

SAS Introduction and Data Management

Jason Brinkley - Department of Biostatistics

January 22, 2009

What is SAS?

- ▶ SAS stands for Statistical Analysis Software.
- ▶ SAS started in the mid-1970's in Raleigh NC, a group of statisticians wanted to develop a software package that would help them do statistical analysis. SAS has since bloomed into one of the largest companies in the world and has sites in well over 100 countries.

Is this workshop right for me?

- ▶ There are many different software packages for statistical analysis available to ECU faculty, staff, and students. SAS (as discussed here) is predominately code based, it can be thought of as a computer language all on it's own. For those that would prefer not to use code-based software ECU has SPSS, JMP, Minitab, and SAS Enterprise Guide that may suit your research needs.

Is this workshop right for me?

- ▶ If you plan on interacting with large government databases, then learning SAS may be a good idea. To access sensitive aspects of government info such as the NHANES, Framingham Cardiovascular, or Census data; researchers sometimes need to write their SAS code in advance and provide it to the government so they know exactly what you are accessing.

Workshop Goals

- ▶ The goal of this workshop series is to introduce participants to some basic SAS coding and give users a foundation for which they can use the software in their own research.
- ▶ Keep in mind that this workshop series is not intended to replace the consulting services that the department of biostatistics already provides.
- ▶ For those of you who are not familiar with our consulting services check us out online at www.ecu.edu/bios/

The SAS Program

- ▶ Hopefully everyone has SAS 9.2 installed on their laptops and we will start by opening the software.
- ▶ Double Click the SAS icon on your desktop or go to Start - All Programs - SAS - SAS 9.2
- ▶ The first thing I want you to notice are the small buttons on the bottom labeled "Output", "Log" and "Editor"
- ▶ The way SAS works is that code goes into the Editor window; after running the code the output appears in the output window. The log window keeps track of all the code that is run and will show you when you make an error.

The SAS Program - Continued

- ▶ To the left you see tabs for "Explorer" and "Results". The Explorer tab lets you access all the saved sas code files as well as temporary datasets that you will create while SAS is open.
- ▶ The results window organizes all of your output.

Writing Programs

- ▶ To start coding, click on the Editor window. SAS code is generally broken up into two broad groups; we create and manipulate data in Data steps and we do analysis on data in procedure (Proc) steps.
- ▶ The best way to learn SAS is by using it so we will start with an example.
- ▶ Suppose you had test scores for 10 male students and you wanted to input that data into SAS.

SAS Example

```
SAS> *Jason Brinkley Jan 2009;
SAS> *SAS test scores example;
SAS>
SAS> Data Men;
SAS> Input Student Gender $ Test1 Test2 F_Exam;
SAS> datalines;
SAS> 1 M 75 78 76
SAS> 2 M 90 85 87
SAS> 3 M 85 76 72
SAS> 4 M 86 88 90
SAS> 5 M 92 95 93
SAS> 6 M 87 74 77
SAS> 7 M 65 76 72
SAS> 8 M 76 78 74
SAS> 9 M 90 92 96
SAS> 10 M 100 96 88
SAS> ;
SAS> run;
SAS>
SAS> proc print Data=Men;
SAS> run;
```

Obs	Student	Gender	Test1	Test2	F_Exam
1	1	M	75	78	76
2	2	M	90	85	87
3	3	M	85	76	72
4	4	M	86	88	90
5	5	M	92	95	93
6	6	M	87	74	77
7	7	M	65	76	72
8	8	M	76	78	74
9	9	M	90	92	96
10	10	M	100	96	88

SAS Example

Now let's say we have a student grader who gives us the homework average for these students and we want to add that to our current data. Then we add:

```
SAS> *add homework grades;
SAS> Data Homework;
SAS> Input Student HW_avg;
SAS> datalines;
SAS> 10 80
SAS> 9 95
SAS> 8 85
SAS> 7 85
SAS> 6 95
SAS> 5 100
SAS> 4 90
SAS> 3 80
SAS> 2 75
SAS> 1 85
SAS> ;
SAS> run;
```

```
SAS> *Data must be sorted to do a merge;
SAS> Proc Sort Data=Homework;
SAS> by Student;
SAS> run;
SAS>
SAS> Data Sample;
SAS> merge Men Homework;
SAS> by Student;
SAS> run;
SAS>
SAS> Proc Print;
SAS> run;
```

Obs	Student	Gender	Test1	Test2	F_Exam	HW_avg
1	1	M	75	78	76	85
2	2	M	90	85	87	75
3	3	M	85	76	72	80
4	4	M	86	88	90	90
5	5	M	92	95	93	100
6	6	M	87	74	77	95
7	7	M	65	76	72	85
8	8	M	76	78	74	85
9	9	M	90	92	96	95
10	10	M	100	96	88	80

Now let's say we want to calculate final grades and assign letter grades to our students based on some weighted sum:

```
SAS> *Find Final Grade = .25*Test1 + .25*Test2 + .10*HW_avg + .40*F_Exam;
SAS> Data Sample;*New dataset Sample;
SAS> set Sample;*Replace the old dataset Sample;
SAS> F_Grade = .25*Test1 + .25*Test2 + .10*HW_avg + .40*F_Exam;
SAS>
SAS> *Letter Grades;
SAS> If F_grade > 89.5 then Letter = "A";
SAS> if 79.5 < F_grade < 89.5 then Letter = "B";
SAS> if 69.5 LT F_grade LT 79.5 then Letter = "C";
SAS>
SAS> run;
SAS>
SAS> Proc Print Data=Sample;
SAS> *Title "Grades for Male Students";
SAS> ID Student;
SAS> Var F_Exam Letter;
SAS> run;
```

Student	F_Exam	Letter
1	76	C
2	87	B
3	72	C
4	90	B
5	93	A
6	77	B
7	72	C
8	74	C
9	96	A
10	88	A

Drop and Keep Statements

- ▶ Say you wanted to create a new dataset with only specific variables that you wanted to do analysis on or give other people.
- ▶ The drop statement in a dataset will eliminate certain variables from your dataset.
- ▶ The keep statement will drop all the variables in the dataset but the ones you want to keep.

Combining Datasets

- ▶ To our all male student dataset let's now add some data on female students
- ▶ Here we have some data on female students in an excel spreadsheet, there are lots of ways to bring this data into SAS. This is just one method I find very easy. We go to the File menu and click on "Import Data". This will take us through a couple of menus and even offer to generate and save code generated to import the Excel data into SAS.
- ▶ Note that the import statement will also bring in data from other programs like Microsoft Access and SPSS.

Now to add the male and female datasets together:

```
SAS> Data Sample;  
SAS> set Sample Female;  
SAS> run;  
SAS>  
SAS> proc print;  
SAS> run;
```

Obs	Student	Gender	Test1	Test2	F_Exam	HW_avg	F_Grade	Letter
1	1	M	75	78	76	85	77.15	C
2	2	M	90	85	87	75	86.05	B
3	3	M	85	76	72	80	77.05	C
4	4	M	86	88	90	90	88.50	B
5	5	M	92	95	93	100	93.95	A
6	6	M	87	74	77	95	80.55	B
7	7	M	65	76	72	85	72.55	C
8	8	M	76	78	74	85	76.60	C
9	9	M	90	92	96	95	93.40	A
10	10	M	100	96	88	80	92.20	A
11	11	F	80	85	82	100	84.05	B
12	12	F	94	95	95	100	95.25	A
13	13	F	85	87	75	85	81.50	B
14	14	F	90	97	89	90	91.35	A
15	15	F	77	76	72	90	76.05	C
16	16	F	92	85	92	95	90.55	A
17	17	F	85	79	87	90	84.80	B
18	18	F	87	85	79	85	83.10	B
19	19	F	95	88	92	95	92.05	A
20	20	F	98	97	90	100	94.75	A

After all the work we have done to with our data, it will all go away unless we save our code. But in addition to this we can also output the dataset we have created to send to others or use for further analysis.

```
SAS> Libname Output 'C:\Documents and Settings\brinkleyj\Desktop';  
SAS> *This is where you want the dataset to go;  
SAS>  
SAS> Data Output.Grades;  
SAS> set Sample;  
SAS> Label Student = "Student ID" Test1 = "Test 1 Score"  
SAS> Test2 = "Test 2 Score" F_Exam = "Final Exam Score"  
SAS> HW_avg = "Homework Average" Letter = "Letter Grade"  
SAS> F_grade = "Final Grade";  
SAS> run;
```

Let's do some simple analysis so you get an idea for procedures (Procs) in SAS:

```
SAS> proc means;
SAS> run;
SAS>
SAS> Proc sort Data=Output.Grades;
SAS> by Gender;
SAS> run;
SAS>
SAS> proc means Data=Output.Grades n mean std;
SAS> by Gender;
SAS> var Test1 Test2 F_Exam HW_avg F_grade;
SAS> run;
SAS>
```

The MEANS Procedure

Variable	Label	N	Mean	Std Dev	Minimum
Student	Student ID	20	10.5000000	5.9160798	1.0000000
Test1	Test 1 Score	20	86.4500000	8.5376873	65.0000000
Test2	Test 2 Score	20	85.6000000	7.8499078	74.0000000
F_Exam	Final Exam Score	20	83.9000000	8.4598121	72.0000000
HW_avg	Homework Average	20	90.0000000	7.4339194	75.0000000
F_Grade	Final Grade	20	85.5725000	7.2451340	72.5500000

Variable	Label	Maximum
Student	Student ID	20.0000000
Test1	Test 1 Score	100.0000000
Test2	Test 2 Score	97.0000000
F_Exam	Final Exam Score	96.0000000
HW_avg	Homework Average	100.0000000
F_Grade	Final Grade	95.2500000

Gender=F

The MEANS Procedure

Variable	Label	N	Mean	Std Dev
Test1	Test 1 Score	10	88.3000000	6.7338284
Test2	Test 2 Score	10	87.4000000	7.1523112
F_Exam	Final Exam Score	10	85.3000000	7.8605909
HW_avg	Homework Average	10	93.0000000	5.8689390
F_Grade	Final Grade	10	87.3450000	6.3496916

Gender=M

Variable	Label	N	Mean	Std Dev
Test1	Test 1 Score	10	84.6000000	10.0465583
Test2	Test 2 Score	10	83.8000000	8.4695533
F_Exam	Final Exam Score	10	82.5000000	9.2165310
HW_avg	Homework Average	10	87.0000000	7.8881064
F_Grade	Final Grade	10	83.8000000	7.9696996

The Where Statement

- ▶ In the last example we used the by statement to illustrate what would happen if we wanted to stratify the means based on gender.
- ▶ Say that we instead were only interested in looking at the scores among females.
- ▶ In the last bit of code if we replace "by Gender;" with "where Gender= 'F' " in the proc means statement we would only get a subset of our data.
- ▶ You need ' ' marks for characters but not for numeric data.

Are there differences?

- ▶ The Proc Means output shows that the average final score among women is higher than men.
- ▶ Let's say we want to test if there is a statistically significant difference.

```
SAS> proc freq Data=Output.Grades;
SAS> *generate a Gender X Letter table;
SAS> tables Gender*Letter;
SAS> run;
SAS>
SAS> proc ttest Data=Output.Grades;
SAS> class Gender;*group variable;
SAS> Var F_grade;*testing variable;
SAS> run;
```

The FREQ Procedure

Table of Gender by Letter

Gender(Gender)	Letter(Letter Grade)			
Frequency				
Percent				
Row Pct				
Col Pct	A	B	C	Total
F	5	4	1	10
	25.00	20.00	5.00	50.00
	50.00	40.00	10.00	
	62.50	57.14	20.00	
M	3	3	4	10
	15.00	15.00	20.00	50.00
	30.00	30.00	40.00	
	37.50	42.86	80.00	
Total	8	7	5	20
	40.00	35.00	25.00	100.00

The TTEST Procedure

Statistics

Variable	Gender	Lower CL		Upper CL		Lower CL	Std Dev
		N	Mean	Mean	Mean	Std Dev	Std Dev
F_Grade	F	10	82.803	87.345	91.887	4.3675	6.3497
F_Grade	M	10	78.099	83.8	89.501	5.4818	7.9697
F_Grade	Diff (1-2)		-3.225	3.545	10.315	5.4445	7.2054

Statistics

Variable	Gender	Upper CL		Minimum	Maximum
		Std Dev	Std Err		
F_Grade	F	11.592	2.0079	76.05	95.25
F_Grade	M	14.55	2.5202	72.55	93.95
F_Grade	Diff (1-2)	10.655	3.2223		

T-Tests

Variable	Method	Variances	DF	t Value	Pr > t
F_Grade	Pooled	Equal	18	1.10	0.2858
F_Grade	Satterthwaite	Unequal	17.1	1.10	0.2865

Equality of Variances

Variable	Method	Num DF	Den DF	F Value	Pr > F
F_Grade	Folded F	9	9	1.58	0.5090

- ▶ SAS has a history of having unattractive output.
- ▶ One thing the makers have done to pretty things up and make it more useful is something called ODS (output delivery service)
- ▶ ODS lets you output files in other formats like html (for the web) or rtf (goes immediately into MS Word) formats.
- ▶ Let's look at the last bit of code in ODS html (and talk about the default directory).

More with ODS

- ▶ ODS also allows you to save particular parts of the output, but you have to know the name of what you are selecting.
- ▶ We can do this with ODS Trace.
- ▶ Let's look at this with our current example.

Online Documentation

- ▶ SAS provides a wonderful resource called the online documentation that can help you troubleshoot SAS code.
- ▶ In addition to providing a little background info, the online documentation can tell you what options you need to for special output.
- ▶ For example we did a t-test today in the ttest procedure, but say we wanted to do a paired t-test instead.
- ▶ Let's see how you would find that information.

- ▶ That's all for today, next time Dr. Qiang Wu will go into a lot more detail about how you can describe data and make comparisons among variables.