SAS Tutorial

SAS – Statistical Analysis System - is the one of the most widely used Statistical Software package by both Industry and Academic circles. SAS software is developed in late 1960s at North Carolina State University and in 1976 SAS Institute was formed . The SAS system is a collection of products , available from the SAS Institute in North Carolina. SAS software is a combination of a statistical package, a data – base management system, and a high level programming language. Most of the pharmaceutical research companies prefer to use SAS and Statisticians with knowledge in SAS has given preference in most of these companies.

Overview of SAS Products

- Base SAS - data management and basic procedures
- SAS/STAT - statistical analysis
- SAS/GRAPH - presentation quality graphics
- SAS/OR - Operations research
- SAS/ETS - Econometrics and Time Series Analysis
- SAS/IML - Interactive Matrix Language
- SAS/SQL – Structural Query Language
- There are other specialized products for access to databases, connectivity between different machines running SAS, etc.

How to Invoke SAS?

SAS is available in UNIX system only in both undergraduate and graduate accounts.
Type the Unix command, 'sas &' and six windows will appear. They are,

SAS: Results
SAS: Explorer
SAS: ToolBox
SAS: Output-Untitled
SAS: Log-Untitled
SAS: Program-Editor-Untitled

You can buy SAS from CHIP. Also to get online help (local UW), visit

http://www.ist.uwaterloo.ca/ew/sas8doc/sashtml/main.htm

Running a SAS Program

You can write your codes in Program Editor and submit. The results will be available in Output window. OR
Write the codes (no need to give run statement at the end) in a text tile with extension sas run as sas filename.sas. Your results will be in filename.lst

Structure of a SAS Program

SAS programs are comprised of SAS statements. Some statements act together to create SAS DATA sets, while other SAS statements act together to run predefined statistical or other routines. Apart from these two, there are OPTIONS statements, which can appear any where in the SAS program (generally appears in the beginning of the program). Some common options are,

nodate (suppresses the date from appearing from the output file)
linesize (linesize=80 restricts the output to 80 columns)
pagesize (pagesize=60 forces a page break after 60 lines)

The DATA steps allow you to read raw/SAS/other-format data, manipulate data by concatenating, merging, recoding and subsetting data. The data step is used to prepare your data for use by one or more of the SAS procedure steps (often called “PROC”).

The PROCedure steps perform analysis on the data, and produce statistical and graphical outputs.

SAS program is not case sensitive, except inside of quoted strings.
Each statement in SAS must end in a semicolon (;)
Many of the Data steps and Proc may need a “run;” to be executed interactively.
Add comments using /* … */ or *…;

SAS DATA Set

A SAS data set contain actual data as well as information on the data variables, variable formats and missing value (generally reported by a period). SAS data sets has Matrix-like Structure

- Rows corresponds to observations
- Columns corresponds to variables

Running a SAS Program

Defining a Data set

A SAS data set may be assigned a single or a two-part name. A single name indicates that the data set is temporary. This type of data set will not be saved after the SAS program is terminated. A two-part name is used for defining a permanent data set. The first part of the name is the library reference, and the second part is the data set name. Each name must be eight characters or less. A period separates the two parts. The library reference represents the physical location of the data set and is defined using the libname statement.

Example for creating a permanent SAS dataset

```
libname mydata ‘stat698";
data mydata.test;
run;
```

To define a temporary data set called test, the syntax is

```
data test;
```

Reading Data from an external file

If we are interested in reading data from an external file, then we need to use the infile statement. For example,

```
data test;
infile ' c:\class\data \test.txt' delimiter=’,’;
input id  sex $ x  y  race $;
logy=log(y);run;
```

The option delimiter is used to define the delimiting criteria in the data file. If the data are delimited by blank, this option is not necessary.

Input statement defines the variable names, in the same order as in the data file.

Creating data set using SAS program

Instead of reading data from an external file, we can directly input the data using SAS itself. The datalines statement (or cards statement) is placed immediately after defining the variables names using input statement. For example,

```
data test;
input id sex $ y x1 x2 ca$;
datalines;
1 M 68.5 155 12 1
2 F 61.2 199 22  2
3 F 63.0 115 29  1
4 M 70.0 205 15 1
5 M 68.6 170 23  2
6 F 65.1 125 30   2
7 M 72.4 220 28  1
8 F 86.2  187 21  1
9 M 87.9 167 19  1
10 F 75.5 156 24 2
;
run;
```

If you need to modify the data set ‘test’ and the modified data set is ‘test1’, then use the following statements :

```
data test1;
set test;
if id=5 then x=69.6;
if id=6 then y=150;
run;
```

The first two statements indicate that we assign the test to test1. In this stage both test1 and test are same. The next two lines indicate that we wish to change the x value for id=5 and y value for id=6.
Creating new variables / modifying existing variables: You can create a new variable by the statement,
Variablename = expression
For example, result1=(x+y)/2;
Similarly you can modify the existing variables.

Arithmetic Operators

**	Exponentiation	x**2
*	Multiplication	x*y
/	Division	x/y
+	Addition	x+y
-	Subtraction	x-y

Comparison Operators

=	EQ	^=	NE
>	GT	>=	GE
<	LT	<=	LE

Logical Operators

&	AND
|	O R
^	NOT

Some Numerical Functions

mean(arglist)	var(arglist)	std (arglist)	sum(arglist)
min (arglist)	ma x(arglist)	abs(arg)	mod(num,div)
log(arg)	exp(arg)	sqrt(arg)	int(arg)
ceil(arg)	floor(arg)		
rannor(seed)	ranuni(seed)	ranbin(seed,n,p)	ranpoi(seed,m)

IF statement

We have already used the if statement in modifying the data files. ‘IF’ statement is generally used to subset data or conditionally assign values to variables.
General format is IF (conditions) THEN (statement); ELSE (statement)

Do Statement

Do statement is similar to FOR statement in Matlab / R
See example, where we are generating 100 random uniform distribution values.

```
data rand1;seed=444;
do i=1 to 100; x=ranuni(seed);
output;end;
proc univariate data=rand1;
run;
```

PROC statements

These statements perform a set of statistical procedures to analyze the data. Some PROC statements are simple but some are very complicated. We will see some simple ones.
We will explain some simple PROC statements using the data 'test'.

Analyzing Univariate Data

a) PROC MEANS

```
proc means data=test;
var y x1 x2;
run;
```

This will give you a summary of variables y, x1 and x2 with details such as sample size (N), mean, std dev, minimum and maximum.

You can get this summary statistics separately for Male and Female, if you first sort the data using PROC SORT and give additional command 'by sex'.

```
proc sort data=test;
by sex;
proc means data=test;
var y x1 x2;
by sex;
run;
```

These statements will give you a separate output of PROC MEANS for both sex.

If you don't want the output files to be printed out and instead you need to save it in another SAS file 'ms', then use the following statements

```
proc means data=test noprint;
var y x1 x2;
output out=ms
N= n_y n_x1 n_x2
mean=m_y m_x1 m_x2
std=s_y s_x1 s_x2;
run;
```

If you need some specific summary measures, you can specify accordingly

b) PROC UNIVARIATE

This PROC will give more extensive list of statistics including tests of normality, stem and leaf plots, box plots, etc.

```
proc univariate data=test;
title ‘More Descriptive Statistics’;
var y x1 x2;
run;
```

By default, you will get 18 statistics for each variables. If you want test of normality and stem and leaf plot and box plot, use the command;

```
proc univaritate data=test normal plot;
var y x1 x2;
run;
```

c) PROC CHART for Bar graph

```
proc chart data=test;
vbar y / levels=2;
run;
```

d) PROC PLOT

```
proc plot data=test;
plot y*x1=’*’;
run;
```

e)PROC FREQ

```
proc freq data=test;
tables sex;
run;
```

Cross Tabulation and testing the association

```
proc freq data=test;
tables sex*ca / chisq;
run;
```

Linear Regression

```
proc reg data=test;
model y=x1 x2 / cp;
run;
```

cp is the model selection criteria. Other criteria includes forward, backward, cp, rsquare etc.

To get additional details, you may need to give some options such as r (residuals), p (predicted values), clm (expected values), cli (individual predicted values) etc.

For logistic regression use PROC LOGISTIC
Other commonly used PROCs are GLM, CATMOD, GENMOD, MIXED, etc.

SAS Assignment

1 Given the data set:

ID	RACE	SBP	DBP	HR	CAT
001	W	130	80	60	1
002	B	140	90	70	1
003	W	120	70	*	0
004	W	150	90	76	1
005	B	160	87	56	0
006	W	120	76	49	0
007	W	159	87	45	0
008	B	140	87	87	1
009	B	135	81	75	1
010	W	125	98	86	1
011	W	145	89	66	0
012	B	129	77	55	0

* missing

a) Compute mean and standard deviation using PROC means separately for two groups of RACE for SBP, DBP and HR

b) Define a new variable LSBP=log(SBP) and fit the multiple regression of LSBP on DBP, HR and CAT

c) Using the cp criterion, select the best model

d) Is there any association between the variables RACE and CAT?

e) Fit a logistic regression: CAT on LSBP, DBP and HR