A2	.	0.9727	A0-A2	.	0.2380	A0-A1	.	0.2490