

Learning SAS® by Diagrams and Examples

Joanne Peng, Indiana University
Shu-Yeng Wong, IUPUI

ABSTRACT

This paper outlines an alternative method of teaching SAS to computer-inexperienced and statistics-phobic students in social sciences. Specifically we are interested in demonstrating how the SAS system and its programming capability can be easily understood through diagrams and examples. We believe diagrams are so effective visually that they assist students in conceptualizing the structure and logic inherent in SAS programming language. Hence, they help to reduce logical errors often made by beginning-level users who lack training in any formal programming language.

Examples of short SAS programs, on the other hand, can be used as templates by students. These templates teach students to "speak" the SAS language correctly, thus, minimizing the likelihood of making syntax errors.

This alternative method of teaching has been tried in an introductory SAS course at Indiana University for four years. Students who completed such a course readily demonstrated an ease with reading SAS manuals and, therefore, were capable of handling their own data analysis tasks independently.

We intend to illustrate our alternative teaching method with examples drawn from our teaching, research, and consulting experiences with a variety of projects.

INTRODUCTION

Over the years, we have witnessed many students trying hard to be initiated into the wonderful network of the "SAS Users Club," and yet stumbling countless times along the way before they finally earned the right to entry. One reason that these students had to struggle so hard is because they typically lacked computer skills in their background—most of them were social science majors, had taken one or two statistics course(s), yet never took a course in a "formal" programming language and probably never would. Another reason is that most SAS manuals are next to impossible to comprehend; they appear to have been written by experts who already know the system and who have difficulty communicating to readers about what they know. Consequently these books are not "user friendly" and answers are seldom found in the most accessible and "logical" places in the manual, except for the SAS Introductory Guide. The type of errors made by entry-level users usually is either (1) the logical error or (2) the syntax error.

To help these users avoid making these types of errors and ease their difficulty of learning SAS, we subsequently developed a one-credit hour mini course. The goal of this course is to train graduate students in social sciences to be well-versed in SAS language so that they can become self sufficient in analyzing data for their dissertations or for research projects.

The particular teaching method we use for this mini course consists primarily of diagrams and examples. Since diagrams are visual, they help students form concepts about the logical structure inherent in the SAS system. They also provide an alternative learning aid for students when dealing with abstract and technical features of SAS.

Examples of simple, short SAS programs, on the other hand, are used as "templates" or "building blocks" for complex programs in the future. These examples are oriented toward solving a real-world problem and, at the same time, teaching students to "speak" the SAS language correctly, thus, minimizing the likelihood of making syntax errors.

After introducing each instructional unit, students engaged in activities such as (a) checking on their own understanding with a set of multiple choice questions, (b) solving two to four problems in ten days, (c) providing feedback on the written materials furnished by the instructor, and (4) submitting their own multiple choice questions for future students enrolled in this course. Throughout the course, the instructor not only assisted students in interpreting various error messages but also compiled them into a Wordperfect file which would benefit prospective students enrolled in the same course.

This alternative method has been tried at Indiana University for the past four years. Students who completed such a course readily demonstrated an ease with reading SAS manuals and, therefore, were able to handle their own data analysis tasks independently. This confidence prepared them for other SAS products such as SAS/Graph, SAS/ETS, etc. Several of them even went on to become SAS consultants at the university computing support center.

We feel this approach to teaching SAS to beginners can be easily modified for alternative user groups or nonacademic settings. It requires only that users be naive and yet motivated. How hard is it to meet this requirement? It cannot be harder than reading a SAS manual!!!

A 3-Stage Concept

At the core of our teaching method is the conceptualization of a 3-stage approach to data analysis/file management within the SAS system. The first stage is referred to as the INPUT stage, the second stage is the PROCESS stage, the third is the OUTPUT stage.

Within each stage, we further introduce detailed concepts such as, Raw Data File and SAS Data Set at the INPUT stage, or Data Processing and Procedure Processing at the PROCESS stage, and Raw Data File, Customized Report, SAS LOG/SAS Listing, and SAS Data Set at the OUTPUT stage. Figure 1 illustrates this 3-step concept.

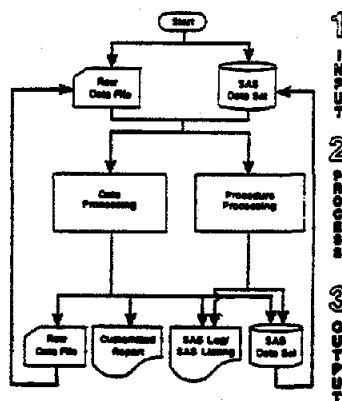
Within the Data Processing, commands are grouped into nine categories due to their similar functions:

Documentation, Transformation, Conditional/repetitive processing, Variable selection, Observation selection, Data set management, User interface, Data indexing and retrieval, and Declaration.

Likewise, various selected statistical procedures are combined into six subtypes, depending on their analysis objectives, which are listed under Procedure Processing. These six subtypes include:

Descriptive analysis, Summative tabulation, Categorical data analysis, Mean analysis, Prediction, and Classification.

3-Stages To Learning SAS



What is so special about this 3-stage concept?

The conceptualization of the 3-stage approach of teaching SAS grew out of our experiences as an instructor and a computer center consultant in an academic setting. The target recipients of this teaching method were students with diverse professional backgrounds; the shared commonality among them was their lack of experience with any computer programming language. For this reason, we relied extensively on a visual presentation format to get many of our ideas across. Thus, the first special feature about this conceptualization is its reliance on graphical representations of concepts and relations implied in SAS system.

The second special aspect about our approach is the circular nature of SAS statements. Thus, Figure 1 shows a **STARTING** point on the top of the diagram but no stopping point. In essence, this figure is to capture the endless possibilities of interweaving commands from all three stages within the same SAS program. Yet commands classified into a particular stage can be interchangeably called up without any particular consideration given to the sequencing effect. Commands belonging to different stages do have to follow the general sequence of **INPUT** --> **PROCESS** --> **OUTPUT** (then back to the INPUT stage again). For example, a SAS Log or SAS listing or a SAS data set has to be created, or output, after the **Procedure Processing**. It is unnecessary, however, to follow the rigid sequence of three stages in their order; it is possible instead to skip a stage in between. For instance, a raw data file (**INPUT**) can be read into a SAS data set (**OUTPUT**) without the intermediate **Data Processing** or the **Procedure Processing**.

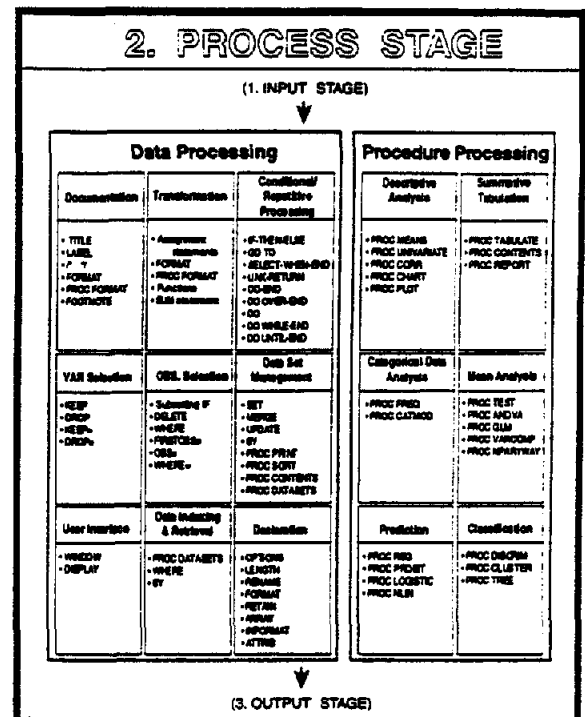
The third characteristic, though minor to this conceptualization, lies in the choice of terms which are standard terminology in computer programming languages. Thus, upon completing this course, if a student wishes to continue pursuing additional computer science courses, he/she will have a minimal transition with concepts and terms.

The fourth and the most important characteristic is our consideration of an entry-level user's point of view. The three-stage approach is more holistic than the traditional dichotomy of DATA step and PROC step, presented in all SAS manuals and publications. Though the separation of DATA step and PROC step is technically correct, it nevertheless is too simplistic to help beginners form a picture of how various commands are interconnected. Furthermore, there are too many rules, too much material to cover within each step. For example, the following commands can all go into the DATA step:

TITLE, LABEL, FORMAT, functions, PUT, INPUT, SUM statements, KEEP, DROP, various DO statements, LINK RETURN, DELETE, OUTPUT, WHERE, SET, MERGE, UPDATE, BY, LENGTH, RENAME, RETAIN, ARRAY, INFORMAT, ATTRIB, and many others.

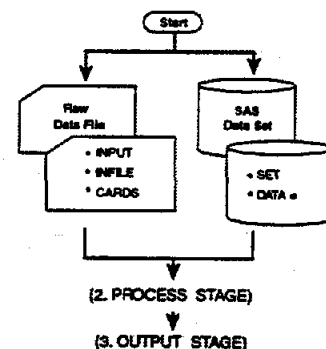
Without a doubt, this list of commands is overwhelming to a beginner; it makes the learning process difficult to even get started. The lack of a proper understanding of how to classify commands is frustrating for user to pass beyond the DATA stage. In frustration, he/she typically resorts to a blind trial-and-error approach in debugging errors which results in more frustration.

Yet our conceptualization of **Data Processing** provides a systematic clustering of commands according to their functions. It is only partially related to the DATA step concept, since **Data Processing** includes all that is necessary or desirable for preprocessing user's data so that they become suitable either for forming a SAS data set or for additional statistical analysis (see Figure 1). Under this definition, **Data Processing**, therefore, includes many of the typical DATA step commands plus sorting, formatting procedures and generic PROC DATASETS, PROC CONTENTS for data indexing and data set management. Figure 2 presents such a nomenclature of various commands and PROC's which we consider relevant and intuitive to user's idea of "data processing."

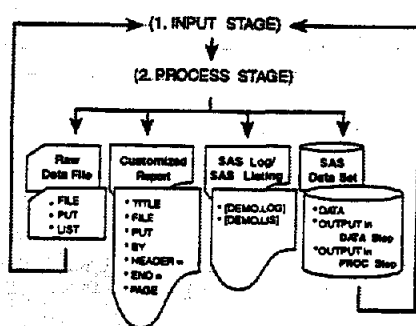


Figures 3 and 4 present in-depth SAS commands classified into **INPUT** stage and **OUTPUT** stage respectively.

1. INPUT STAGE



3. OUTPUT STAGE



How is this approach to teaching/learning implemented?

Our proposed approach to teaching and learning SAS has been implemented in a 1-credit hour course at Indiana University for the past four years. The frequency of course offering has been once each semester; the instructor is the first author. Enrollment of this course is approximately 10 with a slight increase observed in the past two semesters. Since this course is an elective course, enrolled students are truly motivated to learn the materials and have an above average ability/apptitude for computers. The majority of students are education majors; others come from diverse backgrounds such as kinesiology, marketing, economics, statistics, applied math, science education, nursing, and computer science. All students have been graduate students.

The course has been oriented toward mainframe SAS version because of IU's open policy on computing time (what a blessing!!!). Over the years, we have witnessed an increase in students' knowledge about the mainframe operating system, the editor, the mail utility and interface between the mainframe and the personal computer. Consequently we spent no more than half a lecture (or one hour) on these concepts in the current semester.

Teaching Materials

Teaching SAS for one credit hour has not been easy; the format we are presently using is 10 weekly lectures and 5 take-home assignments. Learning materials have been divided into 5 instructional units. Thus, each unit is covered in two lectures, each of which lasts 2 hours. To help students minimize their anxiety about learning, teaching materials were distributed to students as Wordperfect files which students could print into a hard copy prior to each lecture. A table of contents is presented below:

SAS Unit 1

Introduction to SAS

Where do I start ?

3-stages to learning SAS

statements

The general appearance of a SAS command file

Steps for creating and submitting a SAS command file

Input Stage

reading the raw data file

INPUT

INFILE

CARDS or DATALINES

using SAS data set in a DATA step or a PROC step

SET statement in a DATA step

DATA = option in a PROC step

Output Stage

creating a SAS data set

DATA

OUTPUT in DATA step

OUTPUT in PROC step

creating a raw data file

FILE

PUT

LIST

Data Processing : Transformation

Simple Arithmetic Operations: +, -, *, /, **

Constant Assignment

Arithmetic Functions

Trigonometric and Hyperbolic Functions

Functions That Generate Pseudo-random Variables

Selected Functions That Calculate Simple Statistics

SUM and SUM Function

Other Functions

Quiz for Unit 1

SAS Unit 2

Data Processing : Documentation

TITLE

LABEL

FORMAT

PROC FORMAT

/* Comment */

Data Processing : Declaration

FORMAT

INFORMAT

LENGTH

ATTRIB

RENAME

RETAIN

ARRAY

OPTIONS

Data Processing : VAR Selection

KEEP

DROP

KEEP =

DROP =

Data Processing : OBS. Selection

Subsetting IF

WHERE

DELETE

FIRSTOBS =

OBS =

WHERE =

Data Processing : Data Set Management

How and Why Do You Reshape SAS Data Sets ?

SET

MERGE

UPDATE

The Use of IN = variable in a SET, MERGE, or UPDATE statement

PROC SORT

PROC CONTENTS

PROC PRINT

Quiz for Unit 2

SAS Unit 3

Data Processing : Conditional Processing

Comparison Operators and Logical Operators

IF-THEN/ELSE

DO-END

SELECT-WHEN-END
GO TO
LINK-RETURN

Data Processing : Repetitive Processing

The ARRAY Declaration

DO
DO OVER - END
DO WHILE - END
DO UNTIL - END

What is the Generic Logic Behind the Iterative DO, DO WHILE, and DO UNTIL ?

Quiz for Unit 3

SAS Unit 4

Procedure Processing : Descriptive Analysis

PROC MEANS
PROC UNIVARIATE
PROC CHART

Procedure Processing : Categorical Analysis

PROC FREQ

Quiz for Unit 4

SAS Unit 5

Procedure Processing : Mean Analysis

PROC TTEST
PROC ANOVA
PROC NPAR1WAY

Quiz for Unit 5

Appendices

Appendix A : Answer Keys to Quiz Questions

Appendix B : Typical Error Messages on the LOG File

Appendix C : Challenge Questions

Appendix D : Information for Advanced Users

As can be noted from this table of contents, topics grouped into the same instructional unit have relationship with one another and present a coherent picture to the learners. During the lecture, the instructor illustrated SAS commands with examples and diagrams based on the 3-stage concept and principle. Students' learning was reinforced by hands-on practice with interactive SAS, installed on a VAX mainframe, and by discussion about their assignments, all taking place in the classroom.

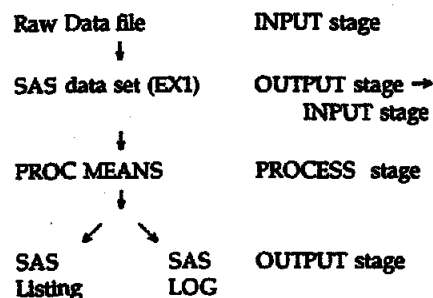
Diagrams/Examples used in Classroom Instruction

Three examples are presented here to highlight the 3-stage concept:

Example 1: [Establishing a SAS data set and carrying out a mean analysis]

```
DATA EX1; /* output stage */
INPUT ID $ SEX $ V1 V2 V3; /* input stage */
CARDS; /*
011 F 1 2 3 /*
012 F 3 4 5 /*
013 M 5 6 7 /*
014 F 7 8 9 /*
015 M . 0 -3 /*
; /*
PROC MEANS DATA=EX1; /* process stage
VAR V1 V2 V3; /*
RUN; /* output stage */
```

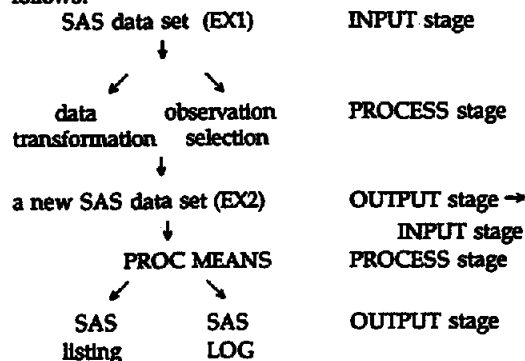
The underlying structure we hope to communicate to students with this example is as follows:



Example 2: [Data transformation and average analysis]

```
DATA EX2; /* output stage */
SET EX1; /* input stage */
IF SEX = 'F'; /* OBS. selection */
V1 = SQRT (V1); /*
V2 = 1/V2; /* data transformation */
V3 = (V1 + V2)/2; /*
PROC MEANS /*
DATA=EX2; /* process stage
VAR V1 V2 V3; /*
RUN; /* lls and log outputs */
```

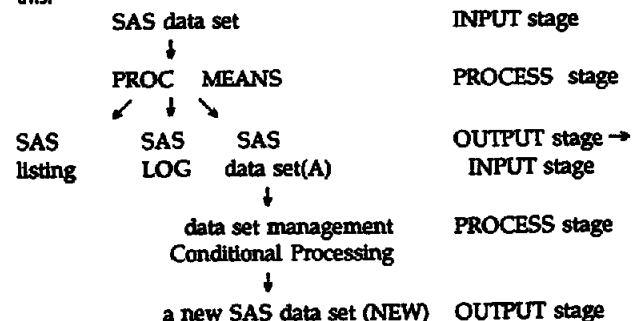
The underlying framework for this example is as follows:



Example 3: [Substituting missing data by group means]

```
PROC MEANS DATA=EX2; /* process stage
VAR V1; /*
OUTPUT OUT=A MEAN=M1; /* output stage
DATA NEW; /* output stage
SET EX2; /* input stage
IF _N_ = 1 THEN SET A; /* process stage
IF V1 = . THEN V1 = M1; /* process stage
```

Obviously this example is more complex than the previous ones; the concept we are hoping to teach to students is this:



These examples were illustrative of the teaching style and the layout of the course materials used throughout the lectures. In other words, these examples were presented in a sequence of increasing difficulty and complexity. In the beginning of the course, simple examples which contain one SAS command or statement are given. As the course progresses, two or more commands/statements are incorporated into the same examples (complex examples). The one-command-per-example approach effectively teaches students the correct syntax of each and every SAS command. The multiple-command-per-example approach, on the other hand, demonstrates how to use SAS to solve a real-world problem. Thus, these complex examples are intended to minimize students' logical errors. The value of accumulating these practical problem-solving skills attracts students to this course over and over again.

For instance, we pose students with this question: "If a questionnaire on *Family Values* contains negatively worded questions and yet the same Likert scale (of 1-5 where 1=strongly disagree, ..., 5=strongly agree) has been used for all questions, how do you reverse scores obtained from these negative items so that they mean the same thing as all other scores?"

We then show six methods for solving this problem--three methods for reversing a single item and three for multiple items. Each solution is in fact a short SAS program with explanations given on the side (see [Case 1](#) and [Case 2](#) below). By studying these solutions and the accompanying explanations, students not only find an answer to a problem they frequently encounter in their own work, but also learn to "talk" like a real SAS user.

Case 1: Scale reversal for a single item

SAS Programs	Explanation
<pre>IF a=5 THEN a=1; ELSE IF a=4 THEN a=2; ELSE IF a=2 THEN a=4; ELSE IF a=1 THEN a=5;</pre>	Use "IF...THEN" statement to set up conditions under which the variable value of "a" changes accordingly.
<pre>SELECT (a); WHEN (5) a=1; WHEN (4) a=2; WHEN (2) a=4; WHEN (1) a=5; OTHERWISE;</pre>	Follow the same logic as "IF ...THEN" statement above.
<pre>a = 6 - a;</pre>	Computationally change the variable value by the proper formula.

Case 2: Scale reversal for multiple items

SAS programs	Explanation
<pre>v1 = 6 - v1; v2 = 6 - v2; v3 = 6 - v3; v5 = 6 - v5; v7 = 6 - v7;</pre>	Reverse five items, identified as v1,v2,v3, v5, and v7 by the computational method illustrated above.
<pre>ARRAY v {5} v1 v2 v3 v5 v7; DO i = 1 TO 5; v(i) = 6 - v(i); END;</pre>	Use the explicitly subscripted ARRAY statement and "DO...END" logic to reverse the same five items as before.
<pre>ARRAY v v1 v2 v3 v5 v7; DO OVER v; v = 6 - v; END;</pre>	Use the implicitly subscripted ARRAY statement and "DO OVER...END" logic to reverse the same five items as before.

Since we believe in the notion of "learning for mastery (LFM philosophy)", there are several learning activities we designed for students which are all aimed at augmenting students' capacity to transfer what they learn from lectures, written examples, or hands-on experience to real-world problems. These activities include (a) answering approximately 20 multiple-choice questions at the end of each unit, (b) completing weekly assignments, and (c) providing feedback on lectures and written coursenote on a biweekly basis.

Quiz Questions

The multiple-choice questions, treated as a QUIZ, were questions which previous students had constructed and which were subsequently edited by the instructor for accuracy and clarity. Below are two multiple-choice questions drawn from Unit 1:

- The DATA Step in SAS is equivalent to all of the following in SPSS except for _____.
 - FILE HANDLE
 - DATA LIST
 - MISSING VALUES
 - VARIABLE LABELS
 - REGRESSION analysis
- Missing values in LIST format are coded by a(an) _____.
 - * (asterisk)
 - . (period)
 - () (parentheses)
 - ; (semicolon)
 - 0 (zero)

During and after presentations of each instructional unit, students were encouraged to check their own understanding with these multiple choice questions and answer keys in the back of the coursenote. The instructor did not score their responses.

Assignment Problems

The second learning activity, i.e., the weekly assignment, asked students to solve 2 or 3 real-world problems each week. The weekly assignment examines students' ability to apply the knowledge and logical thinking to problems they might face someday. For example, the following two problems were assigned on the unit which dealt with file management and categorical data analysis:

- A. Given two data sets below, can you think of a way to combine them so that a travel agent may generate complete information for your spring vacation which consists of the date, flight number, carrier, from (city), to (city), the time of departure and arrival.

		Data NUMBER			
variables:	Date	Number	Carrier		
	030493	23	NW		
	031293	405	U.S.		
	031493	74	American		
	032093	328	TWA		
	032593	18	NW		
	032993	92	TWA		
	033093	266	Eastern		
		Data TIME			
variables:	Date	From	To	Depart	Arrival
	030493	New York	L.A.	9:45am	5:05pm
	031293	Chicago	Miami	3:20pm	9:35pm
	031393	Indy	Twin City	8:30am	10:15am
	032093	Denver	St Louis	1:00pm	6:08pm
	032593	St Louis	Cleveland	4:15pm	7:12pm
	032993	Boston	SF	7:55am	3:16pm
	033093	Madison	Indy	2:18pm	11:43pm

B. DATA A
VARIABLES:
NAME SEX AGE ROOM S.E.S INCOME (in \$1000)

ANDY	M	28	122	H	9.6
SOPHIA	F	42	412	M	7.4
TED	M	68	101	H	8.8
DICKENS	M	37	135	M	6.3
RUTH	F	39	430	L	1.3
CHARLIE	M	54	222	M	4.7
WOLF	M	51	104	H	8.4
ANNE	F	58	404	H	10.2
DAVID	M	47	117	M	5.2
RICHARD	M	28	240	H	9.2
SAM	M	36	231	M	6.4
PETER	M	65	108	L	3.3
LINDA	F	20	425	M	5.5

Given the variables of Data A listed above, please:

- Sort the Data A by name alphabetically.
- Produce two separate percentage barcharts according to sex, and S.E.S.
- Suppose that five age groups are defined according to the following age ranges:
group 1 = 20-29 group 2 = 30-39 group 3 = 40-49
group 4 = 50-59 group 5 = 60-69
Produce a frequency barchart according to age group.
- Assuming that the room numbers represent the floor levels.
For example: Room 404 is a room on the fourth floor and Room 117 is on the first floor.
Sort the data set first by floor level of the rooms and then by sex. Find the relationship between sex and floor level of room.
- Describe the extreme values of INCOME variable. Draw plots and discuss if it's distribution follows a normal curve.

For these weekly assignments, a letter grade is assigned based on three criteria: accuracy, programming style and punctuality.

Students' Feedback

The third learning activity, feedback from students on the course note, is most helpful to the instructor in terms of future course revisions. The feedback is obtained by using a semi-structured form consisting of three questions: (a) What sections of the Unit 1 (or 2,3,4,5) materials are not clear to you? (b) Did you notice errors in the materials? If yes, please describe it/them. (c) What changes would you suggest to improve the materials?

One response we always received on question (c) above has been: Include more examples on topic XXX. Here are some of the direct quotes from students who are enrolled this semester:

"Could you give me more examples to help me understand these materials?"

"page 25-example 11, please give 3 to 4 observations so that this is clear to novice users. Please give example for page 39 as how we can use these internal variables."

"I felt Unit 2 material was very clear and well supported by questions and examples. The only change as I stated with Unit 1 is to include more examples or questions."

"I felt the materials for the most part were very clear."

Incorporating the different examples and questions regarding the material is very helpful. Improving this material would consist of doing more of what is already incorporated--just a few more examples or more in-depth questions. So far I have liked the material and have found it to be helpful."

How do we help learners correct errors?

The best way to help learners correct errors is to remind them of principles used in SAS syntax and logic. For example, the most popular error made during the first week of lecture is caused by forgetting a ";" at the end of one or more SAS statement. As a result, we have to constantly drill into students' mind that a SAS statement is like a complete English sentence. When a statement is completed, it ends with a semi-colon, just like a period (".") in ordinary sentences.

Other rules, such as using "." for missing observations or restricting data set names or variable names to 8 characters or fewer, are likewise repeated over and over again until students could not stand it any longer; consequently they master these rules like a second nature.

Typical error messages and their interpretation/correction have been compiled into an appendix, attached at the end of the course note, for students' reference. A sampler of this appendix is enclosed here:

Appendix B: Typical error messages on the LOG file

- WARNING 14-169: Assuming the symbol CARDS was misspelled as CAR.**

DATA; INPUT S T;

CARD;

14

WARNING 14-169: Assuming the symbol CARDS was misspelled as CAR.

This warning is self-explanatory. The command CARDS; cannot be spelled as a singular word (Sorry!!).

- ERROR 23-2: Invalid option name INPUT.**

DATA; INFILE ASS1

INPUT X \$ 3. Y 3.;

23

ERROR 23-2: Invalid option name INPUT.

This is clearly a beginner's error because [;] is missing from the end of INFILE statement. This problem can be easily erased by the lines below:
DATA; INFILE ASS1; INPUT X \$ 3. Y 3.;

The Measure of our Success

This alternative method of teaching has been tried in an introductory SAS course at Indiana University for four years. Students who completed such a course readily demonstrated an ease with reading SAS manuals, were capable of handling their own data analysis tasks independently, became intrigued by other SAS products such as SAS/Graph, SAS/ETS, etc., and most impressively, were hired by computer centers at Indiana University and Purdue University as consultants on SAS!!!

SAS, SAS/Graph, SAS/ETS are registered trademarks of SAS Institute Inc.

Wordperfect is a registered trademark of WordPerfect corporation.

VAX is a registered trademark of Digital Equipment Corporation.

In the USA and other countries, ® indicates USA registration.

Authors

Joanne Peng

Room 4050, School of Education
201 N. Rose Avenue
Bloomington, IN 47405-1006
peng@ucs.indiana.edu
phone: (812) 856-8337
fax: (812) 856-8440

Shu-Yeng Wong

Integrated Technologies
IUPUI, ET #1023
799 W. Michigan St.
Indianapolis, IN 46202
syw400@indycms
phone: (317) 274-0726
fax: (317) 274-0776