The Application of SAS Macro and Autocall Facility to a Customized Exploratory Data Analyzing System

William A.R. Orban, University of Ottawa
Marc Alary, University of Ottawa

ABSTRACT

The system, utilizing macro, AUTOCALL facility and all Basic procedures under version 5.16 of the SAS product, produces customized outputs for statistical analysis. It was designed to do exploratory data analysis and generate reportable outputs from a number of large data sets. The system can be functionally categorized into the following modules: 1) Primer; 2) Storage; 3) Assembler; 4) Processor.

The Primer module defines the key and current Symbolic macro variables utilizing the system editor. The Storage module contains and catalogues all DATA and PROC step information for a specific predefined output. The Assembler constructs the data set required for processing in either a permanent or temporary form with either standard or arrayed variables. It automatically generates the data set for one of three levels: sample, subsample, and sub-subsample. The Processor modules conditionally select specific SAS statements and options required to execute the desired procedure according to the key macro variables.

The system has seventeen Processor modules each designed to generate a specific output: plots, charts and tables of descriptive and analytic statistics. The main features of this system are its efficiency, manageability, maintainability, and automatic output retrieval capacity:

1. It requires a minimal number of entree changes† to generate a specific output;
2. Modules can be catalogued and easily accessed;
3. Modules can be readily modified and updated;
4. All outputs are efficiently stored and can be easily refined and automatically reproduced.

INTRODUCTION

The purpose of this system is to package all SAS tasks into a system which can automatically execute and generate a selected output for statistical analysis or report rapidly and efficiently. While writing scientific reports it happens that an analysis of some specific data is required to complete the body of evidence. In order to eliminate or at least reduce the loss of continuity of thought in writing a quick and easy access to such results is imperative. This modular system was designed to provide such results. With several single word changes in one file a large range of output results are rapidly available.

The operational (functional) components of this system is one SAS file which primes and drives it, and three types of Macro Modules (Storage, Assembler and Processor) which are members in a MA(6)LIB referenced by the AUTOCALL facility.

THE PRIMER MODULE

The Primer has one basic function: to define macro variables for the global referencing environment. These symbolic variables can be classified by their function into three types: 1) System; 2) DATA step; and 3) PROC step. Their primary function is to make conditional branching decisions, and can also be referenced and resolved to control specific output information.

1. The System symbolic variables, like all macro variables in the Primer, are initialized utilizing the system editor. Since they are global macro variables they can be referenced and resolved in any part of the modular system. They are used as flags for decisions in all job steps. The values of these macro variables identify the project or chapter, the sample and subsample, the item or specific problem, the variable set, and the procedure to be used. These automatically determine the content and form of the output.

2. The second category of symbolic variables can be used to provide fixed and new variable names, operational and subsetting statements for the DATA step. Since this information can be stored in the Storage module, this group of macro variables are only used when an importunate and impromptu question requires a quick answer. Otherwise the Storage module is used by duplicating and modifying a similar previously stored item. This is accomplished by using the system editor to replace, omit, or add the values of the parameters in the basic macro calls stored in the Storage module files. The member must then be replaced in the MA(6)LIB before running the job.

3. Similarly, all the specific parameters and options required for each PROC step are defined by the third type of symbolic variable in the Primer module. For example, PROC ANOVA requires the identification of the independent and dependent variables, the MODEL statement, and the MODEL options to run it. This must be done, unless, as in the DATA step above, the previously stored

† The CMS editor (XEDIT) is used to alter the variable values, however this could be accomplished using the SAS/AF product.
parameter values of the PAREF macro are modified and replaced.

THE STORAGE MODULE

The Storage module is a macro in which are nested macros capable of making conditional branching decisions and invoking macros to generate information required for the DATA and PROC steps. There are two types of macros used to achieve this purpose: 1) Template macro stored in the MACLIB; and 2) Nested macro calls.

1. The template macro contains a %GLOBAL statement and a list of macro statements defining a set of symbolic variables, for example:

```
%LET OPSTM1=OP1;
```

The parameter values are also the symbolic variable values. And the parameters, in the initialized macro, have a null default value. In this way only parameters which have a value in the macro call are effectively utilized. This is essential when a macro variable which is referenced in a SAS statement is not required. For example, a keyword parameter macro, with null default parameter values

```
%OPST(OP1=,...,OP10=);
```

are initialized for 10 operational symbolic variables

```
%LET OPSTM1=OP1;
```

This symbolic variable, referenced in the DATA step (%OPSTM1) must be resolved to prevent an error. A template macro of this type is used to define existing and new variable names, operational and subsetting SAS statements as described in Primer module. The template macro is also used to redefine the symbolic variables initially defined in the Primer module. The parameter values, in this case, are the referenced macro variables as defined in the Primer module. This prevents ambiguity of input information. The template macro, therefore, defines the final global symbolic variables utilized throughout the system.

2. The second type of macro module contains packets of template macro calls. Each packet contains one of each type: 1) Fixed variables names; 2) New variable names; 3) Operational statements; 4) Subsetting statements; and 5) PROC step parameters. The macro parameter values in each packet are related to a specific project and item. A particular packet is conditionally selected and executed by referencing the System symbolic variables defined in the Primer module.

There are six modules of this second type: three for each of the standard and arrayed variables. The three macros nested in the standard or arrayed modules are: 1) Non-stored items, i.e., the values defined in the Primer; 2) Stored but incomplete items, i.e., not finalized; and 3) Stored and completed reportable items.

Since all macros are stored in a MACLIB it is necessary to replace these storage macros after each modification.

THE ASSEMBLER MODULE

The Assembler Module is an orchestrated invocation of macros: 1) to select the source data set(s); 2) To create new group variable names and values; 3) To generate variables for KEEP, BY, CLASS and VAR statements; 4) to subset the variables in sets of four (one independent and 3 dependent) variables for processing; and 5) to run the DATA step and generate a data set for processing.

1. The source data set(s) are automatically selected and flagged by symbolic variables defining the project, sample, and item. The defined source data set name(s) required to satisfy these macro variables are resolved in the SET statement.

2. Symbolic variables for new group variable names and parameters, required to create the variable values, are generated by conditionally branched %DO loops. The symbolic variables which flags the branching is defined in the Primer module.

3. Since the inclusion of BY and CLASS statements vary with each analysis, the BY and CLASS variables are automatically selected and defined by Symbolic variables and included in the KEEP statement.

4. In order to reduce the number of variables to be processed at once, the variables are automatically arranged into sets of 4 variables. This downsizing of the group of variables optimizes the processing.

Creation of small sets of variables is accomplished by redefining the symbolic variable for each one. A template macro is used for this purpose. The parameter values are flagged by a System symbolic variable, defined in the Primer module. Consequently a set of variables, and even a single variable may selected for a particular output procedure.

5. Finally the DATA step generates a permanent or temporary data set either with standard or arrayed variables. These data sets contain only the variables required for the job and only the observations specified by the sample, subsample or sub-subsample selection criteria.

The data set for each level, i.e., sample, subsample or sub-subsample is conditionally selected by subsetting statements in the data step. In addition the sub-sample can be further
subset defining a symbolic variable found in the Primer module, for example

%LET SUBGRP1 = IF AGE = 8;

New variables can be created by defining a symbolic variable in the Primer module and referencing it in a DATA step. If new variables are not created these macro variables must have null values. Similarly functions such as LOG, LAG, MAX, SYMPUT, and EVAL are utilized by using the operational symbolic variables to define the expression. Again the use of the template macro with null default parameters prevents the error of unresolved macro variables from occurring.

A conditional branching of an iterate %DO subroutine using the %EVAL function to generate the new group variables is executed in the DATA step. The symbolic macro variables produced by this subroutine are then referenced and resolved for inclusion in the DATA and PROC step.

The first or sample data set created whether as permanent or temporary, is automatically sorted according to the BY variables which were defined earlier.

THE PROCESSOR MODULE

This modular system contains 17 different Processor modules, each one designed for a specific output and which may include several different PROC steps. The structure of these modules generally include a mixture of SAS model statements, macro statements, macro invocations and macro variable references for the various variables, options, and parameters required to run it. In addition to a macro call for any special system or processing options, there are conditional macro calls to write the following statements; BY, CLASS, VAR, FORMAT, and LABEL.

The main feature in this module is the macro invocation of the TITLE and FOOTNOTE packets. The Title-Footnote macro module is a finely orchestrated series of nested macros, macro statements and definition, referencing, and resolution of macro variables. This module produces a customized title and/or footnotes for any output on any device. It can automatically generate these in an analytic or reportable form.

In the analytic form, the text and information is specific to the data which it contains. All essential information about the number, size and type of observations, the parameters and options which were used to produce the results. This provides a rapid and simple reproduceable referencing for a specific output. The reportable form of the output on the other hand is designed to provide legends and captions with descriptive information relevant to the output table, graph or chart. It is numbered and in an form acceptable for publication.

THE OUTPUT

The output device and the form of the output is very versatile, limited only to the device support by the SAS system. Both are controlled by single symbolic variable values which when referenced automatically branch to the selected device or form.

CONCLUSION

A simple modular system was designed using one SAS file a large number of macro modules for rapid statistical analysis and reporting. The versatility of the system is limited only by the customized results desired. The AUTOCALL facility provides an efficient and unlimited utilization of macro modules, nested macros and macro calls. This in turn provides a limitless capacity to manipulate symbolic variables. The template macro initialized with null default parameter values is the most effective and versatile macro used to optimize user and machine time to obtain rapid and consistent results.

AUTHORS

For more information you may contact:

Marc Alary, (613)564-5965
MMA2F@UOTTAWA.BITNET

William A.R. Orban, (613)564-9125
WAORE@UOTTAWA.BITNET

University of Ottawa,
35 McDougall Lane,
Ottawa, Ontario, Canada
K1N 6N5.

SAS, SAS/AF are registered trademarks of SAS Institute Inc., Cary, NC, USA.