DEVELOPING RELEASE 6.03 SAS/AF® APPLICATIONS AS A STATISTICAL TEACHING AID

L. M. Harschnitz, Dofasco Inc.
L. A. Abell, Dofasco Inc.

Introduction

Dofasco Inc., located in Hamilton, Ontario, is Canada's largest integrated steel producer. The company's steel operations in Hamilton employ approximately twelve thousand people and produces over four million tons of flat rolled steel products per year. In the early 1980's, like most North American manufacturers, Dofasco found that it had to begin to improve quality at a much faster rate than it had in the past. In order to do this, Management realized that it had to provide new tools for the employees to work with. The obvious choices were Statistics and Statistical Process Control techniques. To promote the use of these techniques, a statistical and statistical process control training program was developed. SAS® software was chosen as a data analysis tool.

As this training program developed, several version 6.03 SAS/AF applications were written for Personal Computer workstations that have 640K of main memory. These applications are used to enhance the training program and ease teaching burdens. This paper gives an overview of the development of this program and discusses the development and use of these applications in detail. Special attention is given to tips for developing SAS/AF applications for Personal Computers with only 640K.

The Statistical Training Program at Dofasco

Statistical training began at Dofasco with the creation of the Quality Training Centre in 1984. At that time the Centre had two permanent staff members and a number of part time trainers. Today, there are nine full time staff members, six of whom are active trainers. At the same time, the number of courses taught has gone from two to eight.

Statistical training begins with one of two basic courses in Statistical Process Control (SPC). SPC I is a two day awareness course in SPC techniques. Students learn how to construct Cause and Effect diagrams, plot and interpret Shewhart control charts, and understand the concepts behind these and other problem solving techniques. The other basic course, SPC II, is a 5 day methods course in the same SPC techniques. The extra time is devoted to teaching the mathematical techniques required to construct histograms and control charts. SPC II is a prerequisite for the more advanced statistical courses that are offered.

Advanced statistical training is available in hypothesis testing, confidence intervals, simple and multiple regression, experimental design, measurement system analysis, and statistics for maintenance. These courses are all methods courses which vary from three to six days in length.

While the mathematics required at the basic level is minimal, the advanced courses require much more sophisticated tools for analysis. The choice of the SAS system for data analysis has led to the development of a 5 day SAS Basic Skills course.

Developing an Application as a Teaching Aid

Why Develop an Application

When using the SAS System as an analysis tool, some computer and SAS programming skills are required. For those who are non-programmers, the learning curve to acquire these skills can take a considerable amount of time and effort. This can be especially difficult if the student is also taking a course in an analysis topic such as statistics. Requiring SAS programming skills significantly increases the prerequisite skills for the advanced statistical courses. As well, classroom exercises would have to be developed according to the level of expected programming skill rather than the level of desired statistical knowledge.

In order to continue to use the SAS System for data analysis and allow students to concentrate on their statistical training, several SAS/AF applications were developed to compliment some of the advanced courses.

Developing an Application

SAS/AF applications that are developed are based on a course's content and learning objectives. They are designed to complement the course as much as possible (i.e. menu screens parallel the course outline). Students quickly become comfortable with the use of the computer as an analysis tool, without knowing anything about the SAS Programming Language.

These courses are developed prior to the SAS/AF application, following a standard procedure. Much of the course design is then carried over to the application design. The first step in the applications planning stage is to take the course outline and construct a flow chart, dividing it up into logical menu and program screens. Once this is done, the flow chart is checked to eliminate redundancy and exact screen text requirements are determined. Care is taken at this stage to ensure that

screens contain the minimum amount of information so that all exercises planned by the instructor can be completed. The course content supplies the structure for the menu system, and course exercises and learning objectives set the constraints for the screens. To illustrate this process, figure 1 shows a course outline for the Experimental Design course, figure 2 shows the flow chart, and figure 3 shows the corresponding main menu.

Figure 1: Course Outline - Experimental Design

Introduction
Comparing Two Treatments
Comparison of More than Two Treatments
Analysis of Variance
Randomized Block Experimental Design
Advantages of Factorial Designs
Introduction to Fractional Factorial Designs
Nested Designs
Introduction to Taguchi Methods

Figure 2: Flow Chart - Experimental Design

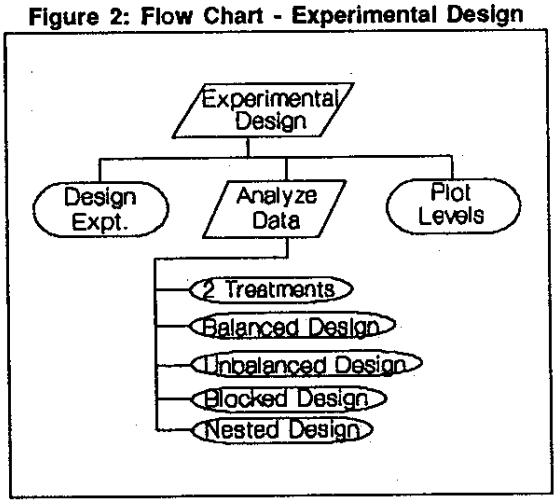
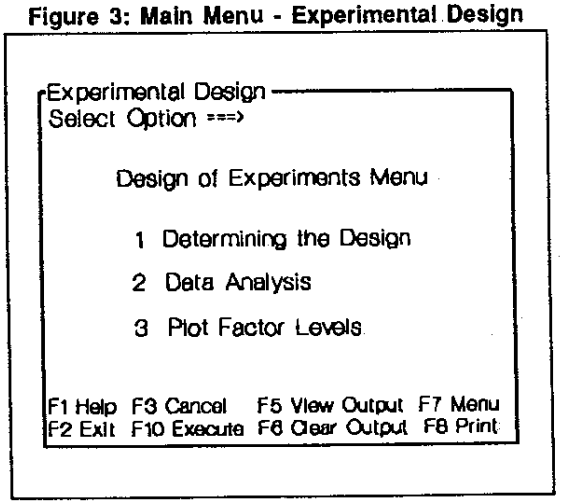


Figure 3: Main Menu - Experimental Design

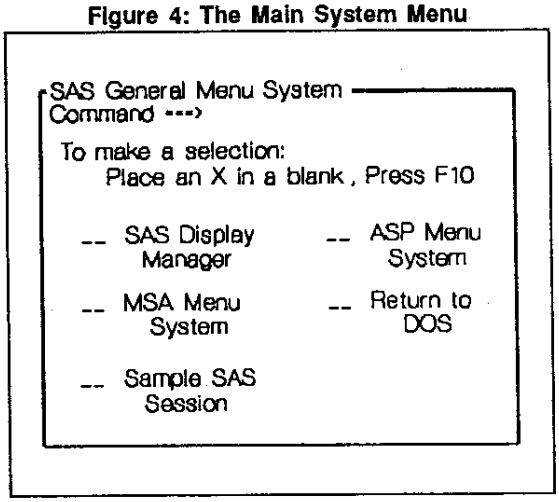


Specific standards are maintained during the programming stage of the development. The screen types within an application are kept consistent. That is, all menu screens look alike and all program screens look and operate alike. When the first applications were developed, this type of consistency was not thought to be a requirement between applications. Since that time, a consistent set of screen EDPARMS have been developed and are now used throughout our application development. Wherever possible, selection lists are used to select data sets and variables. Also, where possible, default values for screen variables are specified in the INIT section of the source code so that the program runs even if executed prematurely. Attribute panels (ATTR) are also carefully set up to provide verification and error checking.

A major restriction to our programming is the necessity of complying with certain hardware restrictions. All of the PC workstations in our classroom have 640K of main memory. The SAS Display Manager's three Primary Windows (program editor, log, and output) use much of this memory. This does not leave enough main memory to run even the simplest of SAS programs from a SAS/AF application. We compensate for the lack of memory using special programming methods which are discussed in detail in the next section.

The computers in the classroom are set up so that by using the DOS command "SAS" with an appropriate AUTOEXEC.SAS file, a main menu is accessed (figure 4) instead of the default Display Manager. From this menu the User makes a selection and is either brought into the SAS Display Manager, to an application menu, or back to DOS.

Figure 4: The Main System Menu



Tips for Programming a 640K Application

- **Check Your AUTOEXEC.BAT and CONFIG.SYS**

The computer's AUTOEXEC.BAT file should not contain any commands that cause unnecessary information to remain memory resident throughout the SAS Session (i.e. mouse software, network software).

Check CONFIG.SYS for memory allocation that is used for caching. Cache is that part of memory that is allocated for effective transfer of data through the use of the FILES= and BUFFERS= statements. SAS Institute Inc. recommends values of files=50 and buffers=17 as minimums. This allows the maximum amount of memory to be reserved for program execution. Using a higher number of files and buffers allows more memory to be reserved for open files and resident software, but memory accessible for data analysis is reduced. (As an aside, the allocation of more memory for these purposes does speed up the use of software.)

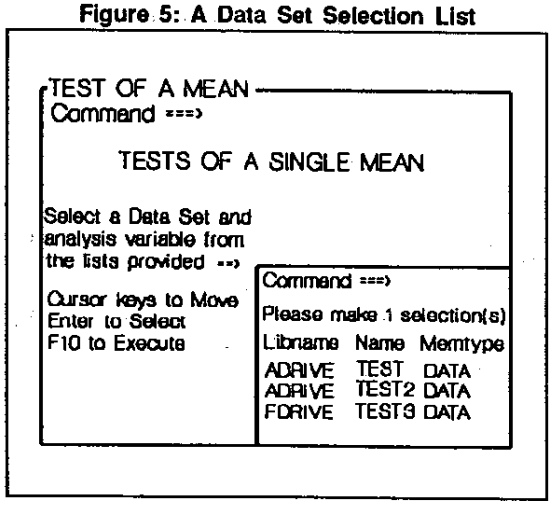
• Use Menus Instead of a Mouse

A mouse, although an effective graphic user interface, requires some of the main memory for the mouse software to reside. In order to use the mouse within a SAS/AF application a program screen must be used for selection instead of a menu screen. The use of a mouse also requires extra Screen Control Language coding so that a User's selection can be executed (SAS/ASSIST® software uses this method). A critical problem with using a program screen is that it will remain open after a selection is made, which takes up some of the main memory. Due to this, in addition to the main memory that the mouse software requires, only a small amount of SAS code will execute free of fatal memory errors. Menu screens however, close as soon as a selection is made. This frees the memory for SAS program execution.

• Use Data Set and Variable Selection Lists

The use of data set and variable selection lists along with consistent program and menu screens make an application very effective and works well within the 640K limitation.

Figure 5: A Data Set Selection List



Selection lists allow a User to choose one or more items (either SAS data sets or variables) from a list. To use a selection list the User must use the cursor keys to locate an item, press enter to make each selection (usually a limited number of choices), and then press F10 to execute the selection(s). A selection list may be used in combination with a LEGEND window to instruct the User on how to make a selection (see below). Figure 5 illustrates the use of a selection list.

• Limit the Use of Legend Windows

SAS/AF Legend windows may be used in an application to give instructions to the User, as long as it is closed promptly after use. In our applications, we use the CALL SOUND function to hold the legend in place for a period of time. In some cases audible sounds are used, in others inaudible sounds. If the LEGEND window is used along with a Selection List, memory errors may occur. This is due to the number of windows that are open (the three primary windows, program display, legend, and selection). Thus, some caution is advisable here.

• Limit the Use of Screen Control Language

SCL commands may be used in the INIT and MAIN section of the source code for data set and variable selection lists, verification of screen variables, and simple data management functions. Data analysis however, should be confined to the SUBMIT blocks.

• Clear Main Memory

The display manager command **CDE P** is an effective way to keep within memory limitations. It frees up all areas of main memory that have been allocated (but are not accessible by SAS) so that it can be used. Placing the Screen Control Language statement:

```
call execcmd('cde p');
```

at the beginning of the program source code's INIT block and the SAS statement:

```
dm 'cde p';
```

at the end of the SUBMIT TERMINATE block (before the ENDSUBMIT statement) automatically takes care of this activity.

• Use SUBMIT TERMINATE Blocks

Any time SAS code is submitted for processing, the TERMINATE option should be used on the on the SCL SUBMIT statement. If more than one submit block is required in a program, the first blocks can be sent to the preview window as a simple SUBMIT (with no option) as long as the amount of code submitted is kept to a minimum. The last submit block in the program must be a SUBMIT TERMINATE. This causes the execution of all of the submit blocks in the order that they were submitted.

By using the TERMINATE option, all windows of the current SAS/AF application that are currently in use are closed before the program begins to execute. This yields more main memory for the execution of the SAS program.

Since the program window has been closed, any parent and child members defined in the GATTR window of the program will be ignored and the user will be brought into the Display Manager when the program is finished executing. To return control to the application, use a display manager statement at the end of the program (**dm** 'command;command;...';). For example, to return control to a menu panel of an application, include the following statement at the end of the program's submit terminate block (just before ENDSUBMIT;):

dm 'af c=libref.catalogname.menuname.**menu' af;**

A display manager statement may also be used to link program entries together, i.e. to call one program from another. Use the following statement at the end of the submit block to call a different program entry:

dm 'af c=libref.catalogname.programname.**program' af;**

- **Limit SAS Data Set Size**

Some SAS procedures require too much memory if the data set is too large (has too many observations). The Screen Control Language function "**nobs( )**" can be used to check the size of a SAS data set. This check should be done in the INIT or MAIN section of the program so that a message can be displayed and the User can make adjustments.

As an example, say for instance the screen variable "&data" holds the name of a SAS data set that the user has selected from a selection list. If the application will only work on SAS data sets having less than 500 observations, the following is included in the MAIN section of the program's source code:

```
MAIN:
 if NOBS(&data) >= 500 then
    _msg_='Selected Data Set is too Large, F1 Help';
    call sound(200,1600);
RETURN;
```

Note: The maximum size of a SAS data set that is allowed is dependent on the application and procedures that are used.

- **QUIT Interactive Procedures**

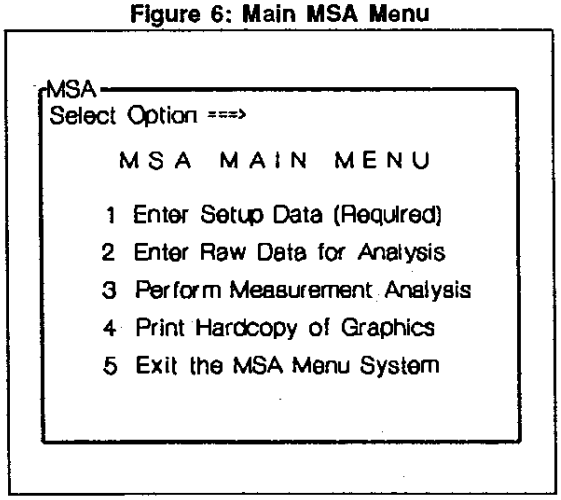
Some SAS procedures are interactive (e.g. GPLOT, REG). An interactive SAS procedure remains memory resident, awaiting further statements, until another procedure or data step is executed or until "QUIT;" is submitted. Terminating an interactive SAS procedure is especially important when it is used at the end of a program: the procedure remains in main memory while another screen is being displayed. A User is likely to get memory errors because the procedure is still using main memory that is needed for the new screen's display and functions.

A run statement at the end of the step does not terminate an interactive procedure. To take the procedure out of main memory, a quit statement (QUIT;) should always be included before the run statement at the end of every interactive SAS procedure.

Example 1: Measurement System Analysis

The Measurement System Analysis Course (MSA) teaches people how to analyze how various types of measurement error affect the measurements that are taken. This topic has become very necessary to manufacturers who must have available demonstrably accurate quality data for their customers. Many would recognize these analysis techniques as R&R studies.

Figure 6: Main MSA Menu



The MSA menu system allows a User to enter their test data in a format similar to the structure of the form on which data is collected and recorded. Menu option 3 generates several pages of printed output (using Base SAS and SAS/STAT® software) as well as two SAS/GRAPH® plots that are displayed on the monitor. The User is also given the option to plot the graphic output at a later time.

Example 2: Advanced Statistical Package

The Advanced Statistical Package (ASP) includes three courses: Hypothesis Testing, Regression Analysis, and Experimental Design. Base SAS, SAS/STAT, and SAS/GRAPH software procedures are used within the ASP application. Figure 7 illustrates the main menu for the ASP application. A selection from the Main Menu directs the User to a submenu (figure 8).

Due to the complex nature of this Application, a help facility is available. By pressing F1 from the main menu, the Help Menu is displayed allowing the User to obtain help on topics that may be giving them trouble (figure 9).

The courses are taught in a practical applications manner and contain much of the statistical theory necessary to exercise good judgement in choosing a type of analysis. The use of a computer for exercises is a requirement, as the concepts and mathematics can quickly become too complex and overwhelming when a hand calculator is used. The courses are designed to run in sequence, taking students from very basic statistical methods to the more complex techniques associated with experimental design.

Figure 7: Advanced Statistical Package Main Menu

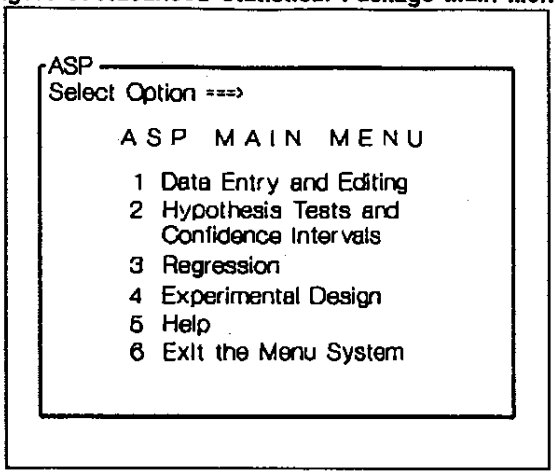


Figure 8: An ASP Submenu

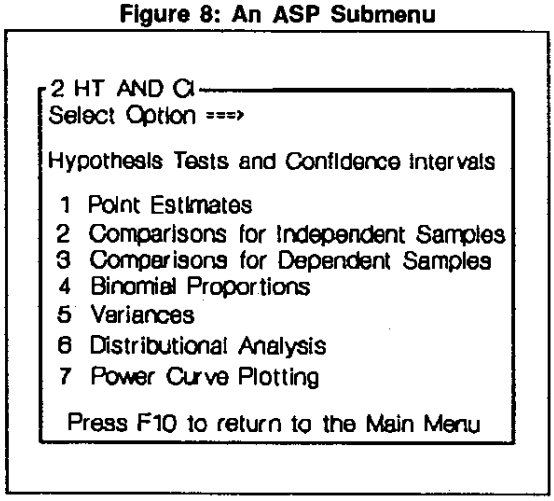
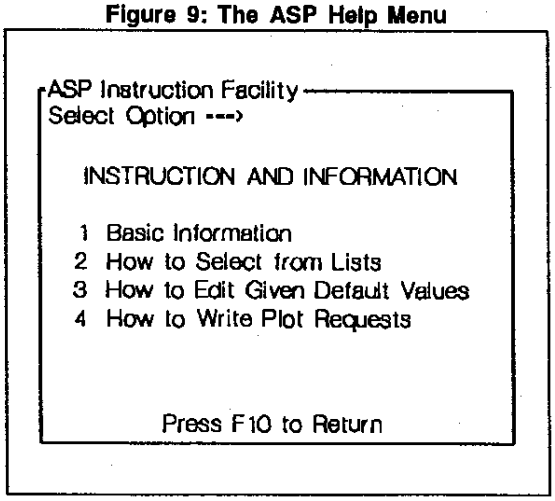


Figure 9: The ASP Help Menu



Using an Application as a Teaching Aid

The menu systems are used in a different manner depending on the course that is being taught. In the MSA Course, the instructor shows students the output before the availability of the menu system is discussed. The students become familiar with SAS output before reaching the computer. Many students in these classes have never used the SAS System before, but since the menu system resembles the course, they generally have little trouble using it.

The menu system also becomes a tool for the instructor, who can produce an analysis of in-class data after class and have it ready for interpretation when the class resumes the next day. This also helps the students who can check the work that they have done by hand.

Near the end of the course, students use the menu system to analyze measurement data that they have collected in a class demonstration. This shows the students the benefits of computer analysis, and allows them some experience with the menu system itself. The ease of the application is confirmed by the many students who return to the Training Centre to perform analyses long after their course has been completed.

The Advanced Statistical Package, which is broken up into three courses (Hypothesis Testing, Regression Analysis, and Experimental Design) has its Menu System structured in this manner (a submenu for each course). The ASP menu system is used in each course at various times to provide the student with a way to perform calculations. Calculating everything by hand tends to cloud the students' interpretation of the results, never mind the fact that the mathematics involved are very tedious. In some cases, the use of computers is necessary to ensure that a single exercise does not take up the entire day.

By using the ASP Menu System, students are able to concentrate on the interpretation of results rather than on the calculation methods. This system, as well, has many return users.

Benefits of SAS/AF Applications

- **Reaching a Larger Audience**

Prior to the development of the SAS applications, our five-day SAS Basic Skills course (or equivalent) was a prerequisite for the advanced courses. For many students, there was no opportunity to become proficient in SAS programming before proceeding to their statistical training courses. Because of this, the analysis methods taught in the advanced courses became frustrating for those who could not use SAS software well enough to do the in class exercises.

By using these Applications Programs as teaching aids, the SAS programming ability that used to be required for our statistical training courses has been eliminated. This increases the potential audience by making the courses available to anyone that has basic Statistical Process Control training (SPC II).

- **Teaching Ease**

When learning statistics, the concepts are often difficult at first, learning is enhanced through the use of many in class exercises. While typical SPC mathematics (pareto analysis, histograms, control charts) are often simple enough to be done by hand, many of the more advanced statistical methods require a computer for data analysis. By developing applications to support the courses that are taught, we have been able to minimize the amount of SAS programming required, while maximizing experience with SAS output from statistical procedures. This has lifted a large burden from the instructor, who can concentrate on concepts and interpretation, instead of SAS programming skills.

Also, the development of more challenging exercises is very feasible given the power of the SAS System, and the convenience of the applications. In addition, the time that would have been devoted to teaching basic programming skills becomes available for additional topics and greater depth of presentation.

- **Learning Effectiveness**

By removing the requirements for SAS programming competency from our statistical courses, we have allowed students to concentrate on learning concepts and interpretation rather than becoming frustrated by SAS programming. This has increased the learning effectiveness by a large amount. Students are able to view SAS output easily and the majority of time given for in-class exercises can be spent on interpretation of the output that the menu system has produced. We see greater understanding during the class and increased implementation of the concepts after training is complete.

Many students continue to use the applications programs for regular data analysis after the course is completed. This has led to the development of a central SAS/AF menu system which has become our basic SAS System Interface. It includes the applications that have been outlined in this paper as well as some general data handling and graphics functions. This interface combines with the established SAS support group to provide students with the ability to gain programming experience as they require, through either the SAS Basics Skills course or self learning.

Conclusion

In summary, the statistical training program at Dofasco has been enhanced through the use of the applications that have been developed. Benefits include teaching ease, ability to reach a larger audience, and increased learning effectiveness.

The development of these applications programs relies on recognizing the constraints in both the course and the users. Once these constraints have been established, the course manual can be used as a guide to development.

Development of the applications for 640K Personal Computer workstations is very possible, once the restrictions are known. Important constraints to keep in mind when include: efficient use of all types of windows, periodic clearing out of unusable portions of the main memory, using the terminate option when submitting blocks of SAS code, quitting interactive SAS procedures, and recognizing that large SAS data sets may require too much memory to process.

Once developed, these applications can form a backbone for a basic menu system which can be made available to our Users for typical data analysis. This minimizes the level of computer programming skills that are required for data analysis. It allows greater concentration on the main objective, which is to improve quality.

The authors may be contacted at:

Dofasco Inc.
Quality Training Centre
P.O. Box 2460,
Hamilton, Ontario
L8N 3J5
Canada