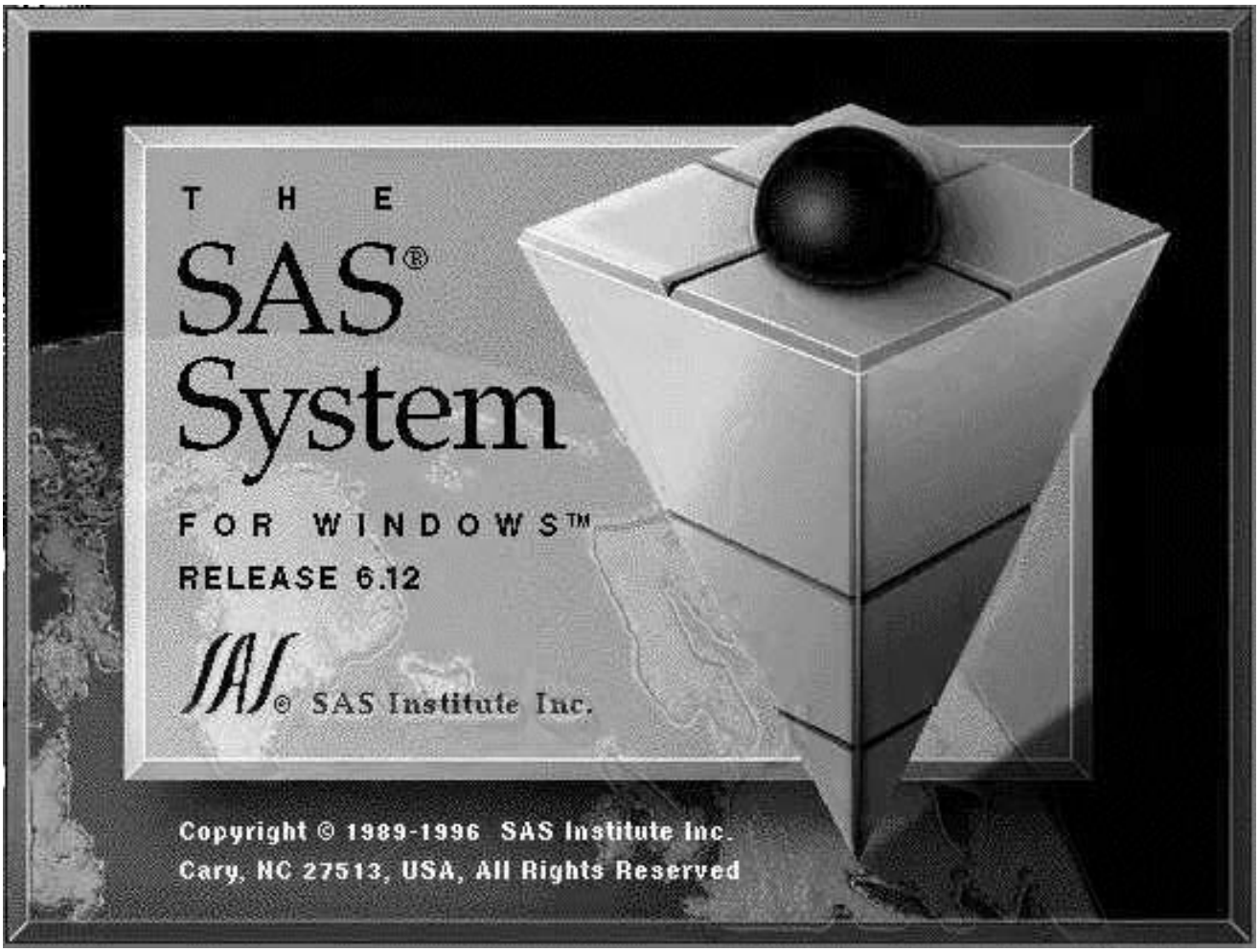

GETTING STARTED WITH SAS FOR WINDOWS

Summer 2000 **June 2000**

Calvin L. Williams, Ph.D.
calvinw@math.clemson.edu
http://www.math.clemson.edu/~calvinw/

Contents

1 Introduction

This document is an introduction to SAS for Windows for IBM compatible computers. SAS is a large software package with scores of modules and utilities. It is impossible to present all its features in a brief document so we focus on the basics of getting started. If you do not own a personal copy of SAS for Windows, you may access the software from various Clemson University Student Computing Labs. If you want to buy a copy of SAS for Windows at an educational discount to install on your own PC's, contact the DCIT and Jon Hoskins.

2 What is SAS

The SAS System is a comprehensive and flexible information delivery system that supports data access, data management, data analysis, and data presentation. At Clemson, SAS is available from several platforms including MS-Windows, HP-UX, DEC OSF/1, and IBM AIX.

3 SAS for Windows

SAS for Windows is a complete implementation of the SAS System. It enables you to perform many analyses on your PC that were once possible only on much larger machines. Windows enables you to communicate between applications. SAS for Windows also reads data files from a variety of file formats including Excel, dBase, Lotus, and SPSS.

4 Windows in SAS for Windows

The SAS System under the Windows environment consists of three types of windows:

1. SAS Application Workspace (AWS) - The SAS Application Workspace (AWS) contains all SAS windows that are open, including those that have been minimized. The only exceptions are the Command dialog box and the Toolbox, which can be moved outside the SAS AWS. The main function of the SAS AWS is to provide a framework for all SAS application windows.

2. Child windows - The child windows are individual windows within the SAS AWS, such as the PROGRAM EDITOR, LOG, and OUTPUT windows. These windows behave like any other windows in that you can maximize, minimize, scroll, and resize them. But because they are child windows of the SAS AWS, you cannot move them outside the boundaries of the SAS AWS window.

3. Dialog boxes - Dialog boxes appear when the SAS System needs more information to complete a task. For example, when you issue a command to exit your SAS session, a dialog box asks if you are sure you want to terminate the session. There are scores of instances a dialog box might appear. When a dialog box appears, you may have to respond to the query before you can proceed.

4.1 SAS AWS menu bar, pull-down menus, and pop-up menus.

When you invoke the SAS is invoked by clicking the SAS icon from the Main window, the session opens with LOG and PROGRAM EDITOR windows within the SAS AWS. The SAS AWS menu bar is just above the LOG window. From the SAS AWS menu bar, you can access the File, Options, Windows, and Help features of the SAS System. Each of these items has a pull-down menu which presents you with other options.

In the following section all the four pull-down menus (File, Options, Windows, Help) are briefly discussed. For detailed information on SAS AWS pull-down menus refer to the vendor-supplied manual, SAS Companion for the Microsoft Windows Environment.

4.2 File Pull-Down Menu

The File pull-down menu shown here offers the following choices:

1. Open enables you to copy a program into the PROGRAM EDITOR window or to immediately run a program. This selection will display an Open dialog box. The Open dialog box, like the Open dialog box in other Windows applications, permits you to select filename, drive, and directory of the file you want to copy into the PROGRAM EDITOR window. Choose OK to copy the program to the PROGRAM EDITOR. Choose Run to run the file immediately.

2. Save As enables you to save a text file to a disk. This option is equivalent to using the display manager FILE command to save text. The Save As dialog box cannot be used to save graphics files.

3. Print enables you to spool output to the Print Manager.

4. Printer Setup enables you to choose and configure your default printer to be used with the SAS System.

5. Run enables you to asynchronously execute an operating system command or another Windows or DOS application (similar to the File Run selection in the Windows Program Manager).

6. Exit SAS enables you to end your SAS session.

   Note that many child windows of the SAS AWS (such as PROGRAM EDITOR and OUTPUT windows) also have a File menu that is similar to that of SAS AWS File pull-down menu.

4.3 Options Pull-Down Menu

The Options pull-down menu shown here offers the following choices:

1. Minimize (Restore) all Windows enables you to minimize all open SAS windows or to restore minimized windows to full size.

2. Fonts enables you to change the text font and point size for text in SAS windows.

3. Preferences enables you to choose and save your display configuration. This is a powerful option to customize your SAS session. You may make selections in the Preference dialog box and choose Save to save your selections.

4. Edit Toolbox enables you to define new toolboxes.

5. DDE Triplet enables you to determine the DDE (Dynamic Data Exchange) triplet for an application by way of the Clipboard.

4.4 Windows Pull-Down Menu

The Windows pull-down menu shown here offers the following choices:

1. Cascade arranges the open SAS windows in layers

2. Tile arranges the open SAS windows in a mosaic format.

3. Resize returns the display configuration to the last saved configuration. A list of all open SAS windows is displayed below the Resize option. The windows in the open window list are in order of the time they were opened (i.e. most recent first). The active window is indicated by a check mark. To change the active window, click on the name of the window you want to activate. If more than nine windows are open concurrently within the SAS session, the last item on the Windows pull-down menu is More Windows...

4.5 Help Pull-Down Menu

The Help pull-down menu shown here offers the following choices:

1. Extended Help displays help for the SAS AWS window.
2. Keys enables you to display and change function key definitions.
3. SAS System provides general information about features of the SAS System.
4. Install invokes the installation process.
5. Utility invokes various utility applications.
6. About includes information about the SAS System copyright, as well as details of your system configuration.

4.6 Pop-up Menus

To see the pop-up menus for a given SAS window, press the right mouse button in while in a SAS window. After the menu appears, you can click your menu selections just as you would with menu bars. Selecting a pop-up menu can lead you to a series of cascading menus. You may choose pop-up menus, menu bars, command line or command dialog box to issue SAS commands. Choose the Preferences in the SAS AWS Options pull-down menu to switch between and combine these methods of issuing commands. Refer to SAS Companion for the Microsoft Windows Environment for further information on pop-up menus, and other methods of issuing commands during a SAS session.

5 Working Within your SAS Session

The section provides some of the basic information you need to use the SAS System efficiently under Windows.

5.1 Using the Cursor

Typeover mode is the default for SAS System text entry. To toggle between insert and typeover mode press the INSERT key. The cursor appears as a rectangular block in typeover mode, whereas in insert mode it appears as a slender bar, called an I-beam. You can change the default cursor setting by opening the Preference dialog box and choosing the option you want and saving it.

5.2 Using Icons

Each SAS window (e.g. PROGRAM EDITOR, OUTPUT) has an icon associated with it. When a window is minimized, its icon appears at the bottom of the SAS AWS. You can activate these windows by clicking the icon. If it is obscured by other open windows you can activate it by selecting its name from the SAS AWS Windows pull-down menu. To make it active, either click open or select it from the SAS AWS Windows menu. You can add your icons to the SAS System using the USERICON system option. Refer to SAS Companion for the Microsoft Windows Environment for details.

5.3 Using the PROGRAM EDITOR

The PROGRAM EDITOR window has been designed to work in a manner similar to other Windows editors such as Notepad. Thus you can edit your SAS code without learning how to use a new text editor. Given below are some of the features of the PROGRAM EDITOR window.

5.4 Line Numbers

In the PROGRAM EDITOR window the line numbers are turned off by default. You can turn the line numbers on by entering the numbers on command from the command line of the PROGRAM EDITOR window, or by selecting the path Edit/Option/Numbers from the PROGRAM EDITOR window pop-up menu. You may also save this option to change the default setting.

5.5 Cursor Movement

The cursor movement keys (e.g., up, down, left, right, PgDn, PgUp) function the same way in the PROGRAM EDITOR window as they do in other Windows applications. Pressing the CTRL key with the right or left arrow key causes the cursor to move forward or backward, one word at a time on a given line. Pressing the HOME key causes the cursor to go to the beginning of the current line. However, if the command line is active, pressing the HOME key will cause the cursor to toggle between the current cursor position in the text and the command line.

5.6 Tab Key

Many text editors retain tab characters, while others expand tabs into space characters. The PROGRAM EDITOR window expands tabs into space characters. Pressing the TAB key inserts spaces and moves any text to the right of the cursor to the right.

5.7 Line Breaks

Pressing the ENTER key creates a line break.

5.8 Marking Text

When the SHIFT key is used with the cursor movement keys, characters are marked. The marking of an area of text continues until a key that is not a cursor movement key is pressed. Pressing an unshifted cursor also ends the marked area.

If characters are marked and you start typing text, the marked area is replaced with the new text. This occurs even if you have moved the cursor away from the marked area.

5.9 Deleting Text

The DELETE key deletes a marked area of text if one exists; otherwise, the character to the right of the cursor is deleted. If the cursor is located past the last text character on the line, the next line is concatenated onto the end of the line containing the cursor.

5.10 Using the Clipboard

The Windows Clipboard utility enables you to exchange text and graphics between applications. You can also submit SAS code stored in the Clipboard. The Clipboard uses DOS memory as an intermediate storage buffer for exchanging text and graphics. With the Clipboard you can move text between windows within the SAS System, and between the SAS System and other Windows applications.

5.11 Marking and Copying Text

For windows that contain text, like the PROGRAM EDITOR, LOG, OUTPUT, KEYS, and OPTIONS windows, you can hold down the left mouse button and drag the mouse to mark the area you want to cut or copy. The text area is immediately marked in reverse video while you are dragging the mouse. In text windows, you can scroll while you are dragging the mouse by moving the cursor beyond the border of the window in the direction you want to scroll. Release the mouse button when you have included all the text you want to copy.

To copy marked text to the Clipboard, select Copy to paste buffer from the Edit pop-up menu. The reverse-video area is restored to its original appearance. To paste text stored in the Clipboard, position the cursor in a text area in a window and select Paste text from the Edit pop-up menu. The text from the Clipboard is pasted to the area you indicate.

For information on marking and copying text in non-text windows (e.g., HELP, GRAPH) refer to SAS Companion for the Microsoft Windows Environment.

5.12 Submitting SAS code from the Clipboard

The SAS code stored in the Windows Clipboard can be submitted. To do so, you can select Locals and then Submit clipboard from the Edit pop-up menu or you can use the GSUBMIT (gsubmit; buf=default) command.

5.13 Using the Toolbox and Toolbar

The SAS System under windows provides two methods of associating icons with display manager commands: the Toolbox and the Tool bar (default tool bar shown below). These two features are similar. The Toolbox is a window containing icons and the Tool bar is a sequence of icons in the menu area of the SAS AWS.

By default, no tools are displayed when the SAS System initializes. To choose between the Toolbox and Tool bar use the Preference dialog box and make your selection and press Save. If you do not want any tools displayed, choose off from the Preference dialog box. The toolclose command from the display manager command line will also turn off the toolbox.

6 Getting Started

The examples given in this document are for working with the SAS System under the Microsoft Windows environment. It is assumed that the reader is familiar with the fundamentals of DOS and Windows systems. It is also assumed that the reader has basic statistical knowledge. This document is not intended to substitute the vendor-supplied SAS manuals. The term SAS refers to the software command language, since the basic command structure is the same across all platforms for SAS products.

6.1 Organizing your data for analysis

SAS uses data organized in rows and columns. Cases are represented in rows and variables are represented in columns. A case contains information for one unit of analysis (e.g., a person, an animal, a machine). Variables

are information collected for each case, such as name, score, age, income, educational level, etc. When data in files are arranged in rows and columns, they are called cases-by-variables, or rectangular data files.

Suppose you have three test scores collected from a class of 10 students (five males, and five females). Each student was assigned an identification number. The information for each student you have is an identification number, gender, and score for test one, test two, and test three. Suppose the data layout is as follows:

```
01 f 83 85 91
02 f 65 72 68
03 f 90 94 90
04 f 87 80 82
05 f 78 86 80
06 m 60 74 64
07 m 88 96 92
08 m 84 79 82
09 m 90 87 93
10 m 76 73 70
```

Notice that in the above data layout at least one blank space is left after each variable. In this instance you do not have to specify in which column a variable appears. SAS automatically considers a blank after each variable as a delimiter. In other words, if the data are space delimited, SAS will automatically know where each variable begins and ends. It is optional whether to leave a space between variable values. For example, you may choose to enter the data as following:

```
01f838591
02f657268
```

In this instance you need to assign column numbers to variables when using SAS to read the data. This is called a fixed format style input (where the variables across subjects are consistently in the same column). Format styles are discussed later in this document. Whichever format you choose, as long as you convey the format correctly to SAS, should not have any impact on the analysis. In the above layout each case/observation has only one line (record) of data. In another situation you may have multiple records per observation.

Your next task is to present the data in a form acceptable to SAS for processing.

6.2 Preparations for Writing a SAS Program

A SAS program consists of two steps: DATA steps and PROC steps. In the DATA step a user may include commands to create data sets, and programming statements to perform data manipulations. The DATA step begins with a DATA statement. In the PROC (Procedure) step you invoke SAS procedures from its library to run statistical analysis on a given data set. The PROC step begins with a PROC statement. These steps contain SAS statements. An important feature of SAS language is that every SAS statement ends with a semicolon (;). Without a semicolon a SAS statement is incomplete.

A. DATA Step

1. DATA dataname;

   The first word, DATA, tells SAS that you want to read a data file and store the data in a SAS data set you specify. Replace dataname with an appropriate SAS name (8 or fewer characters), i.e., trial, company, drug, behavior. In the example given below dataname is replaced by the name ANXIETY. Note the semicolon at the end of the statement.

   ```
   DATA anxiety;
   ```

2. INFILE 'pathname \ filename';

   If the data file is stored in a separate file an INFILE command is used to read the data set into the SAS program. Replace pathname with the name of the directory in which the data are

stored. SAS can read several data files from within the same program file, so you can have multiple DATA steps in a single SAS program file.

The INFILE command is entered immediately after the DATA line.

```
DATA test;
INFILE 'pathname\filename';
```

Replace pathname with the drive and path to your data file, and filename with the name of the file. For example, if you were reading a file called clas.dat from the A: floppy disk drive, the syntax would be:

```
DATA test;
INFILE 'a:\clas.dat';
```

3. INPUT var1 column# var2 column# var3 column# ..... varn column# ;

   The INPUT statement tells SAS the names of the variables and the column numbers that can be read on a specified line. Variable names in SAS can contain from one to eight characters. They may

   contain numbers but must begin with a letter. If your data contains more than one line per case (observation), indicate the line number before specifying the variables on that line, for example:

   ```
   INPUT id 1-3 company 8-10 #2 insal 6-10 finalsal 18-23 #3
   retire 15-19;
   ```

   The above INPUT statement informs SAS that there are 3 lines of data for each subject/observation. The lines are indicated by a # sign.

   The INPUT statement does not have to contain column numbers provided there is a space between each variable value on the data line. This is referred to as free format as opposed to the fixed format in which you specify the column numbers. If a variable contains character value indicate it by a $ sign after the variable name. If you are choosing the free format, a character variable should not exceed 8 characters and should not contain embedded blanks. Free format may not be a good idea if you have a large number of variables.

   If there are decimal points in your data, you may enter the decimal points as they are or leave them out when entering the data and later indicate in the INPUT statement that a given variable has specified number of decimal points. Suppose you have a variable GPA in your study and the value is to be indicated with 3 digits, the last two of which are decimal places. e.g. 3.89 If you decide to enter the decimal points in your data file indicate this in your INPUT statement as: INPUT GPA 1-4; Another choice is to leave out the decimal (389) and later indicate in the INPUT statement that the variable GPA has 2 decimal points: INPUT GPA 1-3 (.2); This means that the variable GPA is given in col. 1-3, and the last 2 places are decimal places.

   The input format can also be written in a shorter form with a mixed style column and formatted input.

   ```
   INPUT ID 1-2 SEX $ 3 (EXP SCHOOL) (1.) (C1-C10) (1.)
   (M1-M10) (1.) (MATHSCOR COMPSCOR) (2.);
   ```

   In this case the program will read the variable ID from columns 1-2 and SEX from column 3. The next two variables, EXP and SCHOOL, have a width of 1 column each and start at column 4. The variables C1 through C10 (10 variables in sequential order) have a width of 1 column each and start immediately after the variable SCHOOL. If you want to read only the variables ID and the last 2 variables (MATHSCOR, COMPSCOR) you could write the INPUT statement as:

   ```
   INPUT ID 1-2 @26 (MATHSCOR COMPSCOR) (2.);
   ```

The @ moves the pointer to column 26 and reads two variables with a width of 2 columns each.

4. CARDS;

   The CARDS statement is used only if the data are embedded within the SAS program file. The CARDS statement tells SAS that data lines are included in the command file. Indicate the end of data lines by a semicolon at the beginning of a new line. For example:

```
CARDS;
25 32 82 32 1
22 42 .  36 2
;
```

   The CARDS statement is entered toward the end of the DATA step.

5. IF-THEN and SAS Functions

   Missing values in a data set can be represented either by a blank or by a period. If you choose a free format (leaving a space after each variable in the data set and not specifying the column numbers in the INPUT statement) make sure you represent missing values with a period. When SAS encounters a blank or a period in a data set the system regards it as missing value. You can assign a missing value to a variable (e.g. 9, 99, 999, 000) and let SAS know which value for a given variable is assigned as missing. Suppose, for a variable MATHSCOR, 99 is assigned as missing value. Immediately after the INPUT statement you may specify:

```
IF test1=99 THEN test1=.;
```

   This statement will assign a missing value whenever it encounters a value of 99 within the variable test1.

   You can also use the IF-THEN commands to recode any variable. For example, if you wanted to collapse the test scores into a dichotomous variable with students who scored 90 points receiving a score of one and the rest being assigned a score of zero, the syntax would be:

```
IF test1 >=90 THEN test1=1;
ELSE test1=0;
```

   There are a number of SAS operators which could be used in a DATA step, e.g. ** (raise to a power), * (multiplication), / (division), + (addition), - (subtraction), = or EQ (equal to), >= or GE (greater than or equal to), AND, OR, NOT.

   In the DATA step you can also use a number of SAS functions to transform existing variables or create new variables. There are too many different SAS function to list them all here, but some of the most commonly used ones are: MEAN (computes arithmetic mean), SUM (calculates sum of arguments), VAR (calculates the variance), ABS (returns absolute value), SIN (calculates sine), LOG (produces the natural logarithm), SQRT (calculates the square root).

   For instance, to create a new variable called final with the arithmetic mean (average) of the 3 scores, you would type:

```
final=MEAN(test1,test2,test3);
```

   For further details refer to SAS Language: Reference Version 6, and SAS Language and Procedures: Usage Version 6.

6. LABEL Statement

   You can use LABEL statements either in the DATA step or in the PROC step to give labels to variables.

```
LABEL variable='variable label';
```

Replace variable with the name of the variable, and variable label with the label you want to assign to the variable. A SAS variable is limited to 8 characters, whereas a label assigned to a SAS variable can have up to 40 characters including blanks. Labels should be enclosed in quotes, and the LABEL step is terminated by a semicolon. For example,

```
LABEL exp='years of computer experience';
LABEL mathscor='score in mathematics';
```

7. PROC FORMAT

   FORMAT statements associate formats with variables in a DATA step. For example, in a data set, the variable SEX has two values (f,m) and the variable SCHOOL has 3 values (1,2,3). To associate these values with appropriate value labels use the format statement. The FORMAT statement may be used in a DATA step or in a PROC step. However, when you define format with PROC FORMAT it appears as the very first line in a SAS program. Note that character variable values are entered in single quotes.

   ```
   PROC FORMAT;
   value $sex 'm'='male'  'f'='female';
   value school 1='rural' 2='city' 3='suburban';
   ```

   Once you defined the format as above, you should associate the format with the variables through a format statement after the INPUT line.

   ```
   PROC FORMAT;
   value $sex 'm'='male'  'f'='female';
   value school 1='rural' 2='city' 3='suburban';
   DATA anxiety;
   INPUT id 1-3 sex $ 4 school 5 test 6-7;
   FORMAT sex $sex. school school.;
   ```

8. RUN Statement

   To execute any series of statements that read or transform data in any way, you must also include a RUN statement to execute those commands unless they are followed by a PROC step (explained below). For example:

   ```
   DATA anxiety;
   INPUT id 1-3 sex $ 4 school 5 test 6-7;
   FORMAT sex $sex. school school.;
   RUN;
   ```

9. Comment Statements

   Comments are provided for documentation purposes. Statements enclosed in /* .... */ or *.....; are ignored by SAS and will not be used while executing the program.

B. PROC Step

The next step is to create PROC (procedure) steps. SAS procedures read the SAS data and perform various computations and print the results of these computations. The FREQ procedure computes the frequencies on specified variables; the T-TEST procedure performs a t-test analysis on the specified variables. In short, the statements that ask SAS to process or analyze a specified data set are known as PROC steps. The DATA steps and PROC steps can be used in any order within a SAS program. As the DATA step starts with a DATA statement the PROC step starts with a PROC statement.

```
PROC PRINT DATA=dataname;
VAR var1 var2;
```

This procedure requests SAS to print data values for variables 1 and 2. If the VAR statement is omitted data values for each variable in the data set will be printed. DATA=*dataname* is an optional statement. If this is omitted SAS selects the most recently created data set within the program. It is a good practice to specify the dataname along with the PROC statement. Replace the *dataname* with the name of the data created in the DATA step. Printing out a few variables before doing any analysis is a good way to check whether the data are being read by SAS as you want them to be read.

1. Frequencies procedure

```
PROC FREQ;
TABLES var1 var2 var1*var2;
```

This statement produces tables showing distribution of variable values. In the above example SAS will display variable values for var1 and var2, and the combined frequency distributions for var1

and var2. For example, if you wanted to get the gender breakdown for the test scores discussed previously, you would use the following command:

```
PROC FREQ;
TABLES sex;
```

The output generated by this command would look like this:

```
Cumulative  Cumulative
SEX   Frequency   Percent   Frequency    Percent
-------------------------------------------------
f            5      50.0           5       50.0
m            5      50.0          10      100.0
```

2. Descriptive Statistics Procedure

```
PROC MEANS;
VAR var1 var2;
```

This statement computes descriptive statistics for specified variables. If the VAR statement is omitted, descriptive statistics for each variable in the data set will be calculated.

Again, referring back to the grades data, if you wanted to generate some basic descriptive statistics for the students' test scores, the commands and output would look like this:

```
PROC MEANS;
VAR test1 test2 test3;
```

```
Variable   N       Mean      Std Dev      Minimum      Maximum
-------------------------------------------------------------
TEST1     10  80.1000000   10.4504120   60.0000000   90.0000000
TEST2     10  82.6000000    8.4616783   72.0000000   96.0000000
TEST3     10  81.2000000   10.6854002   64.0000000   93.0000000
-------------------------------------------------------------
```

3. Correlation Procedure

```
PROC CORR;
VAR var1 var2;
```

A correlation analysis is performed to quantify the strength of association between two numeric variables. For example, if you wanted to see if there were a correlation between student test scores, the commands and output would look like this:

```
PROC CORR;
VAR test1 test2 test3;

Correlation Analysis

Pearson Correlation Coefficients / Prob > |R| under Ho: Rho=0
/ N = 10

TEST1             TEST2             TEST3

TEST1           1.00000           0.75692           0.91522
0.0               0.0113            0.0002

TEST2           0.75692           1.00000           0.86857
0.0113            0.0               0.0011

TEST3           0.91522           0.86857           1.00000
0.0002            0.0011            0.0
```

4. ENDSAS statement

   The last statement in a SAS program is ENDSAS. By this statement you indicate that there are no more data steps or procedure steps to be read or processed. ENDSAS should be followed by a semicolon (e.g., ENDSAS;). However, an ENDSAS statement during a SAS for Windows session will exit the SAS session and take you to the Microsoft Windows environment.

7 Writing a SAS Program

Now that we have looked at various steps in creating a SAS program, the next step is to write one. A SAS statement may begin in any column on a line. However, for the sake of clarity, lines starting with DATA or PROC statements begin in column 1; all other statements are indented. In the following example SAS is asked to create a data set named grades after reading the data steps. The data are embedded within the SAS program. You may use a DOS editor or Windows editor to create the following program, grade1.sas, and save it to a disk.

The ENDSAS statement is omitted from all the following examples as we don't want to quit the SAS session after executing the command lines. At any time, to quit a SAS session, select File/Exit SAS... from the SAS AWS window (or click right mouse button to activate the pop-up menu and select File/Exit).

A SAS program to read the data, and execute some basic descriptive statistics, in a simple form would be:

```
DATA grades;
INPUT id 1-2 sex $ 3 test1 4-5 test2 6-7 test3 8-9;
CARDS;
01f838591
02f657268
03f909490
04f878082
05f788680
;
RUN;

PROC PRINT;
VAR sex test1 test2 test3;
PROC FREQ DATA=grades;
TABLES sex test1 test2 test3;
```

In this case the program will read the variable ID from columns 1-2 and SEX from column 4, and test1 from column 6-7, test2 from column 9-10, and test3 from column 12-13. The procedure PRINT will print out the variables specified with the VAR (abbreviation for VARIABLES) statement.

If your data contain at least one blank space after each variable value, you can skip the column specifications in the INPUT statement.

```
DATA grades;
INPUT id sex $ test1 test2 test3;
CARDS;
01 f 83 85 91
02 f 65 72 68
03 f 90 94 90
04 f 87 80 82
05 f 78 86 80
;
< other command lines >
RUN;
```

This type of format is called free format. Missing values cannot be left blank in this type of data input as the SAS System will fill that blank with the succeeding variable value. Always assign some value to missing values when you use free format. With fixed format, you may leave the missing values blank.

Suppose you decided that you want to keep your data file as an external file, then make the following changes to the above program:

```
DATA grades;
INFILE 'a:\grade.dat';
INPUT id 1-2 sex $ 4 test1 6-7 test2 9-10 test3 12-13;
RUN;
PROC PRINT;
VAR sex test1 test2 test3;
PROC FREQ DATA=grades;
TABLES sex test1 test2 test3;
```

Replace a:\grade.dat with your own parameter. The INFILE command points out the location of the data file during program execution. Note that the CARDS statement followed by the data lines are not needed anymore in the above example because the data file is stored as an external file.

8 Executing a SAS Program

Suppose that you saved the above program into a file, grade1.sas, on your A drive. If you have saved your data file, grade.dat, as an external file let us assume that file is also on drive A.

9 Launching SAS

The first task is to access SAS for Windows. If SAS for Windows is installed on your computer, start the Windows session and click the SAS for Windows icon. If you are accessing SAS for Windows from a computer in one of the Clemson University student Computer Labs:

1. Find an available workstation.
2. From the Main Menu select Windows Applications.
3. Select Y for settings.
4. Place your data disk in drive A (It may take a couple of minutes to get the software applications loaded). Once the applications software is loaded:
5. Double click SAS the for Windows icon.

10 Submitting Commands

This opens the SAS AWS window as described in the earlier section of this document. To retrieve the command file, grade1.sas, you created and saved earlier from the SAS AWS window:

1. select File/Open/Read File

   Note: If you haven't already created and saved the command file you may move the cursor to the PROGRAM EDITOR window and type in the lines. A dialog box appears which enables you to make an appropriate file selection. Once the file is selected click OK. The file will be opened into the PROGRAM EDITOR window. This task can also be accomplished by opening the pop-up menu (click the right mouse button while the cursor is in the PROGRAM EDITOR window) and selecting File/Open/Read file ...

   Next, tell SAS which commands you wish to execute. In SAS you can run an entire program file at once or just portions of it. Select the commands you want by either clicking and dragging over those commands, or select the entire program file by:

2. select Edit/Select All in the Program Editor

   To run the program either open the pop-up menu in the PROGRAM EDITOR window or go to the main menu and:

3. select Locals/Submit

   The program will run and LOG and OUTPUT windows will appear (if there are errors no OUTPUT window will appear).

   If there are errors, return to the PROGRAM EDITOR (select Windows/Program Editor or click anywhere in the PROGRAM EDITOR window), edit the appropriate commands, and resubmit the job.

   Note that you do not have to select the text before you use the submit command, but SAS will clear the PROGRAM EDITOR automatically after the commands are executed. If you wish to retreive a command file after you have submitted it, you can select Locals/Recall text from the main or pop-up menu to retrieve the last commands executed.

If you are entering the command line from the PROGRAM EDITOR window you may execute each command line, instead of waiting to enter the whole program, as described above. You can switch windows using the SAS AWS Windows pull-down menu and making appropriate selections.

You may print the contents of the LOG and OUTPUT window by selecting File/Print from the SAS AWS window menu. You may also save the contents in any of the windows by selecting File/Save from the pop-up menu. If you are in Command mode use the file statement (e..g, file a:\ test.out) to write the contents of the windows to a disk.

11 Sample Program and Data Files

So far we have looked at the SAS System to develop a basic idea of how SAS for Windows works. The next step is to examine a few other data analysis techniques that you might employ for your own data analysis. All the statistical procedures available from the SAS System under other operating systems are also available from SAS for Windows. Refer to the SAS documentation for further information.

The data set we discussed in our earlier example was designed to get you started. A more sophisticated data set and command file with statistical examples is available on the World Wide Web for you to download, peruse, and execute. The program file has many examples of the various procedures and commands discussed previously and you should probably read through this file closely to make sure you understand the syntax. Comments are provided throughout the program file which includes some statistical examples to demonstrate how these procedures work.

12 Obtaining Sample Data and Command Files

If you are interested in obtaining a copy of this data file you may copy it from the Mathematical Sciences 405 web home page's sas directory. (http://math.clemson.edu/∼calvinw/MthSc405/sas).

To obtain a copy of the sample files:

1. Launch a web browser (e.g. Netscape or Lynx)
2. Go to the URL: http://math.clemson.edu/∼calvinw/MthSc405/sas
3. Save this as a text file (e.g., in Netscape go to File->Save as.. and change Save As.. File Type to text) to a floppy disk. Use clas1.sas for the filename.
4. Go to the URL: http://math.clemson.edu/∼calvinw/MthSc405/sas. This time use clas1.dat for the filename. Contact a consultant if you need assistance.

To run the sample program file:

1. Launch SAS for Windows
2. Retrieve the program file by selecting File/Open/Read File
3. Check to make sure the INFILE command corresponds to the location and name of your data file. For example, if the data file is called clas1.dat and it is saved on a floppy disk in the A: drive, the INFILE command should be: INFILE 'a:\ clas1.dat';
4. Highlight the text by selecting Edit/Select all
5. Submit the command file by selecting Local/Submit

Once you submit the commands, SAS takes you to the output window where you can review your results. You can scroll through the output by either using the PgUp or PgDn keys or the scroll bar. To print the results, click the printer icon on the toolbar, make sure the contents are set to OUTPUT, and click OK.

13 SAS Data Sets

13.1 Creating a SAS data set

If you are handling a large data file (large numbers of cases/variables) it is advisable to create a SAS data set and work with that. A SAS data set is a specially structured files readable only by the SAS System from the operating system where the file was created. A stored SAS file cannot be edited. SAS data sets are referenced with a one- or two-level name. The two-level name is of the form libref.member-name, where libref refers to the directory in which the data set is to be stored or read, and member-name refers to the name of the SAS data file to be created or read. If a two-level name is not used, SAS stores files in a temporary work library that is deleted when you exit SAS.

Suppose you wished to read in an external file, and store the data in a permanent SAS data set. The program file you would create would look something like the one below, which is a subset of the sample data and program files described previously. Note the first two lines in the following program where the libname command is issued to reference the directory in which the SAS file is to be created, and the two-level name links the member-name (e.g., anxiety) to the libref (e.g., try1).

```
LIBNAME try1 'A:';
DATA try1.anxiety;
INFILE 'a:\clas1.dat';
INPUT ID 1-2 SEX $ 3  EXP 4 SCHOOL 5
(C1-C10) (1.) (M1-M10) (1.) (MATHSCOR COMPSCOR) (2.);
IF MATHSCOR=99 THEN  MATHSCOR=.;
IF COMPSCOR=99 THEN  COMPSCOR=.;
C3=6-C3; C5=6-C5; C6=6-C6; C10=6-C10;
M3=6-M3; M7=6-M7; M8=6-M8; M9=6-M9;
COMPOPI = SUM (OF C1-C10);
MATHATTI = M1+M2+M3+M4+M5+M6+M7+M8+M9+M10;
LABEL ID='STUDENT IDENTIFICATION' SEX='STUDENT GENDER'
EXP='YRS OF COMP EXPERIENCE' SCHOOL='SCHOOL REPRESENTING'
MATHSCOR='SCORE IN MATHEMATICS'
COMPSCOR='SCORE IN COMPUTER SCIENCE'
COMPOPI='TOTAL FOR COMP SURVEY'
MATHATTI='TOTAL FOR MATHATTI SCALE';
RUN;
```

Replace a: with the appropriate directory. When the job is executed a SAS data set named anxiety.ssd will be stored in the directory assigned.

13.2 Accessing a SAS data set

To read an existing SAS data set, use a two-level name of the form libref.member.name. The following example illustrates how to access the SAS data set (e.g., anxiety.ssd) created and run some SAS procedures with it. (Note that try2 is given as libref and anxiety is given as member-name. The SET command used below reads the SAS data set called try2.anxiety into the data area called test.

```
LIBNAME try2 'A:';
DATA TEST;
SET try2.anxiety;
PROC TTEST;
CLASS SEX;
VAR COMPOPI;
TITLE 'T-TEST';
PROC CORR DATA=ANXIETY2;
VAR COMPSCOR MATHSCOR COMPOPI MATHATTI;
```

It is also possible to retrieve only a subset of the original data using an IF statement. For example, if you wanted to retrieve only the female respondents from the anxiety.ssd file, the DATA step of your program would look something like this:

```
LIBNAME try2 'A:';
DATA TEST;
SET try2.anxiety;
IF sex='f';
RUN;
```

The IF statement in this example tells SAS to only read in those observations where the character variable SEX is equal to "f." All other cases will be ignored. Several conditions can be set with an IF statement. For more information, see SAS Language, and SAS Companion for the Microsoft Windows Environment.

13.3 SAS transport libraries

SAS also handles transport format data sets. You create a transport format file when you want to move your SAS data set to another operating system (e.g., UNIX). Also, if you are bringing SAS data sets from another operating system into SAS for Windows, you must first save the file in transport format.

Suppose you want to create a SAS transport format file from the SAS data file, anxiety.ssd. Define a libname to read the SAS data set, and another libname to write a SAS transport file. The XPORT (for transport engine) parameter is used to indicate that you want to create a transport format file. A transport format file always has a fixed block size with a record length of 80, and a block size of 8000. The select statement may be omitted if there is only one SAS file stored in the directory or if you want to convert all the members of a single SAS data library into SAS transport format library.

```
LIBNAME test1 'A:';
LIBNAME test2 XPORT 'A:\trans.dat';
PROC COPY IN=test1 OUT=test2;
SELECT anxiety;
RUN;
```

Once the job is executed, a file, trans.dat, will be created and stored in the directory specified.

To read a transport format file (e.g., trans.dat) stored on the disk and create a SAS data file, as in the example given above, define two libnames (one for reading, and one for writing) as in:

```
LIBNAME test1 'A:';
LIBNAME test2 XPORT 'A:\trans.dat';
PROC COPY IN=test2 OUT=test1;
SELECT anxiety;
RUN;
```

The select statement may be dropped if you want to convert all the members of the transport library into a SAS file, or if there is only one member in the specified transport library. If you want to read/write SAS transport files involving format library use the CIMPORT/CPORT procedures instead of the COPY procedure. See SAS Language Reference V.6, and SAS Procedures Guide V.6 for further information.

14 Other Sample Files

A number of sample files are available for each of the SAS add-on modules. You can also copy the files to your disk. If you are accessing SAS from personal workstations, you will find the sample files are found with each add-on module subdirectory from the main SAS directory (typically C:\SAS).

15 Using SAS/GRAPH

SAS has excellent graphics capabilities. The manual, SAS/Graph software: Introduction, is a good place to start. To make a hard copy of the graphics output you may create Postscript file and plot it using the laser printers at various computer laboratories at Clemson. You may send your SAS/Graph output from a SAS session directly to a laser printer, or display it on the terminal.

To create a postscript file, type the command lines:

```
FILENAME stat2 'A:\graph1.ps';
GOPTIONS DEV=LJ3PSIPS GSFNAME=stat2 GSFLEN=132 GSFMODE=REPLACE
NODISPLAY;
```

Replace the parameters in lowercase with your own parameters. To display your SAS Graph output on a terminal during a SAS for Windows session, use the WIN driver specification as in:

```
GOPTIONS DEV=WIN;
```

To send SAS/GRAPH output to a laser printer, during a SAS for Windows session, in the computer laboratories at Clemson use the syntax:

```
FILENAME stat2 'A:\graph1.ps';
GOPTIONS DEV=LJ3PSIPS GACCESS='SASGASTD>LPT1:' GSFMODE=PORT;
```

16 Relative Frequency, Distributions, Descriptive Statistics, Correlations, and Plots

A preliminary analysis of the data may include examining relative frequency distributions and descriptive statistics (e.g., mean, median, and standard deviation) for both the dependent and independent variables, correlations between pairs of variables, and exploratory plots of the dependent variable against each of the independent variables as we will see in subsequent chapters.

16.1 Clasprog2.sas

The following programs gives the necessary commands for exercises 2.6 and 2.7 of Bowerman and O'Connell, your text book.

```
DATA VISCOSE;
INPUT VICOSITY @;  /* Give the variable a name */
 /* The following is a sample of "viscosity" readings for a chemical process:*/
CARDS;
1.93 1.96 1.98 1.94 1.99 1.98 2.01 2.01
1.94 1.97 1.98 2.02 2.00 1.98 2.01 1.96
1.97 1.99 2.00 2.02 1.99 1.95 2.02 1.99
2.00 2.01 2.00 2.02 1.99 1.96 2.01 1.99
1.95 1.99 2.02 2.02 1.98 1.98 2.01 2.04
1.99 1.95 2.02
1.99 1.96 2.01
1.98 1.98 2.01
;
PROC PRINT;    /* Prints out the data */
RUN;
PROC UNIVARIATE PLOT NORMAL;  /* Give the descriptive statistics stem */
VAR  VICOSITY;                /* and leaf plot and normal probability plot for */
RUN;                          /* the data */
PROC CHART;
VBAR VICOSITY/TYPE=PERCENT; /*Histogram of data*/
RUN;                          /* the data */
```

16.2 Clasprog3.sas

The following is a sample of "bag weights" (in pounds) for an industrial bagging operation from exercise 2.7.

```
DATA BAGS;
INPUT WEIGHTS @;  /* Give the variable a name */
CARDS;
50.6 49.8
50.6 51.4
50.8 50.8
50.8 50.6
50.8 50.6
49.8 50.8
50.8 50.5
50.4 50.3
50.6 50.8
50.7 50.6
49.1 51.2
49.0 51.1
50.2 50.4
49.9 50.1
52.2 50.7
50.3 49.8
```

```
50.2 52.0
46.8 50.5
;
PROC PRINT;    /* Prints out the data */
RUN;
PROC UNIVARIATE PLOT NORMAL;  /* Give the descriptive statistics stem */
VAR  WEIGHTS;                /* and leaf plot and normal probability plot for */
RUN;                          /* the data */
PROC CHART;
VBAR WEIGHTS/TYPE=PERCENT; /*Histogram of data*/
RUN;                          /* the data */
```

The UNIVARIATE procedure generates descriptive statistics for the quantitative variables specified in the VAR statement.

The CHART procedure is used to generate relative frequency and frequency distributions (histograms) for quantitative data. The key word VBAR followed by the variable name VISCOST produces a frequency histogram for the viscosity measurements.

Relative frequency histograms are produced by adding the option TYPE=PERCENT. SAS will automatically select suitable class intervals for the histogram.

17 Simple Linear Regression

The SAS commands and instructions given in Chapter 5 generate the simple linear regression printouts for the Comp-U-Systems service call data of Section .

17.1 Clasprog4.sas

SAS program for a simple linear regression analysis of the Comp-U-Systems service call data.

```
DATA COMP;        /*Assigns name COMP to file */
INPUT Y X;    /*Assigns variable names. */
LABEL Y = 'service time'
X = 'number of microcomputers';
/*Y = service time; */
/*X = number of microcomputers*/
/* Comp-U-Systems data. see Table 5.6*/
CARDS;
37 1
58 2
68 2
82 3
103 4
109 4
112 4
134 5
138 5
154 6
189 7
.   4
;
/* Decimal point denotes missing value */
/*Used to obtain point prediction when x0 = 4 */
PROC PRINT;    /* Print procedure prints the data*/
RUN;
PROC REG DATA = COMP;     /*Specifies regression procedure */
MODEL Y = X/P CLM CLI;       /* Specifies model yi = b0 + b1xi + ei */
RUN;
/* Note. P = Point predictions desired */
/* CLM = 95% confidence intervals desired */
/* CLI = 95% prediction intervals desired */
/* To change to 90% use ALPHA= 0.1 */
```

The REG procedure performs a complete simple linear regression analysis on the data.

In the MODEL statement, the dependent variable is listed to the left of the equals sign and the independent variable to the right. The option P (following the slash) prints predicted values and residuals, and the option CLI prints corresponding lower and upper 95% prediction limits for all observations in the data set. Specify CLM to obtain 95% confidence intervals for E(y). [Note: To predict y for a value of x that is not included in the data set (e.g., $x = 4$), you must include an "extra" observation in the data set. This observation has the specified value of x (e.g., 4), but a missing value for y (i.e., a single decimal point).]

The optional ID statement identifies the value of x for each 95% prediction (or confidence) interval.

17.2 Clasprog5.sas

SAS program for a regression through the origin of the naval data in Table 5.8.

```
DATA HOURS;   /*Assigns name HOURS to data file*/
```

```
INPUT OBS X Y;   /*Assigns variable names.*/
LABEL Y = 'labor hours'
      X = 'items processed';
CARDS;
1        15        85
2        25       125
3        57       203
4        67       293
5       197       763
6       166       639
7       162       673
8       131       499
9       158       657
10       241       939
11       399      1546
12       527      2158
13       533      2182
14       563      2302
15       563      2202
16       932      3678
17       986      3894
18      1021      4034
19      1643      6622
20      1985      7890
21      1640      6610
22      2143      8522
23       500     .
;
/* Decimal point denotes missing value */
/* Used to obtain point prediction when x0 = 500 */
PROC PRINT;             /* Print procedure. prints the data */
RUN;
PROC REG DATA = HOURS;    /* Specifies regression procedure*/
MODEL Y = X/NOINT P CLM CLI;  /* NOINT omits the intercept in the usual model */
RUN;
/* and thus specifies the model*/
/* Yi = b1xi + ei */
/* Note: P = Point predictions desired */
/*  CLM = 95% confidence intervals desired */
/*  CLI = 95% prediction intervals desired  */
```

18 Residual Analysis and Regression Diagnostics

18.1 Clasprog6.sas

SAS program to obtain residual plots and other diagnostics for the Comp-U-Systems model

$$y = \beta_0 + \beta_1 x_i + \epsilon i.$$

```
DATA COMP;     /* Assigns name COMP to data file */
INPUT Y X;    /*Assigns variable names. */
LABEL Y = 'service time'
X = 'number of microcomputers';
/*Y = service time; */
/*X = number of microcomputers*/
TIME=_N_;   /*  Defines variable TIME to be observation number */
/* Comp-U-Systems data. see Table 5.6*/
CARDS;
37 1
58 2
68 2
82 3
103 4
109 4
112 4
134 5
138 5
154 6
189 7
;
PROC PRINT;    /* Print procedure: prints the data*/
RUN;
PROC REG DATA=COMP;    /* Specifies regression procedure*/
MODEL Y = X/P DW;        /* Durbin-Watson statistic*/
OUTPUT OUT=ONE PREDICTED=YHAT RESIDUAL=RESID;   /* y = b0 + b1x + e  against*/
RUN;
/* define symbol characteristics */
symbol
interpol=NONE  /* regression analysis with */
value=diamond /* plot symbol   */
height=3      /* plot symbol height  */
cv=red        /* plot symbol color  */
ci=blue       /* regression line color */
co=green      /* confidence limits color */
width=2;      /* line width   */
/* produce plot */
PROC GPLOT DATA=ONE;               /* Plot residuals from model*/
/* x, yhat, and time. DW requests*/
PLOT RESID*(X YHAT TIME) / vref=0.0;
RUN;
PROC CHART;
VBAR RESID/TYPE=PERCENT; /*Histogram of residuals*/
PROC UNIVARIATE PLOT DATA=ONE;         /* Requests stem-and-leaf diagram, box plot*/
VAR RESID;                             /* and normal plot of residuals for model*/
RUN;                                   /* y = b0 + b1x + e */
```

19 Multiple Regression

19.1 Clasprog8.sas

SAS program to carry out a regression analysis of the Fresh demand data.

```
DATA DETR;  /* Assigns the name DETR to the data file */
INPUT OBS X1 X2 X4 X3 Y; /* Defines variable names, */

LABEL X1 = 'Price'
X2 = 'Average Price'
X4 = 'price difference'
X3 = 'advertising expenditure'
Y  = 'demand';
X3SQ = X3*X3;          /* Transformation defines the squared term X3^2*/
/* and assigns the name X3SQ to this variable*/
CARDS;
1 3.85 3.8 -0.05 5.5 7.38
2 3.75 4 0.25 6.75 8.51
3 3.7 4.3 0.6 7.25 9.52
4 3.7 3.7 0 5.5 7.5
5 3.6 3.85 0.25 7 9.33
6 3.6 3.8 0.2 6.5 8.28
7 3.6 3.75 0.15 6.75 8.75
8 3.8 3.85 0.05 5.25 7.87
9 3.8 3.65 -0.15 5.25 7.1
10 3.85 4 0.15 6 8
11 3.9 4.1 0.2 6.5 7.89
12 3.9 4 0.1 6.25 8.15
13 3.7 4.1 0.4 7 9.1
14 3.75 4.2 0.45 6.9 8.86
15 3.75 4.1 0.35 6.8 8.9
16 3.8 4.1 0.3 6.8 8.87
17 3.7 4.2 0.5 7.1 9.26
18 3.8 4.3 0.5 7 9
19 3.7 4.1 0.4 6.8 8.75
20 3.8 3.75 -0.05 6.5 7.95
21 3.8 3.75 -0.05 6.25 7.65
22 3.75 3.65 -0.1 6 7.27
23 3.7 3.9 0.2 6.5 8
24 3.55 3.65 0.1 7 8.5
25 3.6 4.1 0.5 6.8 8.75
26 3.65 4.25 0.6 6.8 9.21
27 3.7 3.65 -0.05 6.5 8.27
28 3.75 3.75 0 5.75 7.67
29 3.8 3.85 0.05 5.8 7.93
30 3.7 4.25 0.55 6.8 9.26
31 . .   10 6.80  .
;
/* Decimal point represents a missing */
/* value. Used to obtain point prediction*/
/* when X03 = 6.80 and X04 = 10*/

PROC PRINT; /* Prints the data*/
RUN;
PROC REG DATA = DETR; /* Specifies regression procedure*/
```

```
MODEL Y = X4 X3 X3SQ / P CLM CLI;
RUN;
/* Specifies model */
/* Y=b0+b1X4 + b2X3 + b3X3ˆ2 + e */
/* (intercept b0 is assumed) */
/* P = Predictions desired */
/* CLM = 95% confidence interval desired */
/* CLI = 95% prediction interval desired */

PROC REG DATA = DETR;
MODEL Y = X4 X3 X3SQ/P;
OUTPUT OUT = ONE PREDICTED = YHAT RESIDUAL = RESID; /* Plots residuals from*/
RUN;
/* define symbol characteristics */
symbol
interpol=NONE  /* regression analysis with */
value=diamond /* plot symbol   */
height=3      /* plot symbol height  */
cv=red        /* plot symbol color  */
ci=blue       /* regression line color */
co=green      /* confidence limits color */
width=2;      /* line width   */
/* produce plot */
PROC GPLOT DATA=ONE;               /* Plot residuals from model*/
PLOT RESID * (X4 X3 YHAT)/vref = 0.0;         /* x, yhat, and time. DW requests*/
/*  against X4, X3, and yhat */

PROC UNIVARIATE PLOT DATA = ONE; /* Produces stem-and-leaf display, box plot,*/
VAR RESID;                        /* and normal plot of residuals from the */
RUN;                             /* above model */
```

19.2 Clasprog10.sas

SAS program for a regression of the Fresh demand Data using model:

$$E(y) = \beta_0 + \beta_1 x_4 + \beta_2 x_3 + \beta_3 x_3^2 + \beta_4 x_3 x_4 + \epsilon.$$

```
DATA DETR;  /* Assigns the name DETR to the data file */
INPUT OBS X1 X2 X4 X3 Y; /* Defines variable names, */

LABEL X1 = 'Price'
X2 = 'Average Price'
X4 = 'price difference'
X3 = 'advertising expenditure'
Y  = 'demand';
X43  = X4*X3;
X3SQ = X3*X3;          /* Transformation defines the squared term X3ˆ2*/
/* and assigns the name X3SQ to this variable*/

CARDS;
1 3.85 3.8 -0.05 5.5 7.38
2 3.75 4 0.25 6.75 8.51
3 3.7 4.3 0.6 7.25 9.52
4 3.7 3.7 0 5.5 7.5
5 3.6 3.85 0.25 7 9.33
6 3.6 3.8 0.2 6.5 8.28
7 3.6 3.75 0.15 6.75 8.75
8 3.8 3.85 0.05 5.25 7.87
9 3.8 3.65 -0.15 5.25 7.1
```

```
10 3.85 4 0.15 6 8
11 3.9 4.1 0.2 6.5 7.89
12 3.9 4 0.1 6.25 8.15
13 3.7 4.1 0.4 7 9.1
14 3.75 4.2 0.45 6.9 8.86
15 3.75 4.1 0.35 6.8 8.9
16 3.8 4.1 0.3 6.8 8.87
17 3.7 4.2 0.5 7.1 9.26
18 3.8 4.3 0.5 7 9
19 3.7 4.1 0.4 6.8 8.75
20 3.8 3.75 -0.05 6.5 7.95
21 3.8 3.75 -0.05 6.25 7.65
22 3.75 3.65 -0.1 6 7.27
23 3.7 3.9 0.2 6.5 8
24 3.55 3.65 0.1 7 8.5
25 3.6 4.1 0.5 6.8 8.75
26 3.65 4.25 0.6 6.8 9.21
27 3.7 3.65 -0.05 6.5 8.27
28 3.75 3.75 0 5.75 7.67
29 3.8 3.85 0.05 5.8 7.93
30 3.7 4.25 0.55 6.8 9.26
31 . .   10 6.80  .
;
/* Decimal point represents a missing */
/* value. Used to obtain point prediction*/
/* when X03 = 6.80 and X04 = 10*/

PROC PRINT; /* Prints the data*/
RUN;
PROC REG DATA = DETR; /* Specifies regression procedure*/
MODEL Y = X4 X3 X3SQ X43 / P XPX I CLM CLI;
T1: TEST X3SQ=0, X43=0; /* Performs partial F-test*/
/* of Ho: b3 = b4 = 0 in above model
RUN;
/* Specifies model */
/* Y=b0+b1X4 + b2X3 + b3X3^2 + b4x4x3 + e */
/* (intercept b0 is assumed) */
/* P = Predictions desired */
/* CLM = 95% confidence interval desired */
/* CLI = 95% prediction interval desired */

PROC REG DATA = DETR;
MODEL Y = X4 X3 X3SQ X43/P;
OUTPUT OUT = ONE PREDICTED = YHAT RESIDUAL = RESID; /* Plots residuals from*/
RUN;
/* define symbol characteristics */
symbol
interpol=NONE  /* regression analysis with */
value=diamond /* plot symbol   */
height=3      /* plot symbol height  */
cv=red        /* plot symbol color  */
ci=blue       /* regression line color */
co=green      /* confidence limits color */
width=2;      /* line width   */
/* produce plot */
PROC GPLOT DATA=ONE;               /* Plot residuals from model*/
/* x, yhat, and time. DW requests*/
PLOT RESID * (X4 X3 YHAT)/vref=0.0;    /*  against X4, X3, and yhat */
```

```
PROC UNIVARIATE PLOT DATA = ONE;
VAR RESID;                    /* Produces stem-and-leaf display, box plot,*/
/* and normal plot of residuals from the */
/* above model */
```

19.3 Clasprog11.sas

SAS program to obtain the correlation matrix, variance inflation factors, and partial leverage residual plots in the hospital labor needs problem.

```
DATA HOSP;
INPUT  X1 X2 X3 X4 X5 Y; /* Defines variables */
CARDS;
15.57 2463 472.92 18.0  4.45 566.52
44.02 2048 1339.75 9.5  6.92 696.82
20.42 3940 620.25 12.8  4.28 1033.15
18.74 6505 568.33 36.7  3.90 1603.62
49.20 5723 1497.60 35.7  5.50 1611.37
44.92 11520 1365.83 24.0  4.60 1613.27
55.48 5779 1687.00 43.3  5.62 1854.17
59.28 5969 1639.92 46.7  5.15 2160.55
94.39 8461 2872.33 78.7  6.18 2305.58
128.02 20106 3655.08 180.5  6.15 3503.93
96.00 13313 2912.00 60.9  5.88 3571.89
131.42 10771 3921.00 103.7  4.88 3741.40
127.21 15543 3865.67 126.8  5.50 4026.52
252.90 36194 7684.10 157.7  7.00 10343.81
409.20 34703 12446.33 169.4 10.78 11732.17
463.70 39204 14098.40 331.4  7.05 15414.94
510.22 86533 15524.00 371.6  6.35 18854.45

PROC PRINT;
RUN;
/* define symbol characteristics */
symbol
interpol=NONE  /* regression analysis with */
value=diamond /* plot symbol   */
height=3      /* plot symbol height  */
cv=red        /* plot symbol color  */
ci=blue       /* regression line color */
co=green      /* confidence limits color */
width=2;      /* line width   */
/* produce plot */
PROC GPLOT DATA=HOSP;
PLOT Y * (X1 X2 X3 X4 X5); /* Plots y against X1, X2, X3, X4, and X5. */
RUN;
PROC CORR;  /* Prints correlation matrix */
RUN;
PROC REG;
MODEL Y =  X1 X2 X3 X4 X5/PARTIAL; /*Gives partial leverage residual plots */
RUN;
/* VIF calculates variance*/
PROC REG;                          /*inflation factors for model*/
MODEL Y = X2 X3 X5/P VIF;          /*Y = b0 + b1x2 + b2x3 + b3x5 + e*/
RUN;
```

19.4 Clasprog12.sas

SAS program to compute diagnostics for outlying and influential observations for the model

$$E(y) \quad = \quad \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_2^2 + \epsilon$$

in the Home Real Estate Company problem.

```
DATA HOUSE;      /* Assigns name HOUSE to the data file */
INPUT OBS Y X1 X2;   /*Defines variable names.
LABEL Y = 'selling price'
X1 = 'size'
X2 = 'age';
/* Transformation defines the squared term X22 */
X2SQ = X2 * X2; /* and assigns the name X2SQ to this variable */

CARDS;
1 68.7 2.05 3.43
2 54.9 1.70 11.61
3 51.5 1.47 8.31
4 71.6 1.75 0.00
5 58.4 1.94 7.41
6 40.7 1.19 31.70
7 51.7 1.56 16.10
8 71.9 1.95 2.05
9 57.1 1.60 1.74
10 58.3 1.49 2.76
11 73.5 1.91 0.00
12 58.5 1.38 0.00
13 49.1 1.55 12.61
14 67.5 1.88 2.80
15 53.7 1.60 7.08
16 50.0 1.55 18.00
17 . 1.70 5.0
;

/*  to obtain a point prediction when x0, = 1.70 */
/*  and x02 = 5.00 */

PROC PRINT; /*  Prints the data*/
RUN;
PROC REG;    /*  Specifies regression procedure*/
MODEL Y = X1 X2 X2SQ/P R INFLUENCE CLM CLI;
RUN;

/* Specifies model   P = predictions desired*/
/* Diagnostics for outlying observations*/
/* and influential observations desired*/

/* CLM = 95% confidence interval desired*/
/* CLI = 95% prediction interval desired*/
```

A two-level dummy variable for neighborhood is created using IF, THEN, ELSE statements.

Interactions and squared terms are created using the multiplication operator *.

The REG procedure fits general linear models in addition to simple linear regression models.

In the MODEL statement the dependent variable is listed to the left of the equals sign and the independent variables to the right. The option P (following the slash) prints predicted values and residuals and the option CLI prints corresponding lower and upper 95% prediction limits. Specify CLM to obtain 95% confidence intervals for E(y).

The TEST statement will conduct a nested model partial F test on the model terms specified (e.g. X1X2 and XlSQX2). These terms must be separated by commas. An optional label (INT_TEST) followed by a colon may be specified in front of

the TEST statement. The R option produces a list of residuals predicted values Studentized residuals and Cook's D statistic for all observations in the regression analysis. The INFLUENCE option requests a detailed analysis of the influence of each observation on the β estimates. This includes leverage values and Studentized deleted residuals. The PARTIAL option generates a partial residual plot for each independent variable in the model. The VIF option produces variance inflation factors for the independent variables.

The residuals and predicted values from the regression are saved in it the data set specified after OUT=.

Two plots are produced: residuals versus predicted (9) and residuals versus the quantitative dependent variable DOTEST.

The CHART procedure is used to produce a histogram for the regression residuals.

The PLOT option of the UNIVARIATE procedure produces a stem-and-leaf plot for the specified variable RESID. The NORMAL option produces a normal probability plot for the variable.

20 Stepwise Regression

20.1 Clasprog13.sas

SAS program to implement model building for the hospital labor data.

```
DATA HOSP;
INPUT X1 X2 X3 X4 X5  Y ;   /* Defines variables */
CARDS;
1 15.57 2463 472.92 18.0 4.45 566.52
2 44.02 2048 1339.75 9.5 6.92 696.82
3 20.42 3940 620.25 12.8 4.28 1033.15
4 18.74 6505 568.33 36.7 3.90 1603.62
5 49.20 5723 1497.60 35.7 5.50 1611.37
6 44.92 11520 1365.83 24.0 4.60 1613.27
7 55.48 5779 1687.00 43.3 5.62 1854.17
8 59.28 5969 1639.92 46.7 5.15 2160.55
9 94.39 8461 2872.33 78.7 6.18 2305.58
10 128.02 20106 3655.08 180.5 6.15 3503.93
11 96.00 13313 2912.00 60.9 5.88 3571.89
12 131.42 10771 3921.00 103.7 4.88 3741.40
13 127.21 15543 3865.67 126.8 5.50 4026.52
14 252.90 36194 7684.10 157.7 7.00 10343.81
15 409.20 34703 12446.33 169.4 10.78 11732.17
16 463.70 39204 14098.40 331.4 7.05 15414.94
17 510.22 86.533 15524.00 371.6 6.35 18854.45
;
PROC PRINT;
RUN;
PROC RSQUARE CP;                  /* Calculates R2 and CP for models */
MODEL Y = X1 X2 X3 X4 X5;
RUN;                              /* utilizing all combinations of*/
/* X1 X2 X3 X4 X5 */
/*  PROC GLM is a regression procedure*/
PROC GLM;                         /* similar to (but different from) PROC REG.*/
MODEL Y = X2 X3 X5 / P CLM;       /* PROC GLM along with CLM in*/
RUN;
/* the model statement gives the*/
/* PRESS statistic as part of the output*/
/* Performs stepwise regression*/
PROC STEPWISE;
MODEL Y = X1 X2 X3 X4 X5 / STEPWISE SLE = 0.1 SLS = 0.1;
/* with alpha entry set equal to .1 */
/* to .1 (SLS = 0.1). Default */
/*   (SLE = 0.1) and alpha stay set equal */
/* values for alpha entry and alpha stay are*/
/* 0.5 and 0.15 respectively.*/
/* Performs stepwise regression*/
/* forward selection backward*/

MODEL Y = X1 X2 X3 X4 X5 / STEPWISE; /* elimination and maximum R2 */
FORWARD BACKWARD MAXR;          /* improvement. Default values of a entry and */
                    /* a stay (0.5 and 0.15. respectively) are used here.*/
RUN;
```

The GLM procedure with the /METHOD=STEPWISE option (line 8) performs a stepwise regression on the data.

In the MODEL statement the dependent variable is listed to the left of the equals sign and the independent variables to be analyzed to the right.

Other optional stepwise selection methods include FORWARD (forward selection) BACKWARD (backward selection) and R SQUARE (all-possible regressions selection). The SLE option specifies the significance level α required for a variable to enter into the model. (The default SLE is .15.) The SLS option specifies the significance level α required for a variable to stay in the model. (The default SLS is 0.15).)

20.2 Clasprog14.sas

SAS program to produce the Fresh Detergent dummy variable model output in Figure 12.11

```
DATA DETR;  /* Assigns the name DETR to the data file */
INPUT OBS X1 X2 X4 X3 Y DB DC; /* Defines variable names, */
LABEL X1 = 'Price'
X2 = 'Average Price'
X4 = 'price difference'
X3 = 'advertising expenditure'
Y  = 'demand';
X43  = X4*X3;
X3SQ = X3*X3;          /* Transformation defines the squared term X3^2*/
/* and assigns the name X3SQ to this variable*/
/* DB and DC are names assigned to the dummy */
/*variables for advertising campaigns B and C*/
CARDS;
1 3.85 3.8 -0.05 5.5 7.38 1 0
2 3.75 4 0.25 6.75 8.51 1 0
3 3.7 4.3 0.6 7.25 9.52 1 0
4 3.7 3.7 0 5.5 7.5 0 0
5 3.6 3.85 0.25 7 9.33 0 1
6 3.6 3.8 0.2 6.5 8.28 0 0
7 3.6 3.75 0.15 6.75 8.75 0 1
8 3.8 3.85 0.05 5.25 7.87 0 1
9 3.8 3.65 -0.15 5.25 7.1 1 0
10 3.85 4 0.15 6 8 0 1
11 3.9 4.1 0.2 6.5 7.89 0 0
12 3.9 4 0.1 6.25 8.15 0 1
13 3.7 4.1 0.4 7 9.1 0 1
14 3.75 4.2 0.45 6.9 8.86 0 0
15 3.75 4.1 0.35 6.8 8.9 1 0
16 3.8 4.1 0.3 6.8 8.87 1 0
17 3.7 4.2 0.5 7.1 9.26 1 0
18 3.8 4.3 0.5 7 9 0 0
19 3.7 4.1 0.4 6.8 8.75 1 0
20 3.8 3.75 -0.05 6.5 7.95 1 0
21 3.8 3.75 -0.05 6.25 7.65 0 1
22 3.75 3.65 -0.1 6 7.27 0 0
23 3.7 3.9 0.2 6.5 8 0 0
24 3.55 3.65 0.1 7 8.5 0 0
25 3.6 4.1 0.5 6.8 8.75 0 0
26 3.65 4.25 0.6 6.8 9.21 1 0
27 3.7 3.65 -0.05 6.5 8.27 0 1
28 3.75 3.75 0 5.75 7.67 1 0
29 3.8 3.85 0.05 5.8 7.93 0 1
30 3.7 4.25 0.55 6.8 9.26 0 1
31 . . 10 6.8 . 0 1
;
/*Generates prediction when x04 = 10. x03 = 6.80.*/
/*and advertising campaign C is used*/

PROC GLM DATA = DETR; /*PROC GLM is needed for estimation*/
MODEL Y = X4 X3 X3SQ X43 DB DC/CLI;
```

```
ESTIMATE ’MUDAB-MUDAA’ DB 1; /* Estimates mu[daB] - mu[daA] = b5*/
ESTIMATE ’MUDAC-MUDAA’ DC 1; /* Estimates mu[daC] - mu[daA] = b6*/
ESTIMATE ’MUDAC-MUDAB’ DB -1 DC 1; /*  Estimates mu[daC] - mu[daB] = b6 - b5*/
RUN;
```

21 Transforming to Linearity

21.1 Clasprog15.sas

SAS program to perform a regression analysis of the steakhouse data using the model Ln y = $\alpha_0 + \alpha_1 t + \epsilon_t$.

```
DATA STEAK;
INPUT YEAR T YT LNYT;
CARDS;
1974 0 11 2.398
1975 1 14 2.639
1976 2 16 2.773
1977 3 22 3.091
1978 4 28 3.332
1979 5 36 3.584
1980 6 46 3.829
1981 7 67 4.205
1982 8 82 4.407
1983 9 99 4595
1984 10 119 4.779
1985 11 156 5.050
1986 12 257 5.549
1987 13 284 5.649
1988 14 403 5.999
1989 15  .    .
PROC PRINT;
RUN;
PROC REG ;
MODEL LNYT = T / P CLI ;
RUN;
```

21.2 Clasprog16.sas

```
DATA HARDWARE;
INPUT VALUE UPKEEP;
VALUESQ = VALUE*VALUE;
VALUEINV = 1/(VALUE*VALUE);
TRANSY = UPKEEP/VALUE;
RVALUE = 1/VALUE;
ONE = 1;

CARDS;
1 118.50 706.04
2 76.54 398.60
3 92.43 436.24
4 111.03 501.71
5 80.34 426.45
6 49.84 144.24
7 114.52 644.23
8 50.89 211.54
9 128.93 675.87
10 48.14 189.02
11 85.50 459.04
12 115.51 813.62
13 114.16 602.39
```

```
14 102.95 428.52
15 92.86 387.50
16 84.39 434.63
17 123.53 698.00
18 77.77 355.75
19 112.10 737.59
20 101.02 706.66
21 76.52 424.57
22 116.09 656.92
23 62.72 301.03
24 84.91 321.07
25 88.64 519.40
26 81.41 348.50
27 60.22 162.17
28 95.55 482.55
29 79.39 460.07
30 89.25 475.45
31 136.10 835.16
32 24.45 62.70
33 52.28 239.89
34 143.09 1005.32
35 41.86 184.18
36 43.10 .212.80
37 66.79 313.45
38 106.43 658.47
39 61.01 195.08
40 99.01 545.42
41 110 .
;
/*Predicts upkeep for house value of 110*/

PROC PRINT;
RUN;
/* Produces output in Figure */
/* by fitting transformed model */
PROC REG;
MODEL TRANSY = RVALUE ONE VALUE/NOINT P;
OUTPUT OUT = ONE PREDICTED = YHAT RESIDUAL = RESID;
RUN;
/* define symbol characteristics */
symbol
interpol=NONE  /* regression analysis with */
value=diamond /* plot symbol   */
height=3      /* plot symbol height  */
cv=red        /* plot symbol color  */
ci=blue       /* regression line color */
co=green      /* confidence limits color */
width=2;      /* line width   */
/* produce plot */

PROC GPLOT DATA=ONE;               /* Plot residuals from model*/
PLOT RESID * (VALUE YHAT)/ vref=0.0;         /* value, yhat*/
RUN;

PROC UNIVARIATE PLOT DATA = ONE; /* Produces stem-and-leaf display, box plot,*/
VAR RESID;                        /* and normal plot of residuals from the */
RUN;                             /* above model */
```

```
PROC REG;
MODEL UPKEEP = VALUE VALUESQ/P CLM CLI;
WEIGHT VALUEINV;
RUN;
```

22 Logistic Regression

22.1 Clasprog17.sas

SAS Program to analyze the product name data of Table 13.8 by using logistic regession

```
DATA COMM;
INPUT NRECALL MENTION;
PROP = NRECALL/50;
R = PROP/(1 - PROP);
L = LOG(R);
M = PROP*(1 - PROP);
NM=50*M;
CARDS;
4 1
7 2
20 3
35 4
44 5
46 6
;
PROC PRINT;
RUN;

PROC REG;
MODEL L = MENTION / P CLM;
WEIGHT NM;
RUN;
```

23 Time Series Forecasting Models and Durbin-Watson Test

Several of the forecasting techniques outlined in Chapter 13 can be performed in SAS. The exponential smoothing model, a regression model and a time series autoregressive error model are applied to the quarterly Farm Bureau data load time series of Table 13.9. Moving averages can be computed using SAS and programming commands. (Consult the references at the end of this tutorial.)

23.1 Clasprog18.sas

```
DATA GAS;
INPUT BILL TIME Q1 Q2 Q3;
TIMESQ = TIME*TIME;
CARDS;
1344.39 1 1 0 0
313.45 2 1 0 0
246.84 3 1 0 0
277.01 4 1 0 0
365.1 5 1 0 0
267 6 1 0 0
467.06 7 1 0 0
336.56 8 1 0 0
440 9 1 0 0
434.66 10 1 0 0
246.63 11 0 1 0
189.76 12 0 1 0
209 13 0 1 0
197.98 14 0 1 0
207.51 15 0 1 0
230.28 16 0 1 0
306.03 17 0 1 0
196.67 18 0 1 0
315.04 19 0 1 0
399.66 20 0 1 0
131.53 21 0 0 1
179.1 22 0 0 1
51.21 23 0 0 1
50.68 24 0 0 1
54.63 25 0 0 1
230.32 26 0 0 1
253.23 27 0 0 1
152.15 28 0 0 1
216.42 29 0 0 1
330.8 30 0 0 1
288.87 31 0 0 0
221.1 32 0 0 0
133.89 33 0 0 0
218.08 34 0 0 0
214.09 35 0 0 0
426.41 36 0 0 0
279.46 37 0 0 0
319.67 38 0 0 0
339.78 39 0 0 0
539.78 40 0 0 0
. 41 1 0 0
. 42 0 1 0
. 43 0 0 1
```

```
. 44 0 0 0
;
PROC PRINT;
RUN;

PROC REG;
MODEL BILL = TIME TIMESQ Q1 Q2 Q3/P DW CLM CLI;
RUN;

PROC ARIMA;
IDENTIFY VAR = BILL NOPRINT CROSSCOR = (TIME TIMESQ Q1 Q2 Q3);
ESTIMATE INPUT = (TIME TIMESQ Q1 Q2 Q3) P = 1 PRINTALL PLOT;
FORECAST LEAD = 4;
RUN;

PROC AUTOREG;
MODEL BILL = TIME TIMESQ Q1 Q2 Q3/NLAG = 6 BACKSTEP SLSTAY = 0.05;
RUN;
```

The Durbin-Watson *d* test is produced by specifying the DW option after the slash in the MODEL statement of the REG procedure.

The AUTOREG procedure in SAS fits time series models with autoregressive errors.

In the MODEL statement specify the dependent variable to the left of the equals (=) sign and the independent variables in the deterministic portion of the model to the right. The option NLAG= 1 following the slash (/) specifies a first-order autoregressive model for the random (correlated) error component.

24 Analysis of Variance

ANOVAs for a wide variety of designed experiments can be performed in SAS.

24.1 Clasprog19.sas

The key to performing an ANOVA using SAS is identifying the source(s) of variation for the experiment that is treatments for completely randomized designs treatments and blocks for randomized block designs and main effects and factor interaction for factorial designs.

```
DATA UNLEAD;
INPUT GASTYPE $ MILEAGE @@;
CARDS;
A 24.0 A 25.0 A 24.3 A 25.5
B 25.3 B 26.5 B 26.4 B 27.0 B 27.6
C 23.3 C 24.0 C 24.7
;
PROC GLM;
CLASS GASTYPE;
MODEL MILEAGE = GASTYPE / P CLM;
ESTIMATE 'MUB-MUA' GASTYPE -1 1;
ESTIMATE 'MUC-MUA' GASTYPE -1 0 1 ;
ESTIMATE 'MUC-MUB' GASTYPE 0 -1 1 ;
ESTIMATE 'MUB-(MUC+MUA)/2' GASTYPE -.5 1 -.5;
RUN;

PROC GLM ; CLASS GASTYPE ;
MODEL MILEAGE = GASTYPE / P CLI ;
RUN;
```

24.2 Clasprog20.sas

```
DATA UNLEAD ;
INPUT Y DB DC ;
CARDS;
24.0 0 0
25.0 0 0
24.3 0 0
25.5 0 0
25.3 1 0
26.5 1 0
26.4 1 0
27.0 1 0
27.6 1 0
23.3 0 1
24.0 0 1
24.7 0 1
;
PROC REG;
MODEL Y = DB DC / P;
RUN;

PROC GLM ;
```

```
MODEL Y = DB DC/ P;
ESTIMATE 'MUB - MUA' DB 1 ;
ESTIMATE 'MUC - MUA' DC 1 ;
ESTIMATE 'MUC - MUB' DB -1 DC 1 ;
ESTIMATE 'MUB-(MUC+MUA)/2' DB 1 DC -.5 ;
RUN;
```

24.3 Clasprog21.sas

```
DATA BAKERY;
INPUT HEIGHT $ WIDTH $ DEMAND @@;
CARDS;
B R 58.2 B R 53.7 B R 55.8
B W 55.7 B W 52.5 B W 58.9
M R 73.0 M R 78.1 M R 75.4
M W 76.2 M W 78.4 M W 82.1
T R 52.4 T R 49.7 T R 50.9
T W 54.0 T W 52.1 T W 49.9
;
PROC GLM;
CLASS HEIGHT WIDTH;
MODEL DEMAND = HEIGHT WIDTH HEIGHT*WIDTH/P CLM;
ESTIMATE 'MUM-MUB' HEIGHT -1 1 0;
ESTIMATE 'MUT-MUB' HEIGHT -1 0 1;
ESTIMATE 'MOT-MUM' HEIGHT 0 -1 1;
ESTIMATE 'MUW-MUR' WIDTH -1 1;
ESTIMATE 'MUMW-MUMR' HEIGHT 0 0 0 WIDTH -1 1 HEIGHT*WIDTH 0 0 -1 1 0 0;
RUN;

PROC GLM;
CLASS HEIGHT WIDTH;
MODEL DEMAND = HEIGHT WIDTH HEIGHT*WIDTH/ P CLI;
RUN;
```

24.4 Clasprog22.sas

```
DATA GASONE;
INPUT GASTYPE $ ADDTYPE $ MILEAGE @ @ ;
CARDS ;
A M 19.4 A M 20.6 A M 20.0
A N 25.0 A N 24.0
A 0 24.3 A 0 25.5
B M 22.6 B M 21.6
B N 25.3 B N 26.5
B 0 27.6 B 0 26.4 B 0 27.0
B P 25.4
C M 18.6 C M 19.8
C 0 24.0 C 0 24.7 C 0 23.3
C P 22.0 C P 23.0
RUN;

PROC GLM ;
CLASS GASTYPE ADDTYPE ;
MODEL MILEAGE = GASTYPE ADDTYPE / P CLM ;
```

```
ESTIMATE 'MUB-MUA' GASTYPE -1 1 ;
ESTIMATE 'MUC-MUA' GASTYPE -1 0 1 ;
ESTIMATE 'MUC-MUB' GASTYPE 0 -1 1 ;
ESTIMATE 'MUB-(MUC+MUA)/2' GASTYPE -.5 1 -.5 ;
ESTIMATE 'MUN-MUM' ADDTYPE -1 1 ;
ESTIMATE 'MUO-MUM' ADDTYPE -1 0 1 ;
ESTIMATE 'MOP-MUM' ADDTYPE -1 0 0 1 ;
ESTIMATE 'MUO-MUN' ADDTYPE 0 -1 1 ;
ESTIMATE 'MUP-MUN' ADDTYPE 0 -1 0 1 ;
ESTIMATE 'MUP-MUO' ADDTYPE 0 0 -1 1 ;
RUN;

PROC GLM ; CLASS GASTYPE ADDTYPE ;
MODEL MILEAGE = GASTYPE ADDTYPE / P CLI ;
RUN;
```

24.5 Clasprog23.sas

```
DATA GASTWO;
INPUT GASTYPE $ ADDTYPE $ MILEAGE @@;
CARDS ;
D Q 18.0 D Q 18.6 D Q 17.4
D R 23.0 D R 22.0
D S 23.5 D S 22.3
E Q 23.3 E Q 24.5
E R 25.6 E R 24.4 E R 25.0
E S 23.4
E T 19.6 E T 20.6
F R 24.0 F R 23.3 F R 24.7
F S 23.0 F S 22.0
F T 18.6 F T 19.8
;

PROC GLM;
CLASS GASTYPE ADDTYPE;
MODEL MILEAGE = GASTYPE*ADDTYPE / P CLM;
ESTIMATE 'MUER-MUDS' GASTYPE*ADDTYPE 0 0 -1 0 1 ;
RUN;

PROC GLM;
CLASS GASTYPE ADDTYPE;
MODEL MILEAGE = GASTYPE ADDTYPE GASTYPE*ADDTYPE;
RUN;
```

24.6 Clasprog24.sas

```
DATA REGULAR;
INPUT Y DB DC DN DO DP DAN DAO DBM DBN DBO DBP DCM DCO DCP;
CARDS;
19.4 0 0 0 0 0 0 0 0 0 0 0 0 0 0
20.6 0 0 0 0 0 0 0 0 0 0 0 0 0 0
20.0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
25.0 0 0 1 0 0 1 0 0 0 0 0 0 0 0
24.0 0 0 1 0 0 1 0 0 0 0 0 0 0 0
24.3 0 0 0 1 0 0 1 0 0 0 0 0 0 0
```

```
25.5 0 0 0 1 0 0 1 0 0 0 0 0 0 0
22.6 1 0 0 0 0 0 0 1 0 0 0 0 0 0
21.6 1 0 0 0 0 0 0 1 0 0 0 0 0 0
25.3 1 0 1 0 0 0 0 0 1 0 0 0 0 0
26.5 1 0 1 0 0 0 0 0 1 0 0 0 0 0
27.6 1 0 0 1 0 0 0 0 0 1 0 0 0 0
26.4 1 0 0 1 0 0 0 0 0 1 0 0 0 0
27.0 1 0 0 1 0 0 0 0 0 1 0 0 0 0
25.4 1 0 0 0 1 0 0 0 0 0 1 0 0 0
18.6 0 1 0 0 0 0 0 0 0 0 0 1 0 0
19.8 0 1 0 0 0 0 0 0 0 0 0 1 0 0
24.0 0 1 0 1 0 0 0 0 0 0 0 0 1 0
24.7 0 1 0 1 0 0 0 0 0 0 0 0 1 0
23.3 0 1 0 1 0 0 0 0 0 0 0 0 1 0
22.0 0 1 0 0 1 0 0 0 0 0 0 0 0 1
23.0 0 1 0 0 1 0 0 0 0 0 0 0 0 1
.    1 0 0 1 0 0 0 0 0 1 0 0 0 0
;
PROC REG;
MODEL Y = DB DC DN DO DP / P CLM CLI;
T1: TEST DB = 0, DC = 0;
T2: TEST DN = 0, DO = 0, DP = 0;
RUN;

PROC GLM;
MODEL Y = DB DC DN DO DP / P CLM;
ESTIMATE 'MUB-MUA' DB 1;
ESTIMATE 'MUC-MUA' DC 1;
ESTIMATE 'MUC-MUB' DB -1 DC 1;
ESTIMATE 'MUB-(MUC+MUA)/2' DB 1 DC -.5;
ESTIMATE 'MUN-MUM' DN 1 ;
ESTIMATE 'MUO-MUM' DO 1 ;
ESTIMATE 'MOP-MUM' DP 1;
ESTIMATE 'MUO-MUN' DN -1 DO 1;
ESTIMATE 'MUP-MUN' DN -1 DP 1 ;
ESTIMATE 'MUP-MUO' DO -1 DP 1 ;
ESTIMATE 'MUBO-MUAN' DB 1 DN -1 DO 1 ;
RUN;
PROC GLM;
MODEL Y = DAN DAO DBM DBN DBO DBP DCM DCO DCP / P CLM ;
ESTIMATE 'MUBO-MUAN' DBO 1 DAN -1 ;
RUN;
```

24.7 Clasprog25.sas

SAS code for Nested ANOVA Assembly Data §16.4

```
DATA ASSEMBLY;
INPUT METHOD $ STATION $ UNITS @@;
CARDS;
1 1 16
1 1 7
1 1 7
1 1 13
1 1 16
1 2 14
1 2 24
1 2 13
```

```
1 2 17
1 2 21
2 1 21
2 1 25
2 1 16
2 1 18
2 1 16
2 2 24
2 2 28
2 2 27
2 2 25
2 2 21
3 1 25
3 1 35
3 1 33
3 1 31
3 1 28
3 2 31
3 2 31
3 2 38
3 2 36
3 2 35
;

PROC GLM;
CLASS METHOD STATION ;
MODEL UNITS = METHOD STATION(METHOD) / P CLM ;
ESTIMATE 'MU2-MU1' METHOD -1 1 ;
ESTIMATE 'MU3-MU1' METHOD -1 0 1 ;
ESTIMATE 'MU3-MU2' METHOD 0 -1 1 ;
ESTIMATE 'MU12-MU11' METHOD 0 0 0 STATION(METHOD) -1 1 ;
ESTIMATE 'MU22-MU21' METHOD 0 0 0 STATION(METHOD) 0 0 -1 1;
ESTIMATE 'MU32-MU31' METHOD 0 0 0 STATION(METHOD) 0 0 0 0 -1 1 ;
RUN;
```

24.8 Clasprog26.sas

SAS Codes from Chapter 17.

```
DATA BOXES;
INPUT BLOCK $ METHOD $ PRDN @ @;
CARDS;
ONE 1 9
ONE 2 8
ONE 3 3
ONE 4 4
TWO 1 10
TWO 2 11
TWO 3 5
TWO 4 5
THREE 1 12
THREE 2 12
THREE 3 7
THREE 4 5
;
PROC GLM;
CLASSES BLOCK METHOD;
MODEL PRDN=BLOCK METHOD/P CLM;
ESTIMATE 'MU4-MU1' METHOD -1 0 0 1;
```

```
ESTIMATE 'MU4-MU2' METHOD 0 -1 0 1;
ESTIMATE 'MU4-MU3' METHOD 0 0 -1 1;
ESTIMATE 'MU3-MU2' METHOD 0 -1 1 0;
ESTIMATE 'MU3-MU1' METHOD -1 0 -1;
ESTIMATE 'MU2-MU1' METHOD -1 1;
ESTIMATE 'MUTHREE-MUTWO' BLOCK 0 1 -1;
ESTIMATE 'MUTHREE-MUONE' BLOCK -1 1 0;
ESTIMATE 'MUTWO-MUONE' BLOCK -1 0 1;
RUN;

PROC GLM;
CLASSES BLOCK METHOD;
MODEL PRDN=BLOCK METHOD/P CLI;
RUN;
```

24.9 Clasprog27.sas

```
DATA CARDBD;
INPUT DEFECTS MACHTYPE $ CBGRADE $ OPERATOR $ @@;
CARDS;
9 A Y 1
10 A Y 2
12 A Y 3
8 A Z 1
11 A Z 2
12 A Z 3
3 B Y 1
5 B Y 2
7 B Y 3
4 B Z 1
5 B Z 2
5 B Z 3
;

PROC GLM;
CLASS MACHTYPE CBGRADE OPERATOR;
MODEL DEFECTS=MACHTYPE CBGRADE OPERATOR/P CLM;
ESTIMATE 'MUBJH-MUAJH' MACHTYPE -1 1;
ESTIMATE 'MUIZH-MUIYH' CBGRADE -1 1;
ESTIMATE 'MUIJ3-MUIJ1' OPERATOR -1 0 1;
ESTIMATE 'MUIJ2-MUIJ1' OPERATOR -1 1 0;
ESTIMATE 'MUIJ3-MUIJ2' OPERATOR 0 -1 1;
RUN;
PROC GLM;
CLASS MACHTYPE CBGRADE OPERATOR;
MODEL DEFECTS=MACHTYPE CBGRADE OPERATOR/P CLI;
RUN;
PROC GLM;
CLASS MACHTYPE CBGRADE OPERATOR;
MODEL DEFECTS=MACHTYPE CBGRADE MACHTYPE*CBGRADE OPERATOR/P CLM;
ESTIMATE 'MUBH-MUAH' MACHTYPE -1 1;
ESTIMATE 'MUZH-MUYH' CBGRADE -1 1;
ESTIMATE 'MUIJ3-MUIJ1' OPERATOR -1 0 1;
ESTIMATE 'MUIJ2-MUIJ1' OPERATOR -1 1 0;
ESTIMATE 'MUIJ3-MUIJ2' OPERATOR 0 -1 1;
ESTIMATE 'MUBZH-MUAYH' MACHTYPE -1 1 CBGRADE -1 1
          MACHTYPE*CBGRADE -1 0 0 1 OPERATOR 0 0 0 0;
```

```
RUN;
```

24.10 Clasprog28.sas

```
DATA GASOLINE;
INPUT GASTYPE $ DRIVER $ CAR $ MILES @@;
CARDS;
4 1 1 17.6
2 1 2 37.8
3 1 3 15.3
1 1 4 31.0
2 2 1 20.4
3 2 2 28.7
1 2 3 21.3
4 2 4 24.7
3 3 1 12.7
1 3 2 33.0
4 3 3 19.0
2 3 4 34.4
1 4 1 16.8
4 4 2 36.1
2 4 3 23.8
3 4 4 23.7
;
PROC GLM;
CLASS GASTYPE DRIVER CAR;
MODEL MILES=GASTYPE DRIVER CAR/P CLM;
ESTIMATE 'MUGAS2-MUGAS1' GASTYPE -1 1 0 0;
ESTIMATE 'MUGAS2-MUGAS3' GASTYPE 0 1 -1 0;
ESTIMATE 'MUGAS2-MUGAS4' GASTYPE 0 1 0 -1;
RUN;
```

The REG or GLM procedures performs an analysis of variance. The CLASSES statement identifies the independent variables (factors) for the experiment. In SAS factors can be either quantitative or qualitative variables.

The sources of variation are specified to the right of the equals sign (=) in the MODEL statement the dependent (response) variable to the left.

25 Hardware Requirements

The minimum hardware and software requirements for SAS for Windows are:

DOS 3.1 or later
Windows 3.1 or later
80386 processor or higher
8 MB RAM
A hard disk with at least 23 MB of free disk space (this varies depending on the number of options you want to add).
A high density 3.5" floppy disk drive
A graphics adaptor with 640x480 resolution (VGA) or higher

Recommended hardware and software requirements for SAS for Windows are as following:

33 MHz 80486/DX processor More than 8 MB RAM Windows running in enhanced mode with a 10 MB permanent swap file Smartdrive with a Windows cache size of at least 512K Math coprocessor strongly recommended for significant graphics work.

26 Further Help

The material covered in this document illustrates some of the basic features of SAS for Windows. For further help refer to SAS for Windows documents. SAS for Windows is currently available for public access from all Clemson University Student Labs.

27 Documentation

SAS Version 6 documentation is available for reference at a number of locations on campus. The document SAS Companion for Microsoft Windows Version 6 First Edition contains information relevant to accessing SAS from a Microsoft Windows environment. Except for this document all other SAS documents are independent of platform or operating system. The basic documents include:

References

[1] SAS Language and Procedures Version 6

[2] SAS Language Version 6

[3] SAS Procedures Guide Version 6

[4] SAS/STAT User's Guide Vol. 1 & 2 Version 6

[5] SAS/Graph Software Vol. 1 & 2 Version 6.

You may order documents may be ordered through Clemson University Bookstore or directly from SAS Institute Inc. Book Sales Dept SAS Campus Drive Cary NC 27513 (phone: 919/677-8000).