UNIVERSITI PUTRA MALAYSIA

AGRICULTURE • INNOVATION • LIFE

Introduction to SAS for Data Science

by

LIM FONG PENG

Outline

1) Introduction to the SAS Language
2) The DATA Step
3) The PROC Step
4) Modifying SAS Data
5) SAS Procedures for Data Description & Simple Inference
6) SAS in Several Statistical Analyses

1) Introduction to the SAS Language

Introduction to the SAS Language

- The SAS system is an integrated set of modules for manipulating, analyzing, and presenting data.
- SAS is a programming language composed of statements that specify how data are to be processed and analyzed.
- The statements correspond to operations to be performed on the data or instructions about the analysis.

Introduction to the SAS Language

- A SAS program consists of two steps:

  data step - prepare data for analysis. It creates a SAS data set and may reorganize the data and modify it in the process.

  procedure (proc) step - perform a particular type of analysis, or statistical test, on the data in a SAS data set.

Introduction to the SAS Language

- When SAS is started, there are five main windows open:

- **Editor**: This is a text editor. You can type in, edit and submit SAS program.

- **Log**: It contains notes about your SAS session (error; warning)

- **Output**: If your program generates any printable results then they will appear in this window.

Introduction to the SAS Language

- **Results**: The Results window is effectively a graphical index to the Output window useful for navigating around large amounts of procedure output.

- **Explorer**: The Explorer window allows the contents of SAS data sets and libraries to be examined interactively, by double-clicking on them.

SAS

File Edit View Tools Solutions Window Help

Explorer

Contents of 'SAS Environment'

Libraries File Shortcuts Favorite Folders

Computer

Results Explorer

Output - (Untitled)

```
                                    EXAMPLE ON PROC REG          21:31 Friday, May 14, 2018   9

                                    The ANOVA Procedure

                                    t Tests (LSD) for Y

NOTE: This test controls the Type I comparisonwise error rate, not the experimentwise error rate.

                              Alpha                          0.05
                              Error Degrees of Freedom         28
                              Error Mean Square          0.087054
                              Critical Value of t         2.04841
                              Least Significant Difference  0.3022


                     Means with the same letter are not significantly different.


                         t Grouping          Mean       N    FACA
```

Editor - Untitled1 * PROC ANOVA running

```
DATA LAB7SLR;
INPUT FACA $ @;
DO I=1 TO 8;
INPUT Y @;
OUTPUT;
END;
CARDS;
```

Output - (Untitled) | Log - (Untitled) | Editor - Untitled1 * PR...

C:\Users\User

Basic Language: Rules and Syntax

1.1) Data Values

- Data values are classified as either character values or numeric values.
- A character value may include letters, numbers, blanks, and special characters.
- Examples:

MIG7, D'Arcy, 5678, South Dakota

Basic Language: Rules and Syntax

1.1) Data Values (Cont)

- A standard numeric value is a number with or without a decimal point that may be preceded by a plus or minus sign but may **not** contain commas.

- Examples:

71 0.0038 –4.0

8214.7221 8.546E–2

Basic Language: Rules and Syntax

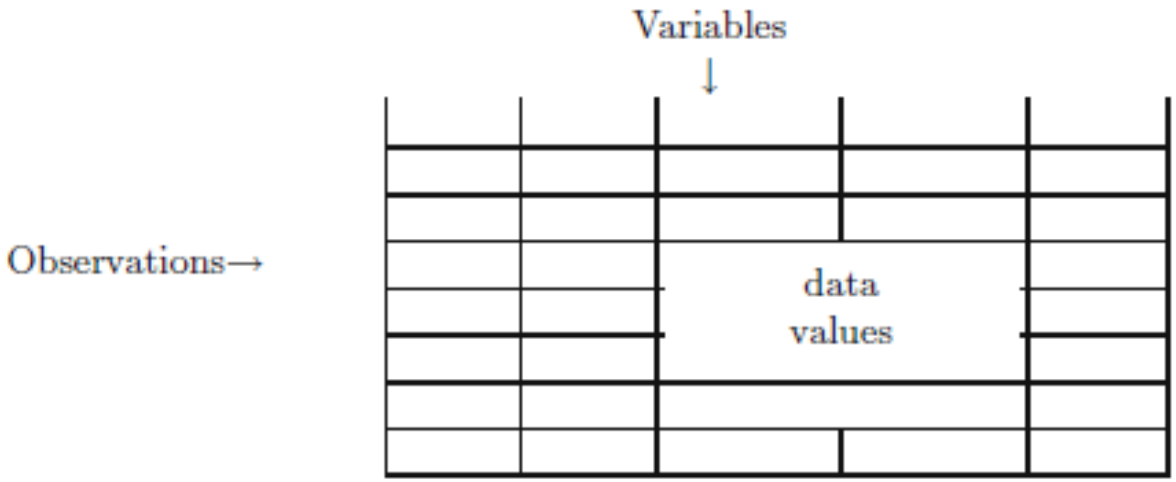
1.1) Data Values (Cont)

- Missing character data are represented by blanks.
- Example: F M □ M M
  F F M F M
- Missing numeric data are represented by a single period (·).
- Example: 50 70 · 49

Basic Language: Rules and Syntax

1.2) *SAS* Data Sets

- SAS data sets consist of *data values* arranged in a rectangular array as displayed in figure below:



Structure of
a SAS data set

- Data values in a column represent a *variable* and those in a row comprise an *observation*.

Basic Language: Rules and Syntax

1.3) Variables

- Variables are of two types: ***numeric*** or ***character***.
- A character variable can include values that are numbers, but they are treated like any other sequence of characters.
- SAS cannot perform arithmetic operations on values of a character variable.

Basic Language: Rules and Syntax

1.4) Observations

- An observation is a group of data values that represent different measurements on the same individual.
- "Individual" here can mean a person, an experimental animal, a geographic region, a particular year, and so forth.

Basic Language: Rules and Syntax

1.5) SAS Names

- The first character in a SAS name **must** be an alphabetic character.
- Embedded blanks are **not** allowed.
- Characters after the first can be alphabetic, numeric, or underscores.
- The alphabetic character can be upper and lowercase.
- Examples: H22A, rep_no, yield

Basic Language: Rules and Syntax

1.6) SAS Variable Lists

- A list of SAS variables consists of the names of the variables separated by one or more blanks.
- Example:

H22A rep_no yield

Basic Language: Rules and Syntax (Cont)

1.6) SAS Variable Lists (Cont)

- A user may define a sequence of variable names in SAS statements by using an abbreviated list of the form

charsxx-charsyy

where "chars" is a set of characters and the "xx" and "yy" indicate a sequence of numbers.

Basic Language: Rules and Syntax

1.6) SAS Variable Lists (Cont)

- Example: the list of indexed variables q2 through q9 may appear in a SAS statement as

q2 q3 q4 q5 q6 q7 q8 q9

- or equivalently as

q2 - q9

Basic Language: Rules and Syntax

1.7) SAS Statements

- Every SAS statement **ends with a semicolon (;)**.
- Can be in upper or lower case.
- Can continue on the next line.
- Can be on the same line as other statement.
- Can start in any column.
- To make the program more understandable, you can insert comments into your programs.

Basic Language: Rules and Syntax (Cont)

1.7) SAS Statements (Cont)

<u>Style of Comments</u>

- Begins with an asterisk (*) and ends with a semicolon.
- Example: *Height of Students in UPM;
- Starts with a slash asterisk (/*) and ends with asterisk slash (*/)
- Example: /*Height of Students in UPM*/

2) The DATA Step

2.1) The *Data* Statement

2.2) The *Cards* Statement or *Datalines* Statement

2.3) The *Infile* Statement

2.4) The *Input* Statement

2.5) Reading Data from an Existing SAS Data Set

2.6) SAS System Options

The Data Step

2) The Data Step

- A “raw” data file can also be referred to as a text file.
- A data step is also used to manipulate, or reorganize the data.
- General form:

```
data …… ;
infile …… ;
input …… ;
```

OR

```
data …… ;
Input …… ;
cards;
…….
…….
;
```

datalines

The Data Step

2) The Data Step (Cont)

- **Example**: The data below shows some hypothetical data on members of a slimming club, giving the membership number, team, starting weight, and current weight.

1023	red	189	165
1049	yellow	145	124
1219	red	210	192
1246	yellow	194	177
1078	red	127	118
1221	yellow	220	.
1095	blue	135	127
1157	green	155	141

The Data Step

2) The Data Step (Cont)

- Program for example:

```
data wghtclub;
infile
'n:\handbook2\datasets\wgtclub1.dat'
;
input idno team $ startweight
weightnow;
```

```
data wghtclub;
input idno team $ startweight
weightnow;
cards;
1023  red      189  165
1049  yellow   145  124
1219  red      210  192
…     …        …    …
;
```

The Data Step

2.1) The *Data* Statement

- The data statement names data set being created, in this case *wghtclub*.

2.2) The *Cards* Statement or *Datalines* Statement

- Put the data set directly into the program editor.
- The *cards* statement in this example:

```
cards;
1023  red      189  165
1049  yellow   145  124
1219  red      210  192
…     …        …    …
;
```

The Data Step

2.3) The *Infile* Statement

- The *infile* statement specifies the file where the raw data are stored.
- The full pathname of the file is given.
- The *infile* statement in this example:

  infile 'n:\handbook2\datasets\wgtclub1.dat';

- If the file is in the current directory, the file name could have been specified simply as '*wghtclub1.dat*'.

The Data Step

2.4) The *Input* Statement

- SAS has only two types of variables: numeric and character.
- The function of the input statement is to name the variables, specify their type as numeric or character, and indicate where in the raw data the corresponding data values are.
- The data values are separated by spaces.
- The variable names are listed in order and character variables are indicated by a dollar sign ($) after their name.

The Data Step

2.4) The *Input* Statement (Cont)

- The correct ways to write a *input* statement:
  - ✓ **list** form
  - ✓ **column** form
  - ✓ **formatted** form

The Data Step

2.4.1) The list form of input:

- Example:

  input idno team$ startweight weightnow;

- Four variables are to be read in from the raw data file: idno, team, startweight, and weightnow.

- The dollar sign ($) after team indicates that it is a character variable.

The Data Step

2.4.2) The column form of input:

- Example:

input name$ 1-18 team$ 20-25 startweight 27-29 weightnow 31-33;

- Can be used when the raw data files do not have spaces between all the values or periods for missing data.

- Each variable are specified after the variable name, or after the dollar in the case of a character variable.

The Data Step

2.4.2) The column form of input (Cont):

- The start and finish columns are separated by a hyphen.
- But for single column variables it is only necessary to give the one column number.
- Make sure that each of the variable's values is **always** found in the same place in the data line.

The Data Step

2.4.2) The column form of input (Cont):

- Advantages of column input compared to *list* input:
  - ✓ Spaces are not required between values.
  - ✓ Missing data can be left blank.
  - ✓ Character data can have embedded spaces.
  - ✓ We can skip unwanted variables.

The Data Step

2.4.1) The formatted form of input: Informat

- Example:

  input name $19. team $7. startweight 4. weightnow 3.;

- To tell SAS how many columns to read.
- The informat for both numeric and character variables is the number of columns occupied by the data values and a period (·).

The Data Step

2.4.1) The formatted form of input (Cont): Informat

- Note that the spaces separating the data values have been taken into account in the informat.

- When numeric data contain an implied decimal point, the informat has a second number after the period to indicate the number of digits to the right of the decimal point.

Example:

5.2 would read five columns of numeric data and move the decimal point two places to the left.

The Data Step

2.4.1) The formatted form of input: Pointer

- The column pointer (@) is to specify the starting column for each variable.

Example:

```
        col1   col4   col9   col11   col14

Data:     001M   191.52
          002F   211.50

        ID   Gender   Age   Height
```

Statement: *input @1 ID 3. @4 Gender $ 1. @9 Age 2. @11 Height 3.2;*

The Data Step

2.4.1) The formatted form of input (Cont): Pointer

- The row pointer (#) specifies which row of data we want to read for a subject having more than one dataline.

Example:

```
        col1     col5   col10
Data:     001 56      72   }  first observation
              130080
          002 45      70   }  second observation
              140070

Statement:    INPUT  #1  ID  1-3  Age  5-6  Height  10-11  #2  SBP  5-7  DBP  8-10;
```

The Data Step

2.4.1) The formatted form of input (Cont): Pointer

- It is possible to move the pointer to any starting column.
- Example:

input @1 ID 3. @9 Age 2. @4 Gender $ 1. @11 Height 3.2;

The Data Step

2.4.1) The formatted form of input (Cont): Pointer

- It is suitable when a row of data consists of variable value with "=" sign.

Example: Name = Md Ali 24 Program = BSK

col. 24 col. 40

```
INPUT @20 Age @1 Name=$ @40 Program=$;
```

The Data Step

2.4.1) The formatted form of input: Informat list

- Place a group of variables together within parenthesis () in input statement and follow this list by one or more informats, also in parenthesis.

Example: INPUT (X Y Z C1 – C3) (1. 2. 1. $3. $3. $3.)

Input ID 1 – 3 @4 (SBP1 – SBP12) (3. +3) @7 (DBP1 – DBP12) (3. +3)

or Input ID 1 – 3 SBP1 4 – 6 DBP1 7 – 9 SBP2 10 –12 DBP2 …

The Data Step

2.4.1) The formatted form of input (Cont): Informat list

- @ also means that SAS will hold that line of data until it reaches either the end of *data* step.
- A single @ sign place at the end of an *input* statement means "to hold the line" which means do not move the pointer to the next record until the end of the *data* step is reached.

The Data Step

2.4.1) The formatted form of input (Cont): Informat list

- The double trailing @ symbol means "hold the line strongly" which means the pointer will not move to next record even if a *cards* statement is encountered. It will move to the next record if there are no more data values to be read in a line.

Example:

```
DATA   EXAMPLE;
INPUT  X Y @@;
CARDS;
1 2 7 9 3 4 10 12
15 18 23 67
```

The Data Step

2.5) Reading Data from an Existing SAS Data Set

- To read data from a SAS data set, the *set* statement is used in place of the *infile* and *input* statements.

Example:

```
data wgtclub2;
set wghtclub;
run;
```

- Create a new SAS data set *wgtclub2* reading in the data from *wghtclub*.
- It is also possible for the new data set to have the same name with the existing SAS data set.

The Data Step

2.6) SAS System Options

- Example: *OPTIONS LINESIZE=80 NODATE;*
- Options linesize = n
  - ✓ It controls the maximum length of output lines, and n from 64 – 256 (by default: varies).
- Options center / nocenter
  - ✓ It controls whether output are centered or left-justified (by default: center).

The Data Step

2.6) SAS System Options (Cont)

- Options date / nodate

✓ It controls whether or not today's date appear at the top of each page of output (by default: date).

- Options number / nonumber

✓ It controls whether or not page numbers appear on each page of SAS output (by default: number)

3) The PROC Step

3) The PROC Statement

3.1) PROC PRINT Procedure

3.2) PROC SORT Procedure

3.3) TITLE Statement

The PROC Step

3) PROC Statement

- The PROC step is a block of statements that specify the data set to be analyzed, the procedure to be used, and any further details of the analysis.

- The step begins with a PROC statement and ends with a RUN statement or when the next DATA or PROC step starts.

- The PROC statement names the procedure to be used and specify the DATA = option for the analysis.

The PROC Step

3.1) PROC PRINT Procedure

- Statement: PROC PRINT DATA = filename;

- To print the data values after reading them to make sure they are correct.
- To print the values for all variables and all observation in a SAS data set.

The PROC Step

3.1) PROC PRINT Procedure (Cont)

- Example 3.1: Write a program to print the values for all variables in the data as below,

sex age height weight

ID of students

A	F	15	2.5	52
B	M	10	3.2	56
C	F	10	5.1	85
D	M	10	2.3	42

The PROC Step

3.1.1) VAR Statement

- The VAR statement specifies the variables that are to be processed by the PROC step.

- Example 3.2: Using the data in Example 3.1, write a program to print the values for variables ID, gender and age only.

The PROC Step

3.1.2) WHERE Statement

- The WHERE statement selects the observations to be processed.
- The keyword WHERE is followed by a logical condition, and only those observations for which the condition is true are included in the analysis.
- Example 3.3: Using the data in Example 1, write a program to print a list of students who more than 15 years old. Print the variables ID, gender and age only.

The PROC Step

3.1.3) BY Statement

- The observations are grouped according to the values of the variable named on the BY statement and a separate analysis is conducted for each group.
- When a BY statement appears as part of the PRINT procedure, the procedure expects the data to be sorted in the order of the BY variables.

The PROC Step

3.1.3) BY Statement (Cont)

- Example 3.4:

Using the data in Example 1, write a program to print all observations according to their gender.

Program:

```
PROC SORT DATA= example1;
BY SEX;
PROC PRINT DATA= example1;
BY sex;
RUN;
```

Output is

---------- sex=F ----------

Obs	name	age	height	weight
1	A	15	2.5	52
2	C	10	5.1	85

---------- sex=M ----------

Obs	name	age	height	weight
3	B	10	3.2	56
4	D	10	2.3	42

The PROC Step

3.1.4) SUM Statement

- To specify variables whose values are to be totaled.
- Example 3.5: Using the data in Example 1, write a program to calculate the total age for all observations and print it.

The PROC Step

3.2) PROC SORT Procedure

Statement:

```
PROC SORT;
BY    Var 1   Var 2   …    Var k;
```

- *BY* statement specifies the variables to be used for sorting.
- can specify as many *BY* variables.

The PROC Step

3.2) PROC SORT Procedure (Cont)

- Observations are first arranged in the increasing order of the values of the first variable specified in the *BY* statement.
- Within each of the resulting groups, observations are arranged in the increasing order of the values of the second variable specified and so on.
- By default, SAS sorts data in ascending order.

The PROC Step

3.3) TITLE Statement

- The TITLE statement tells SAS to put the title on the top of each page of output.

- Without TITLE statement, SAS would put the words 'The SAS System' at the top of each page.

4) Modifying SAS Data

4.1) Creating and Modifying Variables
- IF statement

4.2) Deleting Variables and Observations

4.3) Merging Data Sets

Modifying SAS Data

4.1) Creating and Modifying Variables

- SAS has the normal set of arithmetic operators: +, -, / (divide), *(multiply), and **(exponentiate), plus various arithmetic, mathematical, and statistical functions.

- The result of an arithmetic operation performed on a missing value is itself a missing value.

Modifying SAS Data

IF THEN Statement

- The condition is an expression comparing one thing to another,
- The action is what SAS should do when the expression is true.
- Example:

```
IF gender = 'F' THEN group = 1;
```

- A single IF THEN statement can **only have one action**.

Modifying SAS Data

IF THEN Statement (Cont)

- Basic Comparison Operators:

Symbolic	Meaning
=	equals
^= or ~=	not equals
>	great than
<	less than
>=	greater than or equals
<=	less than or equals

Modifying SAS Data

IF THEN Statement (Cont)

- If you add the keywords DO and END, then you can execute **more than one action.**

Statement:

```
IF condition THEN DO;
action;
action;
END;
```

Example:

```
IF model = 'mustang' THEN DO;
        make = 'ford';
        size = 'compact';
END;
```

Modifying SAS Data

IF THEN Statement (Cont)

- You can also specify **multiple conditions** with the keywords AND and OR.

Statement:

```
IF condition AND condition THEN action;
```

- Example: IF model = 'mustang' AND year < 1975 THEN status = 'classic';

Symbolic	Mnenomic	Meaning
&	AND	All comparisons must be true.
|	OR	Only one comparison must be true.

Modifying SAS Data

IF THEN Statement (Cont)

- You can specify **multiple conditions** with **different actions** using the statement below:

Statement:

```
IF condition THEN action;
        ELSE IF condition THEN action;
        ELSE IF condition THEN action;

                or

IF condition THEN action;
        ELSE IF condition THEN action;
        ELSE action;
```

Example:

```
IF age >= 18 THEN adult = 1;
        ELSE adult = 0;
```

Modifying SAS Data

IF THEN Statement (Cont)

- Example 4.1:

Using the data below, write a program to categorize the students into 2 groups such as:

```
name    gender
 ↓       ↙
ALI  L  ANI  P  CHUA  L
LING  P  AZHAR  L  YAP  P
SANI L  GUNA  L
```

'P' = Group 1 and

'L' = Group 2

Modifying SAS Data

4.2) Deleting Variables and Observations

- Deleting Variables:

DROP Statement

- Example: the statement

```
DROP x, y, z;
```

in a data step results in a data set that does not contain the variables x, y, z.

KEEP Statement

- Example: the statement

```
KEEP x y z ;
```

results in a data set that contains only those three variables.

Modifying SAS Data

4.2) Deleting Variables and Observations (Cont)

- Deleting / Subsetting Observations:

Statement:

```
IF expression THEN DELETE;
```

- To tell SAS which observations to exclude.
- Example: IF gender = 'M' THEN DELETE; or IF gender = 'F';

Modifying SAS Data

4.2) Deleting Variables and Observations (Cont)

- Example 4.2: Use the data in Example 4.1, write a program to print all the observations, excluding those are female.

Output is	ALI	L
	CHUA	L
	AZHAR	L
	SANI	L
	GUNA	L

Modifying SAS Data

4.3) Merging Data Sets

- Two or more data sets can be combined into one by specifying them on a single SET statement.
- A simple way of adding new observations to an existing data set.
- Statement:

```
DATA   prog1;
SET   dataset1    dataset2;
RUN;
```

Modifying SAS Data

4.3) Merging Data Sets

- Example 4.3

```
data third;
input w 1-2 x 3-5 y 6;
datalines;
211023
312034
413045
;
run;
data fourth;
input x y z;
datalines;
14 5 7862
15 6 6517
16 7 8173
;
run;
data fifth;
set third fourth;
run;
proc print;
title 'Combining SAS data sets end-to-end ';
run;
```

Output is

Combining SAS data sets end-to-end

Obs	w	x	y	z
1	21	102	3	.
2	31	203	4	.
3	41	304	5	.
4	.	14	5	7862
5	.	15	6	6517
6	.	16	7	8173

5) SAS Procedures for Data Description & Simple Inference

SAS Procedures for Graphics

5.1) PLOT Procedure

5.2) CHART Procedure

SAS Procedures for Computing Statistics

5.3) PROC MEANS Statement

5.4) PROC UNIVARIATE Statement

5.5) PROC FREQ Statement

5.6) PROC CORR Statement

SAS Procedures for Graphics

5.1) PLOT Procedure

- The general structure of a PROC PLOT step is

```
PROC PLOT < options >;
BY variables;
PLOT plot-requests < / options >;
```

SAS Procedures for Graphics

5.1) PLOT Procedure (Cont)

- PROC statement options:
  - ✓ 'DATA=' for naming the data set to be analyzed,
  - ✓ 'NOMISS' for excluding observations with missing values,
  - ✓ 'NOLEGEND' suppresses the legend that appears on top of the plot by default,
  - ✓ 'HPERCENT=' or 'HPCT=' is used to list percentages of the available horizontal space to be used for each plot (in case of multiple plots that are not overlaid).
  - ✓ 'VPERCENT=' or 'VPCT=' is used to list percentages of the available vertical space to be used for each plot.

SAS Procedures for Graphics

5.1) PLOT Procedure (Cont)

- PLOT statement options:
  - ✓ ‘HAXIS=’ and ‘ VAXIS=’ specify the tick-mark values for the horizontal and vertical axes, respectively.
  - ✓ ‘HREF=’ and ‘VREF=’ specify the positions on the horizontal and the vertical axis, respectively, at which lines will be drawn on the plot perpendicular to the respective axes. By default, the characters / and – respectively, will be used to draw the reference lines. The options ‘HREFCHAR=’ and ‘VREFCHAR=’ may be used to specify alternate choices (e.g. HREFCHAR =’×’).
  - ✓ ‘BOX’ draws a solid line border around the plot.
  - ✓ ‘OVERLAY’ overlays all plots that are specified in the plot statement on one set of axes.

SAS Procedures for Graphics

5.1) PLOT Procedure (Cont)

- Example 5.1: Using the following data

SEX	WEIGHT	HEIGHT
M	68	155
F	61	99
F	63	115
M	70	205
M	69	170
F	65	125
M	72	220

to draw a plot such as OUTPUT Example 5.1a & 5.1b.

SAS Procedures for Graphics

5.2) CHART Procedure

- The general structure of a PROC CHART step is

```
PROC CHART < options >;
        BLOCK variables < / options >;
        BY variables;
        HBAR variable < / options >;
        PIE variable < / option >;
        STAR variable < / option >;
        VBAR variable < / option >;
```

SAS Procedures for Graphics

5.2) CHART Procedure (Cont)

- Options for HBAR, VBAR, BLOCK, PIE, or STAR statements:

  ✓ "LEVELS=" specify the number of bars representing each chart variable when the variables given in the VBAR statement are continuous.

  ✓ "SYMBOL= *character*" defines the symbol to be used in the body of standard HBAR and VBAR charts with no subgrouping.

SAS Procedures for Graphics

5.2) CHART Procedure (Cont)

- Options for HBAR, VBAR, BLOCK, PIE, or STAR statements:

  ✓ "MIDPOINT= *values*" defines the range of value for the chart variable each bar or section represents by specifying the range midpoints.

  ✓ "REF= *n*" (n is integer if frequency on y-axis) or "REF= *p*" ($0 \leq p \leq 100$ if cumulative frequency on y-axis) request that a single reference line be drawn on the response axis.

SAS Procedures for Graphics

5.2) CHART Procedure (Cont)

- Options for HBAR, VBAR, BLOCK, PIE, or STAR statements:

  ✓ "TYPE=*statistics*" specifies what the bars in the chart represent (by default: TYPE=FREQ).

Option	Results
TYPE=FREQ	Frequency count
TYPE=PCT	Percentages
TYPE=CFREQ	Cumulative frequencies
TYPE=CPCT	Cumulative percentages
TYPE=SUM	Totals
TYPE=MEAN	Means

SAS Procedures for Graphics

5.2) CHART Procedure (Cont)

- Options for HBAR, VBAR, BLOCK, PIE, or STAR statements:

  ✓ "GROUP= *variable*" produce side by side charts, with each chart representing the observations having a given value of the GROUP=variable.

  ✓ "SUBGROUP= variable" requests that each bar be subdivided into characters that show the SUBGROUP= variable's contribution to the bar.

SAS Procedures for Graphics

5.2) CHART Procedure (Cont)

- Example 5.2: Using the data set in Example 5.1, draw a chart given that

a)
```
PROC CHART;
HBAR SEX;
```

b)
```
PROC CHART;
VBAR SEX;
```

c)
```
PROC CHART;
BLOCK SEX;
```

d)
```
PROC CHART;
VBAR WEIGHT / MIDPOINTS=50 TO
80 BY 10;
```

e)
```
PROC CHART;
VBAR WEIGHT / GROUP=SEX;
```

SAS Procedures for Computing Statistics

5.3) PROC MEANS Statement

- PROC MEANS provides simple statistic on numerical (quantitative) variables.

Statement:

```
PROC MEANS < options >;
VAR < variable – list >;
```

- The order of the options does not matter.
- Refer to Appendix for the list of the commonly requested options for PROC MEANS.

SAS Procedures for Computing Statistics

5.3) PROC MEANS Statement (Cont)

- Example 5.3: Using the data set in Example 5.1, write a program to obtain some descriptive statistics such as mean, median etc. Use

a) PROC MEANS MAXDEC=2 N MEAN STD STDERR VAR;

b) PROC MEAN DATA=Book N MEAN MEDIAN CLM ALPHA=0.10 ;

SAS Procedures for Computing Statistics

5.3) PROC UNIVARIATE Statement

- PROC UNIVARIATE is best suited for studying the empirical distributions of variables in a data set.
- It can be used to produce graphics such as histograms with overlayed kernel density estimates, quantile-quantile plots, and probability plots

SAS Procedures for Computing Statistics

5.4) PROC UNIVARIATE Statement (Cont)

- The general structure of a PROC UNIVARIATE statement:

```
PROC UNIVARIATE DATA=filename < options >;
VAR variable-list;
```

- Refer to Appendix for the PROC statement.

SAS Procedures for Computing Statistics

5.4) PROC UNIVARIATE Statement (Cont)

- Example 5.4: Using the data set in Example 5.1, write a program to obtain some descriptive statistics such as mean, median etc. Use the PROC UNIVARIATE statement below.

```
PROC UNIVARIATE NORMAL PLOT;
VAR HEIGHT WEIGHT;
```

SAS Procedures for Computing Statistics

5.5) PROC FREQ Statement

- The FREQ procedure in SAS computes many statistics and measures related to the analysis of categorical data.

Statement:

```
PROC FREQ DATA =filename;
TABLES variable-combinations / < options >;
```

- Refer to Appendix for the TABLES options.

SAS Procedures for Computing Statistics

5.5) PROC FREQ Statement (Cont)

- Example 5.5:

```
DATA HTWT;
INPUT SEX $ WEIGHT HEIGHT SMOKE;
CARDS;
M      68     155    1
F      61     99     2
F      63     115    1
M      70     205    2
M      69     170    1
F      65     125    1
M      72     220    1
;
PROC FREQ;
TABLES SEX*SMOKE;
RUN;
```

- Output Example 5.5:

The FREQ Procedure

Table of SEX by SMOKE

SEX	SMOKE		
Frequency Percent Row Pct Col Pct	1	2	Total
F	2 28.57 66.67 40.00	1 14.29 33.33 50.00	3 42.86
M	3 42.86 75.00 60.00	1 14.29 25.00 50.00	4 57.14
Total	5 71.43	2 28.57	7 100.00

SAS Procedures for Computing Statistics

5.6) PROC CORR Statement

- Correlation analysis is to measure the strength or degree of linear association between 2 variables.

Coefficient correlation, $r \Longrightarrow -1 < r < 1$

- If $r = 0$, no relationship between two variables.
- If $r = \pm 1$, perfect relationship.
- Example: $r = 0.9$ means strong relationship.
  $r = 0.2$ means weak relationship.

SAS Procedures for Computing Statistics

5.6) PROC CORR Statement (Cont)

- Statement:

```
PROC   CORR    DATA= filename;
VAR   variable-list;
WITH    variable-list;
```

- Example 5.6: refer to Appendix.

6) SAS in Several Statistical Analyses

6.1) T-Test
6.2) Regression
6.3) Analysis of Variance
6.4) Survival Analysis

SAS in Several Statistical Analyses

6.1) T-Test : One Sample

- PROC MEANS will compute the t-statistic and *p*-value associated with $H_0: \mu = 0 \;\; vs. \; H_a: \mu \neq 0$.

- The option T requests the t-statistic, and the option PRT requests the p-value for the two-sided test.

SAS Procedures for Computing Statistics

6.1) T-Test : One Sample (Cont)

- If you are interested in a μ_0 value other than 0, you can recode the data by subtracting μ_0 from each observation. Testing that the true mean of the recoded data equals 0 is the same as testing whether the true mean of the original data equals μ_0.

- If you want to do a one-sided test instead of a two-sided test, divide the *p*-value on the printout by two. The result is the *p*-value for a one-sided test.

SAS in Several Statistical Analyses

6.1) T-Test : Two Samples

- PROC TTEST general form is

```
PROC   TTEST;
CLASS   variable;
VAR   variables;
```

- Example 6.1: Refer to Appendix.

SAS in Several Statistical Analyses

6.2) Simple & Multiple Linear Regression

- PROC REG general form is

```
PROC REG;
MODEL  dependent variable = independent variables < / options >;
```

- Refer to Appendix for the options of MODEL statement.

- Example 6.2 & 6.3: Refer to Appendix.

SAS in Several Statistical Analyses

6.3) Analysis of Variance

- PROC ANOVA general form is

```
PROC   ANOVA;
CLASS  variables;
MODEL  dependent variable = independent variables;
MEANS   effects < / options > ;
```

- Refer to Appendix for the options of MEANS statement.
- Example 6.4 & 6.5: Refer to Appendix.

SAS in Several Statistical Analyses

6.4) Survival Analysis

- There are three SAS procedures for analyzing survival data: LIFETEST, PHREG, and LIFEREG.
- PROC LIFETEST is a nonparametric procedure for estimating the survivor function, comparing the underlying survival curves of two or more samples, and testing the association of survival time with other variables.
- PROC PHREG is a semiparametric procedure that fits the Cox proportional hazards model and its extensions.
- PROC LIFEREG is a parametric regression procedure for modeling the distribution of survival time with a set of concomitant variables.

SAS in Several Statistical Analyses

6.4) Survival Analysis (Cont)

- The Kaplan-Meier(*K-M*) survival curves and related tests (Log-Rank, Wilcoxon) can be generated using PROC LIFETEST.
- The Cox (proportional hazards) regression is performed using PROC PHREG.
- The accelerated failure time regression is performed using PROC LIFEREG.

Note:

https://support.sas.com/documentation/cdl/en/statug/63033/HTML/default/viewer.htm#phreg_toc.htm

References

- Geoff Der and Brian S. Everitt (2002). *A Handbook of Statistical Analyses using SAS*, Chapman & Hall/CRC.
- Mervyn G. Marasinghe, William J. Kennedy (2008). *SAS for Data Analysis: Intermediate Statistical Methods*. Springer Science+Business Media, LLC.

THANK YOU