INTRODUCTION TO SAS

JangDong Seo
Indiana University
Statistical Consulting Center

TOPICS

1. General understanding of SAS
2. Writing SAS programs
3. Reading data from external files
4. Data management
5. Descriptive statistics
6. Statistical analyses
7. Statistical graphics
8. Advanced SAS programming

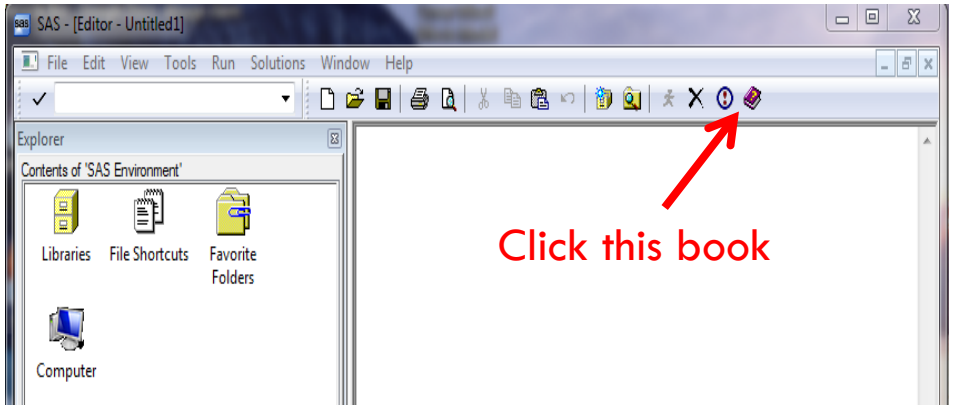
WHAT IS SAS?

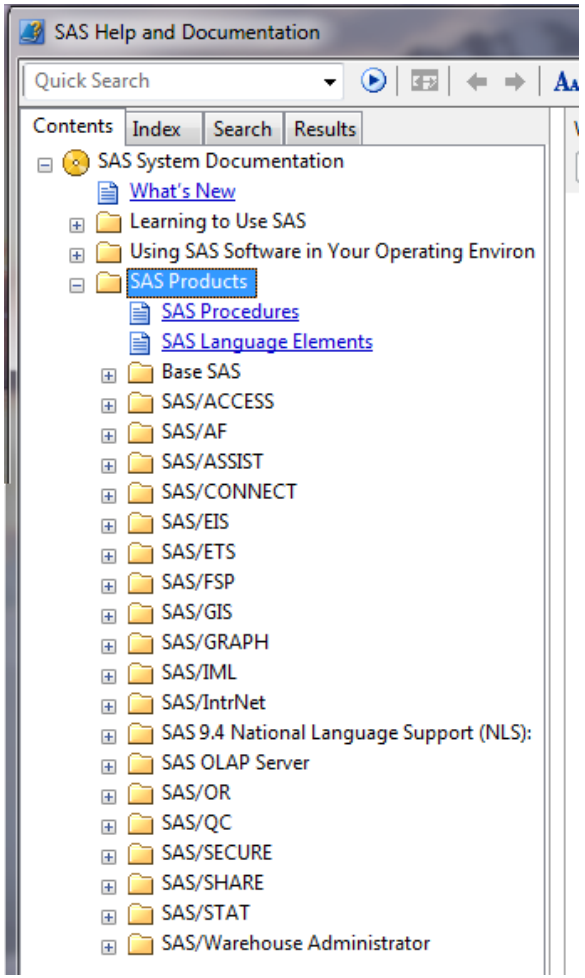
1. Statistical Analysis System
2. Can be used for
   - Data input and management
   - Producing reports & graphs
   - Statistical analysis
3. SAS is the *defacto* standard in industries for data management and analyses

RESOURCES

1. SAS Help and Documentation

   - Lists all of the procedures





2. Books written by users

3. Previously-written codes

WAYS TO USE SAS AT IU

1. Simple and FREE:

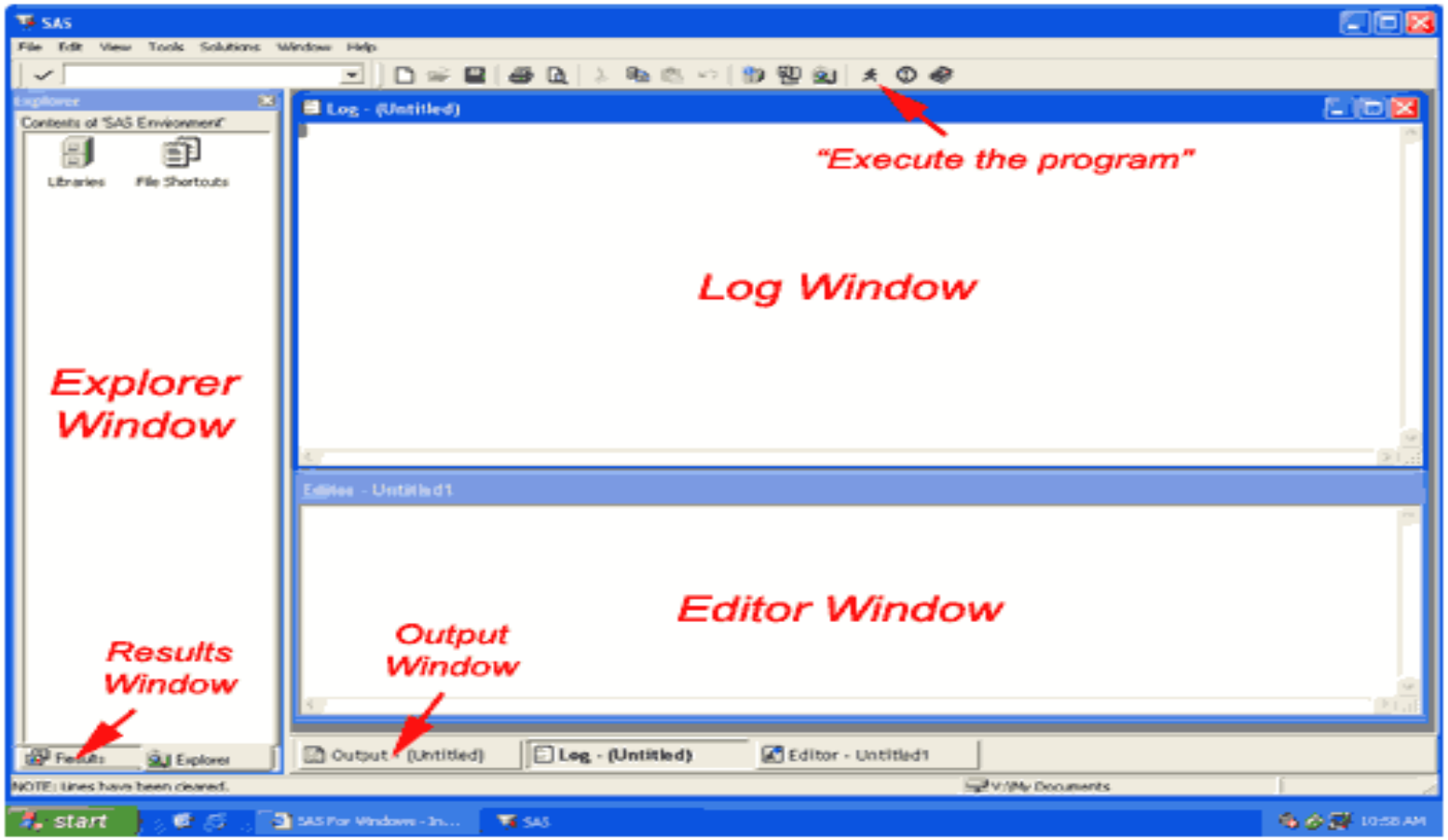
   https://iuanyware.iu.eduPersonal Installation

2. Personal installation:

   Contact Social Science Research Commons

   http://ssrc.indiana.edu/home/

OPENING SAS


"Execute the program"
Log Window
Explorer Window
Editor Window
Output Window
Results Window

IMPORTANT SAS WINDOWS

1. Editor – where you type your code
2. Log
   - Contains a summary of all operations performed by SAS during the execution of your program
   - Where to look to find coding mistakes
3. Output – displays output produced by PROC statements
4. Results – functions like a table of contents for your output
5. Explorer – gives easy access to your SAS files and libraries

THE ENHANCED EDITOR

Most important feature: color-coding scheme

Color	Program Element
Blue text (not bold)	SAS keyword
Navy Text (bold)	PROC name or DATA step
Purple text	string (anything surrounded by matching quotes)
Black text	normal text
Green text	comment
Red text	**INDICATES AN ERROR**
Yellow background	data lines

EXECUTING A PROGRAM

- Click on the "running man" icon at the top of the screen
- If you highlight a section of code, only that section will be run
- Alternatives
  - right-click and choose *Submit All* or *Submit Selection*
  - Use the Run menu

SOME COMMON ERRORS

1. **Missing semicolons**
2. Misspelled PROC statements or SAS keywords
   - Remember, they should be navy or blue in color
3. Misspelled variable names
   - Must match those specified in the DATA step
4. Unmatched quotes
   - This will be clearly visible in the Enhanced Editor as long blocks of purple text

FINDING & ADDRESSING ERRORS

1. Go to the top and look for the first error in the Log Window
   - It will indicate some program code
2. Look for that code in the Editor Window
3. Look for errors on that line and then in above lines
   - An error may not cause a problem until later in the program
   - Fix coding errors from top-to-bottom
   - Often fixing the first error will also fix later ones

SAVING SAS FILES

1. You should always save your program file
   - have extension .sas
2. Generally, you will not save your output
3. File saving tips
   - Use a sensible folder structure
   - Use meaningful program names

```
4     OPTIONS autocorrect; /*default*/
5     PROC FREQ DATE=idre.charities;
                ----
                1
WARNING 1-322: Assuming the symbol DATA was misspelled as DATE.
6           TABLE code;
7     RUN;

NOTE: Writing HTML Body file: sashtml.htm
NOTE: There were 12 observations read from the data set IDRE.CHARITIES.
NOTE: PROCEDURE FREQ used (Total process time):
      real time           4.36 seconds
      cpu time            0.20 seconds


8
9     OPTIONS noautocorrect;
10    PROC FREQ DATE=idre.charities;
                ----
                22
                202
ERROR 22-322: Syntax error, expecting one of the following: ;, COMPRESS, DATA, FC, FORMCHAR,
              NLEVELS, NOPRINT, ORDER, PAGE.
ERROR 202-322: The option or parameter is not recognized and will be ignored.
11          TABLE code;
12    RUN;

NOTE: The SAS System stopped processing this step because of errors.
NOTE: PROCEDURE FREQ used (Total process time):
      real time           0.01 seconds
      cpu time            0.01 seconds
```

WRITING SAS PROGRAMS

Examples

SAS PROGRAMS

1. Program: a mixture of *steps* and *statements*
2. Consider a paper/article as an analogy to a program
   - Step → a paragraph
   - Statement → a sentence

STEPS

1. A set of statements that produce some results when executed together
2. Two types of steps
   - **DATA** step – input & manipulate data sets
   - **PROC** step – (usually) do something with data sets

STATEMENTS

Statement – an instruction

- Begins with an alphabetic character and **ends with a semicolon**
- Some can be used anywhere in the program
  - E.g. TITLE, RUN, FILENAME
  - Called "global" statements
- Some can only be used inside a DATA or PROC step
  - E.g. BY, INPUT, MODEL

AN EXAMPLE PROGRAM

```
DM LOG 'CLEAR';
DM OUTPUT 'CLEAR';

DATA bodyfat;
    INPUT gender $ fatpercent;
    DATALINES;
F 22.0
F 21.0
M 13.3
M 19.0
M 31.0
;
RUN;

TITLE 'Body Fat Data for Fitness Program';
PROC PRINT;
    BY gender;
RUN;
```

Steps

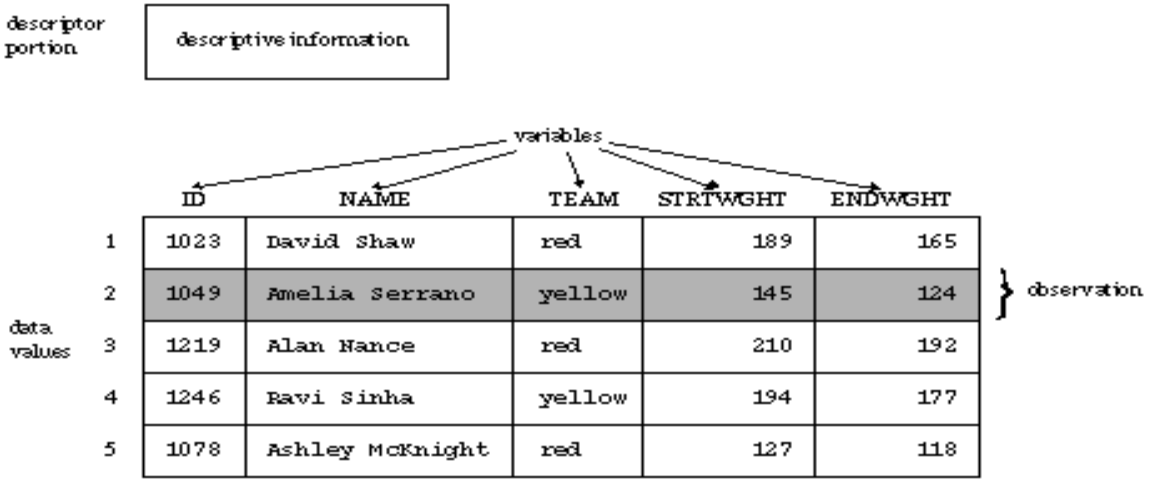
Must have at least one RUN statement at the end of every program

USING WHITE SPACE

1. Spaces, tabs, returns, etc. are ignored by SAS
2. Use white space to make your program more readable
3. Indent all lines below a DATA or PROC statement until the next step
   - Exception: avoid using tabs when typing in data values following a DATALINES or CARDS statement.
4. Returns (Enter Key in your keyboard)
   - Perhaps leave one blank row between each step
   - Separate distinct sections of code with several blank rows

DATA SETS

1. The fundamental data structure in SAS
2. Tabular structure in which rows = observations, columns = variables



THE DATA STEP

1. Usually the first step in a simple program
2. The most basic DATA step does two things:
   - **Create a new data set** with a specified name
   - **Read data into** a new data set
3. Required statements:
   - DATA statement – name the data set
   - INPUT statement – define the variables in the data set.
     - SAS defaults to numeric
     - Use $ to indicate the preceding variable is character in type
   - DATALINES statement
     - Signals SAS to begin reading data values on the next line

DATA STEP SYNTAX

```
DATA mydata;
   INPUT var1 var2 anothervar;
   DATALINES;
<data values>
;
```

Rules for naming data sets and variables

- must begin with a letter
- can contain up to 32 characters consisting of:
  - letters, numbers, and underscores (_)

PROC PRINT

1. Used to display the contents of a dataset in a tabular form
2. Statements
   - PROC PRINT statement – only required statement
   - VAR statement (optional) – specify which variables to display and in what order

COMPLETE SIMPLE PROGRAM

```
DATA fish;
   INPUT species sex weight length;
   DATALINES;
1 1 575 31.3
1 0 920 35
3 0 110 19.1
3 0 120 19.4
3 0 145 20.5
3 0 160 21.1
;
RUN;

PROC PRINT;
   VAR species length weight;
RUN;
```

Obs	species	length	weight
1	1	31.3	575
2	1	35.0	920
3	3	19.1	110
4	3	19.4	120
5	3	20.5	145
6	3	21.1	160

ANOTHER SIMPLE PROGRAM

```
DM LOG 'CLEAR';

DatA bodyfat;
   input gender $ fatpercent;
   DataLineS;
M 13.3
F 22.0
m 19.0
F 21.0
M 31.0
M 12.0
F 12.0
F 23.3
M 20.0
M 24.0
;
RuN;

proc print data=bodyfat;
run;
```

Specifies that gender is a character variable

Case matters here (entering data values)
Ex. $M \neq m$

Obs	gender	fatpercent
1	M	13.3
2	F	22.0
3	m	19.0
4	F	21.0
5	M	31.0
6	M	12.0
7	F	12.0
8	F	23.3
9	M	20.0
10	M	24.0

CREATING NEW VARIABLES (IN A DATA STEP)

Before CARDS (or DATALINES) place a statement of the form:

newvar = [*some expression*];

```
DATA bikerides;
     INPUT miles;
     km = miles*1.6;
     CARDS;
17
34
67
42
;

PROC PRINT;
RUN;
```

Transforms "miles" to "km"

Obs	miles	km
1	17	27.2
2	34	54.4
3	67	107.2
4	42	67.2

COMMENTS

1. Comments are sections of "code" that will be ignored at execution
   - Can include descriptive text in your program
2. Two ways to comment in SAS (will appear in green):
   - Comment a single line with an asterisk (*) and semicolon (;)
   - Embed a comment between /* and */ (block comment)

```
This example uses the *message; format:
* This code finds the number in the BY group;

This example uses the *message; format:
   *----------------------------------------*
   |  This uses one comment statement       |
   |           to draw a box.               |
   *----------------------------------------*;

This example uses the /*message*/ format:
 input @1 name $20.   /* last name    */
       @200 test 8.   /* score test   */
       @50 age 3.;    /* customer age */
```

READING DATA FROM EXTERNAL FILES

Text, csv, and Excel files

READING TEXT FILES

1. The DATA step can read directly from ASCII (text) files
   - usually have extensions .dat or .txt
2. Must use an INFILE statement in the DATA step

Example:

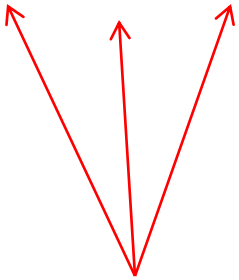
```
INFILE "E:\jangseo\data\airplanes.txt";
```

Location File name

CARS EXAMPLE

```
DATA mileage;
        INFILE 'E:\jangseo\data\cars.dat';
        INPUT model hp mpg;
RUN;

PROC PRINT;
RUN;
```



Variable names

Obs	model	hp	mpg
1	1	155	31
2	1	165	28
3	1	115	30
4	1	170	30
5	1	220	28
6	1	235	26
7	3	275	20
8	3	210	20
9	3	282	16
10	3	220	20

FILENAME STATEMENT

1. Purpose: define *a file reference name* for an external file
2. Use: specify both
   - An "alias" for the file – can be any SAS name
   - The path to the data file
3. Must **precede the DATA step**
4. SYNTAX:

   FILENAME <*fileref*> 'car.dat';

   e.g. `FILENAME cars 'F:\car.dat';`

FILENAME EXAMPLE

Fileref name for cars data

```
filename cars 'F:\cars.dat';
data mileage;
        infile cars;
        input model hp mpg;
proc print;
run;
```

Obs	model	hp	mpg
1	1	155	31
2	1	165	28
3	1	115	30
4	1	170	30
5	1	220	28
6	1	235	26
7	3	275	20
8	3	210	20
9	3	282	16
10	3	220	20

SUGGESTED STRATEGY

1. Use the FILENAME statement rather than including a path in the INFILE statement
2. Place all FILENAME statements at the top of the program
3. Reason: Suppose you have to change the path to a data file
   - Only need to modify one line of code
   - Always know where to do so

USEFUL INFILE OPTIONS

DELIMITER (DLM) – Specify a character that separates data values in the file

- DATA step assumes data are "space"-delimited
- Other common ASCII delimiters are:

Delimiter	Syntax
comma	DLM = ','
tab	DLM = '09'x
"pipe" symbol (|)	DLM = '|'

READING CSV FILES (COMMA DELIMITED)

```
filename rowers 'C:\rowing.dat';
data row;
        infile rowers delimiter=',';
        input wt school $ ;
run;
```

1	189	Yale
2	183	Yale
3	195	Yale
4	185	Yale
5	214	Yale
6	204	Yale
7	186	Yale
8	179	Yale
9	109	Yale
10	186	Oxford
11	185	Oxford
12	204	Oxford
13	185	Oxford
14	196	Oxford
15	203	Oxford
16	174	Oxford
17	183	Oxford
18	110	Oxford

IMPORTING OTHER FILES (EXCEL)

1. Can use **PROC IMPORT** to read data from other types of files
2. Current versions of SAS can read Excel files in file types, such as .xls, .xlsb, .xlsm, and .xlsx
3. Data must be in readable format
   - rows for observations, columns for variables
   - can have *varnames* in first row

IMPORT FROM EXCEL - SYNTAX

```
PROC IMPORT
     DATAFILE='filename.xls'
     OUT=mydata
     DBMS=xls
     REPLACE;
   GETNAMES = NO | YES;
RUN;
```

1. DBMS may not be necessary if SAS can determine it from the file extension (.xls)
2. GETNAMES (optional) - if 'yes', SAS uses values in the first row as variable names. Make sure the names are allowable in SAS

oldfaithful.xls – Excel 2003 File

```
PROC IMPORT datafile='data\oldfaithful.xls'
        OUT=erupt DBMS=xls REPLACE;
        GETNAMES=yes;
run;
proc print data=erupt (obs=8);
run;
```

Print only first 8 observations

Example 5 - Importing an Excel File

Obs	DATE	DURATION	INTTIME
1	1	4.4	78
2	1	3.9	74
3	1	4	68
4	1	4	76
5	1	3.5	80
6	1	4.1	84
7	1	2.3	50
8	1	4.7	93

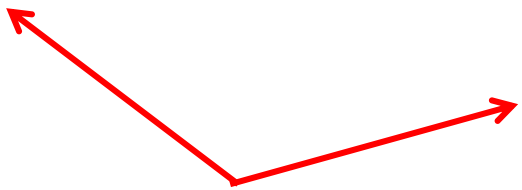
USING ONLY A SUBSET OF DATA

1. To restrict the data being read → add a "subsetting **IF**" or "subsetting **WHERE**" statement to a **DATA** step
2. To restrict the data used by a PROC → add a **WHERE** statement to the **PROC**

SUBSETTING IF

1. Use in a DATA step
   - Any logical expression can be used as the condition
2. Only cases that meet the condition will be read into the data set

```
data row2;
   infile rowers dlm=',';
   input wt school $;
   if wt < 150;
run;
proc print;
run;
```



Only obs that match

Subsetting IF and WHERE Statement

Obs	wt	school
1	109	Yale
2	110	Oxford

WHERE STATEMENT

1. Can be used in PROC or Data Steps
2. Only runs the procedure on matching values
3. Does not affect dataset contents

```
** Original data set (18 obs);
proc print data=row;
        where wt < 150;
run;
```

```
Subsetting IF and WHERE Statement

Obs      wt     school

  9     109     Yale
 18     110     Oxford
```

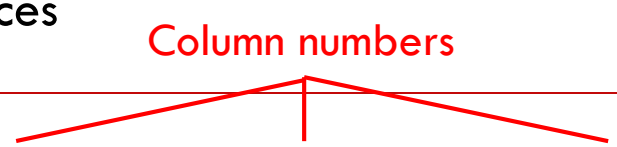
LIST, COLUMN, & FORMATTED INPUT

In all earlier examples, data were read using *list input*

- Values were separated by spaces
- When reading a value, SAS reads digits (or characters) **from left to right** until a space is encountered

COLUMN INPUT

1. Used for "**fixed-width**" data
   - data values are in the same columns for every row
2. Can skip columns, reread columns into a different variable, read character values that include spaces

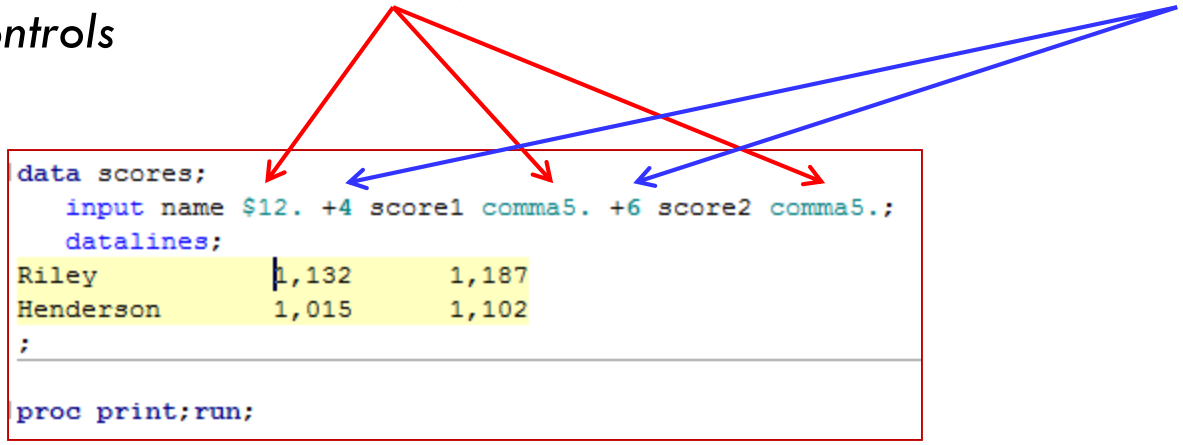



```
data scores;
   input name $ 1-12 score1 27-30 score2 17-20 ;
   datalines;
J Riley          1132        987
E Henderson      1015       1102;
*---+----1----+----2----+----3----+----4;
```

Obs	name	score1	score2
1	J Riley	987	1132
2	E Henderson	1102	1015

FORMATTED INPUT

1. Specify the format of input data, including:
   - the data type (numeric or character)
   - width (# of columns)
   - number of decimal places (for numeric data)
2. Input formats (*informats*) are often used in conjunction with *pointer controls*



```
data scores;
   input name $12. +4 score1 comma5. +6 score2 comma5.;
   datalines;
Riley           1,132      1,187
Henderson       1,015      1,102
;

proc print;run;
```

EXAMPLE

```
data scores;
   input @1 name $12. @15 score1 @25 score2 @35 gpa 3.2;
   datalines;
J Riley         1132        987       327
E Henderson     1015       1102       265
;
proc print;
run;
```

Obs	name	score1	score2	gpa
1	J Riley	1132	987	3.27
2	E Henderson	1015	1102	2.65

OTHER INFORMATS

- Consider reading dates

```
Wyatt Earp              1-13-1929 80
William H Bonney        7-14-1887 21
Doc Holliday           11-08-1887 36
```

1. One method: treat the date as a character value

```
input name $18. +1 death $10. +1 age 3.;
```

2. Another method: use a date *informat*

```
input name $18. +1 death MMDDYY10. +1 age 3.;
```

DATES: METHOD 1 (OK)

```
data west;
  input name $18. +1 death $10. +1 age 3.;
  cards;
Wyatt Earp           1-13-1929 80
William H Bonney     7-14-1887 21
Doc Holliday        11-08-1887 36
;
proc print data=west;
   title 'Famous Wild West Figures';
run;
```

Death is read as a character, instead of a date

You should convert it to a date later, if needed

```
Wild West Figures

Obs     name                    death        age

 1      Wyatt Earp            1-13-1929       80
 2      William H Bonney      7-14-1887       21
 3      Doc Holliday          11-08-1887      36
```

DATES: METHOD 2 (BETTER)

```
data west;
  input name $18. +1 death MMDDYY10. +1 age 3.;
  cards;
Wyatt Earp           1-13-1929 80
William H Bonney     7-14-1887 21
Doc Holliday        11-08-1887 36
;
proc print;
   title 'Famous Wild West Figures';
run;
```

SAS reads death as a date, but saves it as a numeric value

Dates are stored internally in SAS as the number of days from Jan 1, 1960

Negative days mean they were born before 1960

```
Wild West Figures

Obs     name                 death     age

 1      Wyatt Earp           -11310     80
 2      William H Bonney     -26468     21
 3      Doc Holliday         -26351     36
```

FORMATS

1. *Informats* affect **how SAS reads in data**
   - You do not have to use the *informat* statement
2. We can also assign *formats* to variables
   - Change the way data are displayed
   - You must tell SAS (i.e, need to write *format*) **how to display the data values**

FORMAT STATEMENT

```
DATA west;
   INPUT name $18. +1 death MMDDYY10. +1 age 3.;
   FORMAT death worddate18.;
   CARDS;
Wyatt Earp           1-13-1929 80
William H Bonney     7-14-1887 21
Doc Holliday        11-08-1887 36
;

PROC PRINT DATA=west;
```

Obs	name	death	age
1	Wyatt Earp	January 13, 1929	80
2	William H Bonney	July 14, 1887	21
3	Doc Holliday	November 8, 1887	36

NUMBER FORMATS

- 6 or 5 is the total number of digits.
- .2 is the number of decimal places

```
DATA numbers;
     INPUT a b c d e f;
     FORMAT b comma.  c 6.3  d 6.2  e dollar6.2 f dollar5.2;
     CARDS;
1500 1500 23.635 23.635 23.635 23.635
;

PROC PRINT;
```

$ is missing since 5 is not enough

Obs	a	b	c	d	e	f
1	1500	1,500	23.635	23.64	$23.64	23.64

DATA MANAGEMENT

Set, drop, keep, merging files, sorting, format, and creating excel files

SET STATEMENT

- It is common to use several datasets in one SAS program
- Use a SET statement in a DATA step to **read in an existing dataset**
- SET is often used in conjunction with a subsetting IF

SET EXAMPLE

Create a new dataset that will contain all observations from the dataset "row" in which the school was “Yale”.

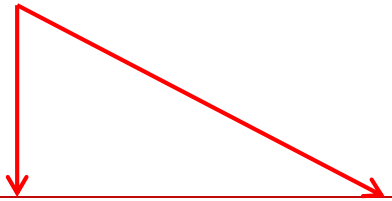
```
data yale;
   set row;
   if school='Yale';
proc print data=yale;
run;
```

Obs	wt	school
1	214	Yale
2	204	Yale
3	195	Yale
4	189	Yale
5	186	Yale
6	185	Yale
7	183	Yale
8	179	Yale
9	109	Yale

DROP & KEEP STATEMENTS

Sometimes we have variables in a dataset that we no longer need

- Consider a data set containing scores from Judges 1 through 5 on some contest
- Maybe we want to throw out scores from Judges 1 and 3



Name	Age	Type	Date	Score1	Score2	Score3	Score4	Score5
Alicia Grossman	13	c	10/28/03	7.8	6.5	7.2	8.0	7.9
Matthew Lee	9	D	10/30/03	6.5	5.9	6.8	6.0	8.1
Elizabeth Garcia	10	C	10/29/03	8.9	7.9	8.5	9.0	8.8
Lori Newcombe	6	D	10/30/03	6.7	5.6	4.9	5.2	6.1
Jose Martinez	7	d	10/31/03	8.9	9.5	10.0	9.7	9.0
Brian Williams	11	C	10/29/03	7.8	8.4	8.5	7.9	8.0

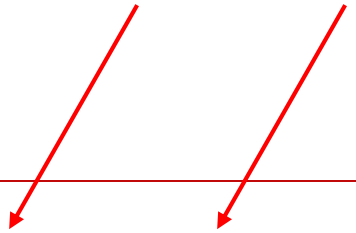
DROP & KEEP STATEMENTS

- SYNTAX:
  - DROP *variable list;*
  - KEEP *variable list;*
- Can only use DROP or KEEP (not both).
- To save typing:
  - Use KEEP if you are dropping more variables than you're keeping.
  - Use DROP if you are keeping more than you are dropping

EXAMPLE: DROP

Let's get rid of the unneeded variables from the *contest* dataset.

```
DATA contest2;
   SET contest;
   DROP score1 score3;
RUN;
PROC PRINT DATA=contest2;
RUN;
```



Name	Age	Type	Date	Score2	Score4	Score5
Alicia Grossman	13	c	10/28/03	6.5	8.0	7.9
Matthew Lee	9	D	10/30/03	5.9	6.0	8.1
Elizabeth Garcia	10	C	10/29/03	7.9	9.0	8.8
Lori Newcombe	6	D	10/30/03	5.6	5.2	6.1
Jose Martinez	7	d	10/31/03	9.5	9.7	9.0
Brian Williams	11	C	10/29/03	8.4	7.9	8.0

COMBINING DATASETS

- We'll focus on two methods
  1. Concatenating (stacking)
  2. Match Merging (joining side-by-side)
- Example: suppose you have two datasets with some university ratings, each from different years

Dataset "school2005"

school	rating	cost	year
IUP	85	mid	2005
IUB	90	cheap	2005
Earlham	86	pricey	2005

Dataset "school2006"

school	rating	year
IUP	89	2006
IUB	83	2006
Earlham	85	2006

CONCATENATING DATASETS

- Add observations from one dataset onto another.
  - Combine "**vertically**"
  - Use the SET statement to concatenate datasets

```
data stacked;
  set school2005 school2006;
  run;

proc print;
  run;
```

Obs	school	rating	cost	year
1	IUP	85	mid	2005
2	IUB	90	cheap	2005
3	Earlham	86	pricey	2005
4	IUP	89		2006
5	IUB	83		2006
6	Earlham	85		2006

MATCH MERGING

- **Add variables** to observations that already exist in both datasets.
- Observations are matched on an "ID variable"
  - one that is common to both datasets and unique to each observation

MATCH-MERGED

ID	*Test1*
4112	87
3285	75
4075	77
3985	98
3222	90

ID	*Test2*
4112	85
3285	82
4075	78
3985	94
3222	91

ID	*Test1*	*Test2*
4112	87	85
3285	75	82
4075	77	78
3985	98	94
3222	90	91

MATCH-MERGE STEPS

1. First, must sort both datasets by the ID variable
2. Then use both a MERGE statement and a BY statement in a DATA step

Example data sets (named *midterm* & *final*)

Midterm Scores

Obs	ID	Test1
1	4112	87
2	3285	75
3	4075	77
4	3985	98
5	4006	82
6	3222	90

Final Scores

Obs	ID	Test2
1	4112	85
2	3285	82
3	4075	78
4	3985	94
5	3222	91

Different variable names

EXAMPLE: MERGE

```
proc sort data=midterm;
   by id;
proc sort data=final;
   by id;
run;
data combined;
   merge midterm final;
   by id;
run;

proc print;
   title 'Scores on Both Exams';
run;
```

Scores on Both Exams

Obs	id	Test1	Test2
1	3222	90	91
2	3285	75	82
3	3985	98	94
4	4006	82	.
5	4075	77	78
6	4112	87	85

SORTING A DATA SET

1. Must use PROC SORT
2. Syntax:

PROC SORT <DATA=*dataset*>
  BY <*DESCENDING*> *varlist;*
RUN;

Asscending is the default

```
title 'Sorted Rowing Data';
PROC SORT DATA=row out = row4
          BY wt;
RUN;
PROC PRINT DATA=row4;
RUN;
```

Sorted Rowing Data

Obs	wt	school
1	109	Yale
2	110	Oxford
3	174	Oxford
4	179	Yale
5	183	Yale
6	183	Oxford
7	185	Yale
8	185	Oxford
9	185	Oxford
10	186	Yale
11	186	Oxford
12	189	Yale
13	195	Yale
14	196	Oxford
15	203	Oxford
16	204	Yale
17	204	Oxford
18	214	Yale

```
title 'Rowers Sorted by School & Weight
(Desc)';
PROC SORT DATA=row out = row4;
          BY school DESCENDING wt;
RUN;
PROC PRINT DATA=row4;
RUN;
```

Rowers Sorted by School & Weight (Desc)

Obs	school	wt
1	Oxford	204
2	Oxford	203
3	Oxford	196
4	Oxford	186
5	Oxford	185
6	Oxford	185
7	Oxford	183
8	Oxford	174
9	Oxford	110
10	Yale	214
11	Yale	204
12	Yale	195
13	Yale	189
14	Yale	186
15	Yale	185
16	Yale	183
17	Yale	179
18	Yale	109

BY STATEMENT IN PROC PRINT

1. Print separate tables for groups defined by the *byvar*
2. Data must first be sorted by that *byvar*

```
proc print data=row3;
  by school;
  var wt;
run;
```

Works because data were sorted by school

school=Oxford

Obs	wt
1	204
2	203
3	196
4	186
5	185
6	185
7	183
8	174
9	110

school=Yale

Obs	wt
10	214
11	204
12	195
13	189
14	186
15	185
16	183
17	179
18	109

LABELING CATEGORICAL VALUES

- Good practice for statistical computing:
  - Always use numerical values
  - Assign labels to categorical values
- Consider the following DATA step:

```
DATA bodyfat;
   INPUT gender $ fatpercent;
   DATALINES;
1 13.3
2 22.0
1 19.0
2 21.0
1 31.0
;

proc freq;
tables gender/nocum;
run;
```

The FREQ Procedure

gender	Frequency	Percent
1	3	60.00
2	2	40.00

You have to remember what 1 or 2 means.

You can, but I can't

LABELING CATEGORICAL VALUES

- We want to use numbers to represent Male and Female
  - 1 = Male
  - 2 = Female

- Must use two steps
  1. Use PROC FORMAT to create a custom format
  2. Assign the format to a variable in a DATA step or a PROC

PROC FORMAT

Syntax:

PROC FORMAT;

VALUE *fmtname* 1='label1' 2='label2' <...>;

RUN;

- This will only define a new format
- You must still assign it to a variable in a DATA step

PROC FORMAT EXAMPLE

Define Format named "gendf"

Assign format to gender

Numeric values

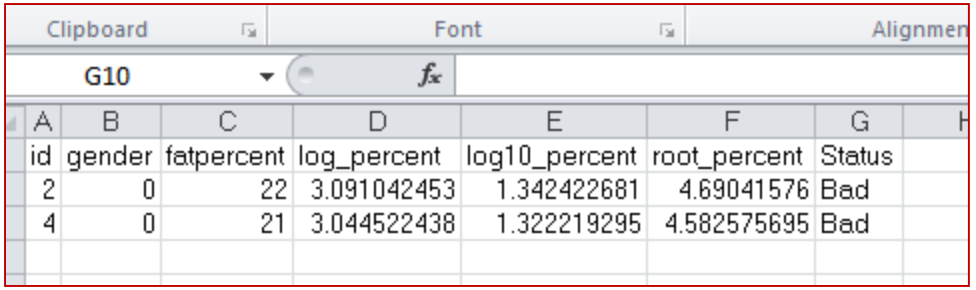
```
PROC FORMAT;
  VALUE gendf 1='Male' 2='Female';
RUN;
DATA bodyfat;
  INPUT gender fatpercent;
  FORMAT gender gendf.;
  DATALINES;
1 13.3
2 22.0
1 19.0
2 21.0
1 31.0
;
PROC PRINT DATA=bodyfat;
RUN;
```

Obs	gender	fatpercent
1	Male	13.3
2	Female	22.0
3	Male	19.0
4	Female	21.0
5	Male	31.0

EXPORTING AS EXCEL (PROC EXPORT)

Finally you want to save your work in an Excel file

```
PROC EXPORT DATA= WORK.MaleOnly
            OUTFILE= "C:\MaleOnly.xls"
            DBMS=EXCEL REPLACE;
     SHEET="maleOnly";
RUN;
```



Clipboard | Font | Alignmen

G10

	A	B	C	D	E	F	G
	id	gender	fatpercent	log_percent	log10_percent	root_percent	Status
	2	0	22	3.091042453	1.342422681	4.69041576	Bad
	4	0	21	3.044522438	1.322219295	4.582575695	Bad

Example:
Table 1 – Sample characteristics

Sample characteristics	Women	Men
Number of subjects, N (%)	219 (47.7)	240 (52.3)
Age, y (mean ± SD)	57.3 ± 14.0	60.8 ± 13.3
BMI, kg/m^2 (mean ± SD)	24.7 ± 4.0	25.2 ± 3.0
Overweight, BMI 25-29.99, N (%)	68 (31.1)	102 (42.5)
Obese, BMI ≥ 30, N (%)	23 (10.5)	12 (5.0)
Waist circumference, cm (mean ± SD)	77.2 ± 9.4	90.4 ± 8.9
Physical activity, kcal/kg (mean ± SD)	15.2 ± 3.9	14.7 ± 3.8
Smoking status, N (%)		
Never smoker	168 (77.0)	181 (75.4)
Current smoker	23 (10.6)	22 (9.2)
Former smoker	27 (12.4)	37 (15.4)
Race/Ethnicity, N (%)		
White	206 (94.1)	231 (96.2)
African American	13 (5.9)	9 (3.8)
Vitamin users, N (%)	124 (56.6)	94 (39.2)

(c) 2004, P.K. Newby, Sc.D., M.P.H.

DESCRIPTIVE STATISTICS

Proc univariate
Proc means
Proc freq

PROC UNIVARIATE

- Generates descriptive statistics for quantitative variables, including:
  - Moments
  - Measures of Center & Variability
  - Quantiles
- Can also request
  - Tests of normality (option: NORMAL)
  - Stem-and-leaf plot, boxplot, & Normality plots (option: PLOT)
  - Histogram
  - Qqplot

PROC UNIVARIATE SYNTAX

PROC UNIVARIATE DATA=*dataset* <*options*>;

BY *variables*;

VAR *variables*;

ID *variables*;

HISTOGRAM;

Two options are:

- normal – compute tests of normality
- plot – produce stem/leaf and box-plots

PROC UNIVARIATE

```
proc univariate data=hw;
id student;
run;
```

- BY produces separate analyses for each BY group.
- VAR statement specifies the variables to be analyzed.
- ID shows IDs for extreme observations for the variables

Extreme Observations					
Lowest			Highest		
Value	student	Obs	Value	student	Obs
125	1023	1	125	1023	1
156	1049	2	156	1049	2
164	1219	3	164	1219	3
182	1246	4	182	1246	4
194	1078	5	194	1078	5

PROC UNIVARIATE EXAMPLE

```
proc univariate data=hw;
    var hw1;
    id student;
    histogram;
run;
```

The UNIVARIATE Procedure
Variable: hw1 (HW1)

Moments			
N	5	Sum Weights	5
Mean	72.2	Sum Observations	361
Std Deviation	32.6527181	Variance	1066.2
Skewness	-0.8630297	Kurtosis	-1.1754824
Uncorrected SS	30329	Corrected SS	4264.8
Coeff Variation	45.2253713	Std Error Mean	14.6027395

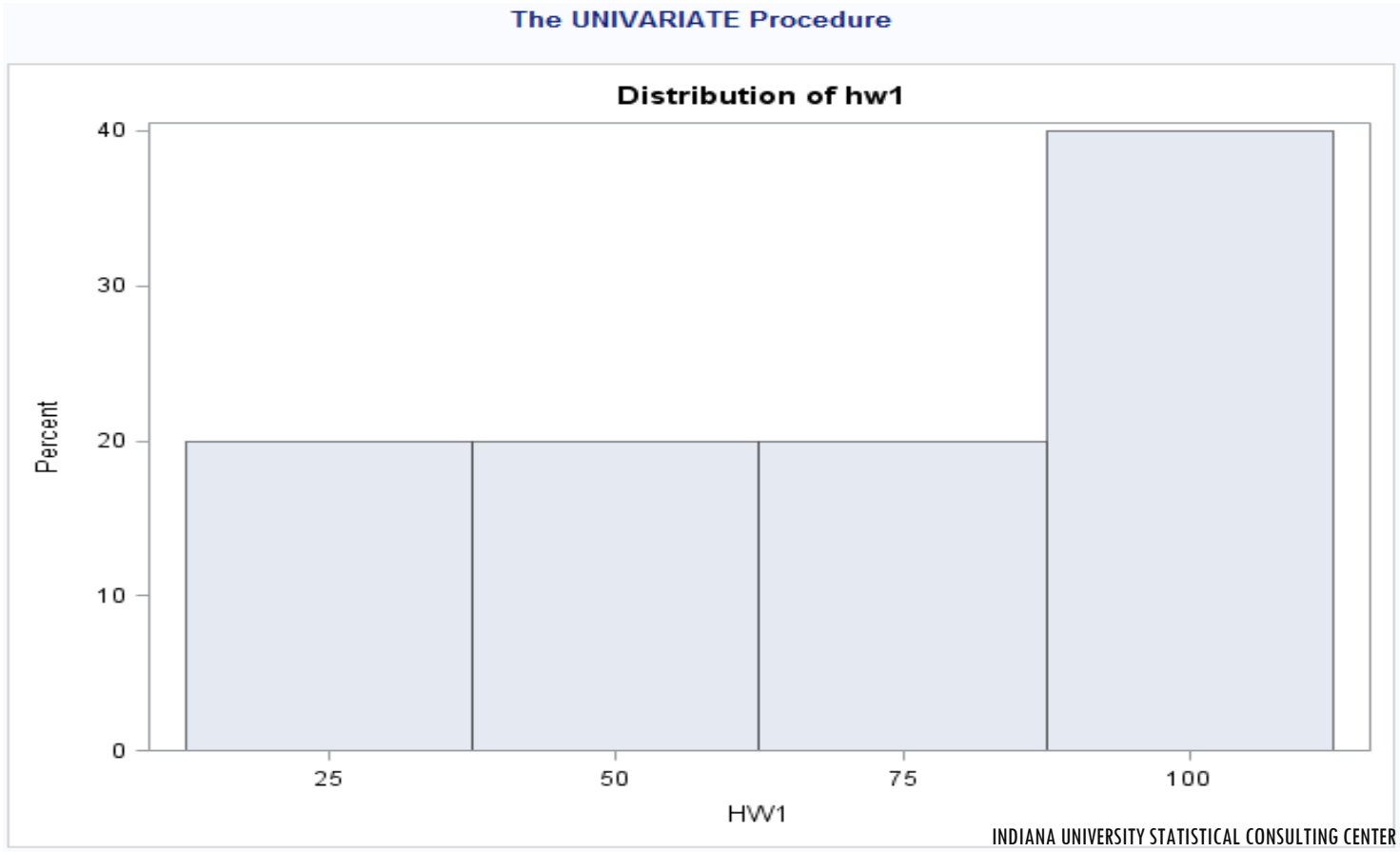
Basic Statistical Measures			
Location		Variability	
Mean	72.20000	Std Deviation	32.65272
Median	86.00000	Variance	1066
Mode	.	Range	75.00000
		Interquartile Range	46.00000

UNIVARIATE EXAMPLE

Tests for Location: Mu0=0				
Test	Statistic		p Value	
Student's t	t	4.944278	Pr > |t|	0.0078
Sign	M	2.5	Pr >= |M|	0.0625
Signed Rank	S	7.5	Pr >= |S|	0.0625

Quantiles (Definition 5)	
Quantile	Estimate
100% Max	100
99%	100
95%	100
90%	100
75% Q3	98
50% Median	86
25% Q1	52
10%	25
5%	25
1%	25
0% Min	25

RESULT OF HISTOGRAM STATEMENT



UNIVARIATE + BY

Can use a BY statement to run UNIVARIATE separately on different levels of a BY variable.

- PLOT option will produce comparative boxplot
- Data must be sorted in advance by the by variable (if BY is used)

```
TITLE 'Bodyfat Data: Univariate Statistics By Gender';
PROC SORT DATA=bodyfat;
   BY gender;
RUN;
PROC UNIVARIATE DATA=bodyfat PLOT;
   BY gender;
RUN;
```

Bodyfat Data: Univariate Statistics By Gender

---------- gender=F ----------

The UNIVARIATE Procedure
Variable: fatpercent

Moments

N	4	Sum Weights	4
Mean	19.575	Sum Observations	78.3
Std Deviation	5.13703871	Variance	26.3891667
Skewness	-1.7990141	Kurtosis	3.36690795
Uncorrected SS	1611.89	Corrected SS	79.1675
Coeff Variation	26.2428542	Std Error Mean	2.56851935

Bodyfat Data: Univariate Statistics By Gender

---------- gender=M ----------

The UNIVARIATE Procedure
Variable: fatpercent

Moments

N	6	Sum Weights	6
Mean	19.8833333	Sum Observations	119.3
Std Deviation	7.02578584	Variance	49.3616667
Skewness	0.57752663	Kurtosis	-0.1306045
Uncorrected SS	2618.89	Corrected SS	246.808333
Coeff Variation	35.3350503	Std Error Mean	2.86826506

PROC MEANS

- Produce summary statistics
- Basic syntax:

```
PROC MEANS DATA=dataset options;
  CLASS classification-var;
  VAR variable-list;
RUN;
```

- Default output for a dataset
  - no options and no CLASS or VAR statements:

The MEANS Procedure

Variable	N	Mean	Std Dev	Minimum	Maximum
Age	19	13.3157895	1.4926722	11.0000000	16.0000000
Height	19	62.3368421	5.1270752	51.3000000	72.0000000
Weight	19	100.0263158	22.7739335	50.5000000	150.0000000

STATISTICS REQUESTS

- You can request particular statistics in the portion of the PROC MEANS statement.
- Some statistics include:

MAX	Maximum
MIN	Minimum
MEAN	
MEDIAN	
N	Number of non-missing values
NMISS	Number of missing values
Q1,Q3, P10, P95, …	Quartiles & Percentiles
RANGE	Skewness
SKEW	Standard Deviation (sample formula)
STD	
SUM	

PROC MEANS EXAMPLE

```
proc means data=sashelp.class mean median std q1 q3;
var weight height;
run;
```

The MEANS Procedure

Variable	Mean	Median	Std Dev	Lower Quartile	Upper Quartile
Weight	100.0263158	99.5000000	22.7739335	84.0000000	112.5000000
Height	62.3368421	62.8000000	5.1270752	57.5000000	66.5000000

CLASS STATEMENT

The CLASS statement requests separate analyses for each level of a *classification variable*.

```
proc means data=sashelp.class mean median std q1 q3;
class sex;
var weight height;
run;
```

The MEANS Procedure

Sex	N Obs	Variable	Mean	Median	Std Dev	Lower Quartile	Upper Quartile
F	9	Weight	90.1111111	90.0000000	19.3839137	84.0000000	102.5000000
		Height	60.5888889	62.5000000	5.0183275	56.5000000	64.3000000
M	10	Weight	108.9500000	107.2500000	22.7271864	85.0000000	128.0000000
		Height	63.9100000	64.1500000	4.9379370	59.0000000	67.0000000

PROC FREQ

- Produce summary statistics
- Basic syntax:

PROC FREQ <options> ;
BY variables ;
TABLES requests </ options> ;
RUN;

- Chi-square test, measure of association, binomial proportion, etc. can be performed

PROC FREQ EXAMPLE

The eye and hair color of children from two different regions of Europe are recorded in the data set *Color*.

Variable label

```
data Color;
    input Region Eyes $ Hair $ Count @@;
     label Eyes  ='Eye Color'
           Hair  ='Hair Color'
           Region='Geographic Region';
      datalines;
1 blue  fair   23  1 blue  red     7  1 blue  medium 24
1 blue  dark   11  1 green fair   19  1 green red     7
1 green medium 18  1 green dark   14  1 brown fair   34
1 brown red     5  1 brown medium 41  1 brown dark   40
1 brown black   3  2 blue  fair   46  2 blue  red    21
2 blue  medium 44  2 blue  dark   40  2 blue  black   6
2 green fair   50  2 green red    31  2 green medium 37
2 green dark   23  2 brown fair   56  2 brown red    42
2 brown medium 53  2 brown dark   54  2 brown black  13
;

proc freq data=Color;
    tables Eyes Hair Eyes*Hair;
    weight Count;
    title 'Eye and Hair Color of European Children';
run;
```

A frequency table for each of Eyes and Hair

A table for Eyes by Hair

The FREQ Procedure

Eye Color

Eyes	Frequency	Percent	Cumulative Frequency	Cumulative Percent
blue	222	29.13	222	29.13
brown	341	44.75	563	73.88
green	199	26.12	762	100.00

Hair Color

Hair	Frequency	Percent	Cumulative Frequency	Cumulative Percent
black	22	2.89	22	2.89
dark	182	23.88	204	26.77
fair	228	29.92	432	56.69
medium	217	28.48	649	85.17
red	113	14.83	762	100.00

"/chisq" can be added in TABLE statement to perform Chi-square Test or Goodness-of-Fit Test

Table of Eyes by Hair

Eyes(Eye Color) Hair(Hair Color)

Frequency Percent Row Pct Col Pct	black	dark	fair	medium	red	Total
blue	6 0.79 2.70 27.27	51 6.69 22.97 28.02	69 9.06 31.08 30.26	68 8.92 30.63 31.34	28 3.67 12.61 24.78	222 29.13
brown	16 2.10 4.69 72.73	94 12.34 27.57 51.65	90 11.81 26.39 39.47	94 12.34 27.57 43.32	47 6.17 13.78 41.59	341 44.75
green	0 0.00 0.00 0.00	37 4.86 18.59 20.33	69 9.06 34.67 30.26	55 7.22 27.64 25.35	38 4.99 19.10 33.63	199 26.12
Total	22 2.89	182 23.88	228 29.92	217 28.48	113 14.83	762 100.00

STATISTICAL ANALYSES

Correlation
Chi-square Test
T-Test
One-Way ANOVA
Two-Way ANOVA
Repeated-Measures ANOVA
ANCOVA
Simple Linear Regression
Multiple Linear Regression
Linear Mixed-Effects Model

CORRELATION USING PROC CORR

Correlation analysis is used to find the statistical relationship between two variables

Let's find the relationship between weight and height

```
proc corr data=sashelp.class;
var height weight;
run;
```

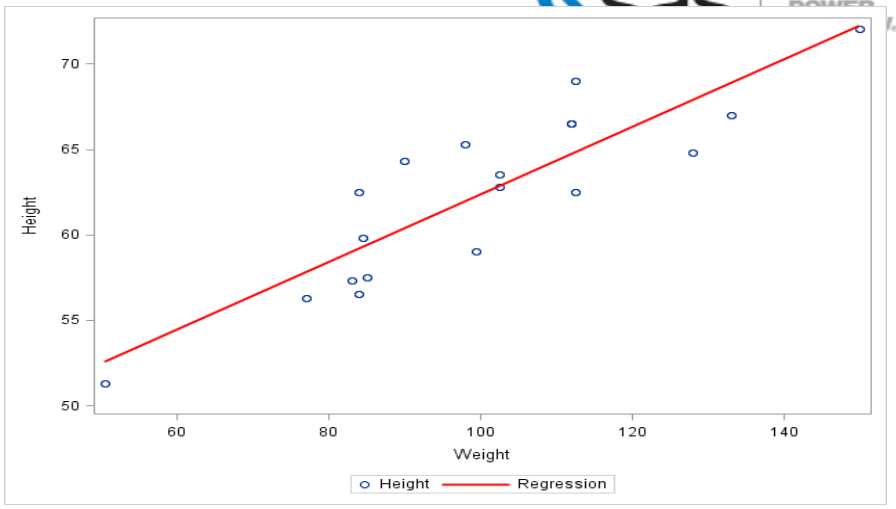
Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69.0	112.5
2	Alice	F	13	56.5	84.0
3	Barbara	F	13	65.3	98.0
4	Carol	F	14	62.8	102.5
5	Henry	M	14	63.5	102.5
6	James	M	12	57.3	83.0
7	Jane	F	12	59.8	84.5
8	Janet	F	15	62.5	112.5
9	Jeffrey	M	13	62.5	84.0
10	John	M	12	59.0	99.5
11	Joyce	F	11	51.3	50.5
12	Judy	F	14	64.3	90.0
13	Louise	F	12	56.3	77.0
14	Mary	F	15	66.5	112.0
15	Philip	M	16	72.0	150.0
16	Robert	M	12	64.8	128.0
17	Ronald	M	15	67.0	133.0
18	Thomas	M	11	57.5	85.0
19	William	M	15	66.5	112.0

The CORR Procedure

2 Variables:	Height Weight

Simple Statistics

Variable	N	Mean	Std Dev	Sum	Minimum	Maximum
Height	19	62.33684	5.12708	1184	51.30000	72.00000
Weight	19	100.02632	22.77393	1901	50.50000	150.00000



Descriptive stats

Pearson Correlation Coefficients, N = 19
Prob > |r| under H0: Rho=0

	Height	Weight
Height	1.00000	0.87779 <.0001
Weight	0.87779 <.0001	1.00000

Default is Pearson Correlation. Spearman' rho and Kendall's tau can be computed

Pearson correlation is 0.88

P-value

CHI-SQUARE TEST USING PROC FREQ

Chi-square test can be used to check the association between two categorical variables

Situation:

A researcher wants to test if there is an association between eye color and hair color

```
proc freq data=Color order=data;
    tables Eyes*Hair / expected norow nocol chisq
                       plots(only)=deviationplot;
    weight Count;
run;
```

Obs	Region	Eyes	Hair	Count
1	1	blue	fair	23
2	1	blue	red	7
3	1	blue	medium	24
4	1	blue	dark	11
5	1	green	fair	19
6	1	green	red	7
7	1	green	medium	18
8	1	green	dark	14
9	1	brown	fair	34
10	1	brown	red	5
11	1	brown	medium	41
12	1	brown	dark	40
13	1	brown	black	3
14	2	blue	fair	46
15	2	blue	red	21
16	2	blue	medium	44
17	2	blue	dark	40
18	2	blue	black	6
19	2	green	fair	50
20	2	green	red	31
21	2	green	medium	37
22	2	green	dark	23
23	2	brown	fair	56
24	2	brown	red	42
25	2	brown	medium	53
26	2	brown	dark	54
27	2	brown	black	13

The FREQ Procedure

Frequency Expected Percent	Table of Eyes by Hair					
	Hair(Hair Color)					
Eyes(Eye Color)	fair	red	medium	dark	black	Total
blue	69 66.425 9.06	28 32.921 3.67	68 63.22 8.92	51 53.024 6.69	6 6.4094 0.79	222 29.13
green	69 59.543 9.06	38 29.51 4.99	55 56.671 7.22	37 47.53 4.86	0 5.7454 0.00	199 26.12
brown	90 102.03 11.81	47 50.568 6.17	94 97.109 12.34	94 81.446 12.34	16 9.8451 2.10	341 44.75
Total	228 29.92	113 14.83	217 28.48	182 23.88	22 2.89	762 100.00

Observed frequency

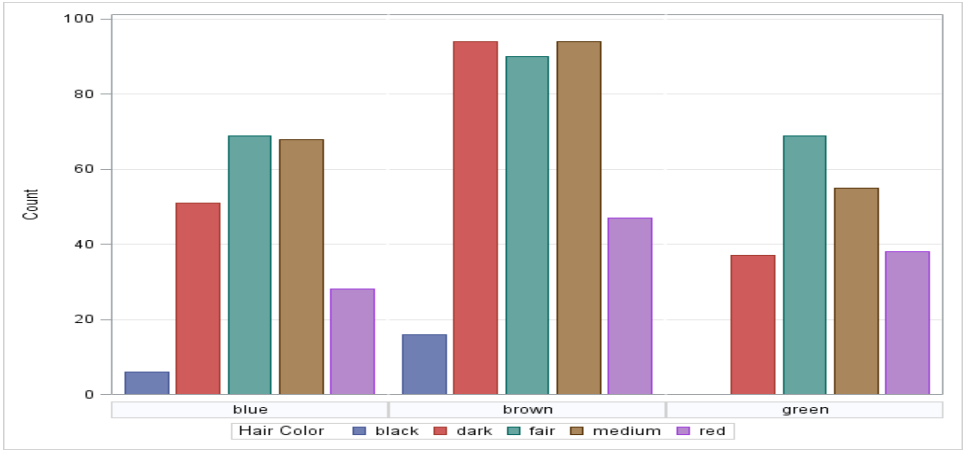
Expected frequency

Percentage

Statistics for Table of Eyes by Hair

Statistic	DF	Value	Prob
Chi-Square	8	20.9248	0.0073
Likelihood Ratio Chi-Square	8	25.9733	0.0011
Mantel-Haenszel Chi-Square	1	3.7838	0.0518
Phi Coefficient		0.1657	
Contingency Coefficient		0.1635	
Cramer's V		0.1172	

We can say there is an association between the two variables



T-TEST USING PROC TTEST

T-Test can be used to compare two groups

Situation: a researcher wants to compare weights of students by gender

```
proc ttest data=sashelp.class;
class sex;
var weight;
run;
```

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69.0	112.5
2	Alice	F	13	56.5	84.0
3	Barbara	F	13	65.3	98.0
4	Carol	F	14	62.8	102.5
5	Henry	M	14	63.5	102.5
6	James	M	12	57.3	83.0
7	Jane	F	12	59.8	84.5
8	Janet	F	15	62.5	112.5
9	Jeffrey	M	13	62.5	84.0
10	John	M	12	59.0	99.5
11	Joyce	F	11	51.3	50.5
12	Judy	F	14	64.3	90.0
13	Louise	F	12	56.3	77.0
14	Mary	F	15	66.5	112.0
15	Philip	M	16	72.0	150.0
16	Robert	M	12	64.8	128.0
17	Ronald	M	15	67.0	133.0
18	Thomas	M	11	57.5	85.0
19	William	M	15	66.5	112.0

The TTEST Procedure

Variable: Weight

Sex	N	Mean	Std Dev	Std Err	Minimum	Maximum
F	9	90.1111	19.3839	6.4613	50.5000	112.5
M	10	109.0	22.7272	7.1870	83.0000	150.0
Diff (1-2)		-18.8389	21.2196	9.7497		

Sex	Method	Mean	95% CL Mean		Std Dev	95% CL Std Dev	
F		90.1111	75.2113	105.0	19.3839	13.0930	37.1351
M		109.0	92.6920	125.2	22.7272	15.6326	41.4910
Diff (1-2)	Pooled	-18.8389	-39.4090	1.7313	21.2196	15.9229	31.8112
Diff (1-2)	Satterthwaite	-18.8389	-39.2325	1.5547			

Descriptive stats

Method	Variances	DF	t Value	Pr > |t|
Pooled	Equal	17	-1.93	0.0702
Satterthwaite	Unequal	16.962	-1.95	0.0680

This is not statistically significant with 0.05, but it is significant with 0.1

Equality of Variances				
Method	Num DF	Den DF	F Value	Pr > F
Folded F	9	8	1.37	0.6645

ONE-WAY ANOVA USING PROC GLM

ANOVA can be used for comparing more than two levels

```
data reading;
input score method$;
cards;
72  A
84  A
80  A
65  A
90  A
50  B
51  B
48  B
60  B
61  B
52  C
57  C
20  C
62  C
60  C
;
```

Situation: 15 students were taught by three different methods to improve their reading scores

```
proc glm data=reading;
class method;
model score=method;
means method/hovtest tukey;
run;
```

- Class: grouping variable
- means: function to generate the mean values of each method
- hovtest: checking assumption of homogeneity of variances
- tukey: multiple comparisons

The GLM Procedure

Dependent Variable: score

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	2	2306.800000	1153.400000	7.99	0.0062
Error	12	1731.600000	144.300000		
Corrected Total	14	4038.400000			

R-Square	Coeff Var	Root MSE	score Mean
0.571216	19.75739	12.01249	60.80000

Source	DF	Type I SS	Mean Square	F Value	Pr > F
method	2	2306.800000	1153.400000	7.99	0.0062

Source	DF	Type III SS	Mean Square	F Value	Pr > F
method	2	2306.800000	1153.400000	7.99	0.0062

- The output "Source" shows sources of variances. "Model" means effects of all of the independent variables (in this case, we have only method).
- One way ANOVA is based on F-distribution and F-statistics value is 7.99 with p-value of 0.0062. This F-statistics is also called "Global test"
- Since the p-value is less than 0.05, we reject the null hypothesis that the teaching methods were all the same

The assumption of equal variances (homogeneity of variances) can be checked with "hovtest" option

The GLM Procedure

Levene's Test for Homogeneity of score Variance
ANOVA of Squared Deviations from Group Means

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
method	2	121066	60533.2	1.21	0.3316
Error	12	599312	49942.7		

- The p-value is 0.3316. There is no evidence to reject the hypothesis of Homogeneity
- If this assumption is violated, "Welch" option can be used to request the variance-weighted one-way ANOVA of Welch
- However, ANOVA is known to be robust to the assumption of equal within-group variances

The GLM Procedure

Tukey's Studentized Range (HSD) Test for score

NOTE: This test controls the Type I experimentwise error rate, but it generally has a higher Type II error rate than REGWQ.

Alpha	0.05
Error Degrees of Freedom	12
Error Mean Square	144.3
Critical Value of Studentized Range	3.77289
Minimum Significant Difference	20.269

Means with the same letter are not significantly different.

Tukey Grouping	Mean	N	method
A	78.200	5	A
B	54.000	5	B
B			
B	50.200	5	C

- SAS provides a variety of options for multiple comparison tests, such as Bonferroni, Duncan, Dunnett, Scheffe, etc.
- Options for multiple comparisons can be useful when F-test statistics is significant
- Under the "Tukey Grouping" column, same letter means no significant effect. For example, method B and C have the same letter of "B", so not significantly different between B and C

TWO-WAY ANOVA USING PROC GLM

Situation :

A researcher is examining the effect of Ritalin and Hyperactivity diagnosis on the level of activity in children

```
data twoway;
 input child$ drug$ activity;
 cards;
 Normal   Placebo 50
 Normal   Placebo 45
 Normal   Placebo 55
 Normal   Placebo 52
 Normal   Ritalin 67
 Normal   Ritalin 60
 Normal   Ritalin 58
 Normal   Ritalin 65
 Hyper    Placebo 70
 Hyper    Placebo 72
 Hyper    Placebo 68
 Hyper    Placebo 75
 Hyper    Ritalin 51
 Hyper    Ritalin 57
 Hyper    Ritalin 48
 Hyper    Ritalin 55
 ;
```

```
proc glm data=twoway;
     class child drug;
     model activity = child drug child*drug;
     lsmeans child drug;
run;
```

- Examining Activity Level
- Factor A: Children (hyperactive vs. Normal)
- Factor B: Drug (Ritalin vs. placebo)
- Interaction between A and B

The GLM Procedure

Dependent Variable: activity

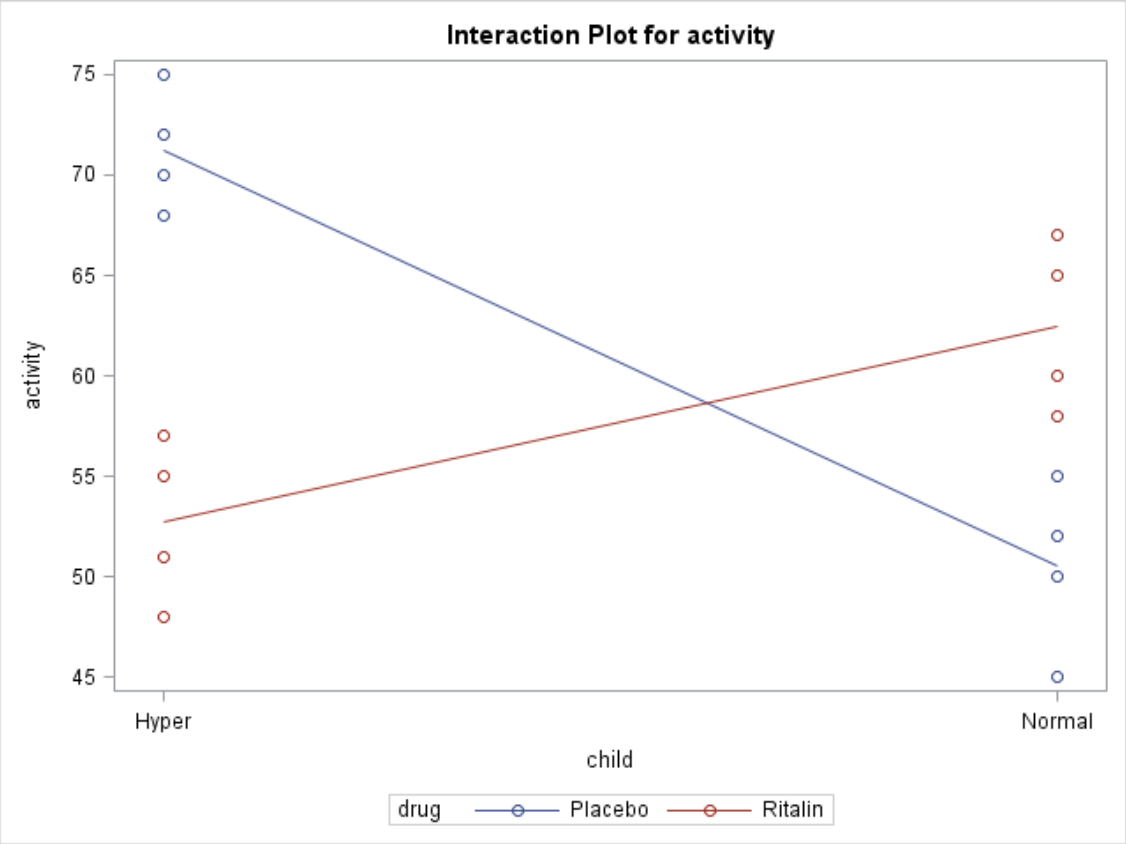
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	1093.500000	364.500000	24.10	<.0001
Error	12	181.500000	15.125000		
Corrected Total	15	1275.000000			

R-Square	Coeff Var	Root MSE	activity Mean
0.857647	6.563860	3.889087	59.25000

Source	DF	Type I SS	Mean Square	F Value	Pr > F
child	1	121.0000000	121.0000000	8.00	0.0152
drug	1	42.2500000	42.2500000	2.79	0.1205
child*drug	1	930.2500000	930.2500000	61.50	<.0001

Source	DF	Type III SS	Mean Square	F Value	Pr > F
child	1	121.0000000	121.0000000	8.00	0.0152
drug	1	42.2500000	42.2500000	2.79	0.1205
child*drug	1	930.2500000	930.2500000	61.50	<.0001

- Global test is significant with F=24.10 and p=<0.0001
- So, we can test effects of A, B, and interaction.



- The interaction is significant with F=61.50 and p=<.0001, which means the effect of "activity" on drug is different at different levels of "child"
- Graph can show why interaction is significant

ONE-FACTOR REPEATED MEASURES ANOVA

Situation:

A school principal is interested in determining whether or not the reading comprehension of dyslexic students is improving. To investigate this question, he selects a random sample of 8 students and tracks their scores on standardized tests for a period of three years. The data, presented as grade equivalent, are shown

Obs	student	grade	score
1	A	3	2.5
2	A	4	3.1
3	A	5	4.1
4	B	3	2.8
5	B	4	4.5
6	B	5	5.2
7	C	3	3.2
8	C	4	4.3
9	C	5	5.3
10	D	3	3.1
11	D	4	4.0
12	D	5	5.1
13	E	3	2.6
14	E	4	3.2
15	E	5	4.7
16	F	3	2.3
17	F	4	3.3
18	F	5	3.4
19	G	3	3.4
20	G	4	3.8
21	G	5	4.8
22	H	3	3.7
23	H	4	3.9
24	H	5	4.1

EXAMPLE CODE USING PROC MIXED

```
PROC MIXED DATA=dysldata;
  CLASS student grade;
  MODEL score = grade;
  REPEATED grade/ TYPE=AR(1) SUBJECT=student;
  LSMEANS grade/PDIFF ADJUST=SIMULATE;
  ODS OUTPUT lsmeans=lsm;
  RUN;


proc sgplot data=lsm;
  series x=grade y=estimate/markers
              markersattrs=(color=red);
  run;
```

Requests that differences of the LS-means be displayed

Requests a multiple comparison adjustment for the p-values for the difference of LS-means.

Covariance structures provided in SAS

Structure	Description	Parms	(i,j)th element
ANTE(1)	Ante-dependence	$2t-1$	$\sigma_i\sigma_j\prod_{k=i}^{j-1}\rho_k$
AR(1)	Autoregressive(1)	2	$\sigma^2\rho^{\lvert i-j\rvert}$
ARH(1)	Heterogeneous AR(1)	$t+1$	$\sigma_i\sigma_j\rho^{\lvert i-j\rvert}$
ARMA(1,1)	ARMA(1,1)	3	$\sigma^2[\gamma\rho^{\lvert i-j\rvert-1}1(i\neq j)+1(i=j)]$
CS	Compound Symmetry	2	$\sigma_1+\sigma^2 1(i=j)$
CSH	Heterogeneous CS	$t+1$	$\sigma_i\sigma_j[\rho 1(i\neq j)+1(i=j)]$
FA(q)	Factor Analytic	$\frac{q}{2}(2t-q+1)+t$	$\Sigma_{k=1}^{\min(i,j,q)}\lambda_{ik}\lambda_{jk}+\sigma_i^2 1(i=j)$
FA0(q)	No Diagonal FA	$\frac{q}{2}(2t-q+1)$	$\Sigma_{k=1}^{\min(i,j,q)}\lambda_{ik}\lambda_{jk}$
FA1(q)	Equal Diagonal FA	$\frac{q}{2}(2t-q+1)+1$	$\Sigma_{k=1}^{\min(i,j,q)}\lambda_{ik}\lambda_{jk}+\sigma^2 1(i=j)$
HF	Huynh-Feldt	$t+1$	$(\sigma_i^2+\sigma_j^2)/2+\lambda 1(i\neq j)$
LIN(q)	General Linear	q	$\Sigma_{k=1}^{q}\theta_k\mathbf{A}_{ij}$
TOEP	Toeplitz	t	$\sigma_{\lvert i-j\rvert+1}$
TOEP(q)	Banded Toeplitz	q	$\sigma_{\lvert i-j\rvert+1}1(\lvert i-j\rvert<q)$
TOEPH	Heterogeneous TOEP	$2t-1$	$\sigma_i\sigma_j\rho_{\lvert i-j\rvert}$
TOEPH(q)	Banded Hetero TOEP	$t+q-1$	$\sigma_i\sigma_j\rho_{\lvert i-j\rvert}1(\lvert i-j\rvert<q)$
UN	Unstructured	$t(t+1)/2$	σ_{ij}
UN(q)	Banded	$\frac{q}{2}(2t-q+1)$	$\sigma_{ij}1(\lvert i-j\rvert<q)$
UNR	Unstructured Corrs	$t(t+1)/2$	$\sigma_i\sigma_j\rho_{\max(i,j)\min(i,j)}$
UNR(q)	Banded Correlations	$\frac{q}{2}(2t-q+1)$	$\sigma_i\sigma_j\rho_{\max(i,j)\min(i,j)}$
UN@AR(1)	Direct Product AR(1)	$t_1(t_1+1)/2+1$	$\sigma_{i_1j_1}\rho^{\lvert i_2-j_2\rvert}$
UN@CS	Direct Product CS	$t_1(t_1+1)/2+1$	$\begin{cases}\sigma_{i_1j_1} & i_2=j_2\\ \sigma^2\sigma_{i_1j_1} & i_2\neq j_2\\ 0\le\sigma^2\le 1\end{cases}$
UN@UN	Direct Product UN	$t_1(t_1+1)/2+$ $t_2(t_2+1)/2-1$	$\sigma_{i_1j_1\sigma_{i_2j_2}}$
VC	Variance Components	q	$\sigma_k^2 1(i=j)$ and i corresponds to kth effect

The Mixe

Model

Covariance P

Cov Parm

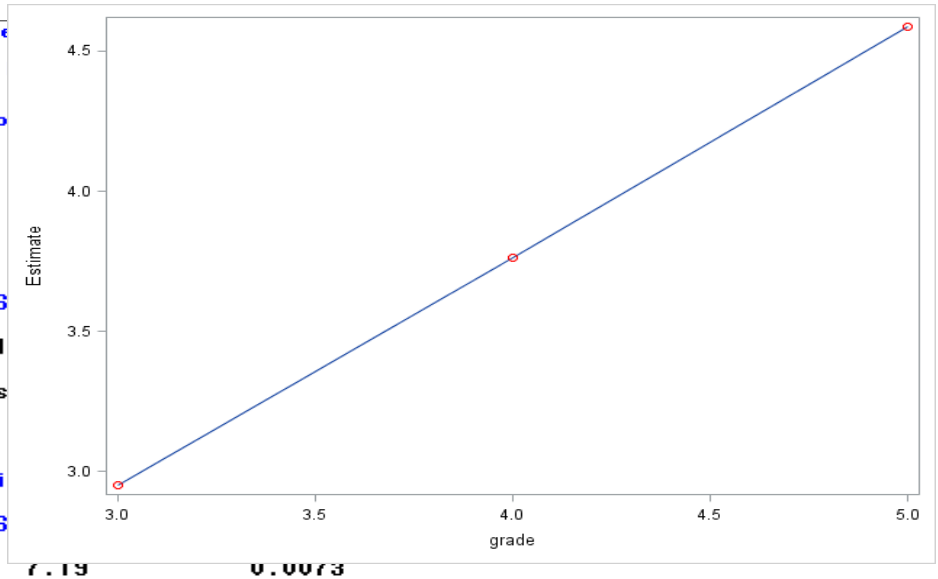
AR(1)
Residual

Fit S

-2 Res Log Likel
AIC (Smaller is
AICC (Smaller is
BIC (Smaller is

Null Model Li

DF	Chi-S	
1	7.19	0.0073



Type 3 Tests of Fixed Effects

Effect	Num DF	Den DF	F Value	Pr > F
grade	2	14	28.90	<.0001

Least Squares Means

Effect	grade	Estimate	Standard Error	DF	t Value	Pr > |t|
grade	3	2.9500	0.2003	14	14.72	<.0001
grade	4	3.7625	0.2003	14	18.78	<.0001
grade	5	4.5875	0.2003	14	22.90	<.0001

Differences of Least Squares Means

Effect	grade	_grade	Estimate	Standard Error	DF	t Value	Pr > |t|	Adjustment	Adj P
grade	3	4	-0.8125	0.1677	14	-4.85	0.0003	Simulate	0.0006
grade	3	5	-1.6375	0.2154	14	-7.60	<.0001	Simulate	<.0001
grade	4	5	-0.8250	0.1677	14	-4.92	0.0002	Simulate	0.0006

ANCOVA USING PROC GLM

Situation :

A researcher is examining the effect of different types of instruction of 5th grade students on reading comprehension. Twenty elementary schools are used for the study–10 schools are considered controls, teaching using traditional curriculum and 10 schools are considered experimental, teaching using computer-assisted curriculum. Fifth grade students were randomly selected from each of the 20 schools. At the end of the school year, all students in all 20 schools are given a common examination designed to measure reading comprehension. The 20 school means were recorded as observations for each of the schools. **Is it reasonable to expect that schools with higher mean IQ scores will tend to have higher mean reading scores**. Average IQ scores will be used as a covariate

EXAMPLE CODE

Data

Obs	group	iqave	readavg
1	e	100.1	78.6
2	e	101.2	75.2
3	e	98.5	68.3
4	e	104.3	79.1
5	e	98.2	59.4
6	e	97.9	56.5
7	e	104.5	88.2
8	e	98.2	72.9
9	e	89.4	48.3
10	e	95.2	65.4
11	c	99.2	65.2
12	c	96.4	44.4
13	c	98.5	49.5
14	c	107.5	85.3
15	c	94.1	45.9
16	c	104.5	72.3
17	c	103.4	72.1
18	c	101.3	43.6
19	c	103.8	69.7
20	c	111.7	70.2

Standard ANOVA

```
proc glm data=readdata;
 class group;
 model readavg=group;
 run;
```

ANCOVA-different intercepts, same slopes

```
proc glm data=readdata;
 class group;
 model readavg=group iqave;
 run;
```

As a covariate

The GLM Procedure

Dependent Variable: readavg

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	271.584500	271.584500	1.50	0.2364
Error	18	3257.665000	180.981389		
Corrected Total	19	3529.249500			

R-Square	Coeff Var	Root MSE	readavg Mean
0.076952	20.53726	13.45293	65.50500

Source	DF	Type I SS	Mean Square	F Value	Pr > F
group	1	271.5845000	271.5845000	1.50	0.2364

Source	DF	Type III SS	Mean Square	F Value	Pr > F
group	1	271.5845000	271.5845000	1.50	0.2364

ANOVA is not able to detect any differences between the two groups

The GLM Procedure

Dependent Variable: readavg

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	2	2403.088582	1201.544291	18.14	<.0001
Error	17	1126.160918	66.244760		
Corrected Total	19	3529.249500			

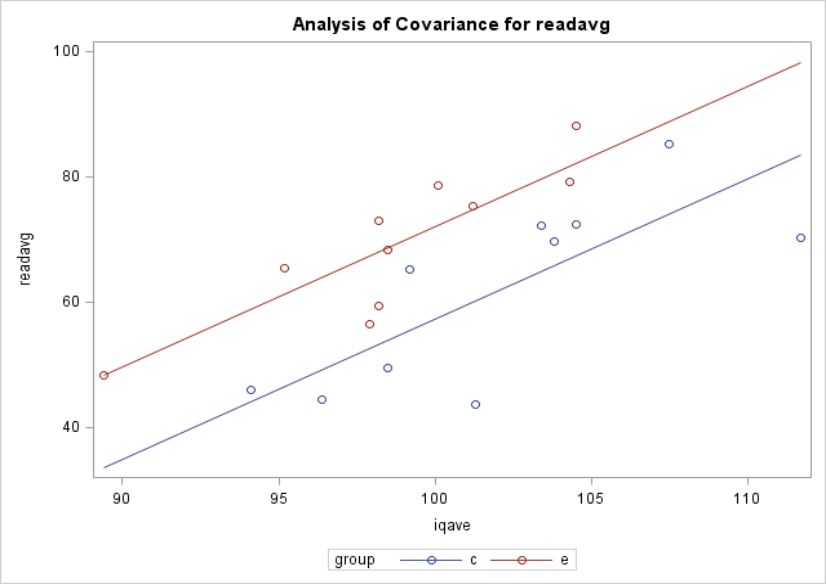
R-Square	Coeff Var	Root MSE	readavg Mean
0.680906	12.42514	8.139088	65.50500

Source	DF	Type I SS	Mean Square	F Value	Pr > F
group	1	271.584500	271.584500	4.10	0.0589
iqave	1	2131.504082	2131.504082	32.18	<.0001

Source	DF	Type III SS	Mean Square	F Value	Pr > F
group	1	964.912776	964.912776	14.57	0.0014
iqave	1	2131.504082	2131.504082	32.18	<.0001

What about ANCOVA?

Both are significant. We can conclude the groups are different. This difference was not visible with one-way ANOVA



SLR USING PROC REG

Situation :

A researcher is examining the relationship between a college entrance exam and freshman GPAs

```
PROC REG DATA=scoregpa;
MODEL gpa=score/CLM CLB CLI;
PLOT gpa*score;
RUN;
```

```
DATA scoregpa;
INPUT gpa score;
CARDS;
3.5 3.2
3.3 2.9
3.4 3.0
3.5 3.1
3.5 3.3
3.5 3.7
3.3 3.1
3.1 3.5
3.2 3.1
3.4 2.9
3.2 2.7
3.4 3.2
3.4 3.0
3.2 2.7
3.0 2.4
3.5 3.3
2.2 2.2
3.3 2.8
3.3 2.7
2.7 2.7
. 3.5
;
```

The REG Procedure
Model: MODEL1
Dependent Variable: gpa

Number of Observations Read	21
Number of Observations Used	20
Number of Observations with Missing Values	1

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	0.91387	0.91387	16.52	0.0007
Error	18	0.99563	0.05531		
Corrected Total	19	1.90950			

Root MSE	0.23519	R-Square	0.4786
Dependent Mean	3.24500	Adj R-Sq	0.4496
Coeff Var	7.24767		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|	95% Confidence Limits	
Intercept	1	1.42338	0.45123	3.15	0.0055	0.47539	2.37138
score	1	0.61231	0.15064	4.06	0.0007	0.29582	0.92879

CLB option computes 100(1-α)% confidence limits for the parameter estimates

The REG Procedure
Model: MODEL1
Dependent Variable: gpa

Output Statistics

Obs	Dependent Variable	Predicted Value	Std Error Mean Predict	95% CL Mean		95% CL Predict		Residual
1	3.5000	3.3828	0.0626	3.2513	3.5142	2.8715	3.8941	0.1172
2	3.3000	3.1991	0.0538	3.0861	3.3121	2.6922	3.7059	0.1009
3	3.4000	3.2603	0.0527	3.1495	3.3711	2.7539	3.7667	0.1397
4	3.5000	3.3215	0.0559	3.2042	3.4389	2.8137	3.8294	0.1785
5	3.5000	3.4440	0.0719	3.2930	3.5950	2.9273	3.9607	0.0560
6	3.5000	3.6889	0.1212	3.4343	3.9436	3.1330	4.2448	-0.1889
7	3.3000	3.3215	0.0559	3.2042	3.4389	2.8137	3.8294	-0.0215
8	3.1000	3.5665	0.0950	3.3669	3.7660	3.0336	4.0993	-0.4665
9	3.2000	3.3215	0.0559	3.2042	3.4389	2.8137	3.8294	-0.1215
10	3.4000	3.1991	0.0538	3.0861	3.3121	2.6922	3.7059	0.2009
11	3.2000	3.0766	0.0669	2.9360	3.2173	2.5629	3.5904	0.1234
12	3.4000	3.3828	0.0626	3.2513	3.5142	2.8715	3.8941	0.0172
13	3.4000	3.2603	0.0527	3.1495	3.3711	2.7539	3.7667	0.1397
14	3.2000	3.0766	0.0669	2.9360	3.2173	2.5629	3.5904	0.1234
15	3.0000	2.8929	0.1013	2.6900	3.1058	2.3549	3.4309	0.1071
16	3.5000	3.4440	0.0719	3.2930	3.5950	2.9273	3.9607	0.0560
17	2.2000	2.7705	0.1280	2.5015	3.0395	2.2079	3.3331	-0.5705
18	3.3000	3.1378	0.0588	3.0143	3.2614	2.6285	3.6472	0.1622
19	3.3000	3.0766	0.0669	2.9360	3.2173	2.5629	3.5904	0.2234
20	2.7000	3.0766	0.0669	2.9360	3.2173	2.5629	3.5904	-0.3766
21	.	3.5665	0.0950	3.3669	3.7660	3.0336	4.0993	.

Sum of Residuals	0
Sum of Squared Residuals	0.99563
Predicted Residual SS (PRESS)	1.52084

SAS prints out a predicted value for the missing gpa

CLM option computes confidence limits for the expected value of the dep. variable

CLI option computes confidence limits for an individual predicted value

MLR USING PROC REG

Situation :

A researcher is examining the relationship between high school math GPA, high school English GPA, SAT math score, SAT verbal score, and freshman GPSs

```
DATA gpa;
INPUT GPA HSmath HSeng SATmath SATverb;
CARDS;
3.5 3.7 3.4 674 662
3.3 3.2 2.7 414 453
3.4 3.5 2.9 490 542
3.5 3.7 3.3 617 601
3.5 3.7 3.8 680 678
3.5 3.8 3.9 796 781
3.3 3.4 2.8 483 528
3.1 2.2 2.2 243 353
3.2 2.5 2.5 366 404
3.4 3.4 2.9 488 537
3.2 2.4 2.2 302 358
3.4 3.6 3.2 594 576
3.4 3.6 3.1 490 560
3.2 2.5 2.4 316 377
3 2.2 2.1 231 329
3.5 3.8 3.9 732 717
2.2 2.1 2.0 213 299
3.3 2.8 2.7 413 431
3.3 2.6 2.6 389 429
2.7 2.2 2.0 231 303
. 3.9 2.3 452 798
;
```

```
proc reg data=gpa;
model gpa=SATmath SATverb HSmath HSeng/P CLB CLM CLI;
run;
```

This code can be used for a general purpose. Output and interpretation will be similar to SLR.

The REG Procedure
Model: MODEL1
Dependent Variable: GPA

Number of Observations Read	21
Number of Observations Used	20
Number of Observations with Missing Values	1

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	4	1.19557	0.29889	6.28	0.0036
Error	15	0.71393	0.04760		
Corrected Total	19	1.90950			

Root MSE	0.21816	R-Square	0.6261
Dependent Mean	3.24500	Adj R-Sq	0.5264
Coeff Var	6.72308		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|	95% Confidence Limits	
Intercept	1	2.01880	0.74486	2.71	0.0161	0.43116	3.60644
SATmath	1	0.00043625	0.00237	0.18	0.8565	-0.00462	0.00549
SATverb	1	-0.00108	0.00309	-0.35	0.7317	-0.00766	0.00550
HSmath	1	0.30257	0.24135	1.25	0.2292	-0.21185	0.81698
HSeng	1	0.22619	0.55775	0.41	0.6908	-0.96263	1.41501

MODEL SELECTION IN PROC REG

```
PROC REG DATA=gpa;
MODEL GPA = SATmath SATverb HSmath HSeng /SELECTION=RSQUARE ADJRSQ BEST=4;
*Best Subset Regression;
run;
```

ocedure
ODEL1
iable: GPA

R-Square Selection Method

Number of Observations Read	21
Number of Observations Used	20
Number of Observations with Missing Values	1

Number in Model	R-Square	Adjusted R-Square	Variables in Model
1	0.6139	0.5924	HSmath
1	0.5844	0.5613	SATmath
1	0.5810	0.5577	HSeng
1	0.5810	0.5577	SATverb
2	0.6230	0.5787	HSmath HSeng
2	0.6210	0.5764	SATmath HSmath
2	0.6182	0.5733	SATverb HSmath
2	0.5867	0.5381	SATmath HSeng
3	0.6253	0.5550	SATverb HSmath HSeng
3	0.6231	0.5524	SATmath HSmath HSeng
3	0.6220	0.5511	SATmath SATverb HSmath
3	0.5869	0.5095	SATmath SATverb HSeng
4	0.6261	0.5264	SATmath SATverb HSmath HSeng

Best 4 models in terms of R-square are chosen with different numbers of variables

LINEAR MIXED-EFFECTS MODEL

The data file is from the 1982 High School and Beyond Survey consisting of 7185 students nested in 160 schools.

The outcome variable of interest is math achievement score (MATHACH). Variable SECTOR is an indicator variable indicating if a school is public or catholic. Variable FEMALE is another grouping variable for gender

A subset of the dataset is presented. As can be seen, we have multiple obs from the same school

SCHOOL	MATHACH	SECTOR	FEMALE
1224	6.714	0	0
1224	0.122	0	0
1224	2.381	0	1
1224	16.336	0	1
1224	6.134	0	1
1224	23.584	0	1
1224	14.053	0	1
1224	2.183	0	1
1288	7.857	0	1
1288	10.171	0	0
1288	15.699	0	0
1288	22.919	0	1
1288	10.664	0	1
1288	13.543	0	1
1288	18.207	0	0
1288	5.552	0	0
1288	7.416	0	0
1288	18.792	0	0
1288	1.575	0	0
1288	3.534	0	1
1288	20.173	0	1

```
proc mixed data = mult;
   class school female sector;
   model mathach = sector female;
   random intercept/ subject = school;
run;
```

School is used as random

This model is called **random intercept model**

Covariance Parameter Estimates		
Cov Parm	Subject	Estimate
Intercept	SCHOOL	6.2327
Residual		38.8551

Variability between school

Variability within school

Fit Statistics	
-2 Res Log Likelihood	47017.8
AIC (Smaller is Better)	47021.8
AICC (Smaller is Better)	47021.8
BIC (Smaller is Better)	47028.0

Fit stats can be used for model comparisons

Type 3 Tests of Fixed Effects				
Effect	Num DF	Den DF	F Value	Pr > F
SECTOR	1	7024	42.96	<.0001
FEMALE	1	7024	64.52	<.0001

STATISTICAL GRAPHICS

Proc Boxplot
Proc Sgplot
Proc Sgpanel
Proc Sgscatter

TOPICS

SAS provides multiple ways to generate graphs. However, we will focus on

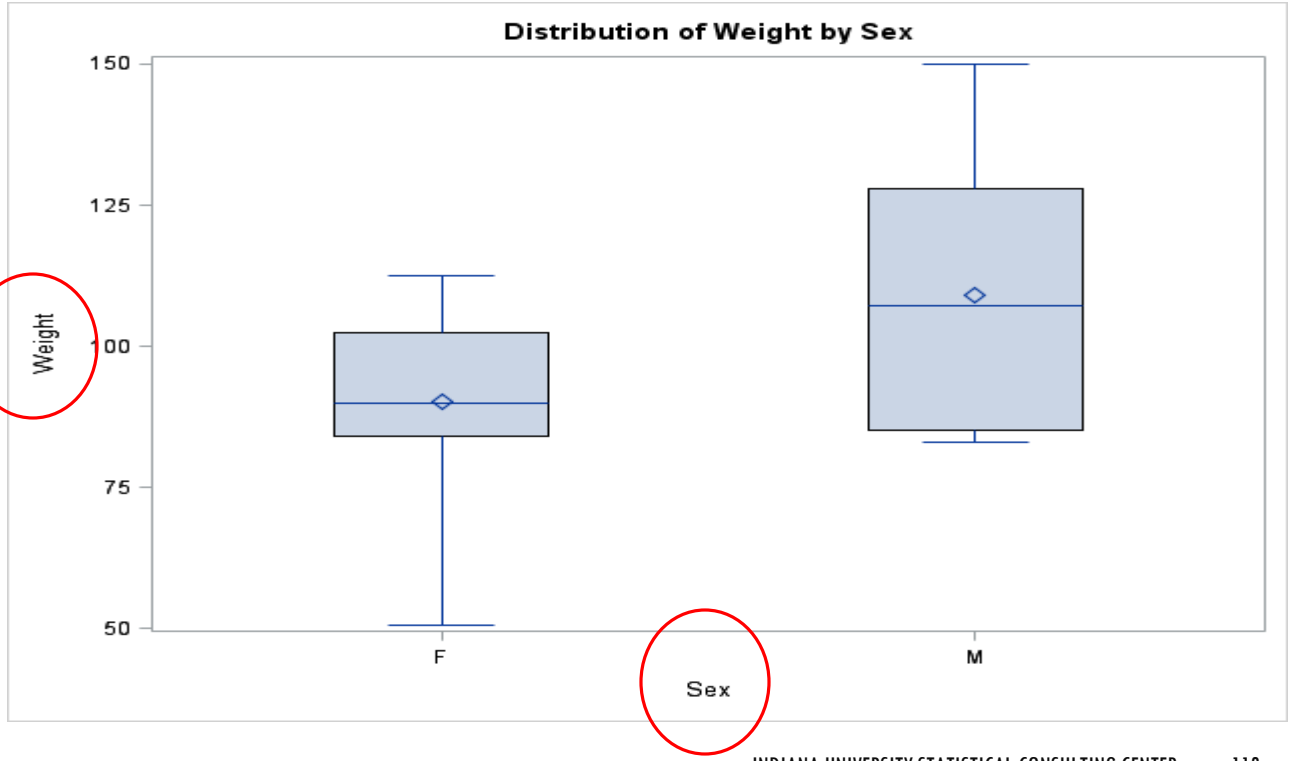
- Proc Boxplot
- Proc Sgplot
- Proc Sgpanel
- Proc Sgscatter

- Some of the examples presented here were borrowed from SAS documentation

PROC BOXPLOT

```
proc boxplot data=sashelp.class;
        plot weight*sex;
run;
```

Weight: outcome
Sex: grouping variable

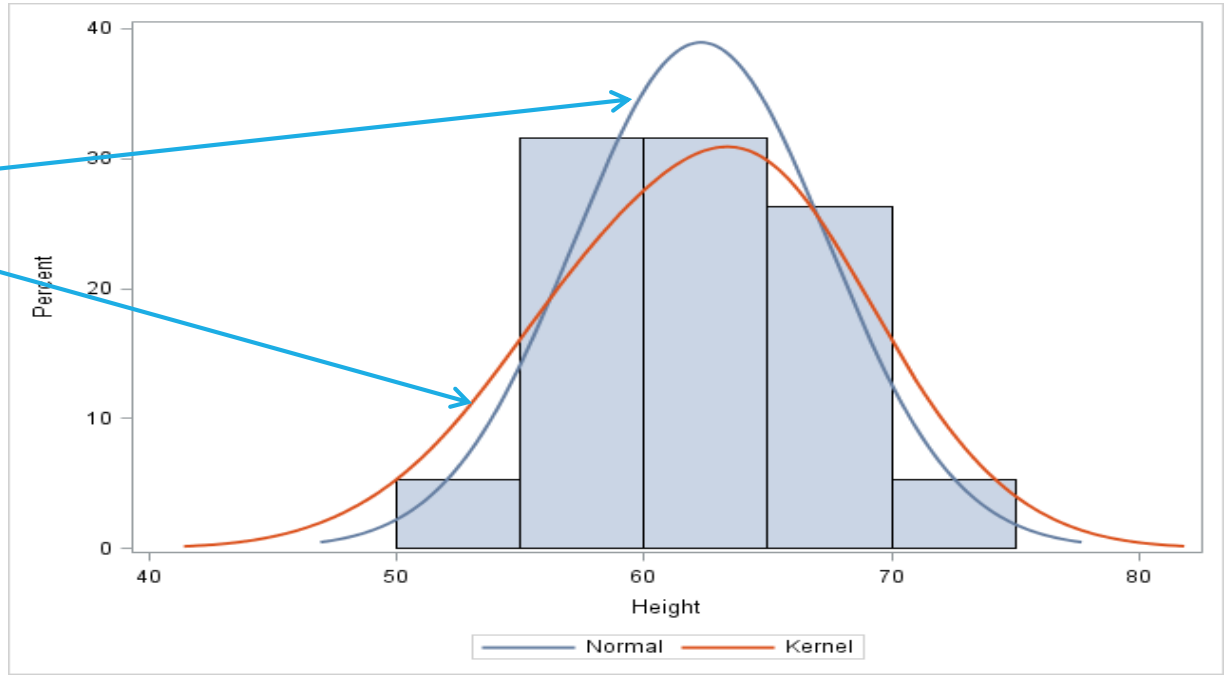


HISTOGRAM (PROC SGPLOT)

Histograms show the distribution of a continuous variable. The following code creates a graph with a histogram, a normal density curve, and a kernel density curve

```
proc sgplot data=sashelp.class;
  histogram height;
  density height;
  density height/type=kernel;
  run;
```

Default type is a normal density curve

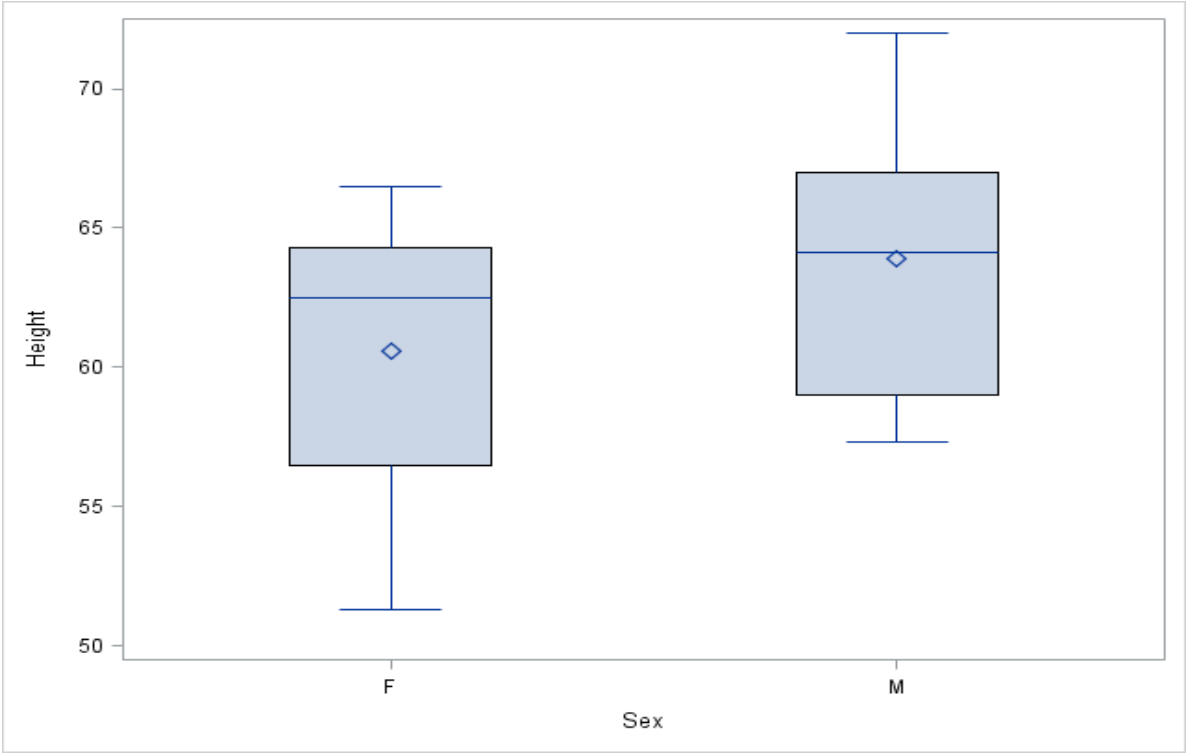


BOXPLOTS (PROC SGPLOT)

Boxplots display the distribution of data based on the five number summary: min, first quartile, median, third quartile, and max

```
proc sgplot data=sashelp.class;
  vbox height/category=sex;
  run;
```

Creates plots by SEX

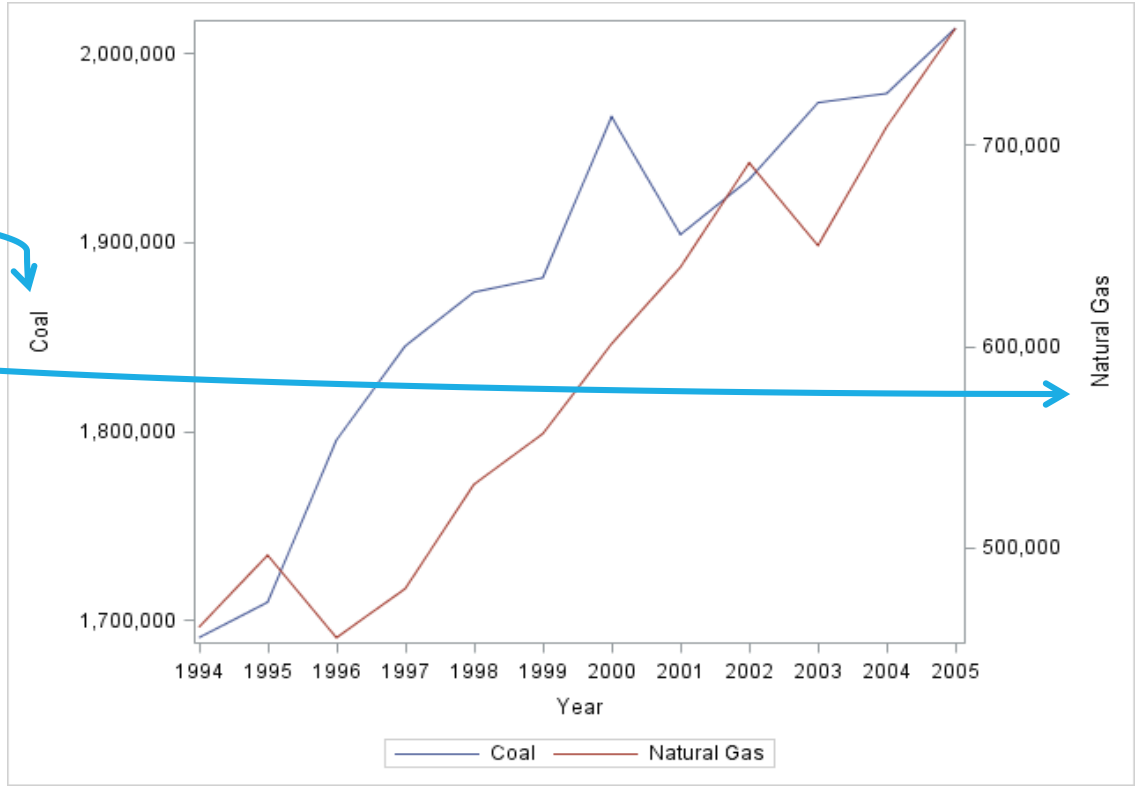


SERIES PLOTS (PROC SGPLOT)

The following code creates a graph with two series plots

```
proc sgplot data=sashelp.electric(
      where=(customer="Residential"));
   xaxis type=discrete;
   series x=year y=coal;
   series x=year y=naturalgas / y2axis;
run;
```

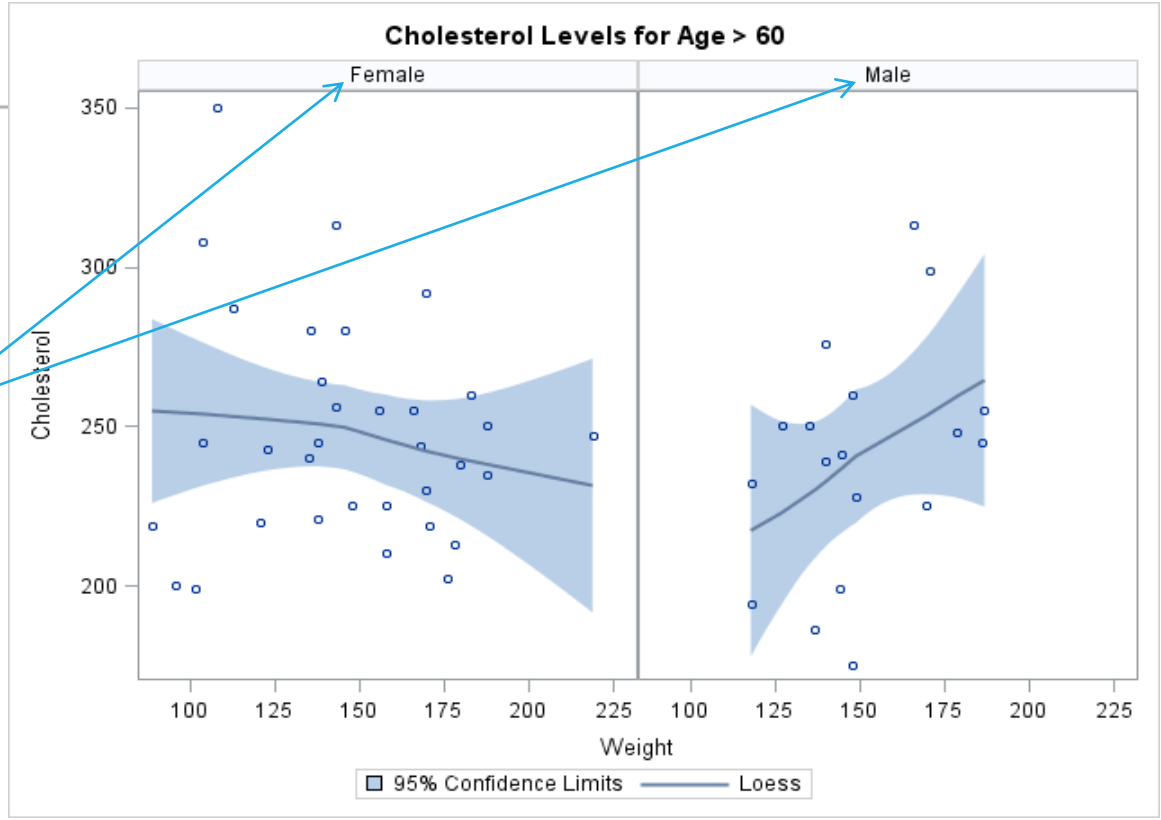
TYPE=discrete specifies an axis with discrete values



LOESS CURVES (PROC SGPANEL)

The following code creates a panel of loess curves

```
title1 "Cholesterol Levels for Age > 60";
proc sgpanel data=sashelp.heart(
    where=(AgeAtStart > 60)) ;
  panelby sex / novarname;
  loess x=weight y=cholesterol / clm;
run;
```



- "novarname" option supresses variable names.
- w/o the option:
  Sex=Female &
  Sex=Male

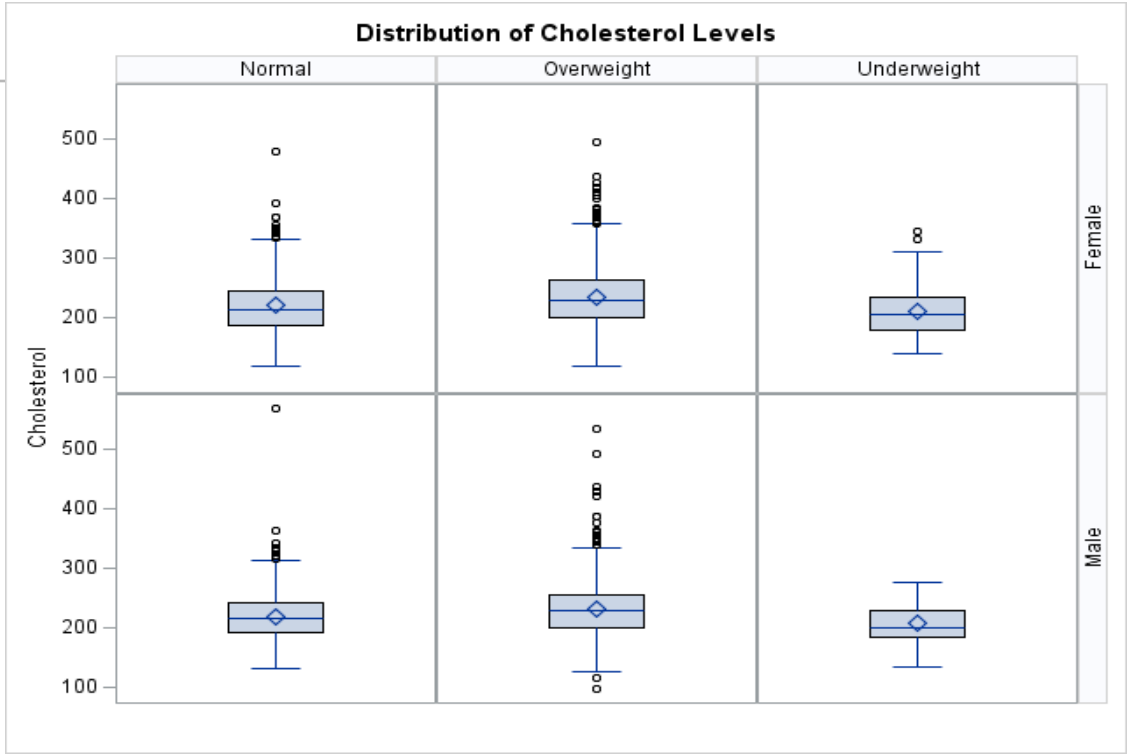
BOX PLOTS (PROC SGPANEL)

The following code creates a panel of box plots in a lattice layout

```
title1 "Distribution of Cholesterol Levels";
proc sgpanel data=sashelp.heart;
  panelby weight_status sex / layout=lattice
                              novarname;

  vbox cholesterol;
run;
```

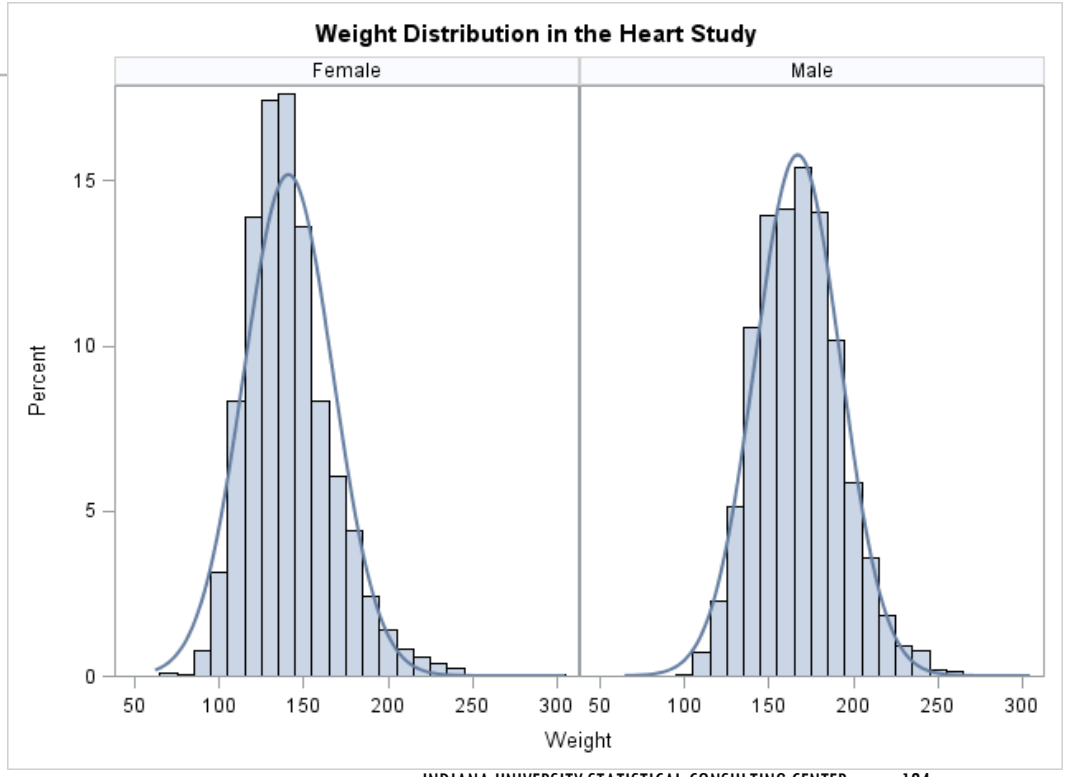
Layout= option controls the format. Lattice type can be used with two classification variables



HISTOGRAM (PROC SGPANEL)

The following code creates a panel of cells with a histogram and a normal density curve

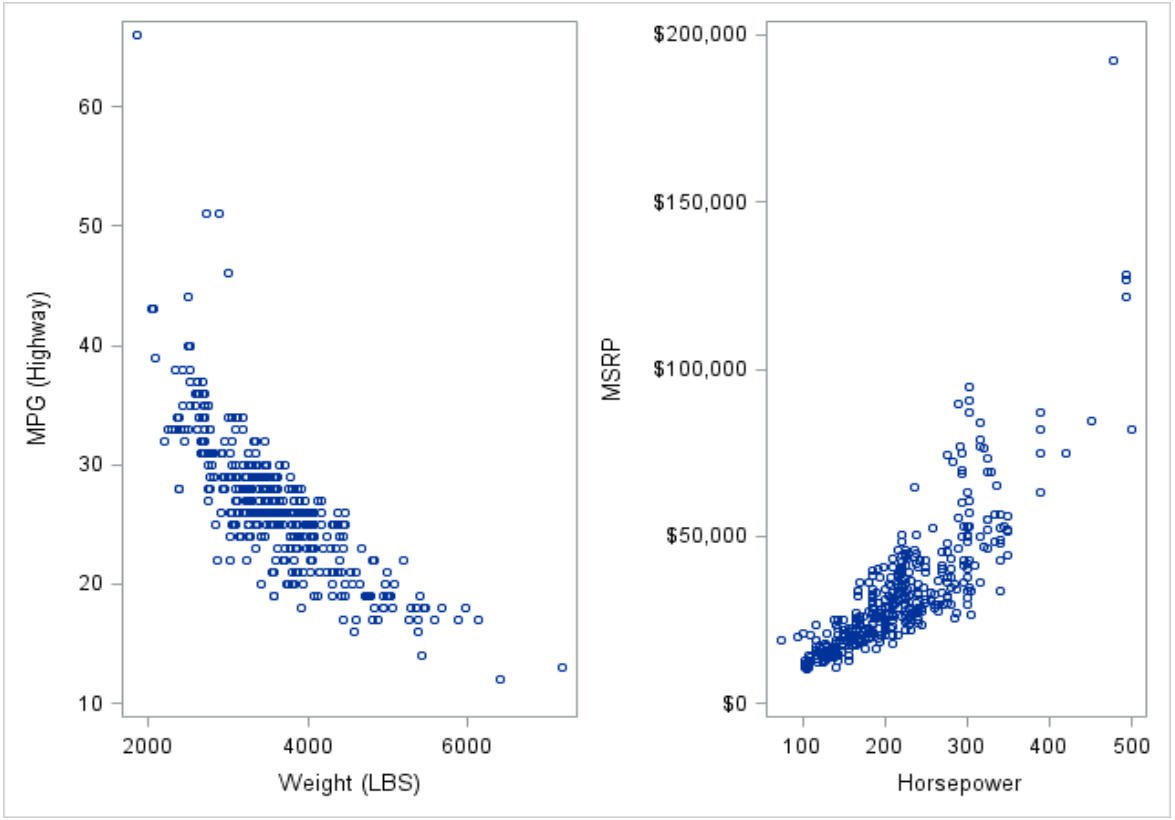
```
Title1 "Weight Distribution in the Heart Study";
proc sgpanel data=sashelp.heart noautolegend;
  panelby sex / novarname;
  histogram weight;
  density weight;
run;
```



SCATTER PLOTS (PROC SGSCATTER)

The following code creates a paneled graph of scatter plots

```
proc sgscatter data=sashelp.cars;
  plot mpg_highway*weight msrp*horsepower;
run;
```

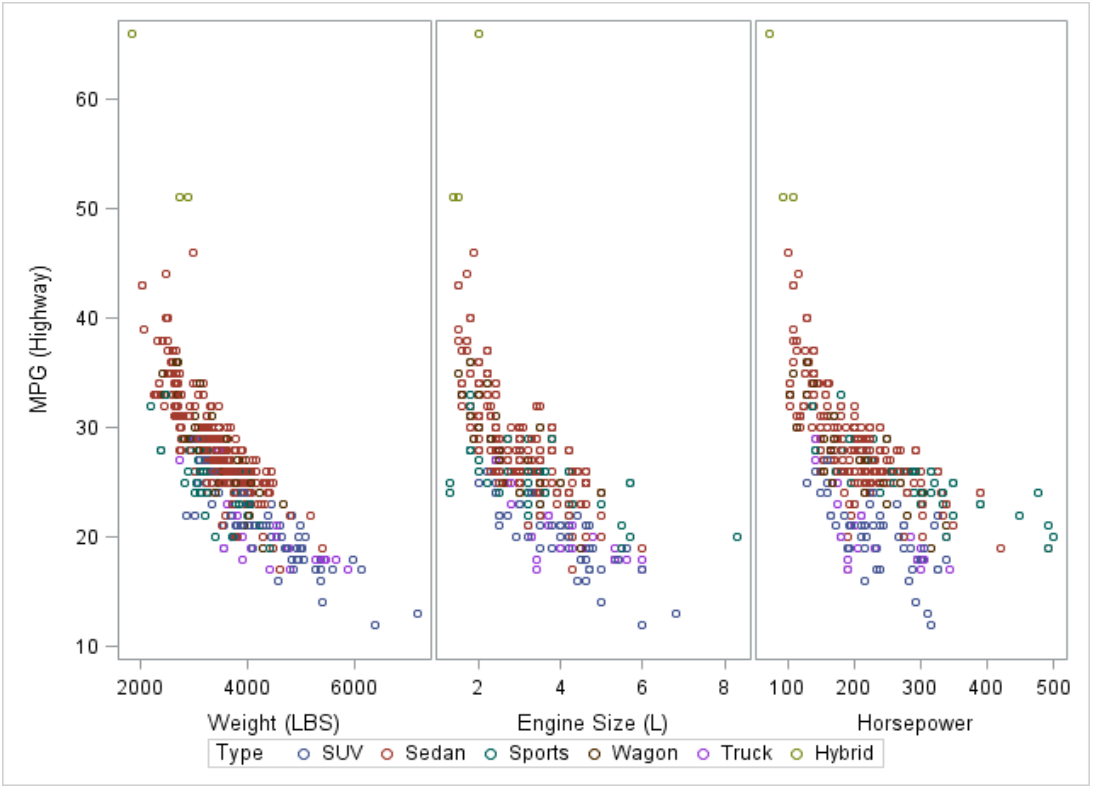


SCATTER PLOTS WITH SHARED AXES

The following code creates a paneled graph of scatter plots with shared axes

```
proc sgscatter data=sashelp.cars;
   compare y=mpg_highway
           x=(weight enginesize horsepower )
           / group=type;
run;
```

Mpg_highway is shared on y axis

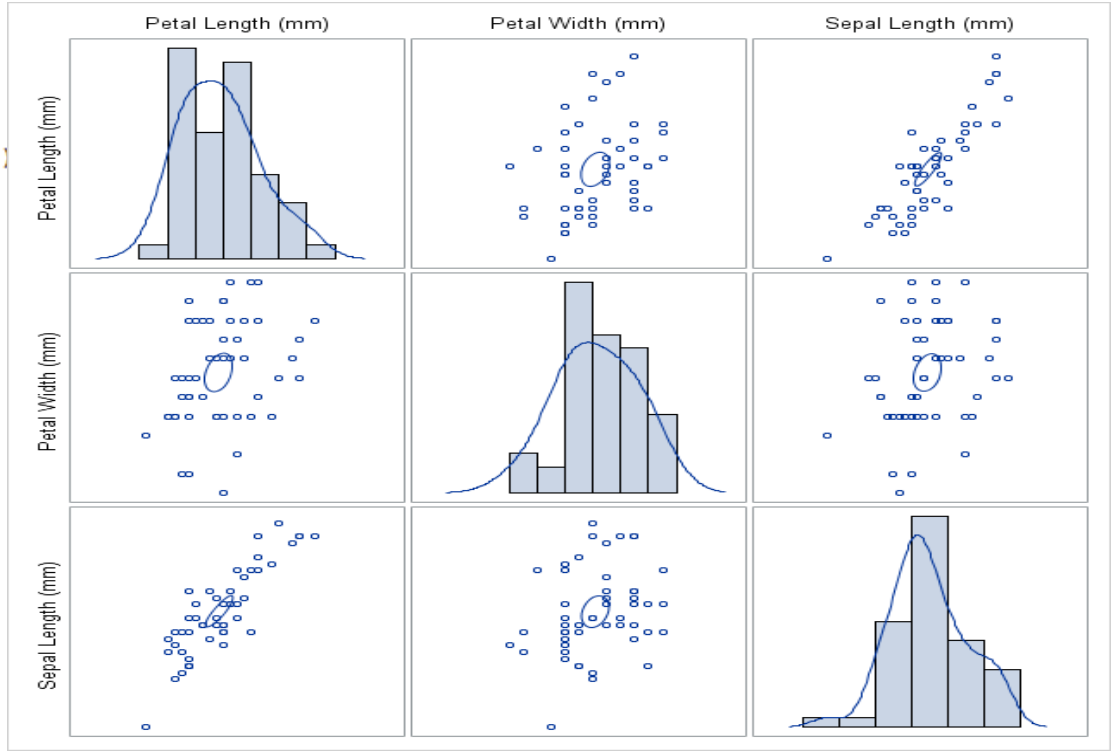


SCATTER PLOT MATRIX (PROC SGSCATTER)

The following code creates a scatter plot matrix with prediction ellipses and a diagonal with histograms and density plots

```
proc sgscatter data=sashelp.iris
               (where=(species eq "Virginica"));
  matrix petallength petalwidth sepallength
       / ellipse=(type=mean)
         diagonal=(histogram kernel);
  run;
```

- Ellipse option adds a confidence or prediction ellipse to each cell that contains a scatter plot
- Type=mean specifies a confidence ellipse for the population mean



ADVANCED SAS PROGRAMMING

Producing reports
Output delivery system

PRODUCING ATTRACTIVE REPORTS

- One of the big issues to researchers is producing fantastic summary reports from big, messy, and big and messy data
- SAS provides multiple ways to do that. So far, we have learned only PROC PRINT and PROC MEANS, which were not pretty enough
- Today, we will talk about
  - PROC REPORT and
  - PROC TABULATE

SIMPLE USE OF PROC REPORT

Let's take a look at the data

```
proc report data=sashelp.shoes(obs=10) nowd;
run;
```

The SAS System

Region	Product	Subsidiary	Number of Stores	Total Sales	Total Inventory	Total Returns
Africa	Boot	Addis Ababa	12	$29,761	$191,821	$769
Africa	Men's Casual	Addis Ababa	4	$67,242	$118,036	$2,284
Africa	Men's Dress	Addis Ababa	7	$76,793	$136,273	$2,433
Africa	Sandal	Addis Ababa	10	$62,819	$204,284	$1,861
Africa	Slipper	Addis Ababa	14	$68,641	$279,795	$1,771
Africa	Sport Shoe	Addis Ababa	4	$1,690	$16,634	$79
Africa	Women's Casual	Addis Ababa	2	$51,541	$98,641	$940
Africa	Women's Dress	Addis Ababa	12	$108,942	$311,017	$3,233
Africa	Boot	Algiers	21	$21,297	$73,737	$710
Africa	Men's Casual	Algiers	4	$63,206	$100,982	$2,221

How can we produce a fancy summary report?

BASIC PROC REPORT SYNTAX

```
PROC REPORT DATA=datasetname <options>;
  COLUMN variable-list <column specifications>;
  DEFINE column / <column-type> <column attributes>;
  DEFINE column / <column-type> <column attributes>;
     ...
RUN;
```

There are five column types

1. DISPLAY
2. ORDER
3. ACROSS
4. GROUP
5. ANALYSIS

The default type is DISPLAY

EXAMPLE 1 - DISPLAY TYPE

```
title 'Proc Report - Basic Syntax';
proc report data=sashelp.shoes nowd headskip headline;
     columns Region Subsidiary Product Stores Sales;
     define region /display format=$15.;
     define Subsidiary /display;
     define Product /display;
     define Stores /display;
     define Sales /display;
run;
```

Headline: underlines all column headings and the spaces between them at the top of each page of the report

Headskip: writes a blank line beneath all column headings at the top of each page of the report

Proc Report - Basic Syntax

Region	Subsidiary	Product	Number of Stores	Total Sales
Africa	Addis Ababa	Boot	12	$29,761
Africa	Addis Ababa	Men's Casual	4	$67,242
Africa	Addis Ababa	Men's Dress	7	$76,793
Africa	Addis Ababa	Sandal	10	$62,819
Africa	Addis Ababa	Slipper	14	$68,641
Africa	Addis Ababa	Sport Shoe	4	$1,690
Africa	Addis Ababa	Women's Casual	2	$51,541
Africa	Addis Ababa	Women's Dress	12	$108,942
Africa	Algiers	Boot	21	$21,297
Africa	Algiers	Men's Casual	4	$63,206
Africa	Algiers	Men's Dress	13	$123,743
Africa	Algiers	Sandal	25	$29,198
Africa	Algiers	Slipper	17	$64,891
Africa	Algiers	Sport Shoe	9	$2,617
Africa	Algiers	Women's Dress	12	$90,648
Africa	Cairo	Boot	20	$4,846
Africa	Cairo	Men's Casual	25	$360,209
Africa	Cairo	Men's Dress	5	$4,051
Africa	Cairo	Sandal	9	$10,532
Africa	Cairo	Slipper	9	$13,732
Africa	Cairo	Sport Shoe	3	$2,259
Africa	Cairo	Women's Casual	14	$328,474
Africa	Cairo	Women's Dress	3	$14,095

EXAMPLE 2 - GROUP TYPE

- The variable *product* has been omitted from the COLUMN statement
- Both *region* and *subsidiary* are defined as GROUP type

```
title 'Summary Report - Omitting "product"';
proc report data=sashelp.shoes nowd headskip headline;
     columns Region Subsidiary Stores Sales;
     define region /group width=15;
     define subsidiary /group;
run;
```

Summary Report - Omitting "product"

Region	Subsidiary	Number of Stores	Total Sales
Africa	Addis Ababa	65	$467,429
	Algiers	101	$395,600
	Cairo	88	$738,198
	Johannesburg	51	$113,008
	Khartoum	71	$186,592
	Kinshasa	56	$196,816
	Luanda	30	$138,115
	Nairobi	70	$106,830
Asia	Bangkok	5	$16,667
	Seoul	59	$442,409
	Tokyo	1	$1,155
Canada	Calgary	29	$61,403
	Montreal	115	$531,028
	Ottawa	29	$115,741
	Toronto	76	$319,772
	Vancouver	193	$3,227,768
Central America	Kingston	218	$2,235,204
	Managua	113	$414,806
	Mexico City	103	$368,588
	San Juan	105	$639,155
Eastern Europe	Budapest	85	$410,529
	Moscow	87	$560,976
	Prague	96	$636,721
	Warsaw	111	$786,714
Middle East	Al-Khobar	110	$1,153,667
	Dubai	153	$1,910,544
	Tel Aviv	134	$2,567,568
Pacific	Auckland	37	$124,424
	Canberra	50	$155,547
	Jakarta	84	$649,044
	Kuala Lumpur	64	$373,130
	Manila	103	$854,904
	Singapore	18	$139,745
South America	Bogota	89	$206,234
	Buenos Aires	66	$118,283
	Caracas	123	$789,323
	La Paz	116	$530,506
	Montevideo	96	$318,646
	Santiago	58	$104,956
	Sao Paulo	84	$366,835
[illegible]	[illegible]	[illegible]	[illegible]

By default, REPORT calculated sums for groups

EXAMPLE 3 – ANALYSIS DEFINE TYPE

- Define a statistic to summarize the variable
  - Calculated for groups created by group & across columns

- Suppose we want a summary report with sum of Stores and mean of Sales

```
title 'Summary Report - Sum of Stores, Mean of Sales';
proc report data=sashelp.shoes nowd headline headskip;
    columns Region Subsidiary Stores Sales;
    define region /group width=30;
    define subsidiary /group;
    define stores / analysis sum;
    define sales / analysis mean 'Average Sales';
run;
```

Sum for Stores

Mean for Sales

Column Heading

EXAMPLE 4 - YOU MAY NEED THIS

```
title 'Summary Report for Canada, US, & Western Europe ';
proc report data=sashelp.shoes nowd ls=120;

columns Region Subsidiary Stores
    ('= Financial Figures (in US Dollars) =/ '
        ('- Sales -' sales sales=salesm)
        ('- Inventory -' inventory inventory=invm)
        ('- Returns -' returns returns=retm)
    );

define region       / group width=14 'Region/--';
define subsidiary / group width=11 'Subsidiary/--';
define stores       / analysis sum width=6 'Number/of/Stores/--';

define sales        / analysis sum f=comma10. 'Total/--';
define salesm       / analysis mean f=comma7. 'Mean/--';

define inventory    / analysis sum f=comma10. 'Total/--';
define invm         / analysis mean f=comma9. 'Mean/--';

define returns  / analysis sum f=comma10. 'Total/--';
define retm         / analysis mean f=comma9. 'Mean/--';

break after region / ol skip summarize suppress;
rbreak after / dol summarize ;

where region in ("Canada", "United States", "Western Europe");
run;
```

Summary Report for Canada, US, & Western Europe

Region	Subsidiary	Number of Stores	Financial Figures (in US Dollars) — Sales Total	Sales Mean	Inventory Total	Inventory Mean	Returns Total	Returns Mean
Canada	Calgary	29	61,403	10,234	259,578	43,263	1,977	330
	Montreal	115	531,028	66,379	1,997,465	249,683	15,784	1,973
	Ottawa	29	115,741	16,534	294,163	42,023	3,949	564
	Toronto	76	319,772	39,972	1,042,519	130,315	9,727	1,216
	Vancouver	193	3,227,768	403,471	9,516,984	1,189,623	97,957	12,245
		442	4,255,712	115,019	13,110,709	354,343	129,394	3,497
United States	Chicago	150	1,565,585	195,698	4,331,155	541,394	57,523	7,190
	Los Angeles	92	737,312	92,164	2,230,790	278,849	25,312	3,164
	Minneapolis	141	1,099,937	137,492	3,488,596	436,075	37,073	4,634
	New York	131	1,489,207	186,151	4,816,974	602,122	46,701	5,838
	Seattle	103	611,945	76,493	1,714,882	214,360	20,893	2,612
		617	5,503,986	137,600	16,582,397	414,560	187,502	4,688
Western Europe	Copenhagen	52	693,116	86,640	1,718,421	214,803	24,523	3,065
	Geneva	95	447,208	55,901	1,499,989	187,499	14,288	1,786
	Heidelberg	105	967,739	120,967	3,356,342	419,543	36,038	4,505
	Lisbon	99	898,345	112,293	2,831,909	353,989	30,431	3,804
	London	113	762,009	95,251	2,506,812	313,352	27,410	3,426
	Madrid	17	200,642	33,440	425,073	70,846	7,334	1,222
	Paris	86	621,877	77,735	1,469,846	183,731	19,324	2,416
	Rome	75	282,064	35,258	1,033,858	129,232	10,407	1,301
		642	4,873,000	78,597	14,842,250	239,391	169,755	2,738
		1701	14,632,698	105,271	44,535,356	320,398	486,651	3,501

PROC TABULATE

- PROC Tabulate is another procedure that can be used to generate tabular summary reports
- Virtually anything that can be done in one can also be done in the other
- However, they both have particular strengths and make some tasks easier or better
- PROC Tabulate is fairly simple yet powerful for producing complex cross-tabulations

SIMPLE EXAMPLE

```
title 'Summary Report for Canada, US & Western Europe';
proc tabulate data=sashelp.shoes;
class region subsidiary ;
var stores sales inventory returns;
tables region*subsidiary,
       Stores *sum
       sales *(Sum mean)
       inventory *(sum mean)
       returns *(sum mean);

where region in ("Canada", "United States", "Western Europe");
run;
```

Finds all combinations of Region by Subsidiary

Finds the total number of Stores

Finds the sum and mean of Sales

Finds the sum and mean of Inventory

Finds the sum and mean of Returns

Subsets data

Simple Summary Report for Canada, US & Western Europe

		Number of Stores	Total Sales		Total Inventory		Total Returns	
		Sum	Sum	Mean	Sum	Mean	Sum	Mean
Region	Subsidiary							
Canada	Calgary	29.00	61403.00	10233.83	259578.00	43263.00	1977.00	329.50
	Montreal	115.00	531028.00	66378.50	1997465.00	249683.13	15784.00	1973.00
	Ottawa	29.00	115741.00	16534.43	294163.00	42023.29	3949.00	564.14
	Toronto	76.00	319772.00	39971.50	1042519.00	130314.88	9727.00	1215.88
	Vancouver	193.00	3227768.00	403471.00	9516984.00	1189623.00	97957.00	12244.63
United States	Chicago	150.00	1565585.00	195698.13	4331155.00	541394.38	57523.00	7190.38
	Los Angeles	92.00	737312.00	92164.00	2230790.00	278848.75	25312.00	3164.00
	Minneapolis	141.00	1099937.00	137492.13	3488596.00	436074.50	37073.00	4634.13
	New York	131.00	1489207.00	186150.88	4816974.00	602121.75	46701.00	5837.63
	Seattle	103.00	611945.00	76493.13	1714882.00	214360.25	20893.00	2611.63
Western Europe	Copenhagen	52.00	693116.00	86639.50	1718421.00	214802.63	24523.00	3065.38
	Geneva	95.00	447208.00	55901.00	1499989.00	187498.63	14288.00	1786.00
	Heidelberg	105.00	967739.00	120967.38	3356342.00	419542.75	36038.00	4504.75
	Lisbon	99.00	898345.00	112293.13	2831909.00	353988.63	30431.00	3803.88
	London	113.00	762009.00	95251.13	2506812.00	313351.50	27410.00	3426.25
	Madrid	17.00	200642.00	33440.33	425073.00	70845.50	7334.00	1222.33
	Paris	86.00	621877.00	77734.63	1469846.00	183730.75	19324.00	2415.50
	Rome	75.00	282064.00	35258.00	1033858.00	129232.25	10407.00	1300.88

APPEARANCE IMPROVEMENT

```
title 'Simple Summary Report for Canada, US & Western Europe';
proc tabulate data=sashelp.shoes ;
class region subsidiary ;
var stores sales inventory returns;
tables region*(subsidiary='' All='Sub Total')
       All = 'Total',
       Stores = ''*sum='Number of Stores'*F=6.
       sales = 'Sales'*(Sum='Total'*F=comma10. mean='Mean'*F=comma7.)
       inventory = 'Inventory'*(sum='Total'*F=comma10. mean='Mean'*F=comma7.)
       returns = 'Returns'*(sum='Total'*F=comma10. mean='Mean'*F=comma7.);

where region in ("Canada", "United States", "Western Europe");
run;
```

Adds Sub Total for each Region

Adds Total across Region

=‘ ‘ changes variable name

Format or F= changes the format to display

Simple Summary Report for Canada, US & Western Europe

		Sales		Inventory		Returns	
		Number of Stores	Total	Mean	Total	Mean	Total	Mean
Region								
Canada	Calgary	29	61,403	10,234	259,578	43,263	1,977	330
	Montreal	115	531,028	66,379	1,997,465	249,683	15,784	1,973
	Ottawa	29	115,741	16,534	294,163	42,023	3,949	564
	Toronto	76	319,772	39,972	1,042,519	130,315	9,727	1,216
	Vancouver	193	3,227,768	403,471	9,516,984	1189623	97,957	12,245
	Sub Total	442	4,255,712	115,019	13,110,709	354,343	129,394	3,497
United States	Chicago	150	1,565,585	195,698	4,331,155	541,394	57,523	7,190
	Los Angeles	92	737,312	92,164	2,230,790	278,849	25,312	3,164
	Minneapolis	141	1,099,937	137,492	3,488,596	436,075	37,073	4,634
	New York	131	1,489,207	186,151	4,816,974	602,122	46,701	5,838
	Seattle	103	611,945	76,493	1,714,882	214,360	20,893	2,612
	Sub Total	617	5,503,986	137,600	16,582,397	414,560	187,502	4,688
Western Europe	Copenhagen	52	693,116	86,640	1,718,421	214,803	24,523	3,065
	Geneva	95	447,208	55,901	1,499,989	187,499	14,288	1,786
	Heidelberg	105	967,739	120,967	3,356,342	419,543	36,038	4,505
	Lisbon	99	898,345	112,293	2,831,909	353,989	30,431	3,804
	London	113	762,009	95,251	2,506,812	313,352	27,410	3,426
	Madrid	17	200,642	33,440	425,073	70,846	7,334	1,222
	Paris	86	621,877	77,735	1,469,846	183,731	19,324	2,416
	Rome	75	282,064	35,258	1,033,858	129,232	10,407	1,301
	Sub Total	642	4,873,000	78,597	14,842,250	239,391	169,755	2,738
Total		1701	14,632,698	105,271	44,535,356	320,398	486,651	3,501

OUTPUT DELIVERY SYSTEM (ODS)

ODS is used to create other forms of output, such as HTML, RTF, or PDF. Sandwich the entire program or specific program

ODS RTF;

PROC print; run;

ODS RTF CLOSE;

- Use ODS HTML statement to create HTML output
- Use ODS RTF statement to create RTF output
- Use PDF statement to create PDF output

- STYLE option is available for different styles. SAS 9.4 provides 52 different styles

```
ods html style=minimal;
proc means data=sashelp.class maxdec=2;
 class sex;
 var age weight height;
 run;
ods html close;
```

Style=minimal

The MEANS Procedure

Sex	N Obs	Variable	N	Mean	Std Dev	Minimum	Maximum
F	9	Age	9	13.22	1.39	11.00	15.00
		Weight	9	90.11	19.38	50.50	112.50
		Height	9	60.59	5.02	51.30	66.50
M	10	Age	10	13.40	1.65	11.00	16.00
		Weight	10	108.95	22.73	83.00	150.00
		Height	10	63.91	4.94	57.30	72.00

Style=sasweb

The MEANS Procedure

Sex	N Obs	Variable	N	Mean	Std Dev	Minimum	Maximum
F	9	Age	9	13.22	1.39	11.00	15.00
		Weight	9	90.11	19.38	50.50	112.50
		Height	9	60.59	5.02	51.30	66.50
M	10	Age	10	13.40	1.65	11.00	16.00
		Weight	10	108.95	22.73	83.00	150.00
		Height	10	63.91	4.94	57.30	72.00

Style=magnify

The MEANS Procedure

Sex	N Obs	Variable	N	Mean	Std Dev	Minimum	Maximum
F	9	Age	9	13.22	1.39	11.00	15.00
		Weight	9	90.11	19.38	50.50	112.50
		Height	9	60.59	5.02	51.30	66.50
M	10	Age	10	13.40	1.65	11.00	16.00
		Weight	10	108.95	22.73	83.00	150.00
		Height	10	63.91	4.94	57.30	72.00

Style=journal

The MEANS Procedure

Sex	*N Obs*	*Variable*	*N*	*Mean*	*Std Dev*	*Minimum*	*Maximum*
F	9	Age	9	13.22	1.39	11.00	15.00
		Weight	9	90.11	19.38	50.50	112.50
		Height	9	60.59	5.02	51.30	66.50
M	10	Age	10	13.40	1.65	11.00	16.00
		Weight	10	108.95	22.73	83.00	150.00
		Height	10	63.91	4.94	57.30	72.00

THANKS