TECHNOLOGY SUPPORT

SAS 9.4- PART II

SHORTCOURSE HANDOUT

Texas Tech University | Heide Mansouri

Table of Contents

SAS 9.4 – Part II

Introduction

In this ShortCourse you will learn more about SAS Syntax-based programming in version 9.4 in Windows environment. Some SAS **mathematical** and **statistical functions**, as well as, some popular **statistical procedures** such as **PROC TTEST, PROC CORR**, and **PROC REG** will be discussed also.

In this ShortCourse it is assumed that you are familiar with **elementary statistics** and you have already taken **SAS ShortCourse – Part I**.

Credit: This document was adapted from **SAS Help and Documentation** and **SAS/STAT Documentation.**

Course Objectives

After completing this ShortCourse, you should be able to write SAS programs that

- Perform **Calculations** using Assignment Statement;
- Calculate the **MEAN**, and the **SUM Functions**;
- Use **Procedures** such as **PROC CORR**, **PROC REG**, and **PROC TTEST**; and
- Enhance your reports with the **Output Delivery System (ODS)**.

Starting SAS

- Click on **Start** button **> ALL Programs > SAS > SAS 9.4 (English)**

Arithmetic Operators in Assignment Statements

Assignment statement evaluates an expression (on the right side of the equal sin) and stores the result in a variable. One way to perform calculations on numeric variables is to write an assignment statement using arithmetic operators. Arithmetic operators indicate addition, subtraction, multiplication, division, and exponentiation.

Operators in Arithmetic Expressions		
Operation	Symbol	Example
addition	+	x = y + z;
subtraction	-	x = y - z;
multiplication	*	x = y * z
division	/	x = y / z
exponentiation	**	x = y ** z

Note: The asterisk (*) is always necessary to indicate multiplication; 2Y and 2(Y) are not valid expressions.

Using arithmetic operations

Remember that you define a variable; before you use it in an assignment statement that is, **order is important**. For example in the following example the syntax is correct, however, the logic is not correct:

```
data roster;
      height =(12*feet) + inches;
      Input First $ Last $ Feet Inches;
datalines;
Tim Smith        6    2
Alice Young      5    4
;
run;
proc print data=roster;
run;
```

Here, the variable "**height**" will not be created since the "height" variable is defined before the variable "**feet**" and "inches" were defined in the **INPUT** statement. However, the following correct program will produce an output for "height":

```
data roster;
      Input First $ Last $ Feet Inches;
      height =(12*feet) + inches;
datalines;
Tim Smith        6    2
```

```
Alice Young     5    4
;
run;
proc print data=roster;
run;
```

Obs	First	Last	Feet	Inches	height
1	Tim	Smith	6	2	74
2	Alice	Young	5	4	64

SAS Functions

A SAS function performs a computation or manipulation on variables (arguments) and returns a value. Most functions use arguments supplied by the user. SAS functions are mainly used in DATA step programming statements.

Note: The argument list can consist of a variable list, which is preceded by **OF**.

Examples:

SAS Statements	Results
`x1=sum(4,9,3,8);`	24
`x2=sum(4,9,3,8,.);`	24

`x1=9;`
`x2=39;`
`x3=sum(of x1-x2);`
	48

`x1=5; x2=6; x3=4; x4=9;`
`y1=34; y2=12; y3=74; y4=39;`
`result=sum(of x1-x4, of y1-y5);`
	183

`x1=55;`
`x2=35;`
`x3=6;`
`x4=sum(of x1-x3, 5);`
	101

`x1=7;`
`x2=7;`
`x5=sum(x1-x2);`
	0

`y1=20;`
`y2=30;`
`x6=sum(of y:);`
	50

Example: **Creating expressions, or New Variables**

```
DATA EXP;
      INPUT SOIL $ TRT COUNT1 COUNT2;
      AVGCNT = (COUNT1 + COUNT2)/2;
      RESPONSE = SQRT(COUNT1 * COUNT2) - LOG(COUNT2);
```

Here, the new variable **AVGCNT** is the computed average of **COUNT1** and **COUNT2**.

By using parentheses, SAS is forced to add COUNT1 and COUNT2 first before dividing by 2.

More Examples:

- Name='Amanda Jones';
- a=a+b;

Exercise#1: Creating a SAS data set "One"

```
data one;
      input x1-x4;
datalines;
1        2     3      4
13.75    .     5      7
0.5      .     .      8
;
run;
proc print data=one;
run;
```

Output:

Obs	x1	x2	x3	x4
1	1.00	2	3	4
2	13.75	.	5	7
3	0.50	.	.	8

Exercise#2: Using the data set "One" and the SUM Function, for a single observation across variables, we can write the following programs:

```
data sums;
     set one;
     total1 = x1 + x2 + x3 + x4;
     total2 = sum(of x1-x4);
     total3 = sum(x1, x2, x3, x4);
run;
proc print data=sums;
run;
```

Output:

Obs	x1	x2	x3	x4	total1	total2	total3
1	1.00	2	3	4	10	10.00	10.00
2	13.75	.	5	7	.	25.75	25.75
3	0.50	.	.	8	.	8.50	8.50

Note: **total1** returns missing values' result for missing values; however, the **SUM** function used for **total2** and **total3** returns the sum of non-missing values. That is if you choose addition, you will get a missing value for the result if any of the fields are missing. Deciding which one of the above functions is appropriate depends upon your needs. However, there is an advantage to using the **SUM** function even if you want the results to be missing.

Exercise#3: Creating a new data set from an existing data set created in exercise #1

```
data means;
   set one;
   mean1 = (x1+x2+x3+x4)/4;
   mean2 = mean(of x1-x4);
   mean3 = mean(x1, x2, x3, x4);
run;
proc print data = means;
run;
```

Output:

Obs	x1	x2	x3	x4	mean1	mean2	mean3
1	1.00	2	3	4	2.5	2.50000	2.50000
2	13.75	.	5	7	.	8.58333	8.58333
3	0.50	.	.	8	.	4.25000	4.25000

Note: To get these results down columns (for a single variable down observations), use **proc univariate**, or **Proc means**, among other things.

Creating High-Resolution Histograms

- A histogram is similar to a vertical bar chart. This type of bar chart emphasizes the individual ranges of continuous numeric variables and enables you to examine the distribution of your data.
- The **HISTOGRAM** statement in **PROC UNIVARIATE** produces **histograms**. PROC UNIVARIATE creates a histogram by dividing the data into intervals of

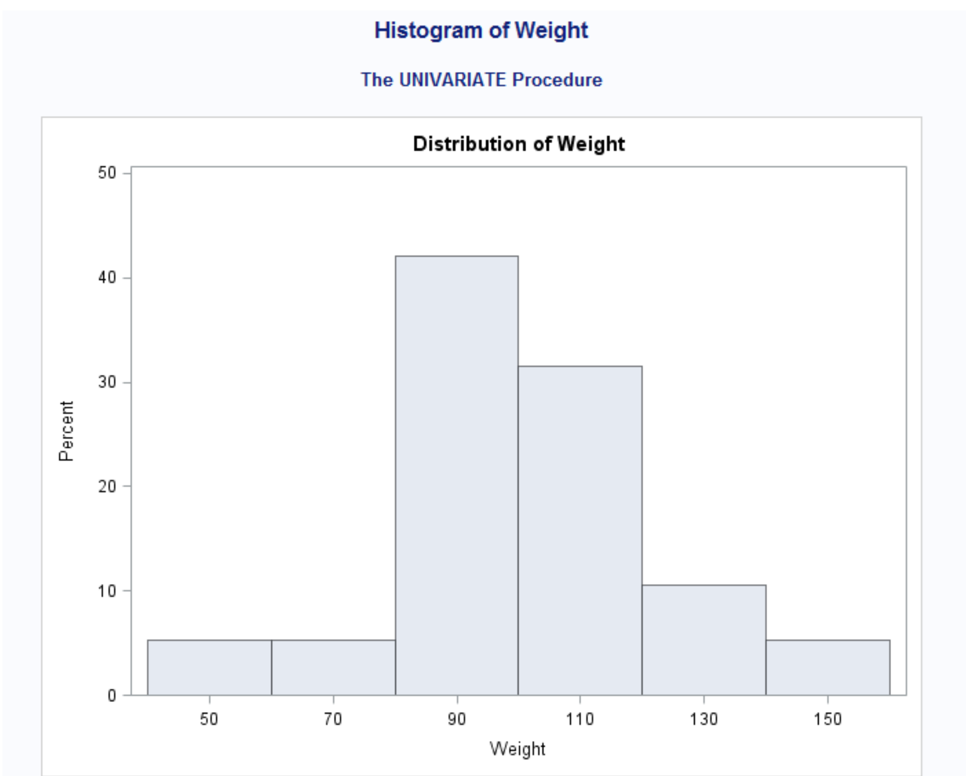
equal lengths, counting the number of observations in each interval, and plotting the counts as vertical bars that are centered on the midpoint of each interval.

- If you use the **HISTOGRAM** statement without any options, then PROC UNIVARIATE automatically does the following:
    - Scales the vertical axis to show the **percentage** of observations in an interval.
    - Labels the axes.

Exercise # 4: Using the SAS sample data set **CLASS** to create a Simple Histogram.

Submit the following program:

```
proc univariate data=sashelp.class noprint;
histogram weight;
title 'Histogram of Weight';
run;
title;* NOPRINT option suppresses the descriptive statistics that the
PROC UNIVARIATE statement creates;
```



ODS Graphics in SAS

ODS Statistical Graphics (or ODS Graphics) is new functionality for creating statistical graphics that is available in a number of SAS software products, including the **SAS/STAT**, and **SAS/GRAPH** products. ODS Graphics is an extension of **ODS** (**Output Delivery System**), which manages procedure output and lets you display it in a variety of destinations, such as **HTML**, **RTF,** and **PDF**.

ODS DESTINATIONS
For most ODS destinations (including **HTML**, **RTF**, and **PDF**), graphs and tables are integrated in the output, and you view your output with an appropriate viewer, such as a Web browser for HTML. If you are using the LISTING destination in the SAS windowing environment, you view you graphs individually by clicking the graph icons in the **Results window**.

STATISTICAL PROCEDURES THAT SUPPORT ODS GRAPHICS IN SAS 9.2

The following statistical procedures have been enhanced to support ODS Graphics in SAS 9.2:

Base SAS	SAS/STAT		SAS/QC	SAS/ETS
CORR	ANOVA	MI	ANOM	ARIMA
FREQ	BOXPLOT	MIXED	CAPABILITY	AUTOREG
UNIVARIATE	CALIS	MULTTEST	CUSUM	ENTROPY
	CLUSTER	NPAR1WAY	MACONTROL	EXPAND
	CORRESP	PHREG	PARETO	MODEL
	FACTOR	PLS	RELIABILITY	PANEL
	FREQ	PRINCOMP	SHEWHART	RISK
	GAM	PRINQUAL		SIMILARITY
	GENMOD	PROBIT		SYSLIN
	GLIMMIX	QUANTREG		TIMESERIES
	GLM	REG		UCM
	GLMSELECT	ROBUSTREG		VARMAX
	KDE	RSREG		X12
	KRIGE2D	SEQDESIGN		
	LIFEREG	SEQTEST		
	LIFETEST	SIM2D		
	LOESS	TCALIS		
	LOGISTIC	TRANSREG		
	MCMC	TTEST		
	MDS	VARIOGRAM		

Source: http://support.sas.com/rnd/app/papers/intodsgraph.pdf

Exercise#5: To create a vertical bar chart where the height of the bars represent the frequency count of the values of the chart variable, for each category, submit the following program:

```
data shirts;
    input size $ @@;
datalines;
medium      large
large       large
large       medium
medium      small
small       medium
medium      large
small       medium
large       large
large       small
medium      medium
medium      medium
medium      large
small       small
;
run;
proc chart data=shirts;
    vbar size;
    title 'Vertical bar of the number of each shirt size sold';
run;
proc gchart data=shirts;
    vbar size;
    title 'Bar Chart of the number of each shirt size sold';
run;
proc gchart data=shirts;
    block size;
    title 'Block Chart of the number of each shirt size sold';
run;
title;
```

Exercise#6: Using the Shirts data set created in previous exercise; create a vertical bar chart such that chart statistic is the percentage for each category of the total number of shirts sold.

```
proc gchart data=shirts;
    vbar size / type=percent;
    title 'Percentage of Total Sales for Each Shirt Size sold';
run;
title;
```

Pearson Correlation Coefficient

Pearson correlation coefficient is a parametric measure of a **linear relationship** between two variables. It measures both the **strength** and **direction** of a linear relationship. If one variable is an exact linear function of another variable, a positive relationship exists if the correlation is 1 and a negative relationship exists if the correlation is -1. If there is no linear relationship between the two variables, the correlation is 0.

You should always verify whether there is a linear relationship between two variables before computing a **Pearson Correlation Coefficient** for those variables. The easiest way to verify that the relationship is linear is to prepare a **scatter plot** of the two variables using **PROC GPLOT** (GPLOT uses ODS and creates high- resolution graphs).

Exercise#7: Using the PROC CORR, compute the Pearson Correlation Coefficients, and plot the given data to verify the linear relationship between variables, and to identify the potential outliers, using the SAS sample Fitness data set.

```
data Fitness;
    input Age Weight Oxygen Runtime @@;
datalines;
44 89.47 44.609 11.37    40 75.07 45.313 10.07
44 85.84 54.297  8.65    42 68.15 59.571  8.17
38 89.02 49.874   .      47 77.45 44.811 11.63
40 75.98 45.681 11.95    43 81.19 49.091 10.85
44 81.42 39.442 13.08    38 81.87 60.055  8.63
44 73.03 50.541 10.13    45 87.66 37.388 14.03
45 66.45 44.754 11.12    47 79.15 47.273 10.60
54 83.12 51.855 10.33    49 81.42 49.156  8.95
51 69.63 40.836 10.95    51 77.91 46.672 10.00
48 91.63 46.774 10.25    49 73.37   .    10.08
57 73.37 39.407 12.63    54 79.38 46.080 11.17
52 76.32 45.441  9.63    50 70.87 54.625  8.92
51 67.25 45.118 11.08    54 91.63 39.203 12.88
51 73.71 45.790 10.47    57 59.08 50.545  9.93
49 76.32   .      .      48 61.24 47.920 11.50
52 82.78 47.467 10.50
;
run;
proc corr data=Fitness plots=matrix(histogram);
run;
```

The CORR Procedure

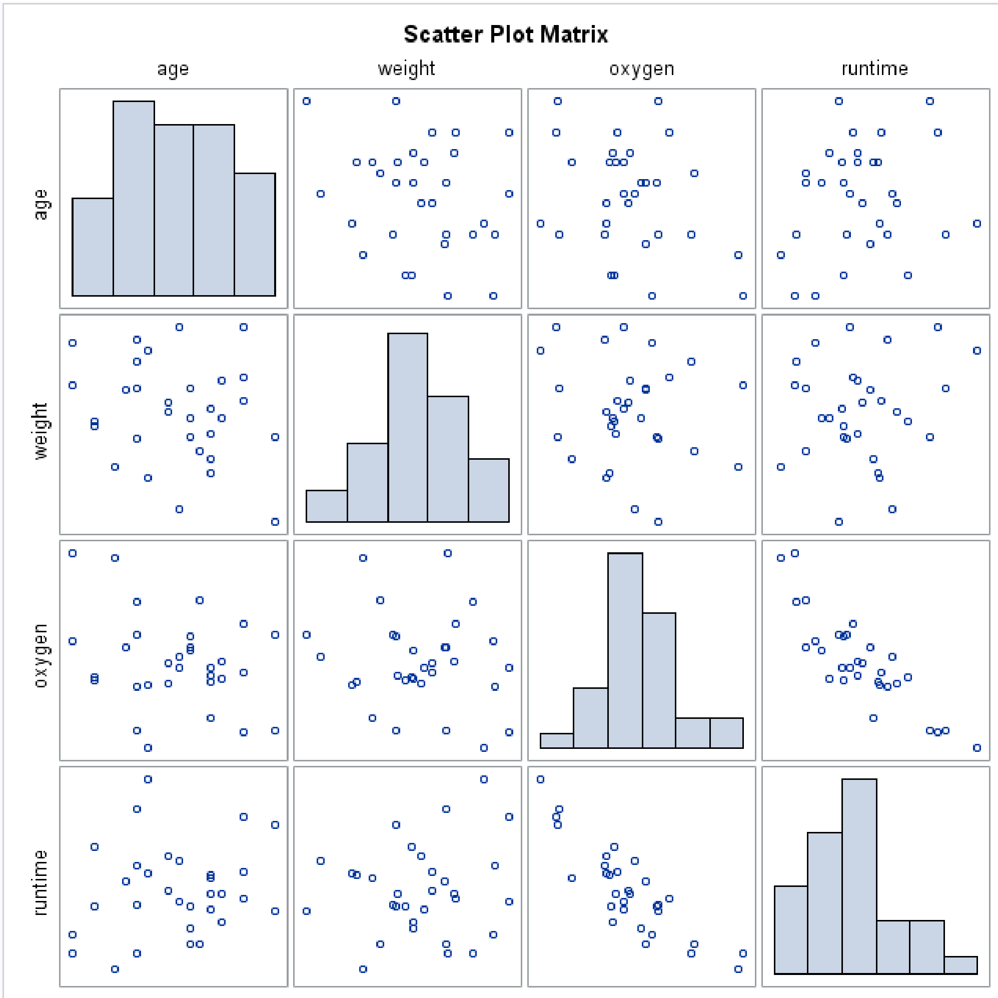
4 Variables:	Age Weight Oxygen RunTime

Simple Statistics

Variable	N	Mean	Std Dev	Sum	Minimum	Maximum
Age	31	47.67742	5.21144	1478	38.00000	57.00000
Weight	31	77.44452	8.32857	2401	59.08000	91.63000
Oxygen	29	47.22721	5.47718	1370	37.38800	60.05500
RunTime	29	10.67414	1.39194	309.55000	8.17000	14.03000

Pearson Correlation Coefficients
Prob > |r| under H0: Rho=0
Number of Observations

	Age	Weight	Oxygen	RunTime
Age	1.00000	-0.23354	-0.31474	0.14478
		0.2061	0.0963	0.4536
	31	31	29	29
Weight	-0.23354	1.00000	-0.15358	0.20072
	0.2061		0.4264	0.2965
	31	31	29	29
Oxygen	-0.31474	-0.15358	1.00000	-0.86843
	0.0963	0.4264		<.0001
	29	29	29	28
RunTime	0.14478	0.20072	-0.86843	1.00000
	0.4536	0.2965	<.0001	
	29	29	28	29



Results: By default, Pearson correlation statistics are computed from observations with non-missing values for each pair of analysis variables.

A correlation of **- 0.86843** between ***Runtime*** and ***Oxygen***, is significant with a p-value less than 0.0001. That is, there exists an inverse linear relationship between these two variables. As ***Runtime*** (time to run 1.5 miles in minutes) increases, ***Oxygen*** (oxygen intake, ml per kg body weight per minute) decreases.

When you use the **PLOTS=MATRIX(HISTOGRAM)** option, the CORR procedure displays a **symmetric matrix** plot for the analysis variables. The histograms for these analysis variables are also displayed on the diagonal of the matrix plot. This inverse linear relationship between the two variables, **Oxygen** and **Runtime**, is also shown in the plot.

Exercise#8: Scatter Plots with Prediction Ellipses

Submit the following program to request a Scatter plot with Prediction Ellipses, Using the SAS Sample dataset, **Fish**.

```
proc corr data=SASHelp.fish plots=scatter (alpha=0.2 0.3);
      var Height Width;
run;
```

The CORR Procedure

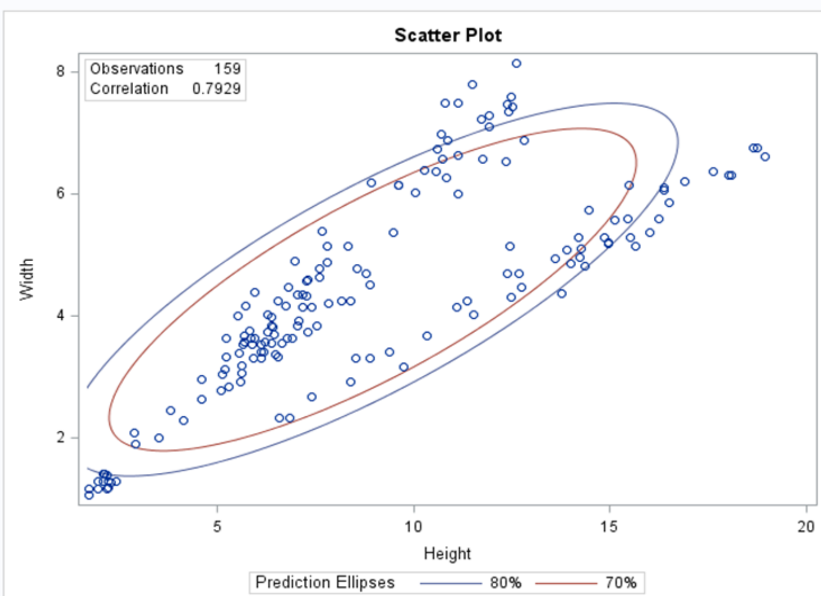
2 Variables:	Height Width

Simple Statistics						
Variable	**N**	**Mean**	**Std Dev**	**Sum**	**Minimum**	**Maximum**
Height	159	8.97099	4.28621	1426	1.72840	18.95700
Width	159	4.41749	1.68580	702.38020	1.04760	8.14200

Pearson Correlation Coefficients, N = 159
Prob > |r| under H0: Rho=0

	Height	Width
Height	1.00000	0.79288 <.0001
Width	0.79288 <.0001	1.00000

The CORR Procedure



Simple Linear Regression

Regression analysis is the analysis of the relationship between one variable and another set of variables.

Suppose that a response variable Y can be predicted by a linear function of a regressor variable X. You can estimate β_0, the intercept, and β_1, the slope, in

$y_i = \beta_0 + \beta_1 x_i + \varepsilon$ for the observations *i=1, 2,…., n*.

For example, you might use regression analysis to find out how well you can predict a child's weight if you know that child's height. Then the equation of interest is

$\text{Weight} = \beta_0 + \beta_1 \text{Height} + \varepsilon$

The variable Weight is the response or dependent variable, and the variable Height is the regressor or independent variable, β_0 and β_1 are the unknown parameters to be estimated, and ε is the unknown error.

For Regression analysis, we use the following MODEL statement, where y is the outcome variable and x is the regressors variable.

```
proc reg;
    model y=x;
run;
```

Exercise#9: Use regression analysis to find out how well you can predict a child's weight if you know that child's height.

The SAS CLASS sample data set is from a study of nineteen children. Height and weight are measured for each child (Source: SAS Help and Documentation).

The equation of interest is $\text{Weight} = \beta_0 + \beta_1 \text{Height} + \varepsilon$

Submit the following program:

```
proc reg data=SASHelp.class;
      model weight=height;
      plot weight*height;
      title link='http://sas.com' ' Simple Linear Regression';
run;
title;
```

Simple Linear Regression

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Number of Observations Read	19
Number of Observations Used	19

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	7193.24912	7193.24912	57.08	<.0001
Error	17	2142.48772	126.02869		
Corrected Total	18	9335.73684			

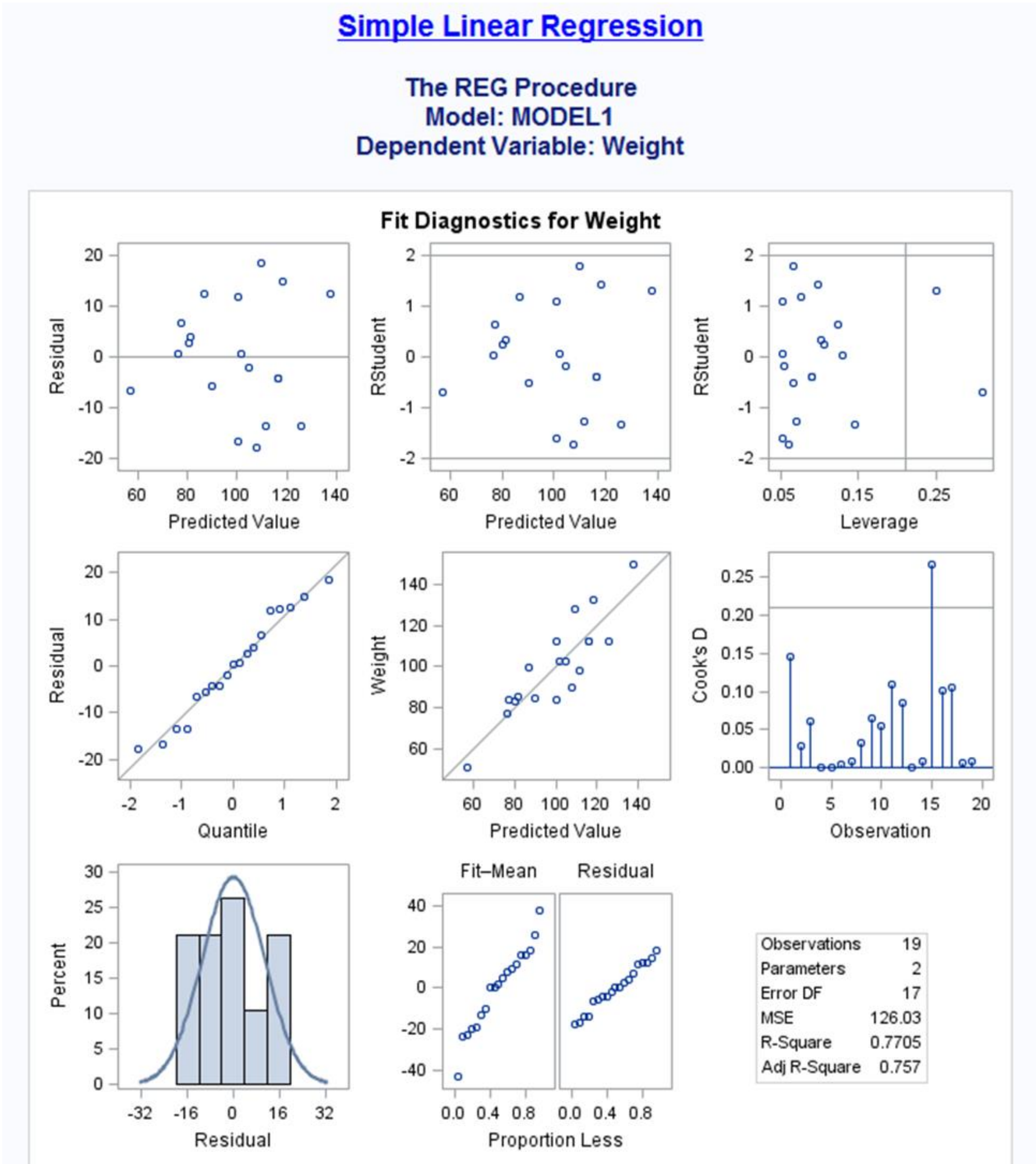
Root MSE	11.22625	R-Square	0.7705
Dependent Mean	100.02632	Adj R-Sq	0.7570
Coeff Var	11.22330		

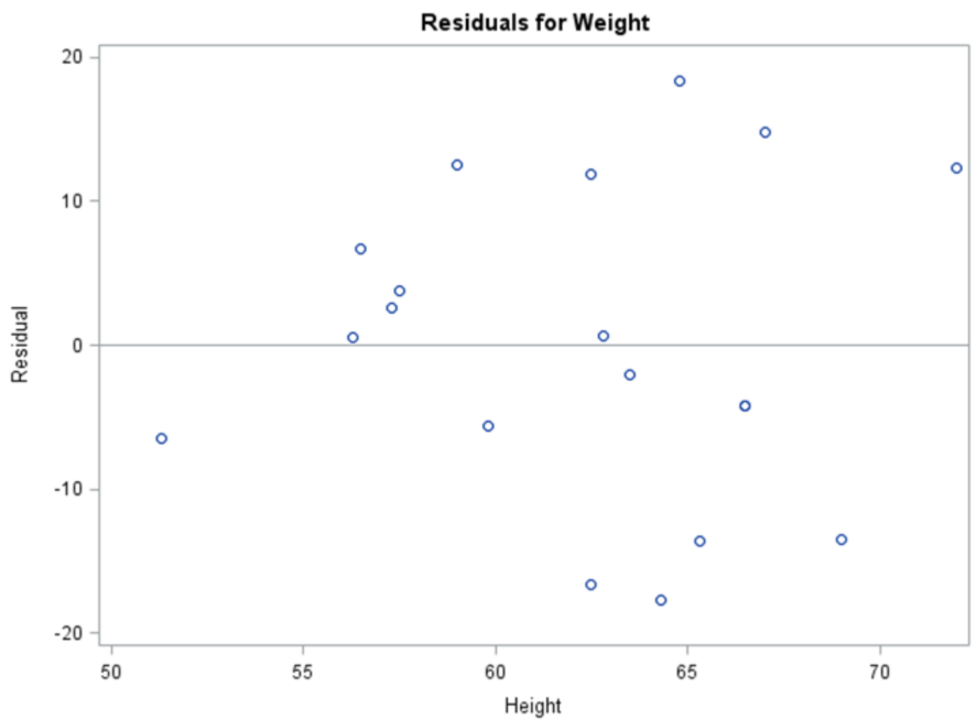
Parameter Estimates

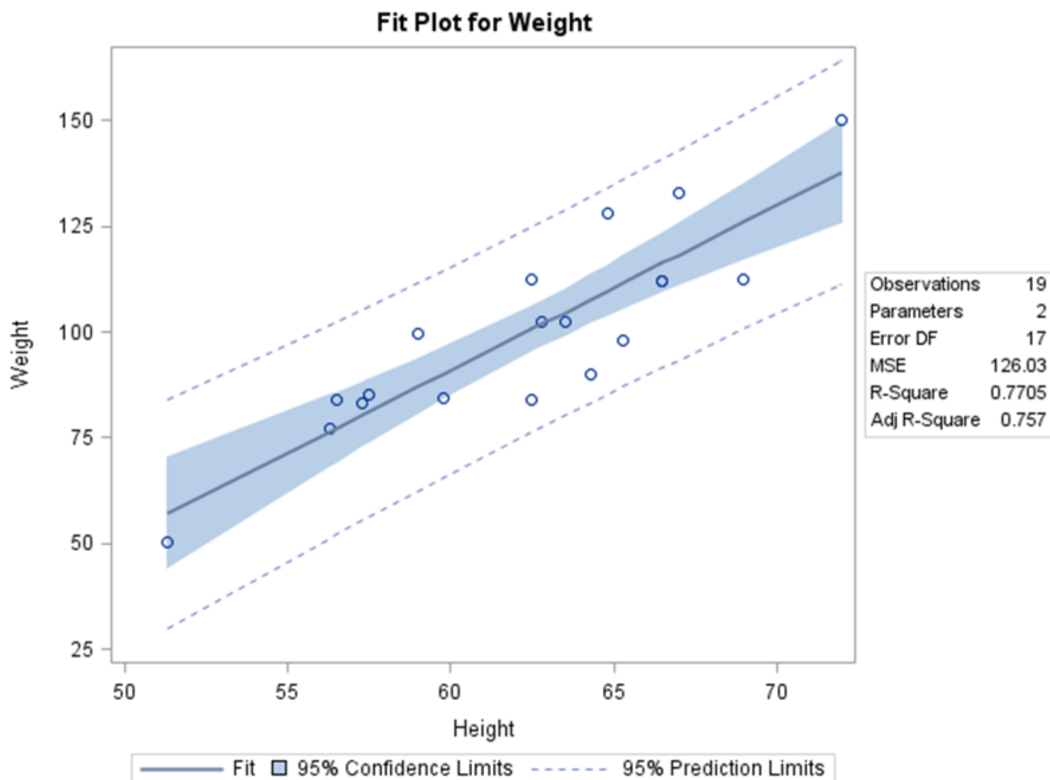
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

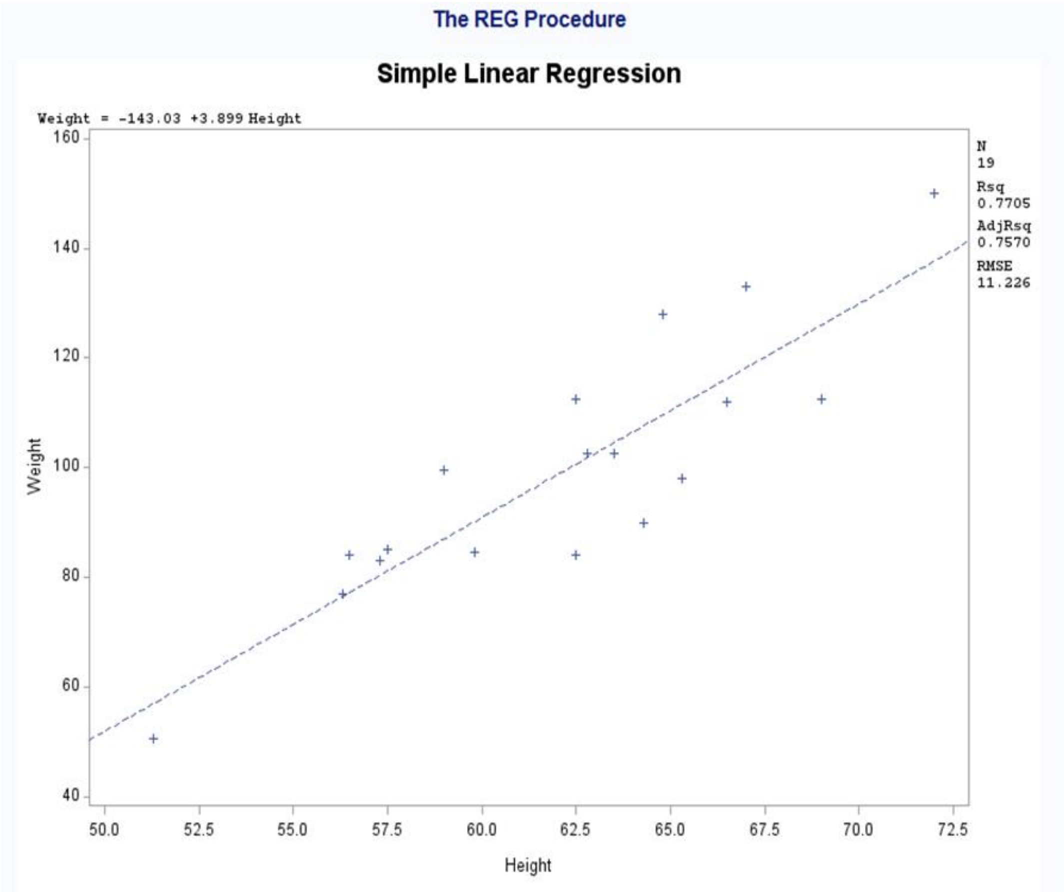
The "**Parameter Estimates**" table contains the t- statistics and the corresponding p-values for testing whether each parameter is significantly different from zero. The p-values (t = - 4.43, p = 0.0004 and t = 7.55, p< 0.0001) indicate that the intercept and *Height* parameter estimates, respectively, are highly significant. From the parameter estimates, the **fitted model** is:

Weight = -143.0 + 3.9 x Height


Simple Linear Regression
The REG Procedure
Model: MODEL1
Dependent Variable: Weight
Fit Diagnostics for Weight
Residual
Predicted Value
RStudent
Leverage
Quantile
Weight
Cook's D
Observation
Percent
Fit–Mean
Proportion Less
Observations 19
Parameters 2
Error DF 17
MSE 126.03
R-Square 0.7705
Adj R-Square 0.757


Residuals for Weight
Residual
20
10
0
-10
-20
50
55
60
65
70
Height


Fit Plot for Weight
Weight
150
125
100
75
50
25
50
55
60
65
70
Height
Observations 19
Parameters 2
Error DF 17
MSE 126.03
R-Square 0.7705
Adj R-Square 0.757
Fit
95% Confidence Limits
95% Prediction Limits



Results: The F- statistic for the overall model is highly significant (F=57.076, P <0.0001), indicating that the model explains a significant portion of the variation in the data.

The model degrees of freedom are one less than the number of parameters to be estimated. This model estimates two parameters β_0, and β_1; thus, the degrees of freedom for model should be 2-1=1. The corrected total degrees of freedom are always one less than the total number of observations in the data set, in this case 19-1=18.

The Root MSE is an estimate of the standard deviation of the error term. The coefficient of variation, or Coeff Var, is a unit-less expression of the variation in the data. The R-square and Adj R-square are two statistics used in assessing the fit of the model; values close to 1 indicate a better fit. The R-square of 0.77 indicates that *Height* accounts for 77% of the variation in *Weight (source: SAS help and Documentation)*.

The T-Test Procedure

The **TTEST procedure** performs ***t*-tests** and computes confidence limits for **one sample**, **two samples**, and **paired** observations.

- The **one-sample *t*-test** compares the mean of the sample to a given number.
- The **two-sample *t*-test** compares the mean of the first sample minus the mean of the second sample to a given number.
- The **paired observations *t*-test** compares the mean of the differences in the observations to a given number.

The underlying assumption of the *t*-test in all three cases is that the observations are random samples drawn from normally distributed populations. In the case of paired t-test, the differences constitute a random sample from a normal distribution. This assumption can be checked using the UNIVARIATE procedure (using **normal probability plot**).

Exercise #10: Submit the following program to perform a **One-Sample t Test**; to test whether the mean length of a certain type of court case is 80 days using 20 randomly chosen cases (time, is assumed to be normally distributed). This example is taken from SAS/STAT Documentation.

```
data Time;
    input time @@;
datalines;
43 90 84 87 116 95 86 99 93 92
121 71 66 98 79 102 60 112 105 98
;
run;
proc ttest data=Time h0=80 plots(showh0) sides=U alpha=0.1; /* sides=U provides upper confidence limit
                                                               (Unbound side of one sided interval) */
    var time;
    title 'One Sample T-test';
run;
title;
```

Here, the only variable in the data set, **time**, is assumed to be normally distributed. The trailing @ signs (@@) indicate that there is more than one observation on a line. The PROC TTEST is used for a one-sample *t* test. The H_0= option specifies that the mean of the time variable should be compared to the value 80. This ALPHA= 0.10 option requests 90% confidence.

One-Sample t Test Results

- Summary statistics appear at the top of output.
- The sample size (N)=20
- Due to the sides=u option, the interval for the mean is an upper one-sided interval of 84.1659
- The standard deviation and its confidence bounds (Lower CL Std Dev and Upper CL Std Dev) and the standard error are displayed with the minimum and maximum values of the time variable.
- The test statistic t=2.30, degrees of freedom df =19, and probability of p=0.0164, at the 10% α-level indicates that the mean length of the court cases are significantly greater than 80 days.

One-Sample t Test

The TTEST Procedure

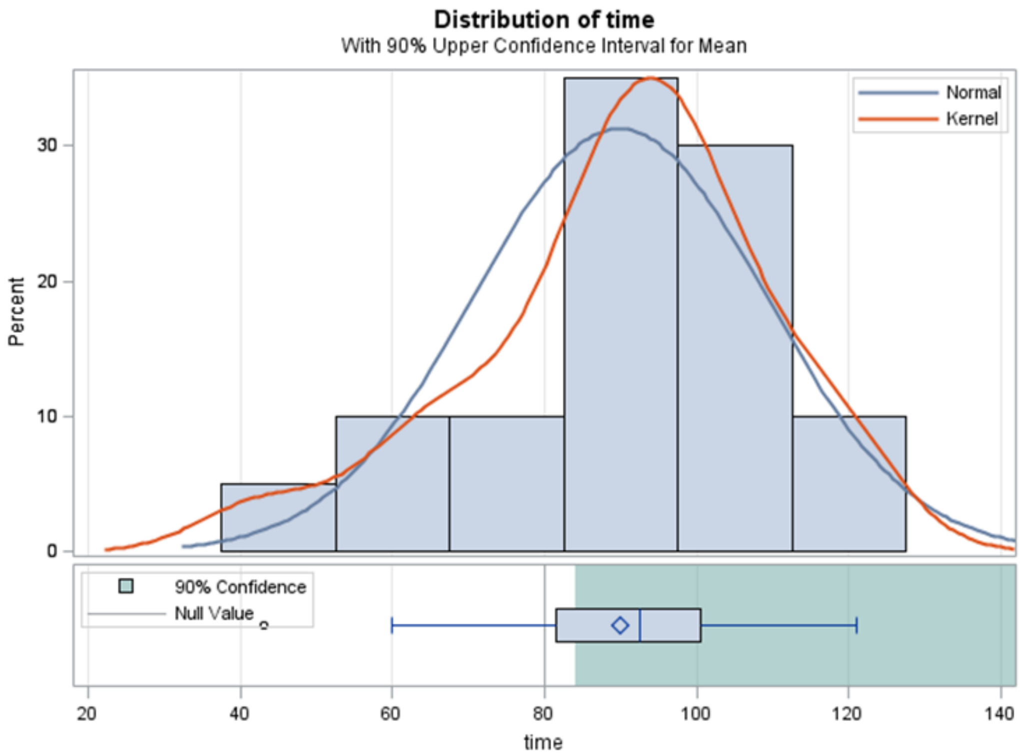
Variable: time

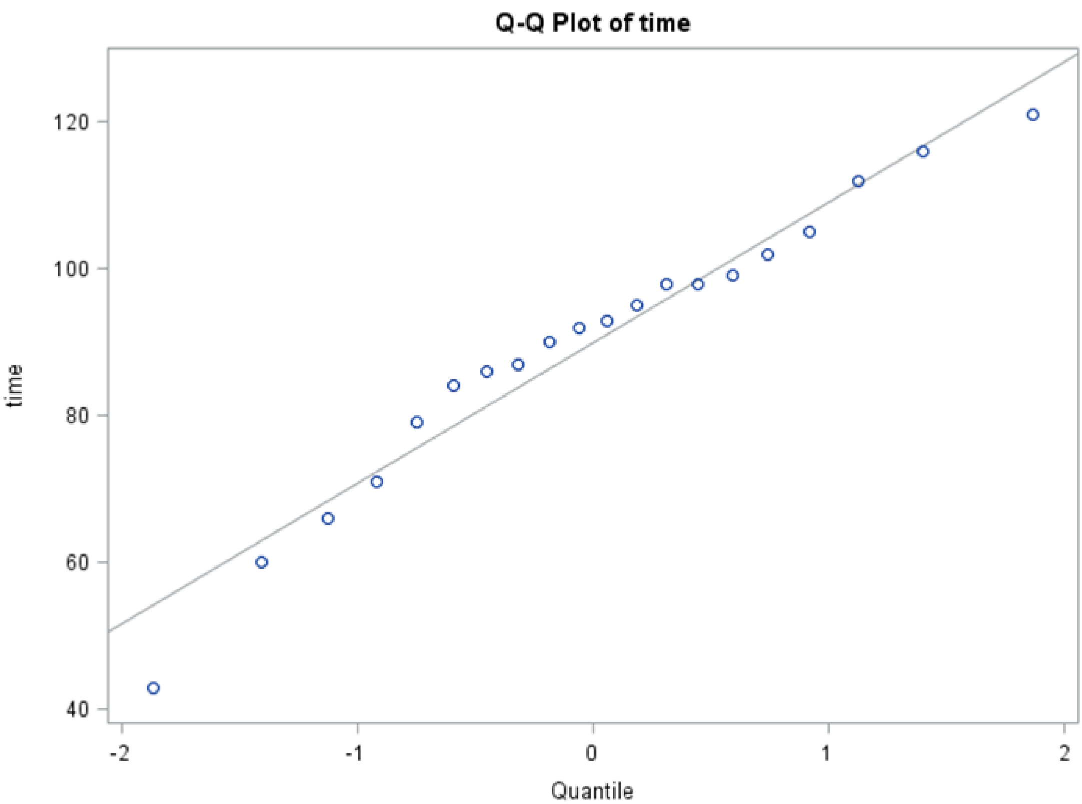
N	Mean	Std Dev	Std Err	Minimum	Maximum
20	89.8500	19.1456	4.2811	43.0000	121.0

Mean	90% CL Mean		Std Dev	90% CL Std Dev	
89.8500	84.1659	Infty	19.1456	15.2002	26.2374

DF	t Value	Pr > t
19	2.30	0.0164

Note: a **quantile-quantile plot (Q-Q plot)**, compares ordered values of a variable with quantiles of a specified theoretical distribution such as the normal. If the data distribution matches the theoretical distribution, the points on the plot form a linear pattern. Thus, you can use a Q-Q plot to determine how well a theoretical distribution models a set of measurements.


Distribution of time
With 90% Upper Confidence Interval for Mean
Normal
Kernel
Percent
0
10
20
30
90% Confidence
Null Value
20
40
60
80
100
120
140
time


Q-Q Plot of time
time
40
60
80
100
120
-2
-1
0
1
2
Quantile

Exercise #11: Paired Comparisons

Suppose that a stimulus is being examined to determine its effect on systolic blood pressure. Twelve men participate in the study. Their systolic blood pressure is measured both before and after the stimulus is applied. The variables **SBPbefore** and **SBPafter** denote the systolic blood pressure before and after the stimulus, respectively.

Submit the following program to test whether the mean change in systolic blood pressure is significantly different from zero. This example is taken from SAS/STAT Documentation.

```
data pressure;
    input SBPbefore SBPafter @@;
datalines;
120 128   124 131   130 131   118 127
140 132   128 125   140 141   135 137
126 118   130 132   126 129   127 135
;
run;
proc ttest data=pressure;
    paired  SBPbefore*SBPafter;
    title 'Paired Comparison';
run;
title;
```

Paired Comparison

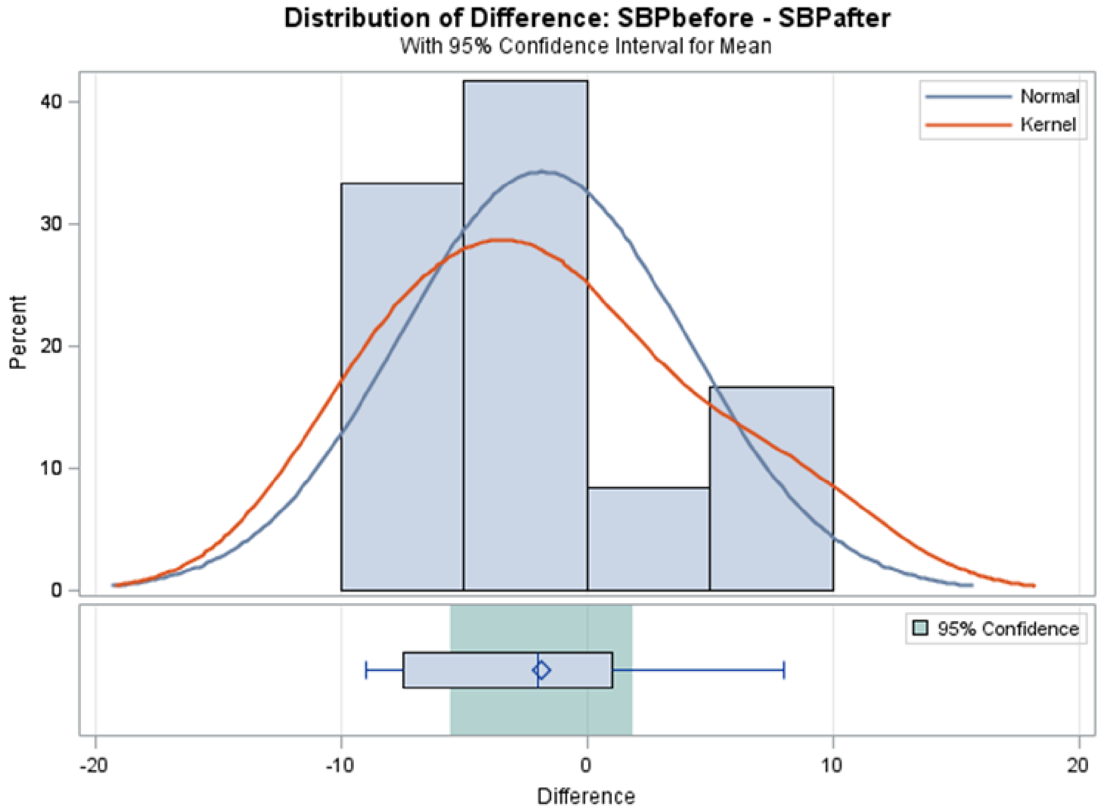
The TTEST Procedure

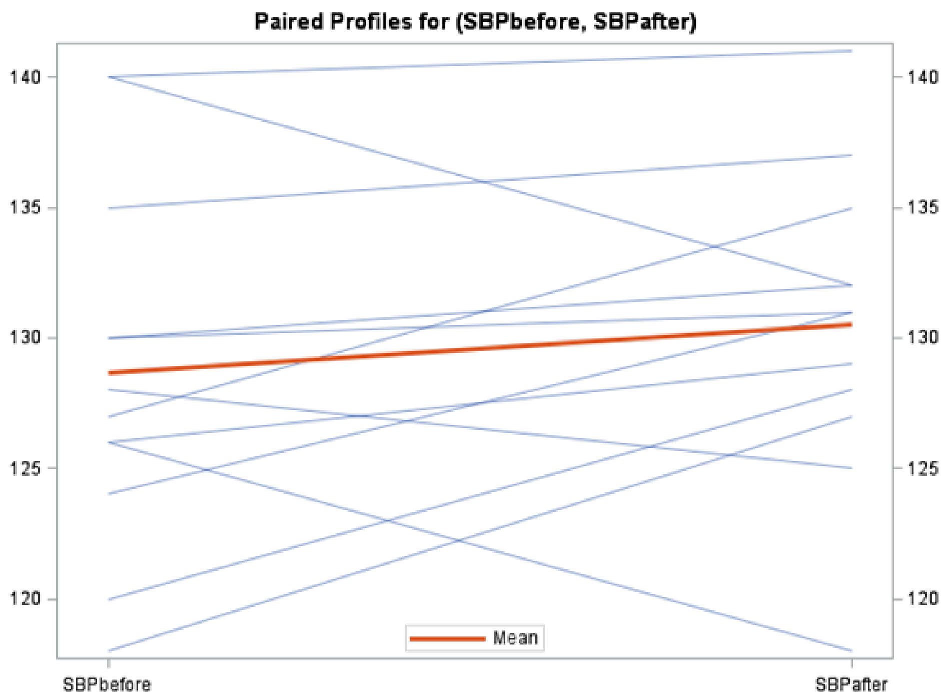
Difference: SBPbefore - SBPafter

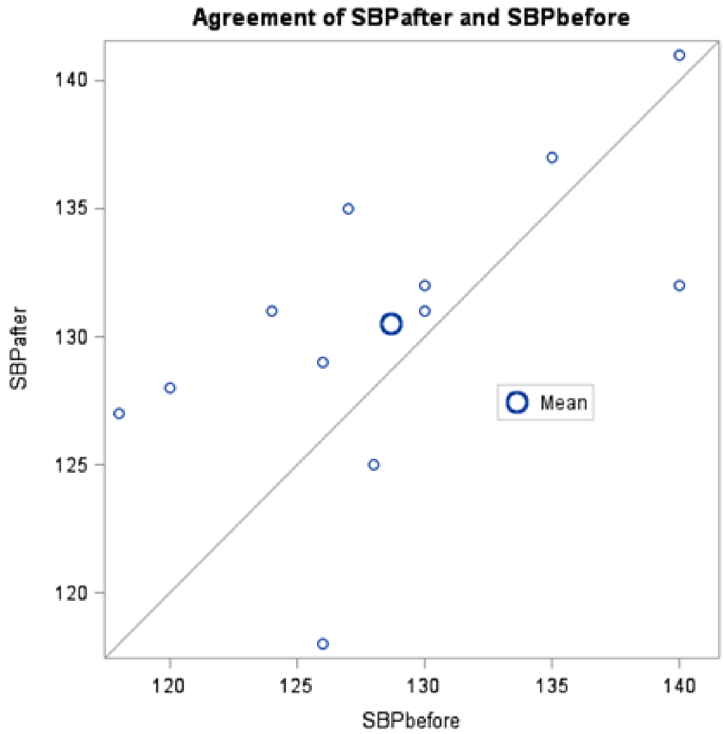
N	Mean	Std Dev	Std Err	Minimum	Maximum
12	-1.8333	5.8284	1.6825	-9.0000	8.0000

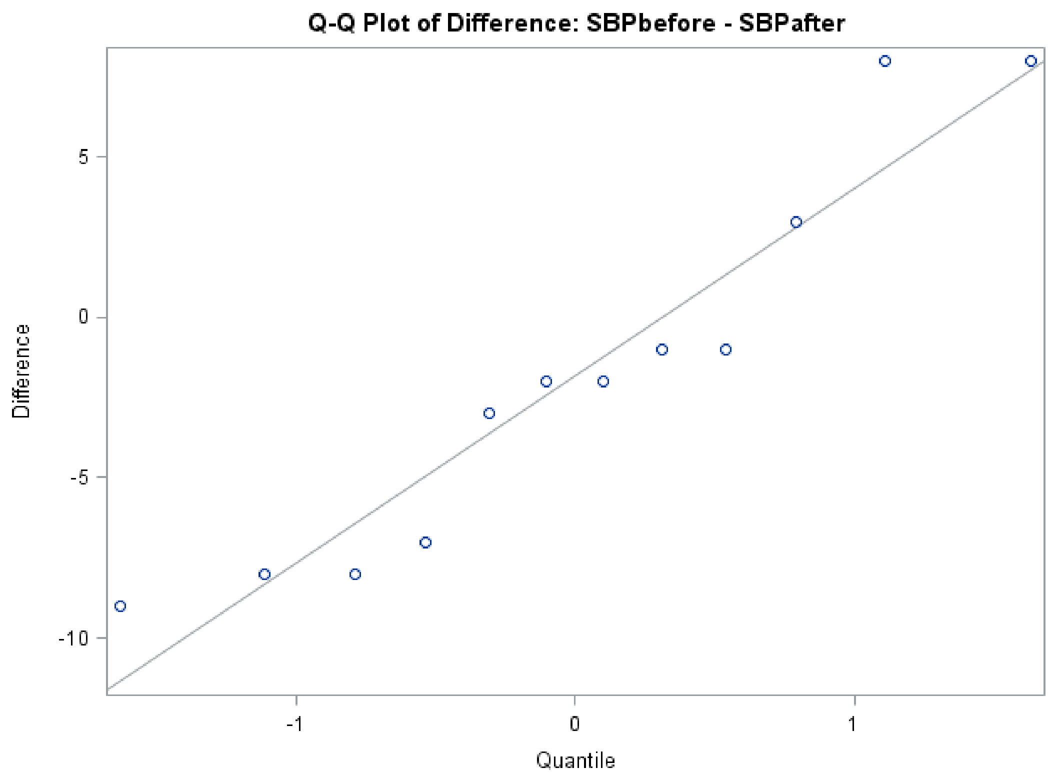
Mean	95% CL Mean		Std Dev	95% CL Std Dev	
-1.8333	-5.5365	1.8698	5.8284	4.1288	9.8958

DF	t Value	Pr > |t|
11	-1.09	0.2992


Distribution of Difference: SBPbefore - SBPafter
With 95% Confidence Interval for Mean
Normal
Kernel
Percent
40
30
20
10
0
95% Confidence
-20
-10
0
10
20
Difference


Paired Profiles for (SBPbefore, SBPafter)
140
135
130
125
120
Mean
SBPbefore
SBPafter


Agreement of SBPafter and SBPbefore
SBPafter
SBPbefore
140
135
130
125
120
120
125
130
135
140
Mean


Q-Q Plot of Difference: SBPbefore - SBPafter
Difference
Quantile
5
0
-5
-10
-1
0
1

Paired Comparison t-Test Results

- The variables SBPbefore and SBPafter are the paired variables with a sample size of 12.
- The summary statistics of the difference are displayed (mean, standard deviation, and standard error) along with their confidence limits.
- The minimum and maximum differences are also displayed.
- The test is **not significant,** indicating that the stimuli did not significantly affect systolic blood pressure.
- The summary panel in shows a histogram, normal densities, box plot, and confidence interval of the SBPbefore SBPafter difference.

Exporting the Output from SAS to an Excel File (only one-way tables)

- Outputs saved in HTML format can be exported to Excel.
- Go to the "**Results Viewer**" to view the html output.
- Right-click on an output, and then use **Export to Excel** option.

Online Resources:

- SAS/STAT Documentations

  http://support.sas.com/documentation/onlinedoc/stat/
- SAS Information Guide http://www.psych.yorku.ca/lab/sas/index.htm#Start
- SAS Training Video Tutorials http://support.sas.com/training/video
- Statistical Software Information http://www.umass.edu/statdata/software/

Please e-mail your comments or suggestions to: heide.mansouri@ttu.edu