

Chapter 2

The SAS Environment

Abstract In this chapter, we begin to become familiar with the basic SAS working environment. We introduce the basic 3-screen layout, how to navigate the SAS Explorer window, the two basic types of SAS programs, the kinds of data with which you will work in SAS, and how to get help.

2.1 The SAS Screen

When you start up SAS, your screen will typically be split up into three frames (Fig. 2.1).

On your left you can tab (at the bottom) between the “SAS Explorer”, to navigate to files and folders, and SAS “Results” which allows you to navigate to the different statistical analyses and graphs you create during a SAS session.

On your right, you will see the Editor window on the bottom half of the screen where you will write and submit your programs and commands and the Log Window on the top portion of the screen, where SAS will give you messages (including flagging errors) about your submitted programs. The Output Window lurks behind the Editor and Log. When you submit syntax to perform analyses, it instantly comes to the fore to display your results. You can, though, bring it up to the front at any time by left clicking on Output on the bottom of your screen.

You left click on the frame of a window or the tab to make a window active. Your toolbar and available commands are context specific, i.e., they change depending on which window is currently active.

2.2 The Program Editor

The Editor window comes in two flavors. You can use the program editor window or the enhanced program editor window.

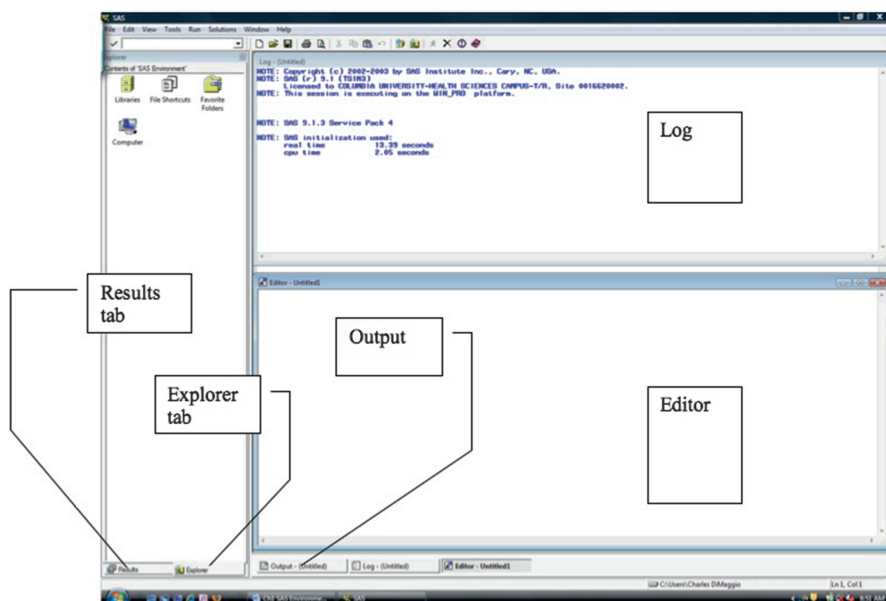


Fig. 2.1 The SAS screen

The program editor window is available across all computing platforms. Only one program editor window can be open at a time, and when you submit a command, it disappears and needs to be brought back up with the “recall” command. Of course since you don’t have an editor window anymore, it’s not easy to submit a command. You will find it again under the View menu at the top of your screen.

The enhanced editor window is available in MS Windows environments, where it is the default and just appears as Editor’. It has some nice features, not the least of which is that it doesn’t disappear when you submit a command. It also color codes your syntax. Data and procedure keywords appear in blue and variables in black. Helpfully, it also flags errors in syntax by coloring them red. It’s the only way to go.

WINDOWS.

If you accidentally close down a window, you can bring it back up by going to the View menu at the top of your screen and clicking on the window you want.

2.3 SAS Statements

You write SAS commands or statements in the Editor window. A line of SAS syntax almost always *starts with a keyword* of some kind, usually **DATA** or **PROC**, (we'll talk about those soon) and *ends with a semicolon*. You can have any number of SAS statements each requesting that SAS do something different. In a very common sense way, SAS will run statements when it sees the word RUN.

Once you've written your commands, make sure *each statement ends in a semicolon* and that all the statements you want to run are followed by a RUN command and that the RUN command also *ends in a semicolon*.¹ You can then submit your syntax. In MS Windows, you submit your statements by (1) highlighting it by left click-dragging your mouse across the syntax and then (2) clicking the little running-man icon on the tool bar. SAS then processes the syntax and then returns your results in the Output window as well as some information, such as how long the process took, if SAS encountered any errors, etc., in the Log window.

RUNNING

If you don't highlight the syntax you want run and just click the running-man icon, SAS will run every command in the Editor window. If you have been working on a lot of code, creating and manipulating files, you may end up destroying a lot of your work.

2.4 Comments

SAS lets you write little notes to yourself or to others who may try to read or run your program. There are two ways to indicate to SAS that what you are typing is not a command and that it should ignore it. You can either enclose the comment with a slash-asterisk asterisk-slash, like so:

```
/* comment */
```

or you can enclose it with an asterisk and a semicolon, like so:

```
* comment ;
```

The second approach is useful if you want to insert a short comment off to the side of a line of syntax or if you want to convert a section of code to text (just insert an * at the beginning of the line of syntax). Then the next time you want to run that syntax, just remove the asterisk.

¹You may be beginning to detect the importance of semicolons in SAS.

I suggest you use comments liberally. They not only help analysts coming after you, but also they help you when you forget what exactly you were trying to accomplish with a particularly gnarly line of syntax.

2.5 Quick Demonstration of an SAS Program

So what does a SAS program look like? Something like this (Fig. 2.2).

Here you see the command syntax written in the Editor window. I've enlarged the default font size so you can read it. Don't worry about what the data represents at this point. This is just a brief, initial, demonstration.

In the Editor window, notice a couple of things. First, look at all those semicolons. Next, notice that the code is color coded. The procedure I'm requesting ("proc univariate") is dark blue, as is the final "run" statement. The subcommands under that procedure, such as specifying the data set to work on ("var") the types of graphs I want for that variable ("histogram" and "probplot") are all light blue. The data set itself ("p8483.demo_1") and the variable name ("age") are in black. The comment I've put after the histogram request is color-coded green. Had I written something incorrectly or omitted a semicolon, SAS would very likely (though not always, depending on the type of error) have flagged the error by coloring the syntax red.

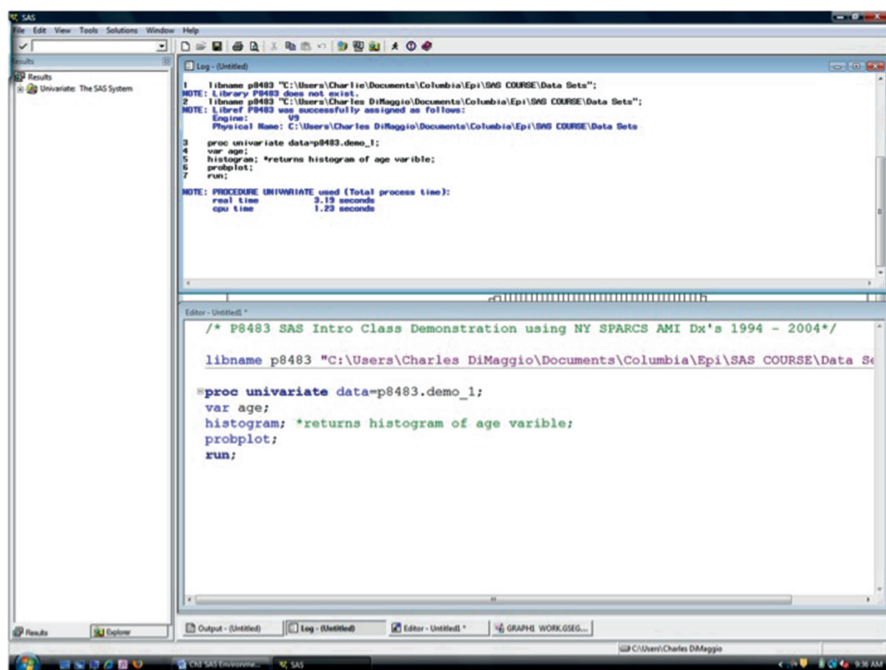


Fig. 2.2 The SAS Editor

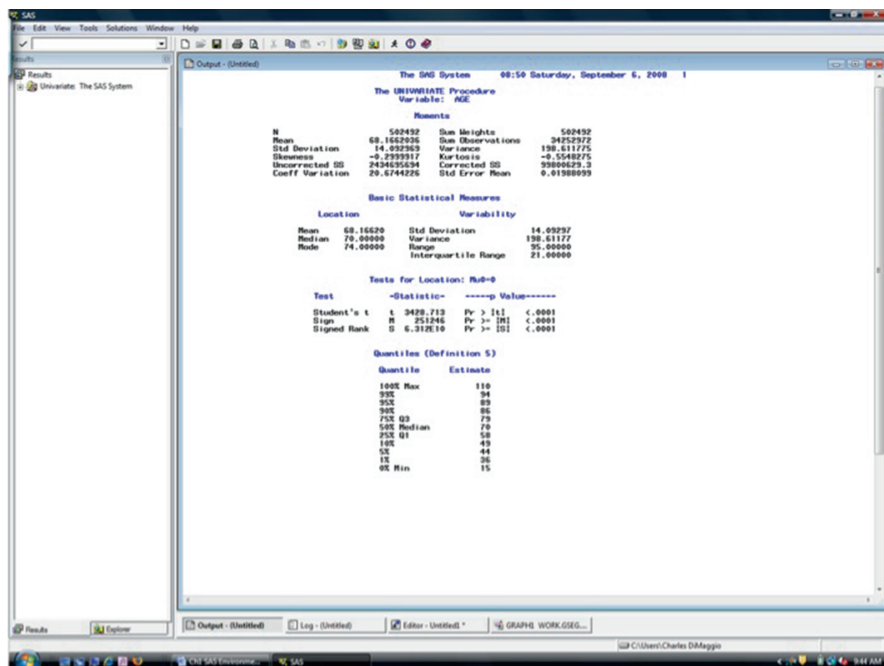


Fig. 2.3 The SAS Results Window

The log window on top gives some information about the syntax I've submitted and how long it took SAS to run the program. If there had been an error message, it would appear here in red. Also, note that the results pane on the left lists the univariate procedure I requested.

If I click on the Output tab at the bottom of the screen, I bring up my results (Fig. 2.3). In this case, I requested both statistical analyses and a graph (Fig. 2.4).

2.6 Two Types of SAS Programs

There are 2 basic types of SAS programs. You may hear them referred to as "steps" by old (and not so old) SAS Pros. They are the DATA step and the PROC step. DATA steps create or manipulate SAS data sets. PROC steps process data sets and conduct analyses. SAS knows a step is beginning when it sees DATA or PROC and knows one is ending when it sees RUN, QUIT, or another DATA or PROC statement. Read this paragraph again. Commit it to memory. It is the center around which all things SAS revolve.

It's helpful to know that DATA steps execute each line of your syntax for each observation in your data set before looping to the next line of syntax. SAS sees only

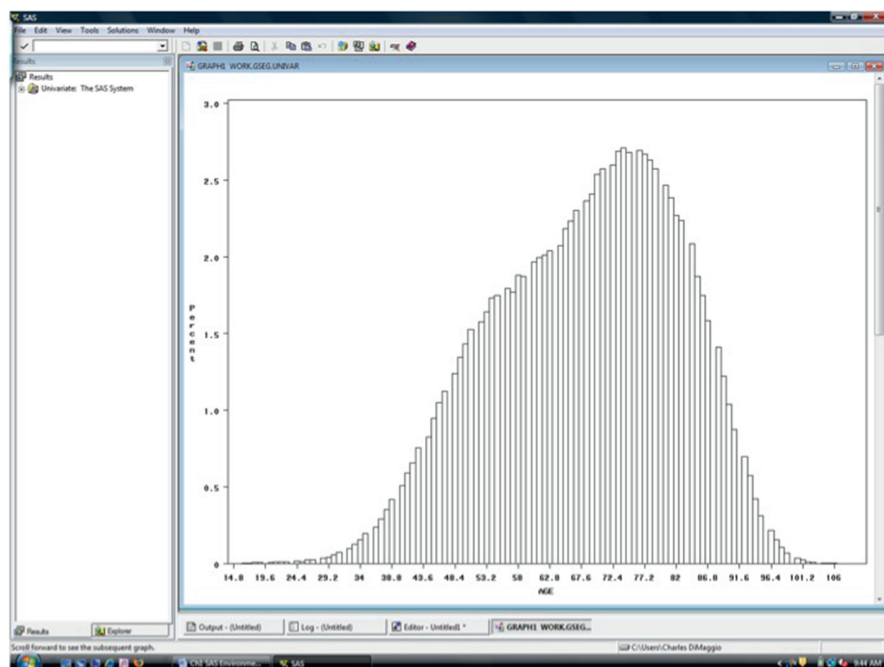


Fig. 2.4 The SAS Graphics Window

one observation of your data at a time and won't go onto the next until finished with first. This may not seem very useful information right now, but when we move on to consider various ways of reading data into SAS, it will help explain some inscrutable DATA step error messages.

As mentioned, SAS needs a RUN statement at the end of a program. But it doesn't really need one for each and every PROC. You can, if you like, string together a bunch of PROCs and then put a RUN statement at the end of them so they all run. I don't recommend this approach. I find the RUN statements a useful way of breaking up long or complex SAS code, making it both easier to read and debug.

SEMICOLONS

In English, a semicolon is used to link two independent clauses and is considered by some to be obsolete; I like them. In SAS, the semicolon is used to demarcate the end of a syntax statement; SAS likes them very, very, very much.

2.7 Two Kinds of SAS Data

Now we know there are (basically) two kinds of SAS programs. There are only two kinds of SAS variables: character and numeric. This distinguishes SAS from other programs that require many specific variable definitions. Even SAS dates (which we'll spend more time considering in later chapters) are numbers. They are based on a reference to January 1, 1960. (Why? Your guess is as good as mine.) So January 2, 1960, is the number 1, December 31, 1959 is the number -1, January 3, 1961 is the number 368, and so on. Missing character variables are denoted by a blank space; missing numbers are denoted by a decimal point.

2.8 Two Parts to a SAS Data Set

A SAS data set file holds (1) the data itself and (2) the descriptor portion that describes the variables.

The data portion of a SAS data file consists of the rows and columns of the data itself. You can view the data itself a couple of ways. You can write a *PROC PRINT* command in the Editor window. Or, you can double click on the physical file in the library folder found in the Explorer window.

The descriptor portion is like meta-data. It contains information on every variable such as the type of variable it is (we now know it can only be numeric or character), its length, any format you've applied, and any label you've applied. We'll discuss formatting and labeling later on. You can access this descriptor information using a *PROC CONTENTS* command.

SAY GOODBYE TO YOUR SPREADSHEET

Perhaps one of the most difficult aspects of the transition to SAS from programs such as Excel and SPSS is letting go of the need to manipulate a physical spreadsheet. SAS data seems to be floating in some netherworld to which you are sending commands.

In fact, SAS data sets very much do exist in a physical form on your hard drive. You just do not have call to work on the physical file directly. This is an asset when you are working with the large files for which SAS is so well suited. It is easy enough to find and call up the file when you have a handle of the SAS filing system of libraries. You can, at any time call the data file up directly. If only to reassure yourself that it exists.

2.9 Some Simple SAS Utilities

Printing and saving in SAS works pretty much the way it does in any Windows program. The default output file format is something called a “list” file, but you can also save your output as file types compatible with Windows Office programs such as Rich Text Format (RTF) for Word text documents and Enhanced Metafile (EMF) files for inserting graphs and charts into windows documents. SAS also has a nice export function, so you can send your data sets to programs like Excel. To print or save just part of your output, highlight the results in the output window, right click, and print, copy, paste, or save.

You can change your system options for things like lines per page, font type and size, centering text under the tools menu:

```
Tools --> Options --> System
```

2.10 Getting Help

OK, SAS is not exactly world renowned for the ease and clarity of their help and documentation. It sometimes seems like you need to know the answer to find the answer. The situation, though, has improved considerably over the years. You can access information on most procedures under the Help menu. You need, though, to go through a couple of steps:

```
Help --> SAS Help and Documentation --> SAS Products
--> Base SAS --> SAS Procedures --> Procedures
```

You can get list of SAS functions (we’ll discuss them later) at:

```
Help --> SAS Help and Documentation ---> SAS products
--> Base SAS --> SAS Language Dictionary -->
Dictionary of Language Elements --> SAS Data Set Options
```

(whew)

You could, alternatively, type “help” and the name of the procedure you want help for (e.g., “help univariate”), in the Command Window. The Command Window is located on the upper left-hand part of your screen and has a little check mark icon next to it. You can also get a list of SAS keyboard shortcuts (“function keys”) by typing “keys” in the command window. This would tell you, for example, that F8 is the keyboard approach to submit a command.

Remember, there’s (almost) always more than one way of doing things in SAS. It is an extraordinarily robust and multifaceted computing language. For example, to clear a your output screen, you can type “clear” in the Command Window or use the short-cut function “CNTRL-E” One of the nice things about SAS’s popularity is

that there is invariable someone who faced the same issue you have, and often, they were nice enough to post something online about how they solved their problem. I answer most of my questions with Google.

Problems

2.1. Submitting Commands and Reading Output

Start up SAS and with the Editor window active, use the “File” menu to navigate to your files. Open the intro–sparcs–ami syntax file by double clicking on it. Scroll down to the code titled “compare ages”.

Type in the full file-path location of the data file

```
demo_1
```

in between quotation marks following the statement

```
libname p8483
```

Don’t forget the semicolon. Submit the t-test statement comparing the age of non-New York City acute myocardial infarction patients (“upstate”) to residents from New York City. What is the mean age of upstate AMI patients? NYC AMI patients? Is this difference statistically significant? Do you think it is clinically important?

2.2. Saving SAS Output to Microsoft Word

Clear your output screen by pressing Cntrl-E. Do the same with your Log screen. Using the intro–sparcs–ami syntax file, run the univariate analysis of age. Save the output of the univariate analysis as a rich text (RTF) file on your desktop, by

1. Saving the RTF SAS file as a MS Word file
2. Using “File - Export” to export your histogram onto your desktop as a Windows Enhanced Metafile (EMF)
3. Opening your saved MS Word file created in step 1 and inserting your histogram in it using “Insert–Insert Picture”

2.3. Using SAS Help

Use the SAS help system to find an example of how to use the “OBS=” data set option to limit the number of observations SAS processes. Copy and paste the example from the help file into the word document you created in Exercise 1.2.

2.4. Using PROC CONTENTS to View Variables

Return to the intro-sparcs-ami syntax file in your editor window. Write in and submit the following syntax:

```
proc contents data = p8483.demo_1;  
run;
```

How many observations and variables are in this data set? Is the data sorted? What type of variable is the DATE variable? What type of variable is the DISPO variable and how long is it?

Use the SAS Explorer (hint: double click on the libraries folder) to find the demo-1 file on your computer system. Double click it to open it. Scroll down and inspect the DATE and the ECODE variables. How does the DATE variable appear? Does the ECODE variable look to be very useful?

Close out of SAS. You do not need to save your changes.

SAS for Epidemiologists

Applications and Methods

DiMaggio, C.

2013, XVII, 258 p., Hardcover

ISBN: 978-1-4614-4853-2