Introduction to SAS

Statistics Outreach Center
Short Course

Topics Covered:

- SAS software and programming language
- SAS Interface
- Basic SAS syntax (DATA and PROC steps)
- Importing Data
- Data manipulation
- Descriptive Statistics
- Inferential Statistics
  - T-tests
  - One way ANOVA
  - Correlation and Regression
- Additional SAS Resources
Introduction
This short course is designed as an introduction to SAS for those who have never used SAS. It will provide basic instruction in the topics listed on the cover page. You will not be an expert at SAS at the end of this course, rather, it is the hope that this presentation will motivate a desire to further pursue SAS. It will also provide many resources for learning additional features of SAS. This short course will use an example dataset that can be downloaded from the SOC website (https://education.uiowa.edu/services/statistics-outreach-center-soc).

Getting Started
Open SAS by clicking on the desktop, searching through start menu, or through Citrix Virtual Desktop. https://virtualdesktop.uiowa.edu/Citrix/VirtualDesktopWeb/

Section 1: SAS Software and Programming Language
SAS is a software suite developed by SAS Institute (since 1976) that is designed to help businesses and organizations solve problems. The SAS programming language is run (or compiled) through the SAS software. SAS is used for many purposes, but in terms of research here are the four main purposes:

- Accessing Data: SAS can read any type of data and can access data remotely
- Managing Data: You can manipulate and reshape data using SAS
- Analyzing Data: Data can be analyzed using a variety of statistical methods
- Presenting Data: SAS can present data in a visually pleasing manner

SAS is the predominant data and statistical software used by many large corporations and drug companies. While it does not have the flexibility of R, SAS is generally considered easier to learn than R, but it is not as easy as SPSS. There is a great deal of SAS documentation online and a google search is often the best way to find what you need. If you are interested in a book, The Little SAS Book (5th Edition by Lora Delwiche & Susan Slaughter) is a fabulous resource. Given SAS’s prominence in the business sector, there is no shortage of well paying SAS related jobs.
Section 2: SAS Interface

SAS is code driven. There are 5 main windows used in SAS. The first three are most commonly used.

1. **Editor window**: This is where you build your program and where you will spend most of your time in SAS. Code to run your program is entered into this window.

2. **Log Window**: This window contains notes concerning your code after you run your code. It is used often to check for errors in your code and to debug your program.

3. **Results Viewer**: Results from analyses procedures (PROC steps) are viewed in this window.

4. **Explorer/Results Tab**: The explorer tab allows you to view SAS datasets and file locations. The results tab keeps a tally of the difference analysis procedures (PROC steps) you run, allowing you to easily reference them by double clicking.

5. **Output Window**: This window displays outputs from certain procedures and is seldom used.

Additionally, if you open up a dataset in SAS by double clicking from the explorer tab, a new dataset window will open.
Section 3: Basic SAS Syntax and a Basic SAS Program

SAS programs generally consist of two important parts:

- **DATA step**: The word `DATA` tells SAS that you want to work with your dataset – either inputting the data or manipulating the data.

- **PROC step**: The word `PROC` tells SAS you want to do something with the data (e.g., print it out, calculate statistics).
  
  - If no data is specified, the last previously used dataset will be invoked.

```sas
DATA CLASSDAT;
  INPUT ID $ NAME $ SEX $ EXAM1 EXAM2 GRADE $;
DATALINES;
S01 Max   M 84 88 A
S02 John  M 89 91 A
S03 Sarah F 86 90 B
S04 Lee   M 85 87 B
S05 Rosa  F 94 99 A
S06 Ming  F 84 77 C
;  
RUN;

PROC CONTENTS DATA=CLASSDAT;
RUN;

PROC MEANS DATA=CLASSDAT;
VAR EXAM1;
RUN;

PROC MEANS;
VAR EXAM2;
RUN;

PROC PRINT;
TITLE 'SAS SHORT COURSE';
RUN;
```

Explanation of Above Program Syntax

The program you see above creates a data set named CLASSDAT using the DATA statement, along with the INPUT and DATALINES statement. The INPUT statement designates the variable (column) names for the dataset. The $ signs following a name (e.g., ID $) designates that variable consists of character data. The DATALINES statement allows you to enter your data directly into SAS. Notice how there needs to be a space between each data point for this statement to work together properly with the INPUT statement. The RUN statement indicates to SAS to run the code that has been passed up until that point. By that definition, technically this program only needs one RUN statement.
This program also includes some PROC steps, which use the PROC statement, along with the MEANS statement to produce means of the variable specified in the VAR statement. A PRINT statement is also used with a TITLE statement to produce a printout of the dataset named CLASSDAT. PROC CONTENTS provides a description of the variables and their data types.

A few additional SAS syntax primers:

- Each SAS statement **must** end with a semicolon “;”
- At the end of your program you must have a run statement, “RUN;”. Otherwise the last SAS data step or SAS procedure will not get executed.
- SAS comments: SAS will pass over anything that is written between “/*” and “*/”. “*” and “;” also works to denote a comment. It is a good idea to use comments to document what you are doing in your program.

Running your program:

- When you want SAS to execute the statements you have written, click the “running man” icon on the toolbar. Or click on the Run pull-down menu and select “submit.” This will run all your code.
- If you only want to run part of your code, highlight the part you want to run and then click Run OR right click and select “Submit Selection.” SAS will only process the part of the program that you have highlighted. If you run your code in this manner, you **must** have a RUN statement included at the end of your selection.
Section 4: SAS DATA step and Variables in SAS

- Include variable manipulation (summing and average of two exam scores), and dataset creation, e.g., subset data by male/female. Use the SET statement.

The SAS DATA step is where data manipulation can occur. SAS can merge and concatenate datasets, create or remove variables, and perform many other manipulations on datasets. This section will introduce you to how SAS structures datasets and variables and will present a couple basic dataset manipulations to give you a preview of what SAS can do.

SAS Datasets

SAS datasets contain columns corresponding to specific variables (e.g., height, weight, etc.) and rows corresponding to specific observations (e.g., persons, clinic sites, etc.). SAS can read data in two different methods:

1. SAS datasets can be directly embedded in the Editor window (as seen in the previous section)
2. SAS datasets can be imported from a file (i.e., text file, excel file, etc; Will be demonstrated in the next section)

SAS Variables

- SAS variables can be in one of two possible formats:
  1. Character: typically letters or strings of letters and numbers, and mathematical operations can not be performed on them. (ID, Name, Gender, Grade)
  2. Numeric: typically numbers, and mathematical operations can be performed on them. (Exam1)

- Some rules about variable names:
  1. Start with a letter or _ (underscore)
  2. Contain only letters, numerals or underscores (_ ) No Spaces!
  3. Are not case sensitive
  4. 32 characters or fewer

Basic Variable Manipulation

To create or change variables, you need to manipulate the data set and variables using a DATA step. You also need to use the SET statement in the DATA step. The DATA statement names the new dataset, while the SET statement identifies which existing dataset you wish to perform manipulations on. The following code also includes PROC PRINT statements so that you can see the product of the various manipulations. Be sure you have run the code in the previous section to create the dataset CLASSDAT before running the code in the following.

*Create a new variable by averaging the sum of two existing variables*

```
DATA EXAMAVG;
    SET CLASSDAT;
```
Exam_Average = (EXAM1 + EXAM2)/2;
RUN;

PROC print data=EXAMAVG;
RUN;

The above code creates a new dataset (EXAMAVG) by using an existing dataset (CLASSDAT) and adding a new variable (Exam_Average) by summing the existing numeric variables and dividing them by 2. You will notice this syntax applies the code (EXAM1 + EXAM2)/2 automatically across every row and puts this product into the new variable, Exam_Average.

Create a subset of a dataset

DATA ExamMales;
   SET CLASSDAT;
   IF SEX = "M";
RUN;

DATA ExamFemales;
   SET CLASSDAT;
   IF SEX = "F";
RUN;

PROC print data=ExamMALES;
run;

The above code creates two datasets, one for males, and one for females using the IF statement.

Concatenate datasets

DATA Newfullset;
   SET ExamMales ExamFemales;
RUN;

PROC print data=newfullset;
RUN;

The above code combines the Male and Female datasets that we created. It works by stacking the datasets on top of one another, which is called concatenation. A MERGE statement would place the datasets next to each other, although this is beyond the scope of this class.

Your Turn!

1. Create a variable that only sums the exam scores.
2. Create a new variable named YEAR and set it to 2015.
3. Create a subset dataset that only includes those who received “A”s for their grade.
4. Extra mile question: Create a subset dataset that only includes those who were Male and who received “A”s. (hint: You need to use an AND logical operator)
Section 5: Importing Data

To import excel data, you need to use the IMPORT statement, but first we need to download the data.

1. Go to SOC website (https://education.uiowa.edu/services/statistics-outreach-center-soc) and click on SOC Short Courses.

2. Download the dataset from the SPSS course (NOT SAS course) and save to your H drive.

3. Run the following PROC IMPORT statement. You may need to change the file path in the DATAFILE statement to match your file location if you are not on a campus computer and/or you did not save to your H drive. The following file path may be used if the file is saved on the local desktop, however, be sure to change the ‘username’ to your own hawk id or user name. (\Client\C:\Users\username\Desktop\employee_data.xls)

```
PROC IMPORT
   OUT=EMPDATA
   DATAFILE="H:\employee_data.xls"
   DBMS=EXCEL REPLACE;
   RANGE="Employee data$";
   GETNAMES=YES;
   MIXED=NO;
   SCANTEXT=YES;
   USEDATE=YES;
   SCANTIME=YES;
RUN;
```

This code will import your excel file and create a SAS dataset called EMPDATA, which we will use for the remainder of this short course. The GETNAMES option takes the first row of the excel sheet and uses it for column names when importing to SAS. There are a number of different options for importing excel files that can also vary depending on the excel file version being imported. Additionally, importing excel files will not always import the way you want them, so it is always important to double check your files using PROC CONTENT to ensure the variables were imported correctly.
Section 6: SAS PROC step: Descriptive Statistics
This section demonstrates a few descriptive statistic SAS procedures on the employee dataset we imported in the last section.

**Calculate mean, standard deviation, minimum, and maximum**

```sas
PROC MEANS data=EMPDATA;
VAR salary;
RUN;
```

**Calculate mean, standard deviation, minimum, and maximum by groups**

```sas
PROC MEANS data=EMPDATA;
CLASS gender;
VAR salary;
RUN;
```

**Calculate only mean and standard deviation**

```sas
PROC MEANS data=EMPDATA mean std;
VAR salary;
RUN;
```

**Create a histogram with extended descriptive statistics**

```sas
PROC UNIVARIATE data=EMPDATA;
VAR salary;
HISTOGRAM;
RUN;
```

**Create a plot**

```sas
PROC PLOT data=EMPDATA;
plot salary*salbegin;
RUN;
```

Your Turn!

1. Calculate mean and standard deviation for both salary and salbegin
2. Create histograms for salary for men and women (hint: use the CLASS statement)
3. Extra mile question: Find out how much education has the highest mean level of salary.
Section 7: SAS PROC step: Inferential Statistics

**t-test**

```sas
PROC TTEST data=EMPDATA;
   CLASS gender;
   VAR salary;
RUN;
```

The amount of information given for the standard independent group t-test is incredible. SAS provides you with assumption checks for equal variance and normality, and then runs both the regular t test, and the Welch-Satterwaite test (which is more robust to assumption violations).

**One-Way ANOVA**

```sas
PROC ANOVA data=EMPDATA;
   CLASS jobcat;
   model salary = jobcat;
RUN;
```

Running this code all together will provide both an omnibus ANOVA test, and a tukey post-hoc comparison test.

**Bivariate Correlation**

```sas
PROC CORR data=EMPDATA;
   VAR salary salbegin;
RUN;
```

This code produces a bivariate correlation between salary and beginning salary.

**Multiple Regression**

```sas
PROC REG data=EMPDATA;
   model salary = salbegin;
RUN;
```

```sas
PROC REG data=EMPDATA;
   model salary = salbegin educ;
RUN;
```

The above code produces two regression models with output related to the regression assumptions and model fit. Notice that adding education as a variable in the second model positively predicts current salary, and uniquely predicts ~3% of the variance above and beyond starting salary (gained by subtracting the adjusted r square in model 1 from model 2).
Additional SAS Resources
The SAS website, www.SAS.com has a number of free tutorials available to get you started.

A great resource is the free SAS eLearning courses available through ITS. Start at the UI Software Download site (https://its.uiowa.edu/campus-software-program/sas). After you sign in, go to “SAS Self-Paced e-Learning. Even if you are an intermediate level programmer, the SAS Programming 1 course can be informative.

The university library has SAS books available for checkout. One excellent resources is The Little SAS Book (5th Edition by Lora Delwiche & Susan Slaughter).

Once you have a basic understanding of the language, google and the official SAS documentation will become your best resource.