Introduction to SAS

SAS (https://en.wikipedia.org/wiki/SAS_(software)) was developed in the early 1970s at North Carolina State University. It is originally intended for management and analysis of agricultural field experiments and now it is the most widely used statistical software. SAS is used to stand for "Statistical Analysis System", but now it is not an acronym for anything. SAS is pronounced "sass", not spelled out as three letters.

SAS is the most widely used statistical software due to the following several reasons:

- It is often better with very large datasets and memory;
- It can deal with multiple datasets at the same time.
- It is better for data manipulation.
- It is better on the Job Market.

The SAS software suite has more than 200 components (https://support.sas.com/software/). Some of the SAS components include:

- Base SAS – Basic procedures and data management
- SAS/STAT – Statistical analysis
- SAS/IML – Interactive matrix language

We will use Base SAS and SAS/STAT in this course to manipulate data and conduct basic statistical data analysis. SAS/IML will be used a little bit in our course.

SAS/IML (https://support.sas.com/) software, is like statistical software R, gives you access to a powerful and flexible programming language (Interactive Matrix Language) in a dynamic, interactive environment. The fundamental object of the language is a data matrix. You can use SAS/IML software interactively (at the statement level) to see results immediately, or you can store statements in a module and execute them later. The programming is dynamic because necessary activities such as memory allocation and dimensioning of matrices are done automatically. SAS/IML software is of interest to users of SAS/STAT software because it enables you to program your methods in the SAS System.

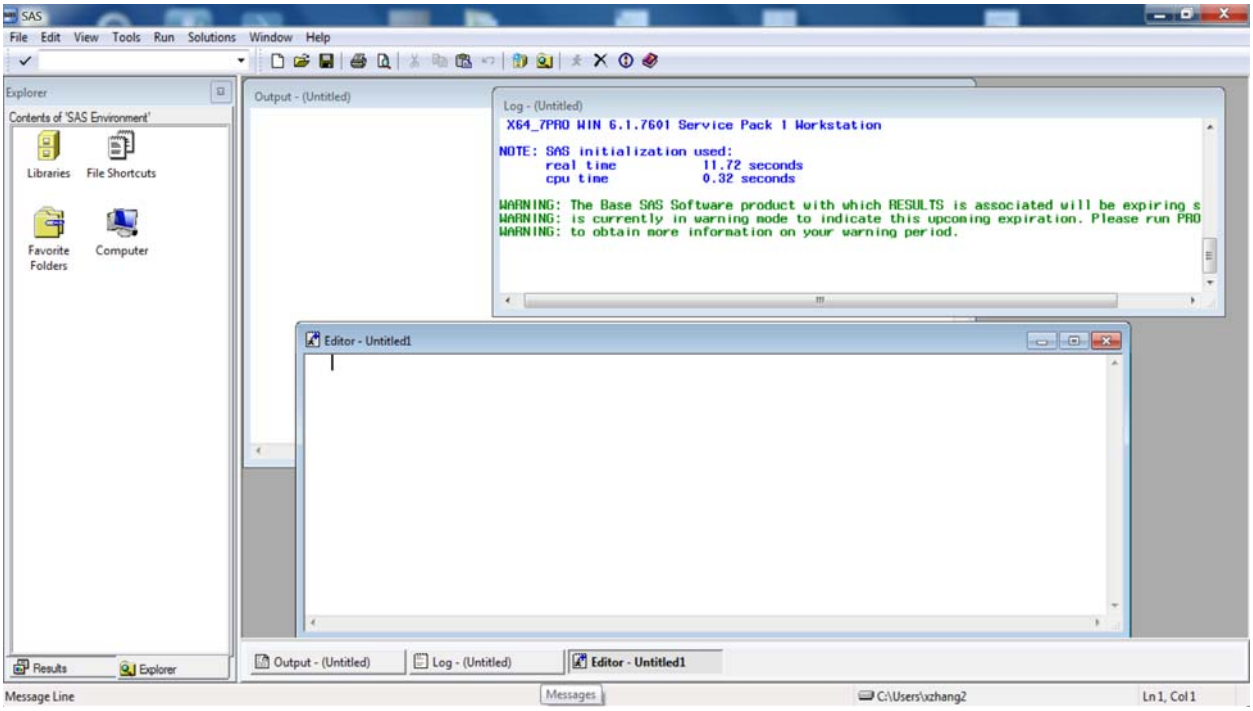
R is a matrix-based programming language that allows you to program statistical methods reasonably quickly. It's open source software, and many add-on packages for R have emerged, providing statisticians with convenient access to new research. Many new statistical methods are first programmed in R. SAS/IML Studio (https://support.sas.com/rnd/app/studio/) also provides the capability to interface with the R language. R will be used in our course MATH 402- Applied Statistical Methods.

SAS University Edition: A single-user license of the basic windows SAS Analytics package can be several thousand dollars. The good news is that SAS University Edition (https://www.sas.com/en_us/software/university-edition.html#) is free for you to use from your PC, Mac or Linux. The University Edition features Base SAS, SAS/STAT, SAS/IML, SAS/ACCESS, and SAS Studio (https://www.sas.com/en_us/offers/14q1/122603-sas-for-academia/overview.html). SAS works through virtualization software via your browser in standalone mode once you download it to your PC, Mac or Linux workstation.

SAS Windows: How do we download and install the SAS University Edition? Because SAS University Edition is a virtual application (or vApp), you need virtualization software to run it. There will be instructions and a list of steps to follow on this webpage: https://www.sas.com/en_us/software/university-edition/download-software.html. For example, or windows users, you can follow this Installation Guide

http://support.sas.com/software/products/university-edition/docs/en/SASUniversityEditionInstallGuideWindows.pdf

The following is the interface of SAS 9.4 PC version.



The SAS System for Windows consists of the 4 main windows:

1. Results and Explorer window

a. Explorer pane is browsing tool for SAS libraries

b. Results pane shows a tree-like summary of the output window. You can select, delete, or edit the output before printing, saving, or copying the results

2. Enhanced Program Editor window – Used to create, edit and execute SAS programs

3. Output Window – Displays output from SAS program. Can view, print, copy or

save information in the output window

4. Log Window – Reports on progress of SAS procedures (BLUE). Displays error messages (RED) and warnings (GREEN).

General Information

- All statement lines must end with a semi-colon.
- Comments are indicated in 2 ways.

  1. Start line with * and end with a semi-colon

  2. Enclose with as /* put comment text here */

- End all procedures with RUN.

To start our first SAS program, we need to get data into SAS first. There are several ways to get data into SAS: 1) Read in-stream data, 2) Use INFILE statement, and 3) Import the data from Excel. Let's use several examples to show how to read raw data into SAS.

SAS Data Step:

Example: Reports of results from clinical trials often include statistics about "baseline characteristics", so we can see that different groups have the same basic characteristics. It is of interest to see the difference between the mean age of men ($\mu 1$) and the mean age of women ($\mu 2$). A sample of 40 men and a sample of 40 women are randomly and independently selected and the ages are recorded in the files MandFBODY.txt and MandFBODY.xls.

Method 1(Read in-stream data or copy the data to the SAS code):

One of the most common ways to read data into SAS is by reading the data instream in a **data step** - that is, by **typing the data directly** into the syntax of your SAS program. This approach is good for relatively small datasets. Spaces are usually used to "delimit" (or separate) free formatted data.

```
data age1;
input male female@@;
cards; /*or use datalines;*/
18    60
20    24
```

```
43     49
39     62
60     53
18     18
57     41
27     21
20     21
18     19
63     19
20     58
24     44
46     52
29     48
63     36
21     48
45     34
40     22
50     61
48     21
64     33
18     32
50     37
20     19
20     51
47     35
19     18
55     60
23     58
21     60
19     48
64     31
30     29
43     46
23     18
64     50
40     20
23     56
44     18
;
run;
```

After reading in the data with a data step, it is usually a good idea to print the first few (or all) cases of your dataset to check that things were read correctly. For example, we print the first 10 cases of the data set.

```
TITLE "Age data";
```

```
proc print data= age1 (obs=10);
run;
```

Method 2(use INFILE statement in the data step):

```
data age2;
infile "E:\ESUTeaching\2019Fall\MATH405\SASHandout\MandFBODY.txt" firstobs=2;
/*change the directory if necessary*/
input male female;
run;

TITLE "Age data";
proc print data= age2 (obs=10);
run;
```

Method 3 (use IMPORT procedure):

```
proc import
datafile=" E:\ESUTeaching\2019Fall\MATH405\SASHandout\MandFBODY.xls"
out=age3 dbms=xls replace;
getnames=yes;
run;

TITLE "Age data";
proc print data= age3 (obs=10);
run;
```

The SAS Import wizard can be used to access spreadsheets (Excel, Lotus) and database (Access) files as well. It is most convenient if the variable names are in the first line of the Excel spreadsheet and comply with SAS naming conventions – no more than 8 characters long, no spaces in the middle, and start with a letter. Go to "File" -> "Import Data..." … We do not use this method in this course.

For more information, please go to http://statistics.ats.ucla.edu/stat/sas/modules/input.htm

Variable Transformations:
Transforming data is an important technique in exploratory data analysis. For example, centering and scaling are simple examples of transforming data. There is no need to transform your raw data outside of SAS. You can use Data step to transform your data.

```
data data1;
  input x y gender$ @@;
  cards;
4.5   17.5 M
2.3   20.5 F
6.7   13.6 F
8.4   17.8 M
-2.0  14.7 M
;
run;

proc print data=data1;
run;
```

Now add a variable z, a log transformation of y, to the data set.

```
data data2;
  input x y gender$ @@;
  z=log(y);
  cards;
4.5   17.5 M
2.3   20.5 F
6.7   13.6 F
8.4   17.8 M
-2.0  14.7 M
;
run;
```

Equivalently, we can do this as well.

```
data data3;
set data1;  /* create data3 from data1 */
z=log(y);
run;
```

Deleting rows: We can delete some rows based on a control condition.

```
data data4;
set data3;
   if x > 0;  /*  keep rows with x>0 only */
run;

proc print data=data4;
run;
```

Deleting variables: We can delete a column as well.

```
data data5;
set data4;
   drop y;  /*  remove variable y*/
run;
```

'do … end' loop: It is similar as the "for" loop in R which is used for iterating over a sequence.

```
data uniform;
   do i = 1 to 10;
      x = ranuni(0); /* random uniform number between 0 & 1 */
      output;
   end;

proc print data=uniform;
run;
```

Note the use of a do loop, which is ended by an end; phrase. The output forces creation of a new case for each uniform number. Each case in set a will have the variables x and i. Here is a list of random number generators:

```
   x = ranuni(seed)           /* uniform between 0 & 1 */
   x = a+(b-a)*ranuni(seed);  /* uniform between a & b */
   x = ranbin(seed,n,p);      /* binomial size n prob p */
   x = rancau(seed);          /* cauchy with loc 0 & scale 1 */
   x = a+b*rancau(seed);      /* cauchy with loc a & scale b */
   x = ranexp(seed);          /* exponential with scale 1 */
   x = ranexp(seed) / a;      /* exponential with scale a */
   x = a-b*log(ranexp(seed)); /* extreme value loc a & scale b */
   x = rangam(seed,a);        /* gamma with shape a */
   x = b*rangam(seed,a);      /* gamma with shape a & scale b */
   x = 2*rangam(seed,a);      /* chi-square with d.f. = 2*a */
   x = rannor(seed);          /* normal with mean 0 & SD 1 */
   x = a+b*rannor(seed);      /* normal with mean a & SD b */
   x = ranpoi(seed,a);        /* poisson with mean a */
   x = rantri(seed,a);        /* triangular with peak at a */
   x = rantbl(seed,p1,p2,p3); /* random from (1,2,3) with probs p1,p2,p3*/
```

Descriptive Statistics Using SAS

Before doing any statistical inferences, we must be able to describe the data in a straight-forward, easy-to-comprehend fashion. One way is to display the data graphically and the second way is to show the descriptive summary statistics: sample size, mean, median, minimum, maximum, variance and standard deviation etc. These can be done by using the MEANS, UNIVARIATE and PLOT procedures in SAS.

Example. Consider the data set **foot.xls** with 6 variables and 40 observations each.

```
data foot;
infile "foot.txt" firstobs=2;
input Sex$ Age FootLength ShoePrint ShoeSize Height; /*Sex is categorical or
nominal*/
run;

proc print data= foot;
run;

proc means data=foot;
TITLE "Simple Descriptive Statistics";
run;
```

We can specify **which statistics** we want to compute by specifying options for PROC MEANS.

```
proc means data=foot N MEAN MEDIAN VAR;
run;
```

We may also wish to specify for which numerical variables in our data set we want to compute descriptive statistics.

```
proc means data=foot N MEAN MEDIAN VAR;
VAR Age FootLength Height;
run;
```

For options of PROC MEANS, please go to the following link for more information.
http://support.sas.com/documentation/cdl/en/proc/61895/HTML/default/viewer.htm#a000146729.htm

If we would like a more extensive list of statistics, including tests of normality, stem-and-leaf plots, and boxplots, PROC UNIVARIATE is the way to go.

```
proc univariate data=foot plot;
TITLE "More Descriptive Statistics";
VAR Age FootLength Height;
run;
```

BOXPLOT can be used to generate a box-plot as well. However, BOXPLOT is used to create side-by-side box-and-whiskers plots of measurements organized in groups. To get a single box plot for a variable you need to create a constant grouping variable. For example, to create a box-plot for the variable Height, we

```
data foot;
set foot;
group=1;
run;
```

```
 proc boxplot data=foot;
 plot Height*group/ BOXSTYLE=SCHEMATIC; /* BOXSTYLE=SCHEMATIC specifies to
draw the modified box plot identifying outliers */
 run;
```

For more information about PROC BOXPLOT, please go to

https://support.sas.com/documentation/cdl/en/statug/63962/HTML/default/viewer.htm#boxplot_toc.htm

To request, additionally, a test of normality (Shapiro-Wilk test, Kolmogorov-Smirnov test, Cramer-von Mises test and Anderson-Darling test), a stem-and-leaf plot, and a boxplot, we would add the options NORMAL and PLOT as follows.

```
proc univariate data=foot NORMAL PLOT;
TITLE "More Descriptive Statistics";
VAR Height;
run;
```

We can ask PROC UNIVARIATE to produce histograms, QQ (Quantile-Quantile) plots, and probability plots by adding appropriate statements.

```
proc univariate data=foot;
TITLE "Histogram for Height";
histogram Height;
run;
```

```
proc univariate data=foot;
TITLE "QQ-plot for Height";
qqplot Height;
run;
```

```
proc univariate data=foot;
TITLE "Probability-plot for Height";
probplot Height;
run;
```

For more information about PROC UNIVARIATE, please go to
http://support.sas.com/documentation/cdl/en/procstat/63104/HTML/default/viewer.htm#procstat_univariate_sect008.htm

The PROC PLOT (or PROC GPLOT) can be used to generate a scatter plot for us to investigate the relationship between two variables.

```
proc plot data=foot;
TITLE "Scatter-plot of Show Size by Height";
plot ShoeSize*Height;
run;
```

For more information about PROC PLOT, please check SAS support document
http://support.sas.com/documentation/cdl/en/proc/61895/HTML/default/viewer.htm#a002473551.htm

Probability Distributions Using SAS

SAS can calculate quantiles and cumulative distribution values as well as generate random numbers for a large number of distributions. SAS includes comprehensive random number generation through the RAND function.

1. CDF(Cumulative Distribution Function)

A **cumulative probability** of a random variable X is the probability of that X is less than or equal to a specified value x_0, denoted by $F(x_0)=P(X \le x_0)$.

Please read SAS support document for the CDF function.
http://support.sas.com/documentation/cdl/en/lefunctionsref/63354/HTML/default/viewer.htm#n0n7cce4a3gfqkn1vr0p1x0of99s.htm

Example 1. $X_1 \sim$ *binomial* $(n=12, p=0.67)$. Find $P(X_1 \le 5)$.

```
data a1;
y = cdf('BINOMIAL', 5, 0.67, 12);
run;

proc print data=a1;
run;
```

Example 2. $X_1 \sim$ *binomial* $(n=12, p=0.67)$. Find $P(X_1 = 5)$.

```
data a2;
y1 = cdf('BINOMIAL', 5, 0.67, 12);
y2 = cdf('BINOMIAL', 4, 0.67, 12);
y = y1 - y2;
run;

proc print data=a2;
run;
```

Example 3. $X_2 \sim$ *Poisson* $(\mu=4.35)$. Find $P(X_2 \le 5)$.

```
data a3;
y = cdf('POISSON', 5,  4.35);
run;

proc print data=a3;
run;
```

Example 4. *Y ~ Normal (μ=0.8, σ=1.15).* Find P($Y \leq 2.2$) and P($Y \geq 1.8$).

```
data a4;
y1 = cdf('NORMAL', 2.2,  0.8, 1.15);
y2 = 1- cdf('NORMAL', 1.8,  0.8, 1.15);
run;

proc print data=a4;
run;
```

2. Find QUANTITLEs of a continuous random variable.

Given a cumulative probability (Lower tail) or 1- cumulative probability (Upper tail), we want to see the corresponding value of a random variable. This is the problem of finding the quantitle of a random variable. For example, find a z-score or a score of a non-standard normal random variable. We are especially interested in finding quantitles of a continuous random variable. The quantitles of a discrete random variable can be found as well. http://support.sas.com/documentation/cdl/en/lefunctionsref/63354/HTML/default/viewer.htm#n0uhywbqfucg6qn18woziy41flqp.htm Please read this SAS support document for the **QUANTITLE** function. The **QUANTITLE** function helps us to find critical values for various probability distributions in statistical inferences.

Example 1. *Y ~ Normal (μ=0.8, σ=1.15).*

(1) If $P(Y \leq y_1) = 0.775$, y_1=?

```
data b1;
y=quantile('NORMAL',0.775, 0.8, 1.15);
y2= quantile('NORMAL',0.975, 0, 1);
run;

proc print data=b1;
run;
```

(2) If $P(Y \geq y_2) = 0.662$, y_2=?

```
data b2;
y=quantile('NORMAL',1-0.662, 0.8, 1.15);
run;

proc print data=b2;
run;
```

Example 2. $Y \sim t\ (df=15)$. If $P(Y \leq y) = 0.975$, y=?

```
data b3;
y=quantile('T',0.975, 15);
run;

proc print data=b3;
run;
```

3. **Random number generation using RAND function in SAS** (http://support.sas.com/documentation/cdl/en/lrdict/64316/HTML/default/viewer.htm#a001466748.htm).

Example. Generate 100 standard normal random numbers.

```
data normal (keep=x); /* keep the random numbers only */
call streaminit(4321); /*set the seed value using STREAMINIT function*/
do i=1 to 100;
x=rand('NORMAL', 0, 1);
output; /* output the random numbers */
end;   /* do loop */
run;

proc print data=normal;
run;
```

4. **Conduct simulations using Proc IML**
   https://support.sas.com/documentation/cdl/en/imlug/63541/PDF/default/imlug.pdf

Example. Consider repeating this process: **Roll a balanced die 5 times.** Find the mean of the results.

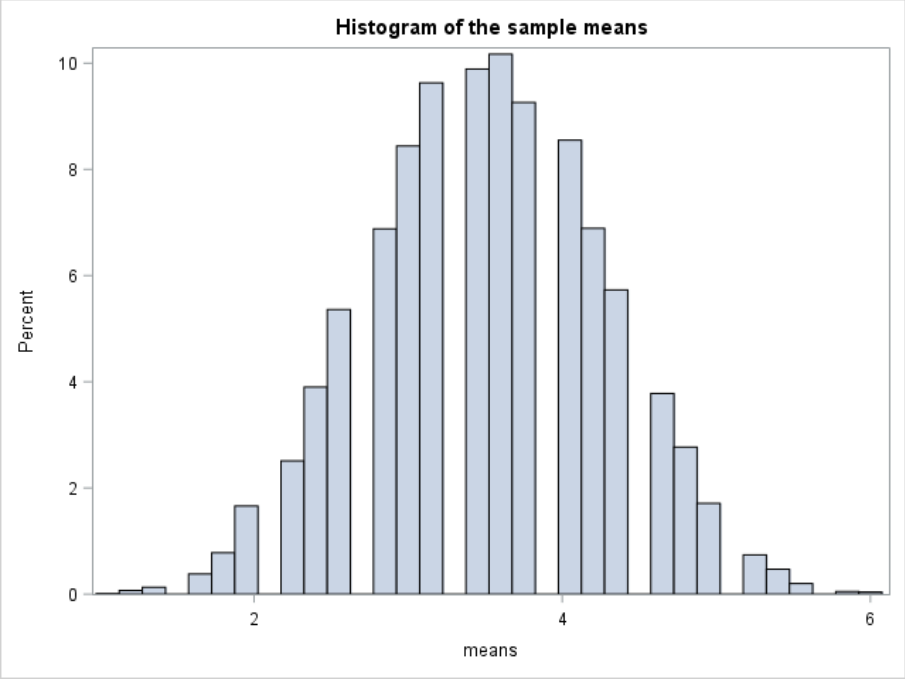
Let X be the random variable of the results of rolling the die. Before studying the distribution of X, we conduct a Monte Carlo simulation of repeating the process (of rolling the die 5 times) 10,000 times to study the sampling distribution of the sample mean.

```
proc iml;

iter = 10000;      /*iteration times*/
n=5;              /*sample size*/
means=j(iter, 1, 0);   /* The sample means in the form of iter*1 matrix*/
```

```
Min = 1; Max = 6;
Do i=1 to iter;
 x=j(n,1,0);   /* The sample data in the form of n*1 matrix*/
  do j = 1 to n;
  x[j]=min + floor((1+Max-Min)*rand("Uniform")); /* random integer values in
Min..Max */
  end;
means[i]=mean(x);
End;

xbar_bar=mean(means);
print xbar_bar;
title "Histogram of the sample means";
call Histogram(means);
quit;
```



We then compare the simulation results with the theoretical results by using the probability distribution of X. Since the die is balanced, we have

$$P(x) = P(X = x) = \frac{1}{6}, \quad x = 1, 2, 3, 4, 5, 6.$$

Thus,

$$\mu = \sum xP(x) = 21/6 = 3.5.$$

The population mean is 3.5; the mean of the 10,000 trials in the simulation was 3.50872. If continued **indefinitely**, the sample mean will be 3.5. Also, notice the distribution is "approximately normal."

T-test in SAS

Let's use one example to show how to make statistical inferences about a single population mean using SAS.

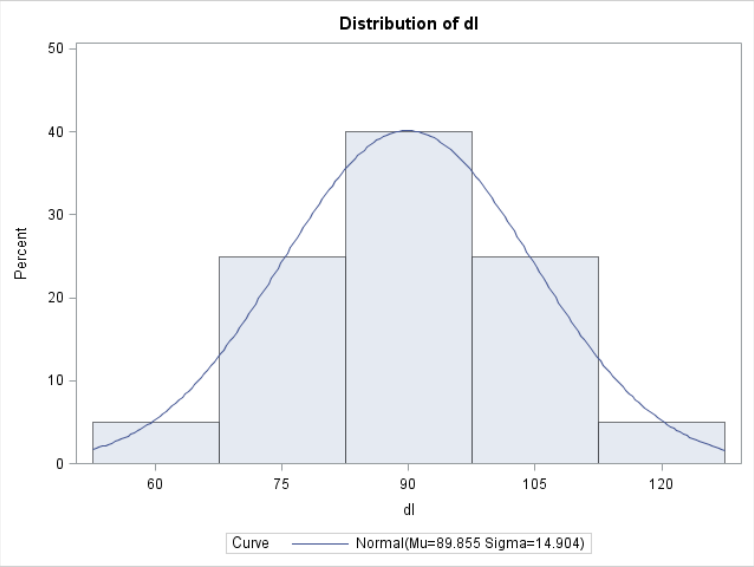
Example. Researchers have shown that cigarette smoking has a deleterious effect on lung function. In their study of the effect of cigarette smoking on the carbon monoxide diffusing capacity (DL) of the lung, Ronald Knudson, W. Kaltenborn and B. Burrows found that current smokers had DL readings significantly lower than either ex-smokers or nonsmokers. The carbon monoxide diffusing capacity for a random sample of current smokers was as follows:

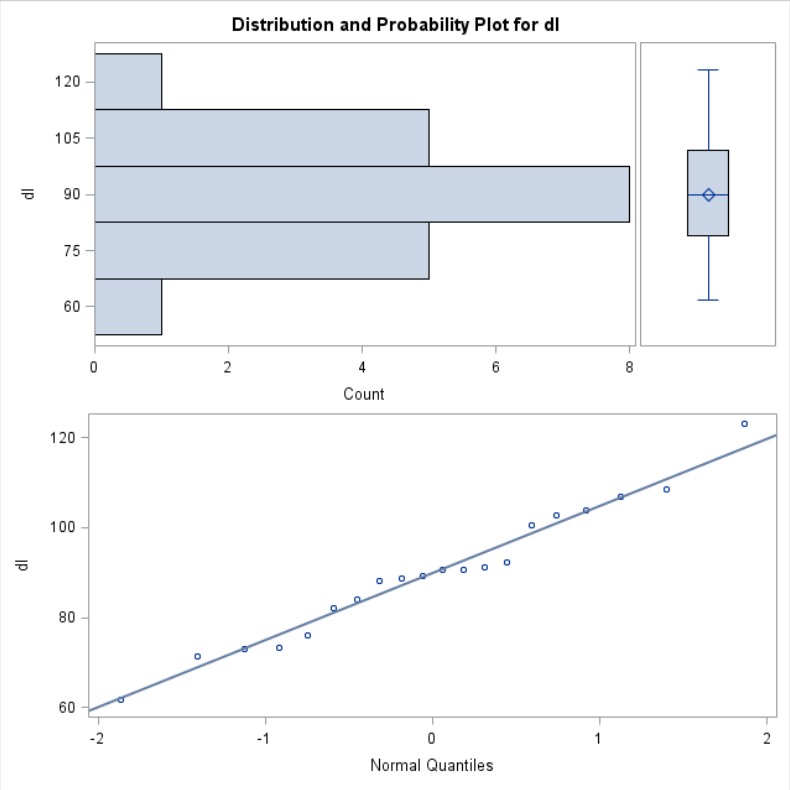
103.768	88.602	73.003	123.086	91.052
92.295	61.675	90.677	84.023	76.014
100.615	88.017	71.210	82.115	89.222
102.754	108.579	73.154	106.755	90.479

Do these data indicate that the mean DL reading for current smokers is lower than 100, the average DL reading for nonsmokers ?

(1) Determine if the data set is from a normal population.
(2) Test at the $\alpha = 0.05$ level. What is the *p*-value of the test?
(3) Use a two-sided symmetric confidence interval to conduct the test.

(1) **ANS.** The histogram of the variable Home has two modes (peaks). Thus the data cannot be from a normal population. It can be further verified by QQ plot where some points are far away from the straight line.





Now we can check the normality by conducting formal statistical tests by testing

H_0: data are from a normal population versus H_a: data are not from a normal population.

The p-values of the test Shapiro-Wild, Kolmogorov-Smirnov, Cramer-von Mises and Anderson-Darling are all large shown in the following table. These large p-values let us fail to reject H_0: data are from a normal population.

Tests for Normality				
Test	**Statistic**		**p Value**	
Shapiro-Wilk	**W**	0.977903	**Pr < W**	0.9042
Kolmogorov-Smirnov	**D**	0.134969	**Pr > D**	>0.1500
Cramer-von Mises	**W-Sq**	0.045419	**Pr > W-Sq**	>0.2500
Anderson-Darling	**A-Sq**	0.254421	**Pr > A-Sq**	>0.2500

Therefore, we conclude that the data are from a normal population. The following is the SAS code.

```
data cig;
input dl@@;
datalines;
103.768
88.602
73.003
123.086
91.052
92.295
61.675
90.677
84.023
76.014
100.615
88.017
71.210
82.115
89.222
102.754
108.579
73.154
106.755
90.479
;
run;

proc univariate data=cig normal plot;
var dl;
histogram dl/normal;
run;
```

(2) **ANS.** We are testing

$$H_0: \mu=100 \text{ versus } H_a: \mu<100.$$

The test statistic of the t-test is -3.04 with a *p*-value = 0.0033 which is smaller than the significance level 0.05. Thus, H_0 is rejected. There is sufficient evidence to show that that the mean DL reading for current smokers is lower than 100.

The following is the SAS code.

```
proc ttest sides=L data=cig H0=100;
var dl;
run;
```

For more information about TTEST in SAS, please go to

https://support.sas.com/documentation/cdl/en/statug/63033/HTML/default/viewer.htm#statug_ttest_a0000000113.htm

(3) **ANS.** Use a confidence interval to conduct the test.

The 90% confidence interval (84.09, 95.62) constructed does not contain the μ value of 100 under the null hypothesis. Thus we reject the null hypothesis. Since the significance level for the one-sided test is 0.05, the conclusion from the 90% two-sided symmetric confidence interval of μ is the same as that by the p-value method in (2).

To construct a 90% two-sided symmetric confidence interval of μ, we can use two-sided TTEST in SAS

```
proc ttest sides=2 data=cig H0=100 alpha=0.1;
var dl;
run;
```

or MEANS procedure in SAS.

```
proc means data=cig clm alpha=0.1;
var dl;
run;
```

For more information about proc MEANS in SAS, please go to

http://support.sas.com/documentation/cdl/en/proc/61895/HTML/default/viewer.htm#a000146729.htm

Linear Regressions in SAS

Let's use one example to show how to fit a multiple linear regression model and check the model assumptions.

Example. The article "How to Optimize and Control the Wire Bonding Process: Part II" (*Solid State Technology*, Jan. 1991: 67–72) described an experiment carried out to assess the impact of the variables X_1 = force (gm), X_2 = power (mW), X_3 = tempertaure (°C), and X_4 = time (msec) on Y = ball bond shear strength (gm). The data WireBonding.xls was simulated to be consistent with the information given in the article. The purpose of the experiment is to see which explanatory variables are most useful in predicting Y.

(1) Assume that a multiple linear regression model is appropriate and include all four explanatory variables in the first order model, fit the multiple linear regression model and construct an ANOVA table.

ANS. The fitted multiple linear regression model is:

$$\widehat{\text{Strength}} = -37.47667 + 0.21167\text{Force} + 0.49833\text{Power} + 0.12967\text{Temp} + 0.25833\text{Time}.$$

And the ANOVA table is

Analysis of Variance					
Source	**DF**	**Sum of Squares**	**Mean Square**	**F Value**	**Pr > F**
Model	4	1660.14000	415.03500	15.60	<.0001
Error	25	665.11867	26.60475		
Corrected Total	29	2325.25867			

(2) What is the observed unbiased estimator of the common variance in the model?

ANS. The common variance σ^2 is estimated by MSE = 26.60 copied from the ANOVA table..

(3) Find the following two values.

Coefficient of determination (R^2) = 0.714

Adjusted coefficient of determination (R_a^2) = 0.6682

(4) Is the regression model in (1) useful in predicting the Y = ball bond shear strength? Justify your answer.

ANS. The answer is ___YES_____ because the p-value of the F-test of usefulness of the model

H_0: __$\beta_1 = \beta_2 = \beta_3 = \beta_4 = 0$ against

H_a: __At least one $\beta_i \neq 0$, i=1,2,3,4_____

is equal to _____1.592×10^{-6}_____ which is very small.

(5) Does "force" contribute to the model significantly at significance level 0.1? Why?

ANS. The answer is ___NO_____ because the *p*-value of the test of

H_0: _____$\beta_1 = 0$_____ against H_a: _____$\beta_1 \neq 0$_____

is equal to _____0.32444___>0.1_____.

(6) Does "time" contribute to the model significantly at significance level 0.1? Why?

ANS. The answer is ___NO_____ because the *p*-value of the test of

H_0: _____$\beta_4 = 0$___ against H_a: _____$\beta_4 \neq 0$_____

is equal to ___0.23132_>0.1_____.

(7) Remove "force" and "time" variables from the model and keep the remaining two independent variables only in the model. Denote the **new model** be our best model. Fit this model using the remaining two explanatory variables.

ANS. The best multiple linear regression model is fitted as:

$\widehat{\text{Strength}} = -24.90167 + 0.49833\text{Power} + 0.12967\text{Temp}.$

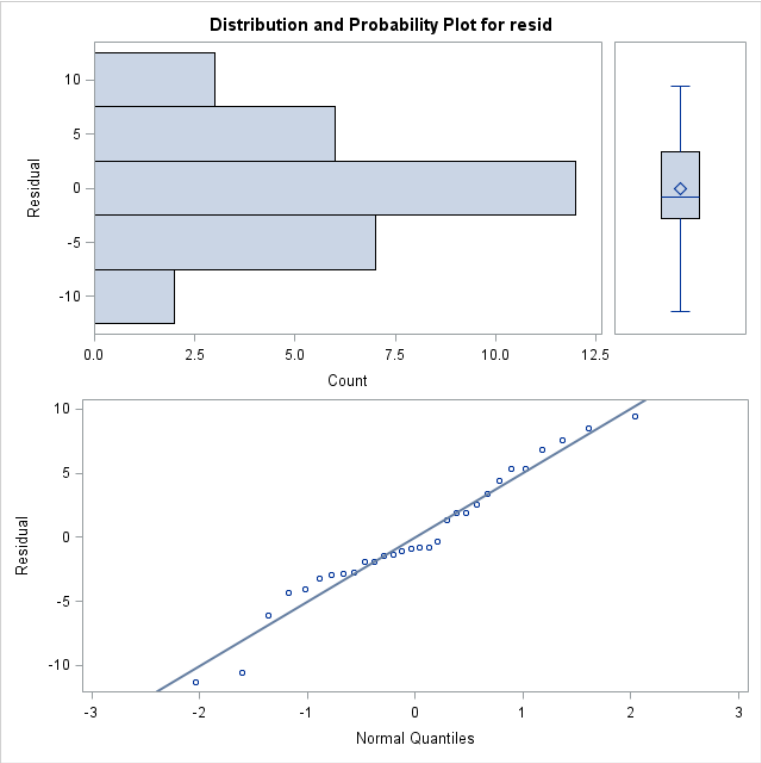
(8) Find the following two values for the fitted model in (7).

Coefficient of determination (R^2) = __0.6852_____
Adjusted coefficient of determination (R_a^2) = __0.6619_____

(9) Check if the normality assumption of the best model in (7) is adequate.

ANS.

The histogram is close to a bell-shape and most points are very close the straight line in the QQ plot. Furthermore, the three tests for normality provides large p-values (>0.15) which lead no rejection of normality assumption. Thus the normality assumption is satisfied.

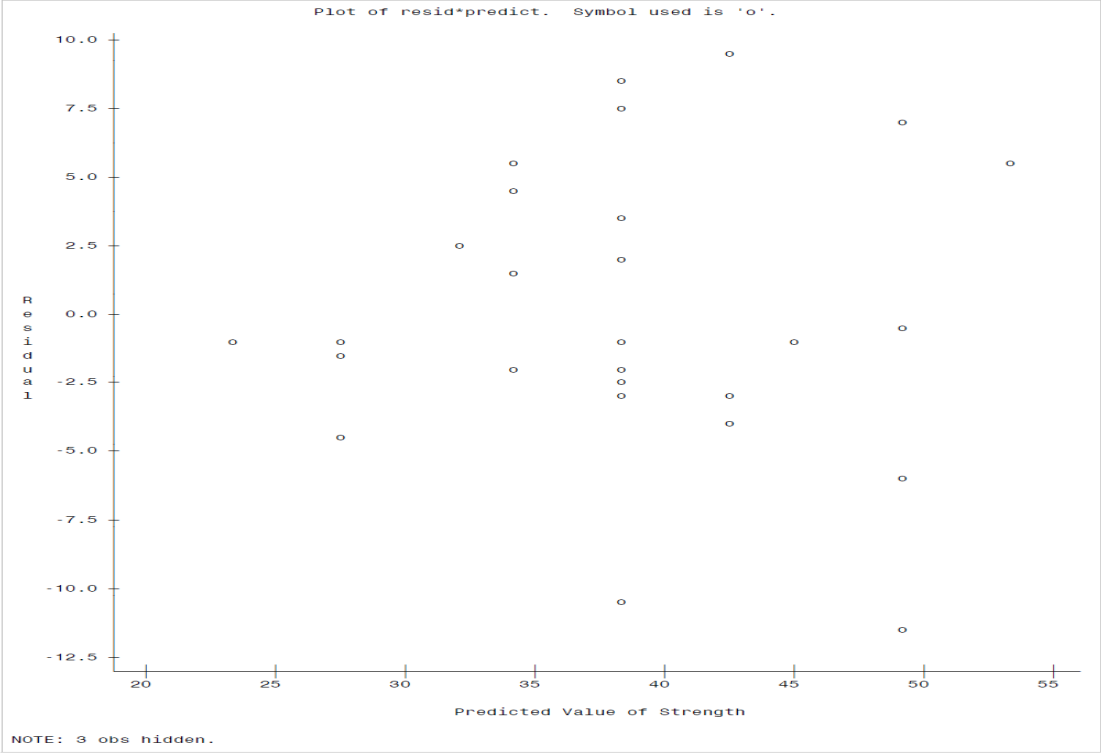


(10) Construct a residual plot and determine if the two model assumptions for the best model in (7): independence and constant variance are adequate.

ANS.

From the following scatter plot, it can be seen that the residuals are randomly distributed and thus no pattern can be found. Therefore, the independence assumption seems reasonable.

Furthermore, it seems that the spread of the residuals has an increasing trend. Constant variance assumptions is not reasonable.




```
proc import datafile="WireBonding.xls" out=wire dbms=xls;
getnames=yes;
run;

proc reg data=wire;
model Strength = Force Power Temp Time;
run;

proc reg data= wire;
model Strength = Power Temp;
output out=new p=predict r=resid;
run;

proc univariate data=new normal plot;
var resid;
histogram resid/normal;
run;

proc plot data=new;
plot resid*predict='o';
run;
```

Simple Random Sampling in SAS

Example 2.5 in the textbook (page 34).

The U.S. government conducts a Census of Agriculture every five years, collecting data on all farms (defined as any place from which $1000 or more of agricultural products were produced and sold) in the 50 states.[2] The Census of Agriculture provides data on number of farms, the total acreage devoted to farms, farm size, yield of different crops, and a wide variety of other agricultural measures for each of the $N = 3078$ counties and county-equivalents in the United States. The file agpop.dat contains the 1982, 1987, and 1992 information on the number of farms, acreage devoted to farms, number of farms with fewer than 9 acres, and number of farms with more than 1000 acres for the population.

To take an SRS of size 300 from this population, I generated 300 random numbers between 0 and 1 on the computer, multiplied each by 3078, and rounded the result up to the next highest integer. This procedure generates an SRSWR. If the population is large relative to the sample, it is likely that each unit in the sample only occurs once in the list. In this case, however, 13 of the 300 numbers were duplicates. The duplicates were discarded, and replaced with new randomly generated numbers between 1 and 3078 until all 300 numbers were distinct; the set of random numbers generated is in file selectrs.dat, and the data set for the SRS is in agsrs.dat.

Comments about the data and sampling method:

The counties selected to be in the sample may not "feel" very random at first glance. For example, counties 2840, 2841, and 2842 are all in the sample while none of the counties between 2740 and 2787 appear. The sample contains 18% of Virginia counties, but no counties in Alaska, Arizona, Connecticut, Delaware, Hawaii, Rhode Island, Utah, or Wyoming. There is a quite natural temptation to want to "adjust" the random number list, to spread it out a bit more. If you want a random sample, you must resist this temptation. Research, beginning with Neyman (1934), has repeatedly demonstrated that purposive samples often do not represent the population on key variables. If you deliberately substitute other counties for those in the randomly generated sample, you may be able match the population on one particular characteristic such as geographic distribution; however, you will likely fail to match the population on characteristics of interests such as number of farms or average farm size. If you want to ensure that all states are represented, do not adjust your randomly selected sample purposively but take a stratified sample (to be discussed in Chapter 3).

Let's look at the variable *acres92*, the number of acres devoted to farms in 1992. A small number of counties in the population are missing that value—in some cases, the data are withheld to prevent disclosing data on individual farms. Thus we first check to see the extent of the missing data in our sample. Fortunately, our sample has no missing data (Exercise 23 tells how likely such an occurrence is). Figure 2.4 displays a histogram of the acreage devoted to farms in each of the 300 counties. ■

Software packages such as SAS that calculate estimates for survey samples use the weight variable to find point estimates of means and totals. Here is output from SAS PROC SURVEYMEANS. The variable *acres92* is the number of acres devoted to farms in 1992, and the variable *lt200k* takes on the value 1 if the county has less than 200,000 acres in farms and takes on the value 1 if the county has greater than 200,000 acres in farms.

```
/* Analyzes the data in Examples 2.5, 2.6, 2.7, and 2.10 of Sampling: Design
and Analysis, 2nd edition
   by Sharon L. Lohr
   Copyright 2009 by Sharon Lohr */

filename agsrs 'agsrs.csv';
options ls=78 nodate;

data agsrs;
   infile agsrs dsd delimiter= ','  firstobs = 2;
     /* The dsd option allows SAS to read missing values between successive
delimiters */
   input county $ state $ acres92 acres87 acres82 farms92
                          farms87 farms82 largef92 largef87
                          largef82 smallf92 smallf87 smallf82;
   if acres92 < 200000 then lt200k = 1;
      else if acres92 >= 200000 then lt200k = 0; /*  counties with fewer than
200000 acres in farms */
   if acres92 = -99 then acres92 =  . ;  /* check for missing values */
   if acres92 = -99 then lt200k =  . ;
   sampwt = 3078/300;  /* sampling weight is same for each observation */
   run;


    /* Draw a histogram of the data */
proc univariate data = agsrs noprint;
   histogram acres92;
   run;

 /* proc surveymeans calculates summary statistics, and adjusts for fpc. */

 /* Note that proc surveymeans gives the 95% confidence interval for
    the mean as [260706, 335088].  Proc surveymeans uses the t percentile
    with 299 degrees of freedom, which is 1.96793.   */
```

```
 /* To estimate population total, use 'sum' in options; 'clsum' gives a CI
for the sum. */
 /* The 'class' statement treats a variable as a categorical variable */

proc surveymeans data=agsrs total=3078 sum clsum mean clm;
   weight sampwt;
   var acres92;
run;


proc surveymeans data=agsrs total=3078 sum clsum mean clm alpha=0.04;
   class lt200k;
   weight sampwt;
   var lt200k;
run;


proc surveymeans data=agsrs total=3078 sum clsum mean clm;
   class lt200k;
   weight sampwt;
   var acres92 lt200k;
run;

 /* ALWAYS use a weight statement in proc surveymeans.  If you omit it, SAS
    assumes that all weights are 1.  For an SRS, the estimated mean will be
    correct, but estimated totals will be wrong. */
```

Ratio Estimation

```
/* Analyzes the data in Examples 4.3 of
   Sampling: Design and Analysis, 2nd ed. by S. Lohr
   Copyright 2008 by Sharon Lohr */

filename agsrs 'agsrs.csv';

/* options ls=78 nodate nocenter;*/

data agsrs;
   infile agsrs delimiter= ','  firstobs = 2;
   input county $ state $ acres92 acres87 acres82 farms92
                          farms87 farms82 largef92 largef87
                          largef82 smallf92 smallf87 smallf82;
   if acres92 < 200000 then lt200k = 1;
      else lt200k = 0; /*  counties with fewer than 200000 acres in farms */
   if acres92 = -99 then acres92 =  . ;  /* check for missing values */
   if acres92 = -99 then lt200k =  . ;
   sampwt = 3078/300;  /* sampling weight is same for each observation */

    /* Define domains of interest:  west and nonwest */
   if state in ('AK', 'AZ', 'CA', 'CO', 'HI', 'ID', 'MT', 'NV', 'NM',
      'OR', 'UT', 'WA', 'WY') then west = 1;
   else west=0;
   acres92w = acres92*west;
run;

   /* We want to do ratio estimation this time. Let's look
      at the correlation between acres92 and acres87 */
proc corr data = agsrs;
   var acres92 acres87;

proc gplot data = agsrs;
   plot acres92 * acres87;
run;

 /* proc surveymeans will estimate ratios with keyword 'ratio'  */

 /* Note that the estimated means and totals of acres87 and acres92
    do not use ratio estimation, however---these are calculated
    using SRS formulas */

proc surveymeans data=agsrs total=3078 mean stderr clm sum clsum ratio ;
   var acres92 acres87;  /* need both in var statement */
   ratio 'acres92/acres87' acres92/acres87;
   weight sampwt;
   ods output Statistics=statsout Ratio=ratioout;
   run;

/* Can get ratio estimates of totals by taking output from
   proc surveymeans and multiplying by N */
```

```
proc print data=ratioout;
run;
data ratioout1;
   set ratioout;
   xtotal = 964470625;
   ratiosum = ratio*xtotal;
   sesum = stderr*xtotal;
   lowercls = lowercl*xtotal;
   uppercls = uppercl*xtotal;

proc print data = ratioout1;
run;
```

Stratified Sampling in SAS

Example 3.6

The sample in Example 3.2 was designed so that each county in the United States would have approximately the same probability of appearing in the sample. To estimate the total number of acres devoted to agriculture in the United States, we create the variable *strwt* in file agstrat.dat with the sampling weights; it contains the value 220/21 for counties in the Northeast stratum, 1054/103 for the North Central counties, 1382/135 for the South counties, and 422/41 for the West counties. We can use (3.8) to estimate the population total by forming a new column containing the product of variables *strwt* and *acres92*, then calculating the sum of the new column. In doing so, we calculate $\hat{t}_{str} = 909{,}736{,}035$, the same (up to roundoff error) estimate as obtained in Example 3.2. Note that even though this sample is approximately self-weighting, it is not exactly self-weighting because the stratum sample sizes must be integers. When calculating estimates, use the exact weights from each stratum.

The variable *strwt* can be used to estimate population means or totals for every variable measured in the sample, and most computer packages for surveys use the weight variable to calculate point estimates. Note, however, that you cannot calculate the standard error of $\hat{t}_{str}$ unless you know the stratification—you need to use (3.4) to estimate the variance. Partial output from SAS PROC SURVEYMEANS for the variable *acres92* is given in below.

Data Summary

Number of Strata	4
Number of Observations	300
Sum of Weights	3078

Statistics

Variable	N	DF	Mean	Std Error of Mean	95% CL for Mean	
acres92	300	296	295561	16380	263325.000	327796.530

Statistics

Variable	Sum	Std Dev	95% CL for Sum	
acres92	909736035	50417248	810514350	1008957721

```
/* Analyzes the data in Examples 3.2 and 3.6 of Sampling: Design and
Analysis, 2nd ed. by S. Lohr
   Copyright 2009 by Sharon Lohr *   Copyright 2009 by Sharon Lohr */

filename agstrat 'agstrat.csv';
options ls=78 nodate;

data agstrat;
   infile agstrat dsd delimiter= ','  firstobs = 2;
     /* The dsd option allows SAS to read missing values between successive
delimiters */
input county $ state $ acres92 acres87 acres82 farms92
                          farms87 farms82 largef92 largef87
                          largef82 smallf92 smallf87 smallf82
                          region $ rn strwt;
   if acres92 < 200000 then lt200k = 1;
      else if acres92 >= 200000 then lt200k = 0; /*  counties with fewer than
200000 acres in farms */
   if acres92 = -99 then acres92 =  . ;  /* check for missing values */
   if acres92 = -99 then lt200k =  . ;
   run;


proc sort data=agstrat;
  by region;


proc univariate data=agstrat plot;
  /* creates crummy looking line printer boxplots */
   var acres92;
   by region;
run;


proc boxplot data = agstrat;
  /* creates pretty boxplots if you have SAS for Windows */
   plot acres92 * region/boxstyle=schematic;
run;


/* Create dataset containing strata totals. If you have small sampling
fractions,
   you need this extra dataset to be able to use the fpc. */

data strattot;
   input region $  _total_;
   cards;
NE 220
NC 1054
S  1382
W  422
;
run;

proc print data=strattot;
run;
```

```
/* Important:  You need BOTH the weight statement AND the stratum statement!
   If you omit the weight statement, SAS assigns weight 1 to every
observation;
   if you have disproportionate allocation, estimates of the mean will be
biased.
   If you omit the stratum statement, the variances will be wrong.
   Try it without one of these statements and see what happens. */

/* The following gives the output printed in Example 3.6 */

proc surveymeans data=agstrat total = strattot nobs mean sum clm clsum df;
   stratum region ;
   var acres92 ;
   weight strwt;
run;


/* We can also add more variables and list the details of the stratification.
*/

proc surveymeans data=agstrat total = strattot nobs mean sum clm clsum df;
   class lt200k;
   stratum region /list;
   var acres92 lt200k;
   weight strwt;
   ods output Statistics=myout;
run;

proc print data=myout;
run;
```

Cluster Sampling in SAS

Example 5.2

A student wants to estimate the average grade point average (GPA) in his dormitory. Instead of obtaining a listing of all students in the dorm and conducting an SRS, he notices that the dorm consists of 100 suites, each with four students; he chooses 5 of those suites at random, and asks every person in the 5 suites what her or his GPA is. The results are as follows:

Person Number	Suite (psu) 1	2	3	4	5
1	3.08	2.36	2.00	3.00	2.68
2	2.60	3.04	2.56	2.88	1.92
3	3.44	3.28	2.52	3.44	3.28
4	3.04	2.68	1.88	3.64	3.20
Total	12.16	11.36	8.96	12.96	11.08

```
Data Summary

Number of Clusters                  5
Number of Observations             20
Sum of Weights                    400

                                Statistics

                                          Std Error
Variable                N          Mean     of Mean      95% CL for Mean
-------------------------------------------------------------------------
gpa                    20      2.826000    0.163665   2.37159339 3.28040661
-------------------------------------------------------------------------
```

```
/* Analyzes the data in Example 5.2 of
   Sampling: Design and Analysis, 2nd ed. by S. Lohr
   Copyright 2008 by Sharon Lohr */

/* options ls=78 nodate nocenter; */
```

```
data gpa;
   input suite gpa;
   wt = 20;  /* every person has weight 100/5 = 20 */
   datalines;
1 3.08
1 2.60
1 3.44
1 3.04
2 2.36
2 3.04
2 3.28
2 2.68
3 2.00
3 2.56
3 2.52
3 1.88
4 3.00
4 2.88
4 3.44
4 3.64
5 2.68
5 1.92
5 3.28
5 3.20
;
proc print data=gpa;

proc boxplot data=gpa;
   plot gpa*suite;
run;


 /* To analyze a cluster sample, need statements for cluster and weight.*/
 /*  Note that total (for the fpc) is to specify the number of psu's, not
     the number of observation units */
 /* As always, need weight variable to get correct answer for sum */

proc surveymeans data=gpa total = 100 nobs mean sum clm clsum;
   cluster suite;
   var gpa;
   weight wt;
run;


/* What do we get in an erroneous analysis that ignores the clustering?*/

 /* DO NOT USE THE FOLLOWING 2 LINES; THEY ARE INCLUDED TO SHOW YOU WHAT
HAPPENS
    WITH AN INCORRECT ANALYSIS */

proc surveymeans data=gpa ;   /* This is wrong since it does not include
                                 cluster or weight statements */
   var gpa;
 /* This analysis assumes that we have a SRS!  WRONG WRONG WRONG! */
```

Example 5.6

One-stage cluster samples are often used in educational studies, since students are naturally clustered into classrooms or schools. Consider a population of 187 high school algebra classes in a city. An investigator takes an SRS of 12 of those classes and gives each student in the sampled classes a test about function knowledge. The (hypothetical) data are given in the file algebra.dat, with the following summary statistics.

Class Number	M_i	$\bar{y}_i$	t_i	$M_i^2(\bar{y}_i - \hat{\bar{y}}_r)^2$
23	20	61.5	1,230	456.7298
37	26	64.2	1,670	1,867.7428
38	24	58.4	1,402	9,929.2225
39	34	58.0	1,972	24,127.7518
41	26	58.0	1,508	14,109.3082
44	28	64.9	1,816	4,106.2808
46	19	55.2	1,048	19,825.3937
51	32	72.1	2,308	93,517.3218
58	17	58.2	989	5,574.9446
62	21	66.6	1,398	7,066.1174
106	26	62.3	1,621	33.4386
108	26	67.2	1,746	14212.7867
Total	299		18,708	194,827.0387

Data Summary

Number of Clusters	12
Number of Observations	299
Sum of Weights	4659.41667

Statistics

Variable	N	DF	Mean	Std Error of Mean
score	299	11	62.568562	1.491578

```
/* Analyzes the data in Example 5.6 of Sampling: Design and Analysis, 2nd ed.
   by S. Lohr. Copyright 2008 by Sharon Lohr */

filename algebra 'C:\math405\algebra.csv';

/* options ls=78 nocenter nodate;*/
data algebra;
   infile algebra delimiter= ',' firstobs = 2;
   input class Mi score;
   sampwt = 187/12;
run;

proc surveymeans data=algebra total = 187 nobs mean sum clm clsum df;
   cluster class;
   var score;
   weight sampwt;
   ods output Statistics=myout;
run;

proc print data=myout;
run;

proc glm data = algebra;
   class class;
   model score = class;
   means class;
run;
```

Two-sample T-test in SAS

Example (Exercise 2.30) Front housings for cell phones are manufactured in an injection molding process. The time the part is allowed to cool in the mold before removal is thought to influence the occurrence of a particularly troublesome cosmetic defect, flow lines, in the finished housing. After manufacturing, the housings are inspected visually and assigned a score between 1 and 10 based on their appearance, with 10 corresponding to a perfect part and 1 corresponding to a completely defective part. An experiment was conducted using two cool-down times, 10 and 20 seconds, and 20 housings were evaluated at each level of cool-down time. All 40 observations in this experiment were run in random order. The data are as follows.

10 seconds		20 seconds	
1	3	7	6
2	6	8	9
1	5	5	5
3	3	9	7
5	2	5	4
1	1	8	6
5	6	6	8
2	8	4	5
3	2	6	8
5	3	7	7

(1) Is there evidence to support the claim that the longer cool-down time results in fewer appearance defects? Use $\alpha=0.05$.
(2) What is the *p*-value for the test conducted in part (1)?
(3) Find a 95 percent confidence interval on the difference in means. Provide a practical interpretation of this interval. (Hint: we have to use a 1-sided confidence interval)
(4) Check the assumption of normality for the data from this experiment

Before using PROC TTEST in SAS, we need to re-arrange the data using the two variables: group and score. See exercise0230.txt.

```
data score;
infile "exercise0230.txt" firstobs=2;
input group$ score; /*group is categorical*/
run;
```

(1) **ANS.** We are testing

$$H_0: \mu_1 = \mu_2 \text{ versus } H_a: \mu_1 < \mu_2.$$

The test (with equal variance assumption) statistic of the t-test is -5.57 with a *p*-value <0.0001 which is smaller than the significance level 0.05. Thus, H_0 is rejected. There is sufficient evidence to show that the longer cool-down time results in fewer appearance defects. The following is the SAS output.

Method	**Variances**	**DF**	**t Value**	**Pr < t**
Pooled	Equal	38	-5.57	<.0001
Satterthwaite	Unequal	35.601	-5.57	<.0001
Equality of Variances				
Method	**Num DF**	**Den DF**	**F Value**	**Pr > F**
Folded F	19	19	1.70	0.2559

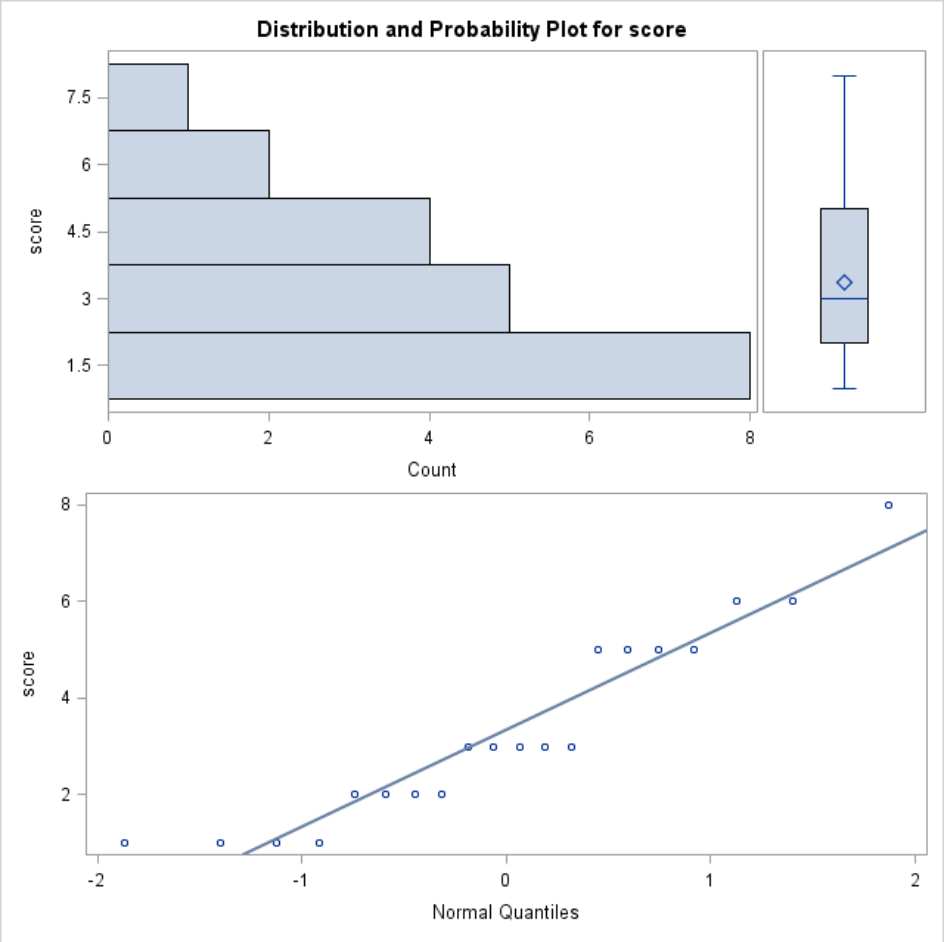
We use the t-test with equal variance assumption because the *p*-value for the test of H_0: the two population variances are equal versus H_a: the two population variances are not equal is large and thus H_0 is not rejected.

(2) **ANS**. From SAS output, *p*-value <0.0001.

(3) **ANS**. From the SAS output, $\mu_1 - \mu_2 < -2.1965$. This upper confidence bound is less than 0. The two samples are different. The 20 second cooling time gives a cosmetically better housing.

(4) **ANS**.

(a) For the group of 10 seconds, the histogram of the score is highly skewed. Thus the data cannot be from a normal population. It can be further verified by QQ plot where some points are far away from the straight line.



Now we can check the normality by conducting formal statistical tests by testing

H_0: data are from a normal population versus H_a: data are not from a normal population.

The p-values of the test Shapiro-Wilk, Kolmogorov-Smirnov, Cramer-von Mises and Anderson-Darling are all very small shown in the following table. These small p-values let us reject H_0: data are from a normal population.

Therefore, we conclude that the data are NOT from a normal population. But notice that the departure from normality is not very significant (p-values>0.01). So we still can use t-test.

Tests for Normality				
Test	**Statistic**		**p Value**	
Shapiro-Wilk	**W**	0.903449	**Pr < W**	0.0479
Kolmogorov-Smirnov	**D**	0.219213	**Pr > D**	0.0128
Cramer-von Mises	**W-Sq**	0.132008	**Pr > W-Sq**	0.0394
Anderson-Darling	**A-Sq**	0.747852	**Pr > A-Sq**	0.0442

(b) For the group of 20 seconds, the histogram of the score is about bell-shaped and symmetric. Thus the data are from a normal population. It can be further verified by QQ plot where most points are close to the straight line.

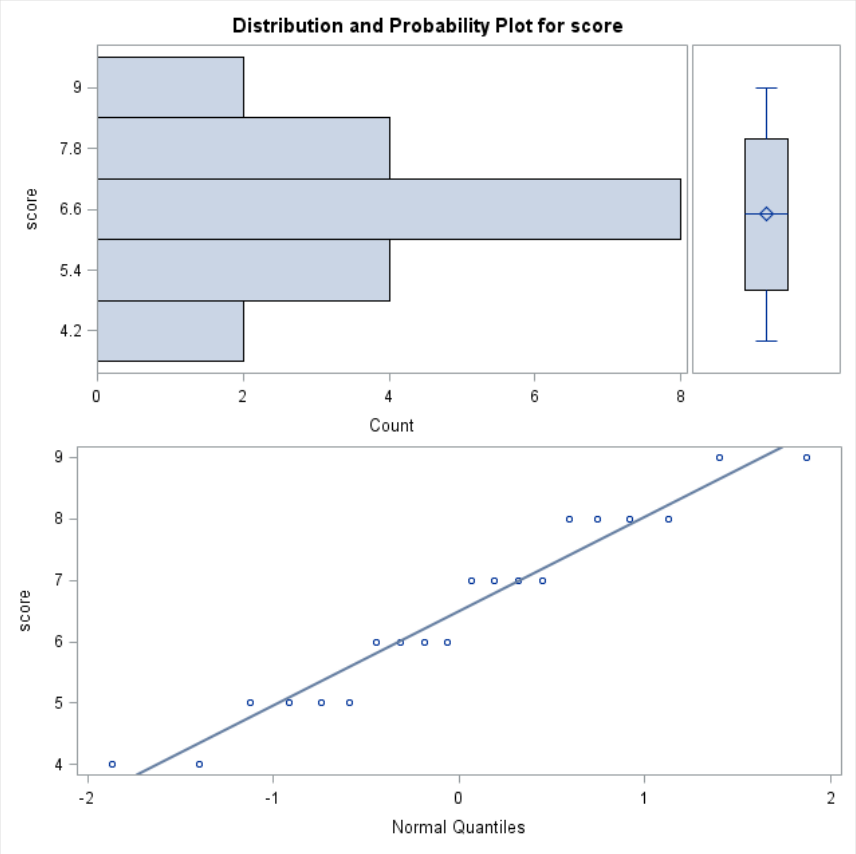
Now we can check the normality by conducting formal statistical tests by testing

H_0: data are from a normal population versus H_a: data are not from a normal population.

The p-values of the test Shapiro-Wilk, Kolmogorov-Smirnov, Cramer-von Mises and Anderson-Darling are all large shown in the following table. These large p-values let us fail to reject H_0: data are from a normal population.

Therefore, we conclude that the data are from a normal population.

Tests for Normality				
Test	**Statistic**		**p Value**	
Shapiro-Wilk	**W**	0.939823	**Pr < W**	0.2379
Kolmogorov-Smirnov	**D**	0.13514	**Pr > D**	>0.1500
Cramer-von Mises	**W-Sq**	0.072556	**Pr > W-Sq**	0.2478
Anderson-Darling	**A-Sq**	0.456584	**Pr > A-Sq**	0.2431



The following is the SAS code.

```
data score;
infile "exercise0230.txt" firstobs=2;
input group$ score; /*group is categorical */
run;

 proc ttest data=score SIDES=L;
      class group;
      var score;
 run;

proc univariate data=score normal plot;
      class group;
      var score;
histogram score/normal;
run;
```

For more information about TTEST in SAS, please go to

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

One-way ANOVA in SAS

Example. Is the attention span of children affected by whether or not they had a good breakfast? Twelve children were randomly divided into three groups and assigned to a different meal plan. The response was attention span in minutes during the morning reading time. The data are as follows.

No Breakfast	Light Breakfast	Full Breakfast
8	14	10
7	16	12
9	12	16
13	17	15

```
proc import
datafile="breakfast.xls"
out=breakfast dbms=xls replace;
getnames=yes;
run;

/*calculate descriptive statistics*/
proc means data=breakfast N MEAN MEDIAN VAR;
class group;
VAR score;
run;

/*plot the data by groups*/
proc plot data=breakfast;
plot score*group;
run;

/*compare three box-plots*/
 proc boxplot data=breakfast;
 plot score*group/ BOXSTYLE=SCHEMATIC;
 run;

/*1-way ANOVA*/
 proc GLM data=breakfast;
class group;
model score=group;
run;
```

The one-way ANOVA can be done by PROC ANOVA as well.

```
PROC ANOVA data=breakfast;
class group;
model score=group;
run;
```

```
/*add output statement to output the residuals*/

proc GLM data=breakfast;
class group;
model score=group;
output out=new p=predict r=resid;
run;

/*residual plot*/
proc plot data=new;
plot resid*predict;
run;

/*check normality of residuals */
PROC univariate data=new normal plot;
Var resid;
run;

/*Statistical Tests of Equality of Variance */
proc GLM data=breakfast;
class group;
model score=group;
MEANS group/HOVTEST=BARTLETT HOVTEST=LEVENE;
run;



proc GLM data=breakfast;
class group;
model score=group;
means group/LSD bon Tukey; /*multiple comparisons*/
run;



proc GLM data=breakfast;
class group;
model score=group;
CONTRAST 'contrast1' group 1 -1 0;
CONTRAST 'contrast2' group 0  1 -1;
CONTRAST 'contrast3' group 1  -2 1; /*contrasts*/
means group/scheffe; /*scheffe method*/
run;
```

For more information about PROC GLM, go to

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

```
/*calculate sample size when given alpha=0.05, power=0.8 and sigma^2=6.472*/
proc GLMPOWER data=breakfast;
class group;
model score=group;
power stddev = 2.54
alpha=0.05
ntotal = .
power=0.8;
run;
```

For more information about PROC GLMPOWER, go to

https://support.sas.com/documentation/cdl/en/statug/63347/HTML/default/viewer.htm#statug_glmpower_a0000000154.htm

```
/* random effects model analyzed using the RANDOM statement*/
proc GLM data=breakfast;
class group;
model score=group;
random group;
run;

/* proc varcomp gets the “variance components”*/
/*proc varcomp assumes all factors are random*/
proc varcomp data=breakfast;
class group;
model score=group;
run;


/* As an alternative to using proc glm with a random statement, and proc
varcomp, you could
instead use proc mixed, which has some options specifically for mixed
models.*/

proc mixed data=breakfast cl; /* cl option asks for the confidence limits.*/
class group;
model score=; /*lists only the fixed effects; there are no fixed effects in
the model*/
random group/vcorr;

run;



/*Kruskal-Wallis Test*/
proc npar1way data=breakfast wilcoxon;
/*WILCOXON requests a box plot of Wilcoxon scores*/
      class group;
      var score;
run;
```

For more information about PROC NPAR1WAY, go to

https://support.sas.com/documentation/cdl/en/statug/63033/HTML/default/viewer.htm#npar1way_toc.htm

The Regression Approach to the Analysis of Variance

The following is the SAS code for the regression approach to ANOVA.

```
proc glm data=breakfast;
      class group;
     model score=group/xpx inverse solution;
run;


data breakfast;
set breakfast;
if (group='no') then x=1;
else if (group='light') then x=2;
else x=3;
tau1=(x eq 1)-(x eq 3);
tau2=(x eq 2)-(x eq 3);
run;

proc print data=breakfast;
run;

proc reg data=breakfast;
model score=tau1 tau2;
run;
```

§3.10 The Regression Approach to the Analysis of Variance.

Denote the response variable by Y and the explanatory variable by $x_1, x_2, \ldots x_a$ which are fixed by the treatment design.

Example. For the breakfast problem,

$$y_{ij} = \mu + \tau_i + \varepsilon_{ij}, \quad i = 1, 2, 3, \quad j = 1, 2, 3, 4$$

parameters: $\mu, \tau_1, \tau_2, \tau_3$, and σ^2

we are testing $H_0: \mu_1 = \mu_2 = \mu_3$ or

$$H_0: \tau_1 = \tau_2 = \tau_3 = 0 \quad \left(\sum_{i=1}^{3} \tau_i = 0\right)$$

We write the model in linear Regression form

$$Y = X\beta + \varepsilon.$$

$$\begin{bmatrix} y_{11} \\ \vdots \\ y_{14} \\ y_{21} \\ \vdots \\ y_{24} \\ y_{31} \\ \vdots \\ y_{34} \end{bmatrix} = \begin{bmatrix} \mu & \tau_1 & \tau_2 & \tau_3 \\ 1 & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & 0 & 1 \end{bmatrix}_{12\times 4} \begin{pmatrix} \mu \\ \tau_1 \\ \tau_2 \\ \tau_3 \end{pmatrix} + \varepsilon$$

It is easy to see that the design matrix is singular because the model is over-parameterized. we have the constraint $\sum_{i=1}^{3} \tau_i = 0$. Thus, a generalized inverse can be used to solve the normal equations.

SAS code:

```
proc glm data=breakfast;
class group;
model score=group/xpx inverse solution;
run;
```

To solve the singularity, we should use the constraint $\sum_{i=1}^{3} \tau_i = 0$ ($\tau_3 = -\tau_1 - \tau_2$)

$$\begin{bmatrix} y_{11} \\ \vdots \\ y_{14} \\ y_{21} \\ \vdots \\ y_{24} \\ y_{31} \\ \vdots \\ y_{34} \end{bmatrix} = \begin{bmatrix} \mu & \tau_1 & \tau_2 \\ 1 & 1 & 0 \\ \vdots & \vdots & \vdots \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ \vdots & \vdots & \vdots \\ 1 & 0 & 1 \\ 1 & -1 & -1 \\ \vdots & \vdots & \vdots \\ 1 & -1 & -1 \end{bmatrix} \begin{pmatrix} \mu \\ \tau_1 \\ \tau_2 \end{pmatrix} + \varepsilon \qquad (*)$$

To fit the linear regression model, we need to input the design matrix to SAS and fit the model

$$Y_{ij} = \beta_0 + \beta_1 \tau_{1ij} + \beta_2 \tau_{2ij} + \varepsilon_{ij}$$

Note that $\beta_0 = \mu$, $\beta_1 = \tau_1$ and $\beta_2 = \tau_2$

see SAS code on the next page.

$\Rightarrow$ $\hat{\mu} = 12.4167$, $\hat{\tau}_1 = -3.167$, $\hat{\tau}_2 = 2.333$

and $\hat{\tau}_3 = -(\tau_1 + \tau_2) = 0.834$

Thus. $\hat{\mu}_1 = \hat{\mu} + \hat{\tau}_1 = 9.25$

$\hat{\mu}_2 = \hat{\mu} + \hat{\tau}_2 = 14.75$

$\hat{\mu}_3 = \hat{\mu} + \hat{\tau}_3 = 13.25$

Block Design ANOVA in SAS

Example. We want to investigate the effect of 3 methods of soil preparation on the growth of seedlings. Each method is applied to seedlings growing at each of 4 locations and the average first year growth is recorded.

Soil Prep	Location			
	1	2	3	4
A	11	13	16	10
B	15	17	20	12
C	10	15	13	10

```
DATA seedlings;
INPUT soil$ location y@@;
CARDS;
A 1 11
A 2 13
A 3 16
A 4 10
B 1 15
B 2 17
B 3 20
B 4 12
C 1 10
C 2 15
C 3 13
C 4 10
;
run;

/* The SAS Type I analysis gives the correct F = 10.06 with a p-value of
0.0121*/
PROC GLM data=seedlings;
CLASS soil location;
MODEL y = soil location;
run;
```

An incorrect analysis of the data using a one-way ANOVA set up (ignoring the blocking factor) is

```
PROC GLM data=seedlings;
CLASS soil;
MODEL y = soil;
run;
```

```
/*Model Adequacy Checking and multiple comparison*/
PROC GLM data=seedlings;
CLASS soil location;
MODEL y = soil location/p;
```

```
means soil location/LSD Tukey scheffe bon;
output out=new p=predict r=resid;
run;

PROC univariate normal data=new;
Var resid; /* By soil; */
run;

proc plot data=new;
plot resid*predict;
plot resid*soil;
plot resid*location;
run;


/*random effect model; block (location) is random*/
proc mixed data=seedlings cl; /* cl option asks for the confidence limits.*/
class soil location;
model y= soil; /*lists only the fixed effects */
random location/vcorr;
run;


/*random effect model using proc GLM*/
proc glm data=seedlings;
class soil location;
model y= soil location;
random location/test;
run;


/*Sample size determination alpha=0.05, power=0.8 and sigma^2=1.89*/
proc GLMPOWER data=seedlings;
class soil location;
model y = soil location;
power stddev = 1.375
alpha=0.05
ntotal = .
power=0.8;
run;
```

Regression Approach to Block Design

Example. We want to investigate the effect of 3 methods of soil preparation on the growth of seedlings. Each method is applied to seedlings growing at each of 4 locations and the average first year growth is recorded.

Soil Prep	Location			
	1	2	3	4
A	11	13	16	10
B	15	17	20	12
C	10	15	13	10

The following is the SAS code for the regression approach to ANOVA for the block design.

```
proc glm data=seedlings;
      class soil location;
model y = soil location/xpx inverse solution;
run;



data seedlings2;
input mu tau1 tau2 beta1 beta2 beta3 y@@;
cards;
1  1  0  1  0  0 11
1  1  0  0  1  0 13
1  1  0  0  0  1 16
1  1  0 -1 -1 -1 10
1  0  1  1  0  0 15
1  0  1  0  1  0 17
1  0  1  0  0  1 20
1  0  1 -1 -1 -1 12
1 -1 -1  1  0  0 10
1 -1 -1  0  1  0 15
1 -1 -1  0  0  1 13
1 -1 -1 -1 -1 -1 10
;
run;

proc reg data=seedlings2;
model y=tau1 tau2 beta1 beta2 beta3 ;
run;
```

§4.1.4 The Regression Approach to RCBD.

For the seedlings example, the statistical model is $y_{ij} = \mu + \tau_i + \beta_j + \varepsilon_{ij}$, $i = 1, 2, 3$, $j = 1, 2, 3, 4$

with $\sum_{i=1}^{3} \tau_i = 0$ and $\sum_{j=1}^{4} \beta_j = 0$

Thus, we consider the linear regression model with parameters τ_1, τ_2, β_1, β_2, and β_3:

$$y_{11} = \mu + \tau_1 + \beta_1 + \varepsilon_{11}$$
$$y_{12} = \mu + \tau_1 + \beta_2 + \varepsilon_{12}$$
$$y_{13} = \mu + \tau_1 + \beta_3 + \varepsilon_{13}$$
$$y_{14} = \mu + \tau_1 - (\beta_1 + \beta_2 + \beta_3) + \varepsilon_{14}$$
$$y_{21} = \mu + \tau_2 + \beta_1 + \varepsilon_{21}$$
$$y_{22} = \mu + \tau_2 + \beta_2 + \varepsilon_{22}$$
$$y_{23} = \mu + \tau_2 + \beta_3 + \varepsilon_{23}$$
$$y_{24} = \mu + \tau_2 - (\beta_1 + \beta_2 + \beta_3) + \varepsilon_{24}$$
$$y_{31} = \mu - (\tau_1 + \tau_2) + \beta_1 + \varepsilon_{31}$$
$$y_{32} = \mu - (\tau_1 + \tau_2) + \beta_2 + \varepsilon_{32}$$
$$y_{33} = \mu - (\tau_1 + \tau_2) + \beta_3 + \varepsilon_{33}$$
$$y_{34} = \mu - (\tau_1 + \tau_2) - (\beta_1 + \beta_2 + \beta_3) + \varepsilon_{34}$$

In matrix notation, we fit the linear regression model

$$\begin{bmatrix} y_{11} \\ y_{12} \\ y_{13} \\ y_{14} \\ y_{21} \\ y_{22} \\ y_{23} \\ y_{24} \\ y_{31} \\ y_{32} \\ y_{33} \\ y_{34} \end{bmatrix} = \begin{bmatrix} \mu & \tau_1 & \tau_2 & \beta_1 & \beta_2 & \beta_3 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & -1 & -1 & -1 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & -1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 0 & 0 \\ 1 & -1 & -1 & 0 & 1 & 0 \\ 1 & -1 & -1 & 0 & 0 & 1 \\ 1 & -1 & -1 & -1 & -1 & -1 \end{bmatrix} \begin{pmatrix} \tau_1 \\ \tau_2 \\ \beta_1 \\ \beta_2 \\ \beta_3 \end{pmatrix} + \varepsilon$$

$\Rightarrow \hat{\mu} = 13.5, \quad \hat{\tau}_1 = -1.0, \quad \hat{\tau}_2 = 2.5 \quad (\text{thus } \hat{\tau}_3 = -1.5)$

$\hat{\beta}_1 = -1.5, \quad \hat{\beta}_2 = 1.5 \quad \hat{\beta}_3 = 2.833 \quad (\text{thus } \hat{\beta}_4 = -2.833)$

$\Rightarrow \hat{\mu}_1 = \hat{\mu} + \hat{\tau}_1 = 12.5, \quad \hat{\mu}_2 = \hat{\mu} + \hat{\tau}_2 = 16, \quad \hat{\mu}_3 = \hat{\mu} + \hat{\tau}_3 = 12.$

Latin Square Design ANOVA in SAS

Example 4.3. Suppose that an experimenter is studying the effects of five different formulations of a rocket propellant used in aircrew escape systems on the observed burning rate. Each formulation is mixed from a batch of raw material that is only large enough for five formulations to be tested. Furthermore, the formulations are prepared by several operators, and there may be substantial differences in the skills and experience of the operators. Thus, it would seem that there are two nuisance factors to be "averaged out" in the design: batches of raw material and operators. The appropriate design for this problem consists of testing each formulation exactly once in each batch of raw material and for each formulation to be prepared exactly once by each of five operators. The resulting design is shown in Table 4.9.

■ TABLE 4.9
Latin Square Design for the Rocket Propellant Problem

Batches of Raw Material	Operators				
	1	2	3	4	5
1	$A = 24$	$B = 20$	$C = 19$	$D = 24$	$E = 24$
2	$B = 17$	$C = 24$	$D = 30$	$E = 27$	$A = 36$
3	$C = 18$	$D = 38$	$E = 26$	$A = 27$	$B = 21$
4	$D = 26$	$E = 31$	$A = 26$	$B = 23$	$C = 22$
5	$E = 22$	$A = 30$	$B = 20$	$C = 29$	$D = 31$

SAS Handout_15: Latin Square Design ANOVA

```
data Rocket;
input batch operator trt$ y@@;
cards;
1 1 A 24
2 1 B 17
3 1 C 18
4 1 D 26
5 1 E 22
1 2 B 20
2 2 C 24
3 2 D 38
4 2 E 31
5 2 A 30
1 3 C 19
2 3 D 30
3 3 E 26
4 3 A 26
5 3 B 20
1 4 D 24
2 4 E 27
3 4 A 27
4 4 B 23
5 4 C 29
1 5 E 24
2 5 A 36
3 5 B 21
4 5 C 22
5 5 D 31
;
run;

PROC GLM data=Rocket;
CLASS batch operator trt;
MODEL y = batch operator trt;
MEANS trt/LSD bon Tukey scheffe;
RUN;
```

Factorial Design ANOVA in SAS

Example 5.1

As an example of a factorial design involving two factors, an engineer is designing a battery for use in a device that will be subjected to some extreme variations in temperature. The only design parameter that he can select at this point is the plate material for the battery, and he has three possible choices. When the device is manufactured and is shipped to the field, the engineer has no control over the temperature extremes that the device will encounter, and he knows from experience that temperature will probably affect the effective battery life. However, temperature can be controlled in the product development laboratory for the purposes of a test.

The engineer decides to test all three plate materials at three temperature levels—15, 70, and 125°F—because these temperature levels are consistent with the product end-use environment. Because there are two factors at three levels, this design is sometimes called a **3^2 factorial design**. Four batteries are tested at each combination of plate material and temperature, and all 36 tests are run in random order. The experiment and the resulting observed battery life data are given in Table 5.1.

In this problem the engineer wants to answer the following questions:

1. What effects do material type and temperature have on the life of the battery?
2. Is there a choice of material that would give *uniformly long life regardless of temperature*?

■ **TABLE 5.1**
Life (in hours) Data for the Battery Design Example

Material Type	Temperature (°F)					
	15		70		125	
1	130	155	34	40	20	70
	74	180	80	75	82	58
2	150	188	136	122	25	70
	159	126	106	115	58	45
3	138	110	174	120	96	104
	168	160	150	139	82	60

This last question is particularly important. It may be possible to find a material alternative that is not greatly affected by temperature. If this is so, the engineer can make the battery **robust** to temperature variation in the field. This is an example of using statistical experimental design for **robust product design**, a very important engineering problem.

This design is a specific example of the general case of a two-factor factorial. To pass to the general case, let y_{ijk} be the observed response when factor A is at the ith level ($i = 1, 2, \ldots, a$) and factor B is at the jth level ($j = 1, 2, \ldots, b$) for the kth replicate ($k = 1, 2, \ldots, n$). In general, a two-factor factorial experiment will appear as in Table 5.2. The order in which the abn observations are taken is selected at random so that this design is a **completely randomized design**.

The observations in a factorial experiment can be described by a model. There are several ways to write the model for a factorial experiment. The **effects model** is

$$y_{ijk} = \mu + \tau_i + \beta_j + (\tau\beta)_{ij} + \epsilon_{ijk} \quad \begin{cases} i = 1, 2, \ldots, a \\ j = 1, 2, \ldots, b \\ k = 1, 2, \ldots, n \end{cases} \tag{5.1}$$

where μ is the overall mean effect, τ_i is the effect of the ith level of the row factor A, β_j is the effect of the jth level of column factor B, $(\tau\beta)_{ij}$ is the effect of the interaction between τ_i and β_j, and ϵ_{ijk} is a random error component. Both factors are assumed to be **fixed**, and the treatment effects are defined as deviations from the overall mean, so $\sum_{i=1}^{a}\tau_i = 0$ and $\sum_{j=1}^{b}\beta_j = 0$. Similarly, the interaction effects are fixed and are defined such that $\sum_{i=1}^{a}(\tau\beta)_{ij} = \sum_{j=1}^{b}(\tau\beta)_{ij} = 0$. Because there are n replicates of the experiment, there are abn total observations.

```
data battery;
input type temp life@@;
cards;
1 15 30
1 15 74
1 15 155
1 15 180
1 70 34
1 70 80
1 70 40
1 70 75
1 125 20
1 125 82
1 125 70
1 125 58
2 15 150
2 15 159
2 15 188
2 15 126
2 70 136
2 70 106
2 70 122
2 70 115
2 125 25
```

```
2 125 58
2 125 70
2 125 45
3 15 138
3 15 168
3 15 110
3 15 160
3 70 174
3 70 150
3 70 120
3 70 139
3 125 96
3 125 82
3 125 104
3 125 60
;
run;

proc means data=battery;
by type;
var life;
run;

proc sort data=battery out=battery;
 by temp;
run;

proc means data=battery;
by temp;
var life;
run;

proc GLM data=battery;
class type temp;
model life = type temp type*temp;
output out=new p=predict r=resid;
run;

PROC univariate normal data=new;
Var resid;
run;

proc plot data=new;
plot resid*predict;
plot resid*type;
plot resid*temp;
run;
```

Factorial Design ANOVA in SAS

Example 13.1

EXAMPLE 13.1 A Measurement Systems Capability Study

Statistically designed experiments are frequently used to investigate the sources of variability that affect a system. A common industrial application is to use a designed experiment to study the components of variability in a measurement system. These studies are often called **gauge capability studies** or **gauge repeatability and reproducibility (R&R) studies** because these are the components of variability that are of interest (for more discussion of gauge R&R studies, see the supplemental material for this chapter).

A typical gauge R&R experiment from Montgomery (2009) is shown in Table 13.1. An instrument or gauge is used to measure a critical dimension on a part. Twenty parts have been selected from the production process, and three randomly selected operators measure each part twice with this gauge. The order in which the measurements are made is completely randomized, so this is a two-factor factorial experiment with design factors parts and operators, with two replications. Both parts and operators are random factors. The variance component identity in Equation 13.1 applies; namely,

$$\sigma_y^2 = \sigma_\tau^2 + \sigma_\beta^2 + \sigma_{\tau\beta}^2 + \sigma^2$$

where σ_y^2 is the total variability (including variability due to the different parts, variability due to the different operators, and variability due to the gauge), σ_τ^2 is the variance component for parts, σ_β^2 is the variance component for operators, $\sigma_{\tau\beta}^2$ is the variance component that represents interaction between parts and operators, and σ^2 is the random experimental error. Typically, the variance component σ^2 is called the gauge repeatability because σ^2 can be thought of as reflecting the variation observed when the same part is measured by the same operator, and

$$\sigma_\beta^2 + \sigma_{\tau\beta}^2$$

is usually called the reproducibility of the gauge because it reflects the additional variability in the measurement system resulting from use of the instrument by the operator. These experiments are usually performed with the objective of estimating the variance components.

Table 13.2 shows the ANOVA for this experiment. The computations were performed using the Balanced ANOVA routine in Minitab. Based on the P-values, we conclude that the effect of parts is large, operators may have a small effect, and no significant part–operator interaction takes place. We may use Equation 13.7 to estimate the variance components as follows:

$$\hat{\sigma}_\tau^2 = \frac{62.39 - 0.71}{(3)(2)} = 10.28$$

$$\hat{\sigma}_\beta^2 = \frac{1.31 - 0.71}{(20)(2)} = 0.015$$

$$\hat{\sigma}_{\tau\beta}^2 = \frac{0.71 - 0.99}{2} = -0.14$$

and

$$\hat{\sigma}^2 = 0.99$$

The bottom portion of the Minitab output in Table 13.2 contains the expected mean squares for the random model, with numbers in parentheses representing the variance components [(4) represents σ^2, (3) represents $\sigma_{\tau\beta}^2$, etc.]. The estimates of the variance components are also given, along with the error term that was used in testing that variance

component in the analysis of variance. We will discuss the terminology **unrestricted model** later; it has no relevance in random models.

Notice that the estimate of one of the variance components, $\sigma^2_{\tau\beta}$, is negative. This is certainly not reasonable because by definition variances are nonnegative. Unfortunately, negative estimates of variance components can result when we use the analysis of variance method of estimation (this is considered one of its drawbacks). We can deal with this negative result in a variety of ways. One possibility is to assume that the negative estimate means that the variance component is really zero and just set it to zero, leaving the other nonnegative estimates unchanged. Another approach is to estimate the variance components with a method that assures nonnegative estimates (this can be done with the maximum likelihood approach). Finally, we could note that the P-value for the interaction term in Table 13.2 is very large, take this as evidence that $\sigma^2_{\tau\beta}$ really is zero and that there is no interaction effect, and then fit a **reduced model** of the form

$$y_{ijk} = \mu + \tau_i + \beta_j + \epsilon_{ijk}$$

that does not include the interaction term. This is a relatively easy approach and one that often works nearly as well as more sophisticated methods.

Table 13.3 shows the analysis of variance for the reduced model. Because there is no interaction term in the model, both main effects are tested against the error term, and the estimates of the variance components are

$$\hat{\sigma}^2_\tau = \frac{62.39 - 0.88}{(3)(2)} = 10.25$$

$$\hat{\sigma}^2_\beta = \frac{1.31 - 0.88}{(20)(2)} = 0.0108$$

$$\hat{\sigma}^2 = 0.88$$

Finally, we could estimate the variance of the gauge as the sum of the variance component estimates $\hat{\sigma}^2$ and $\hat{\sigma}^2_\beta$ as

$$\begin{aligned}\hat{\sigma}^2_{\text{gauge}} &= \hat{\sigma}^2 + \hat{\sigma}^2_\beta \\ &= 0.88 + 0.0108 \\ &= 0.8908\end{aligned}$$

The variability in the gauge appears small relative to the variability in the product. This is generally a desirable situation, implying that the gauge is capable of distinguishing among different grades of product.

■ **TABLE 13.1**

The Measurement Systems Capability Experiment in Example 13.2

Part Number	Operator 1		Operator 2		Operator 3	
1	21	20	20	20	19	21
2	24	23	24	24	23	24
3	20	21	19	21	20	22
4	27	27	28	26	27	28
5	19	18	19	18	18	21
6	23	21	24	21	23	22
7	22	21	22	24	22	20
8	19	17	18	20	19	18
9	24	23	25	23	24	24
10	25	23	26	25	24	25
11	21	20	20	20	21	20
12	18	19	17	19	18	19
13	23	25	25	25	25	25
14	24	24	23	25	24	25
15	29	30	30	28	31	30
16	26	26	25	26	25	27
17	20	20	19	20	20	20
18	19	21	19	19	21	23
19	25	26	25	24	25	25
20	19	19	18	17	19	17

```
options ls=80;
data RR;
input part operator y @@;
cards;
1  1  21  1  1  20  1  2  20  1  2  20  1  3  19  1  3  21
2  1  24  2  1  23  2  2  24  2  2  24  2  3  23  2  3  24
3  1  20  3  1  21  3  2  19  3  2  21  3  3  20  3  3  22
4  1  27  4  1  27  4  2  28  4  2  26  4  3  27  4  3  28
5  1  19  5  1  18  5  2  19  5  2  18  5  3  18  5  3  21
6  1  23  6  1  21  6  2  24  6  2  21  6  3  23  6  3  22
7  1  22  7  1  21  7  2  22  7  2  24  7  3  22  7  3  20
8  1  19  8  1  17  8  2  18  8  2  20  8  3  19  8  3  18
9  1  24  9  1  23  9  2  25  9  2  23  9  3  24  9  3  24
10  1  25  10  1  23  10  2  26  10  2  25  10  3  24  10  3  25
11  1  21  11  1  20  11  2  20  11  2  20  11  3  21  11  3  20
12  1  18  12  1  19  12  2  17  12  2  19  12  3  18  12  3  19
13  1  23  13  1  25  13  2  25  13  2  25  13  3  25  13  3  25
14  1  24  14  1  24  14  2  23  14  2  25  14  3  24  14  3  25
15  1  29  15  1  30  15  2  30  15  2  28  15  3  31  15  3  30
16  1  26  16  1  26  16  2  25  16  2  26  16  3  25  16  3  27
17  1  20  17  1  20  17  2  19  17  2  20  17  3  20  17  3  20
18  1  19  18  1  21  18  2  19  18  2  19  18  3  21  18  3  23
19  1  25  19  1  26  19  2  25  19  2  24  19  3  25  19  3  25
20  1  19  20  1  19  20  2  18  20  2  17  20  3  19  20  3  17
;
run;

proc anova data=RR;
class part operator;
model y= part operator part*operator;
run;

proc varcomp data=RR;
class part operator;
model y= part operator part*operator;
run;

/*random effect model using proc GLM*/
proc GLM data=RR;
class part operator;
model y= part operator part*operator;
random part operator part*operator/test;
output out=new p=predict r=resid;
run;

proc print data=new;
run;

PROC univariate normal data=new;
var resid;
run;

proc plot data=new;
plot resid*predict;
run;
```

```
/*mixed model Example 13.2: operators fixed*/
proc glm data=RR;
class part operator;
model y=operator|part;
random part part*operator/test;
run;

proc mixed data=RR;
class part operator;
model y= operator;  /*lists only the fixed effects */
random part part*operator;
run;
```

/* **Example 5.5***/

```
data life0;
input angle speed y @@;
cards;
15 125 -2 15 125 -1
15 150 -3 15 150 0
15 175  2 15 175 3
20 125  0 20 125 2
20 150  1 20 150 3
20 175  4 20 175 6
25 125 -1 25 125 0
25 150  5 25 150 6
25 175  0 25 175 -1
;
run;

data life1;
set life0;
x1=angle;
x2=speed;
x1x2=x1*x2;
x12=x1*x1;
x22=x2*x2;
x12x2= x1*x1*x2;
x1x22= x1*x2*x2;
x12x22= x1*x1*x2*x2;
run;

data life2;
set life0;
x1=angle-20;  x2=speed-150;
x10=angle;
x20=speed;
x1x2=x1*x2;
x12=x1*x1;
x22=x2*x2;
x12x2= x1*x1*x2;
x1x22= x1*x2*x2;
x12x22= x1*x1*x2*x2;
run;
```

```
proc reg data=life1;
model y = x10 x20 x1x2 x12 x22;
run;

proc reg data=life2;
model y = x10 x20 x1x2 x12 x22 x12x2 x1x22 x12x22;
output out=new p=predict r=resid;
run;

PROC univariate normal data=new plot;
Var resid;
run;

proc plot data=new;
plot resid*predict;
run;


goptions reset=all border;
data one;
do x10 = 15 to 25 by 0.5;
x1=x10-20;
do x20 = 125 to 175 by 2.5;
x2=x20-150;
y=-24+0.7*(x1+20)+0.08*(x2+150)-0.008*x1*x2-0*x1*x1-0.0016*x2*x2-
0.0016*x1*x1*x2 -0.00128* x1*x2*x2 -0.000192*x1*x1*x2*x2;
output;
end;
end;
run;

proc gcontour data=one;
plot x20*x10=y;
run;

proc g3d data=one;
   plot x20*x10=y;
run;




/******Example 5.4***/
data battery0;
input type temp life@@;
cards;
1 15 30 1 15 74
1 15 155 1 15 180
1 70 34 1 70 80
1 70 40 1 70 75
1 125 20 1 125 82
1 125 70 1 125 58
2 15 150 2 15 159
2 15 188 2 15 126
2 70 136 2 70 106
```

```
2 70 122 2 70 115
2 125 25 2 125 58
2 125 70 2 125 45
3 15 138 3 15 168
3 15 110 3 15 160
3 70 174 3 70 150
3 70 120 3 70 139
3 125 96 3 125 82
3 125 104 3 125 60
;
run;

data battery;
set battery0;
if temp = 15 then A=-1;
if temp = 70 then A=0;
if temp = 125 then A=1;
/*if type = 1 then B1=1;  else B1=0;*/
/*if type = 2 then B2=1;  else B2=0;*/
if type = 1 then B1=1;   if type = 2 then B1=0;   if type = 3 then B1=-1;
if type = 2 then B2=1;   if type = 1 then B2=0;   if type = 3 then B2=-1;

A2= A*A;
AB1=A*B1;
AB2=A*B2;
A2B1= A*A*B1;
A2B2= A*A*B2;
run;


proc glm data=battery;
model life = A B1 B2 A2 AB1 AB2 A2B1 A2B2;
output out=new p=predict r=resid;
run;


proc GLM data=battery PLOTS=INTPLOT;
class temp type;
model life = temp type temp*type;
run;
```

Nested Design

Q1. Read the Nested Design in the text (page 604 -612) and reproduce the results of **example 14.1** using SAS.

(1) Proc GLM produce the results in Table 14.4

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	11	84.9722222	7.7247475	2.93	0.0135
Error	24	63.3333333	2.6388889		
Corrected Total	35	148.3055556			

Source	DF	Type I SS	Mean Square	F Value	Pr > F
supplier	2	15.05555556	7.52777778	2.85	0.0774
batch(supplier)	9	69.91666667	7.76851852	2.94	0.0167

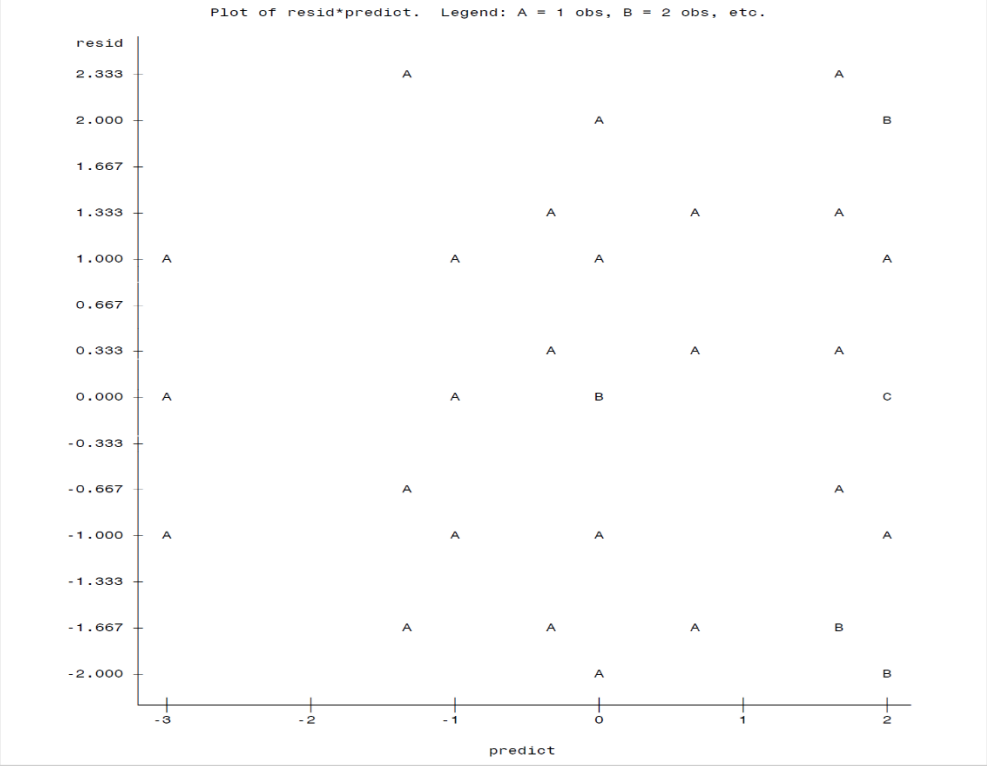
Tests of Hypotheses Using the Type I MS for batch(supplier) as an Error Term					
Source	**DF**	**Type I SS**	**Mean Square**	**F Value**	**Pr > F**
supplier	2	15.05555556	7.52777778	0.97	0.4158

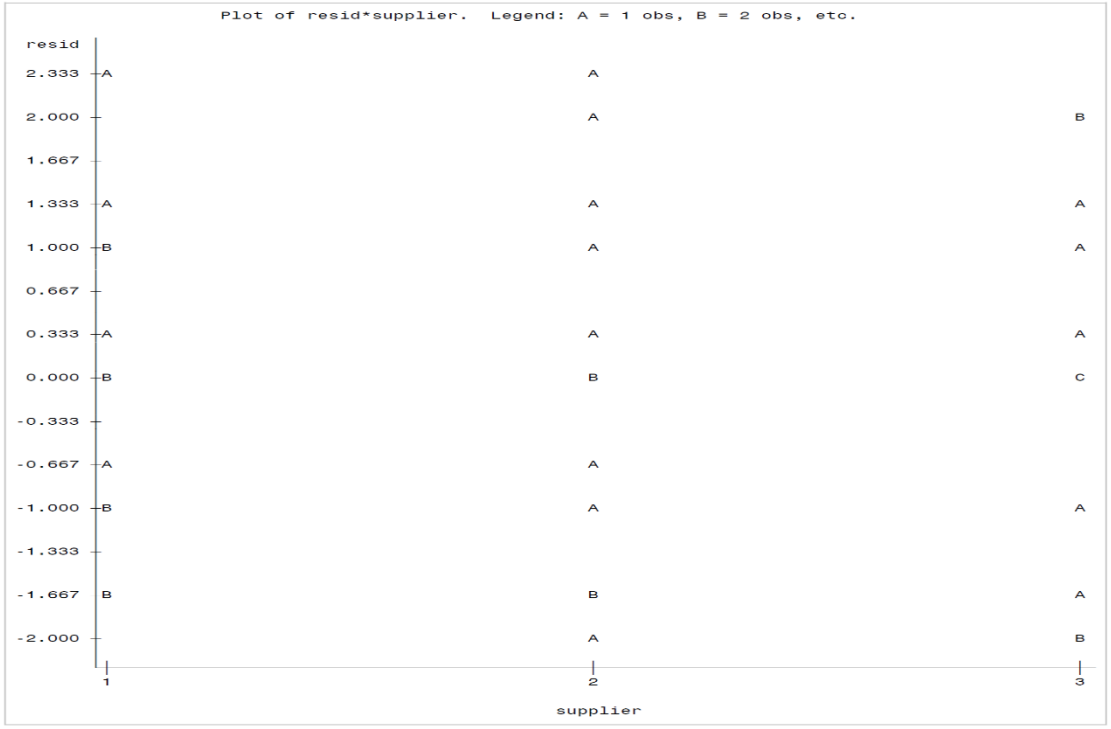
(2) Proc VARCOMP produce the results in Table 14.6

Type 1 Analysis of Variance				
Source	**DF**	**Sum of Squares**	**Mean Square**	**Expected Mean Square**
supplier	2	15.055556	7.527778	Var(Error) + 3 Var(batch(supplier)) + Q(supplier)
batch(supplier)	9	69.916667	7.768519	Var(Error) + 3 Var(batch(supplier))
Error	24	63.333333	2.638889	Var(Error)
Corrected Total	35	148.305556		

Type 1 Estimates	
Variance Component	**Estimate**
Var(batch(supplier))	1.70988
Var(Error)	2.63889

(3) The following are the two residual plots:






```
data example141;
input batch supplier resp@@;
datalines;
1 1 1 1 1 -1 1 1 0
2 1 -2 2 1 -3 2 1 -4
3 1 -2 3 1 0 3 1 1
4 1 1 4 1 4 4 1 0
1 2 1 1 2 -2 1 2 -3
2 2 0 2 2 4 2 2 2
3 2 -1 3 2 0 3 2 -2
4 2 0 4 2 3 4 2 2
1 3 2 1 3 4 1 3 0
2 3 -2 2 3 0 2 3 2
3 3 1 3 3 -1 3 3 2
4 3 3 4 3 2 4 3 1
;
run;

proc anova data= example141;
class supplier batch;
model resp= supplier batch(supplier);
run;
```

```
proc glm data= example141;
class supplier batch;
model resp = supplier batch(supplier);
random batch(supplier);
test h=supplier e=batch(supplier)/htype=1 etype=1;
output out=new p=predict r=resid;
run;

proc plot data=new;
plot resid*predict;
plot resid*supplier;
run;

proc varcomp data= example141 method=type1;
class supplier batch;
model resp = supplier batch(supplier)/fixed=1;
run;

proc mixed data= example141;
class supplier batch;
model resp = supplier;
random batch(supplier);
run;
```