

Cut the Key Entry and Contain the Corrections: Automating SAS Systems with OUTPUT and MERGE features in MACROS.

David J. Ross
Lupetin Consultants, Inc.

INTRODUCTION

SAS[®] systems developed by novice users frequently involve manual entry as an intermediate step between what appears to be disparate processes. With a clear set of design goals, an understanding of the OUTPUT statement of procedures, the MERGE statement, and the MACRO language, you can implement a "seamless" processing of complex information.

The goal of the original process was to generate estimated regression coefficients for a given model, the estimated intercept, the predicted values, and the standard error of the mean predicted value using Stepwise Regression of PROC REG. These values are based on the questionnaires received during the previous year (approximately 11,000). These values are used as a basis for Period Reports for individual clients so that they can compare their present performance to the previous year.

The processing was:

1. Input Raw Data and Validate Data Entry
2. Create SAS Data Set of Scale Variables
3. Perform Regressions
4. Print Regression Results
5. Print Standard Error of Mean Predicted
6. Values, Summarize, and Average
7. Evaluate Regression Results
8. Data Entry of Estimated Regression
9. Coefficients for each model
10. Data Entry of Intercept for each model
11. Data Entry of Average Standard Error of Mean Predicted Value for each model

Not surprisingly, all of the Data Entry steps required careful proofing for possible numeric errors and transformations. Furthermore, only those variables which contributed to the equations for that year were included in the final computations, leading to questions concerning which variables were actually available in the original computations. Finally, considerable staff time was required for all of these tasks.

DESIGN GOALS

It is important to establish what the goals are: first, to generate an equation for dependent variables for each category, and second, to eliminate the Data Entry steps, reducing clerical errors, staff time, and frustration.

To accomplish this, a system was designed to output a SAS program which could be included into the table generation program used to generate Period Reports for clients. Individual models were needed for each. In this way, a separate SAS program makes it possible to generate a new set of equations at any time without modifying the table generation program.

Consequently, the Design Specification Tasks were:

1. Generate regression equation for a dependent variable
2. Generate output for each category
3. Summarize Standard Error of Mean Predicted Values
4. Merge Estimated Regression Coefficients for each model
5. Attach values for multiple dependent variables
6. Output predictive equations for each category and multiple dependent variables into a SAS program

IMPLEMENTATION

The use of the SAS Macro Language lets us create structured groups of SAS code for each of the tasks outlined.

Processing is performed for multiple categories. This accomplished by:

```
*** Set for Each Category of Values ;
*** CATONE*** ;
    %MAINMACR((DX=9 OR DX=13),CATONE) ;
*** CATWO *** ;
    %MAINMACR((DX=11 OR DX=12),CATWO) ;
```

where the first parameter (DX=11 OR DX=12) is used in an IF statement to select the category, and the second parameter is used to define a SAS Data Set for a category.

```
**** MAIN PROCESSING MACRO ;
    %MACRO MAINMACR(PS1,PS2) ;
    *** USE FOR SAS DATA SET FOR CATEGORY;
        %GLOBAL WHICHDS ;
    *** USE FOR IF STATEMENT CATEGORY SELECTION;
        %GLOBAL DXSELECT ;
        %LET WHICHDS=&PS2;
        %LET DXSELECT=&PS1;
    *** DESIGN SPECIFICATION TASKS 1 - 5;
        %REGMACR ;
    *** DESIGN SPECIFICATION TASK 6;
        %PRNTMACR ;
    %MEND MAINMACR ;
```

Although the macro %REGMACR does most of the work, the tasks accomplished deal with file creation and data manipulation. The macro

creation and data manipulation. The macro %PRNTMACR deals with the creation of a SAS program for other uses. Thus they are treated separately in the program and in this paper.

REGRESSION AND FILE MANIPULATION (%REGMACR)

The regression and file manipulation tasks are accomplished by:

```
%MACRO REGMACR ;
*** MAKE SURE THE TARGET SSD IS CLEAR;
    data TEST.&whichds ;
        set TEST.&WHICHDS;
        delete ;
*** BEGIN PROCESSING ;
DATA REG;
    SET temp.finyr ;
TITLE "processing for &WHICHDS where
DX = &dxselect";
*** SELECT ONLY CASES FOR THE DESIRED CATEGORY ;
IF &DXSELECT ;
*** CALCULATE MEANS OF VALUE AND SAVE ;
%RDEPVAR1 ;
    %SUMM;
    %MERGEALL;
    %ATTACH;
RUN;

%RDEPVAR2;
    %SUMM;
    %MERGEALL;
    %ATTACH;
RUN;
```

Although these statements could be placed in a %DO loop, different processing may be required for each dependent variable. Consequently, additional macros could be included here. For presentation, only 2 dependent variables are used, with the following describing one of the dependent variables.

Macro %RDEPVAR1

The regression and output files are generated for each dependent variable by the following:

```
*** Generate coefficients for PREDV1 ;
%MACRO RDEPVAR1 ;
    PROC reg DATA = REG outest = est;
    MODEL DEPVAR1 = indvar1 indvar2 indvar3 indvar4
indvar5
    / selection = stepwise ;
    output out=pred p=pred stdp=prederr ;
%MEND RDEPVAR1;
```

The outest = est statement places the parameter estimates, model name, dependent variable name, and the estimated intercept into a SAS data set. In this example, the contents are:

```
Data Set Name: WORK.EST
Observations: 1
```

```
Variables: 11
# Variable Type Len Pos Label
11 DEPVAR1 Num 8 84 DEPENDENT VARIABLE
6 INDVAR1 Num 8 44 INDEPENDENT VARIABLE
7 INDVAR2 Num 8 52 INDEPENDENT VARIABLE
8 INDVAR3 Num 8 60 INDEPENDENT VARIABLE
9 INDVAR4 Num 8 68 INDEPENDENT VARIABLE
10 INDVAR5 Num 8 76 INDEPENDENT VARIABLE
5 INTERCEP Num 8 36 Intercept
3 _DEPVAR_ Char 8 20 Dependent variable
1 _MODEL_ Char 8 4 Label of model
```

```
OBS _MODEL_ _DEPVAR_ INTERCEP INDVAR1 INDVAR2
1 MODEL1 DEPVAR1 2.34851 0.018590 0

INDVAR3 INDVAR4 INDVAR5 DEPVAR1
-.0000022914 0 0 -1
```

The output out=pred p=pred stdp=prederr ; statement places the predicted values and standard error of the mean predicted value for each observation of a category into a SAS data set. In this example, the contents are:

```
Data Set Name: WORK.PRED1
Observations: 39
Variables: 2
#Variable Type Len Pos Label
1 PRED Num 8 4 Predicted Value of
DEPVAR1
2 PREDERR Num 8 12 Standard Error of Mean
Predicted Value
```

Macro %SUMM

It is necessary to reduce the variable PREDERR to a single value across all observations of a category. This is accomplished by using PROC SUMMARY to calculate the actual mean. This is accomplished by:

```
*** Macro to summarize PRED dataset ;
*** P EQUALS PREDICTED VALUES ;
*** STDP STANDARD ERROR OF MEAN PRED VALUE ;
%MACRO SUMM ;
data pred1 ;
    set pred ;
    keep pred prederr ;
proc summary ;
    var pred prederr ;
    output out=pred1 mean= ;
run;
    data pred ;
        set pred1 ;
        drop _type_ _freq_ ;
run;
%MEND SUMM;
```

Macro %MERGEALL

The next task is to merge these values into a single SAS data set for further processing. This is accomplished by:

```
*** MERGE Coefficient file with error estimate file
;
```

```
%MACRO MERGEALL ;
  data mergeall ;
    merge est pred ;
    drop _rmse_ _type_ ;
run;
%MEND MERGEALL ;
```

In this example, the contents are:

```
Data Set Name: WORK.MERGEALL
Observations: 1
Variables: 11
# Variable Type Len Pos Label
9 DEPVAR1 Num 8 68 DEPENDENT VARIABLE
4 INDVAR1 Num 8 28 INDEPENDENT VARIABLE
5 INDVAR2 Num 8 36 INDEPENDENT VARIABLE
6 INDVAR3 Num 8 44 INDEPENDENT VARIABLE
7 INDVAR4 Num 8 52 INDEPENDENT VARIABLE
8 INDVAR5 Num 8 60 INDEPENDENT VARIABLE
3 INTERCEP Num 8 20 Intercept
10 PRED Num 8 76 Predicted Value
11 PREDERR Num 8 84 Standard Error
2 _DEPVAR_ Char 8 12 Dependent variable
1 _MODEL_ Char 8 4 Label of model
```

```
OBS _MODEL_ _DEPVAR_ INTERCEP INDVAR1 INDVAR2
1 MODEL1 DEPVAR1 2.34851 0.018590 0

INDVAR3 INDVAR4 INDVAR5 DEPVAR1 PRED PREDERR
-.0000022914 0 0 -1 2.83258 0.082752
```

Macro %ATTACH

A single SAS data set of all the dependent variables is required for processing a category. This is accomplished by:

```
*** build a single sas file with all the variables ;
%MACRO ATTACH ;
  data TEST.&whichds ;
    set TEST.&whichds mergeall ;
run;
%MEND ATTACH ;
```

This last macro accomplishes all of the necessary data file manipulations.

SAS PROGRAM CREATION (%PRNTMACR)

A SAS program is created with the use of PUT statements. This is accomplished by:

```
%MACRO PRNTMACR ;
  data coeff ;
    set TEST.&whichds ;
    FILENAME FORPRN "/wisas/allcoef.sas";
    file FORPRN NOPRINT notitle ps=60 mod ;
    whchone = SYMGET('whichds');
    dselect = SYMGET('dxselect');
    mkname = right(WHCHONE)||'.SAS';
    call symput('mkname',mkname);
  if _n_ = 1 then DO;
    put '*** ' whchone '***' ;
    put 'IF ' dselect ' THEN DO;' ;
```

```
  end;
  if _depvar_ = "DEPVAR1 " then DO;
    %DEPVAR1 ;
  end;
  else if _depvar_ = "DEPVAR2 " then DO;
    %DEPVAR2 ;
  end;
  if _n_ = 2 then DO;
    put 'END;' ;
  end;
RUN ;
%MEND PRNTMACR ;
```

Although these statements could be placed in a %DO loop, different processing may be required for each dependent variable. Consequently, additional macros could be included here.

Macro %DEPVAR1 and %DEPVAR2

The SAS program file "ALLCOEF.SAS" is generated using the following PUT statements:

```
%MACRO DEPVAR1;
  put "**** COEFFICIENTS FOR DEPVAR1 ; " ;
  put @1 'PREDV1 = ' ;
  put @5 '(' INDVAR1 '*indvar1) +' ;
  put @5 '(' INDVAR2 '*indvar2) +' ;
  put @5 '(' INDVAR3 '*indvar3) +' ;
  put @5 '(' INDVAR4 '*indvar4) +' ;
  put @5 '(' INDVAR5 '*indvar5) +' ;
  put @5 '(' INTERCEP ')+ ' ;
  put @5 '-1*(' PREDERR '*1.96) ; ' ;
%MEND DEPVAR1;
```

```
%MACRO DEPVAR2 ;
  put "**** COEFFICIENTS FOR DEPVAR2 ; " ;
  put @1 'PREDV2 = ' ;
  put @5 '(' INDVAR3 '*indvar3) +' ;
  put @5 '(' INDVAR4 '*indvar4) +' ;
  put @5 '(' INDVAR5 '*indvar5) +' ;
  put @5 '(' INDVAR6 '*indvar6) +' ;
  put @5 '(' INDVAR7 '*indvar7) +' ;
  put @5 '(' INTERCEP ')+ ' ;
  put @5 '-1*(' PREDERR '*1.96) ; ' ;
%MEND DEPVAR2 ;
```

The "END" statement is placed at the end of the DO function by the if _n_ = 2 then DO; code in %PRNTMACR. The resultant SAS program contains the following:

```
*** CATONE *** ;
IF (DX=9 OR DX=13) THEN DO;
*** COEFFICIENTS FOR DEPVAR1 ;
PREDV1 =
(0.0185899599 *indvar1) +
(0 *indvar2) +
(-2.291441E-6 *indvar3) +
(0 *indvar4) +
(0 *indvar5) +
(2.3485118119 )+
-1*(0.08275162 *1.96) ;
*** COEFFICIENTS FOR DEPVAR2;
PREDV2 =
(0 *indvar3) +
(0 *indvar4) +
```

```

(0 *indvar5) +
(-1.127375E-6 *indvar6) +
(0 *indvar7) +
(9.6091952509 )+
-1*(0.0821171282 *1.96) ;
END;

```

This SAS program is incorporated into the table generation program with an %INCLUDE ALLCOEF statement for processing.

CONCLUSION

The system documented above has accomplished the stated goals of the project: to generate an equation for dependent variables for each category which can be inserted into another SAS program to create Period Reports for clients, and second, to eliminate the Data Entry steps, reducing clerical errors, staff time, and frustration. By implementing the solution with MACRO statement, the process has been transformed into discreet functional components. This makes the program more maintainable and understandable.

Finally, the following suggestions can guide you in evaluating present systems and implementing better ones.

1. State a few goals; even if they are very general, they form a clear reference point to strive for.
2. Refer to these goals every time you consider a data step or procedure. The MERGE statement lets you simplify data for a variety of uses. The OUTPUT statement lets you pass information between apparently disparate processes.
3. If you need to perform the same task more than once, create a MACRO.
4. Don't over MACRO! Plan for future changes by keeping similar code separate if there is a chance it will change. Don't force others to draw an elaborate road map connecting you MACRO statements. Elegance should be defined by clarity and not reduction.
5. When reviewing previous systems, separate Form from Function, and then focus on Function. "ifit ain't broke, don't fix it" is a Functional issue, whereas the Form may all too often convince you it really is broken and needs fixing.

TRADEMARK NOTICE

SAS is a registered trademark of SAS Institute, Inc., Cary, N.C. USA

AUTHOR

David J. Ross
 Lupetin Consultants, Inc.
 5351 West Winona Street
 Chicago, IL 60630
 Telephone: 312.736.4059