STAT 8678 - SAS Programming & Data Analysis

Chi-Kuang Yeh

2026-05-03

Table of contents

Preface

Thank you

The course is **finished**. Thank you all for the participation and attention. Hope you all have learned something from this course, and have fun. Wish everyone has a good summer vacation!

Description

This course covers programming using the SAS statistical software package, and it provides an introduction to data analysis stressing the implementation using SAS.

Topics include two main parts:

1) **SAS Programming**: data management and manipulation, basic procedures, macro programming;
2) **Data Analysis**: descriptive statistical analysis, one- and two-sample inference, basic categorical data analysis, regression analysis, and other selected topics.

Prerequisites

MATH 4544/6544 – Biostatistics, or equivalent.

Instructor

Chi-Kuang Yeh, Assistant Professor in the Department of Mathematics and Statistics, Georgia State University.

- Office: Suite 1407, 25 Park Place.
- Email: cyeh@gsu.edu.

Office Hour

10:00–13:00 on Monday, or by appointment.

Grade Distribution

- Assignments: 60%
- Exam: 20%
- Project: 20%

Assignment

- ☒ A1, due on Jan 28, 2026
- ☒ A2, due on Feb 8, 2026
- ☒ A3, due on Feb 15, 2026
- ☒ A4, due on Feb 28, 2026
- ☒ A5, due on Mar 18, 2026
- ☒ A6, due on Apr 18, 2026

Midterm

- ☒ March 4, 2026

Final Project

- ☒ Due on May 1, 2026

More information can be found on the project page

Status	Lecture	Topic

Topics and Corresponding Lectures

Those chapters are based on the lecture notes. This part will be updated frequently.

Status	Lecture	Topic
	1	Welcome and Overview
		Part 1: Basics
	2	Basic SAS Operation
	3	SAS Syntax
	4	Import and Export Data
	5	Random Variable
		Part 2: Statistical Analysis
	6	Introduction to Statistical Inference I
	7	Introduction to Statistical Inference II
	8	One Sample Nonparametric Test
	9	One Sample Proportion Test
	10	Introduction to SAS Marco
	11	Chi-Square GoF test
	12	Power Analysis and Calculation (Recorded)
	13	Class Activity I
	14	**Exam**
	15	Two Sample (t)-test for Independent and Paired Datasets
		Part 3: Regression Analysis
	16	Analysis of Variance
		Spring Break
	17	Class Activity II
	18	Intro to Regression Analysis
	19–20	Regression with Interaction Effects
	20	Class Activity III
	21	Linear Mixed Effects Model I
	22	Linear Mixed Effects Model II
	23	Class Activity IV
	24	Model Selection
	25	Nested Model
		Part 4: Advanced Topic
	26	Predictive Modelling
	27	Machine Learning

Recommended Textbooks

- Statistics 480: Introduction to SAS, The Pennsylvania State University.
- SAS Training, SAS Institute.
- SAS Resources, University of California, Los Angeles.

Acknowledgments

Special thanks to Li-Hsiang Lin for providing the base materials given on this website.

Part I

Introduction

1 Introduction to Basic SAS Operation

Learning objective:

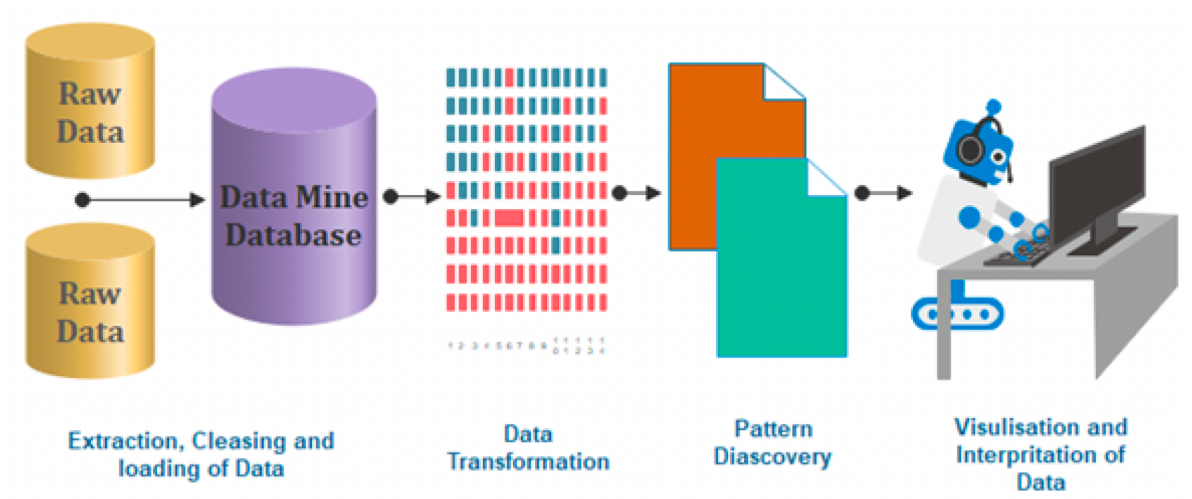
1. Familiarize ourselves with SAS windows (editor, log, output)
2. Create a dataset
3. Sorting Data (by 1 or more variables)
4. Obtain summary statistics of variables

1.1 Introduction to SAS

Q: What is SAS?

SAS (Statistical Analysis Software) is a prominent tool in the field of Data Analytics, offering a comprehensive suite for data manipulation, mining, management, and retrieval across various sources, coupled with robust statistical analysis capabilities. It excels in a range of functions including *data management, statistical analysis, report generation, business modelling, application development, and data warehousing.* SAS is user-friendly, featuring a point-and-click interface for those without technical expertise, while also providing deeper functionality through the SAS programming language. This software is instrumental in employing qualitative methods and processes that enhance employee productivity and business profitability.

Within SAS, data extraction and categorization into tables are pivotal for identifying and understanding data trends. This versatile suite supports advanced analytics, business intelligence, predictive analysis, and data management, facilitating effective operation in dynamic and compet- itive business environments. Additionally, SAS's platform-independent nature allows it to operate seamlessly across various operating systems, including Linux, Windows, Mac, and Ubuntu. SAS provides extensive support to programmatically transform and analyze data in the comparison of drag and drop interface of other Business Intelligence tools. It provides very fine control over data manipulation and analysis.



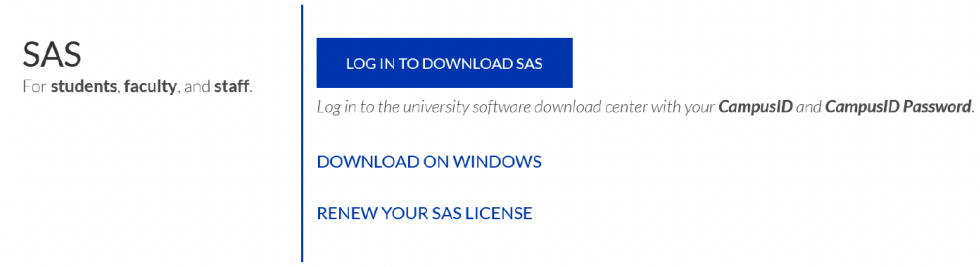
1.1.1 SAS Installzation

Georgia State University (GSU) has purchased license, so we can access SAS University Edition for free!

To install SAS University Edition, choose from the following options:

- **Option 1**:

  Download on your personal PC: Free SAS license available to GSU students, faculty, and staff via Technology Services (download required; check system requirements): Download from https://technology.gsu.edu/technology-services/software-equipment/university-licensed-software/ (Need to log-in from your GSU Account)



`Get Help for the Installation from <https://gsutech.service-now.com/sp>`

- **Option 2**:

- On Campus Access: SAS can be found on all GSU Library PCs: Floors 1-4 (not available on Library Macs, because there is no Mac version of SAS)
- Graduate Biostatistics Computer Lab (SPH): 6th floor of the Urban Life building (swipe card access required)
- Common MILE Lab whose opening time is
  - Monday & Wednesday: 9 – 18
  - Tuesday & Thursday: 9 – 17
  - Friday: 9 – 15

- **Option 3**:

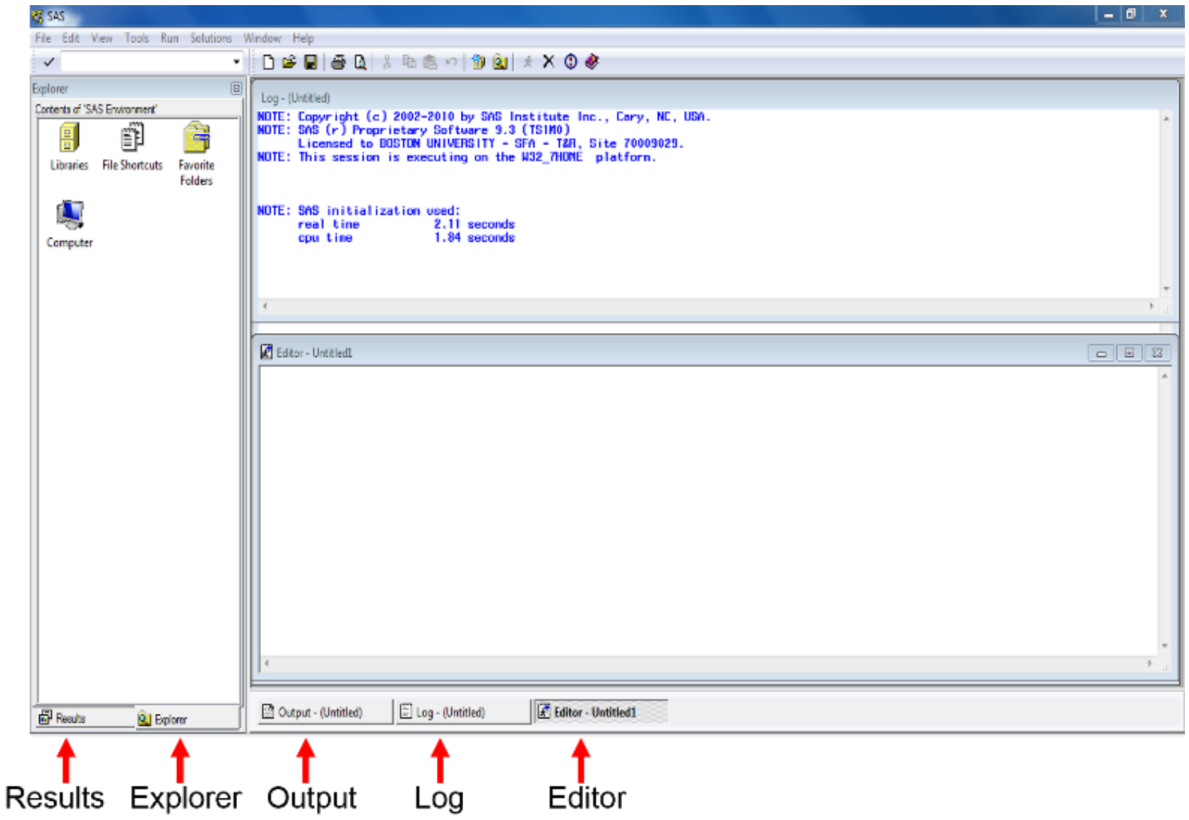
  Access via VLab, GSU's Remote Desktop Environment. Download and Connect to Cisco AnyConnect Client to connect to GSU's VPN (secureaccess.gsu.edu). Once connected to the VPN, login to VLab at: https://vlab.gsu.edu/ to access SAS.

- **Option 4**:

  Access via SAS OnDemand for Academics/SAS Studio. If you do not already have one, create a SAS profile at https://welcome.oda.sas.com/ Then, sign in with credentials and click SAS®Studio to access the web-based SAS environment.

1.1.2 SAS Windows

Once SAS has started, the screen will look similar to the following: The main SAS window is divided into several sub-windows:



- The menu and toolbar along the top of the window
- The **explorer/results browser** along the left hand side, where you can a listing of the results of successful SAS program.
- The **log** to the top right. This gives you information about possible errors after you have run your SAS program.
- The **program editor** below the log on the bottom right, where you create your SAS program.
- The windows bar along the bottom for you to switch all windows.

The **Editor (Program Editor)** window is a text editor that facilitates writing SAS programs (code). The Log window displays system messages, errors, and resource usage and is thus used to review program statements. The Output window displays output from statistical procedures run within the SAS program; however this is no longer the default. In SAS 9.3 output is sent to the Results Viewer which opens automatically when you run a procedure that generates output. The Results window displays a map of the Output window, and is useful for navigating the results of complicated analyses. Finally, the Explorer window contains all of the data sets in the current SAS session.

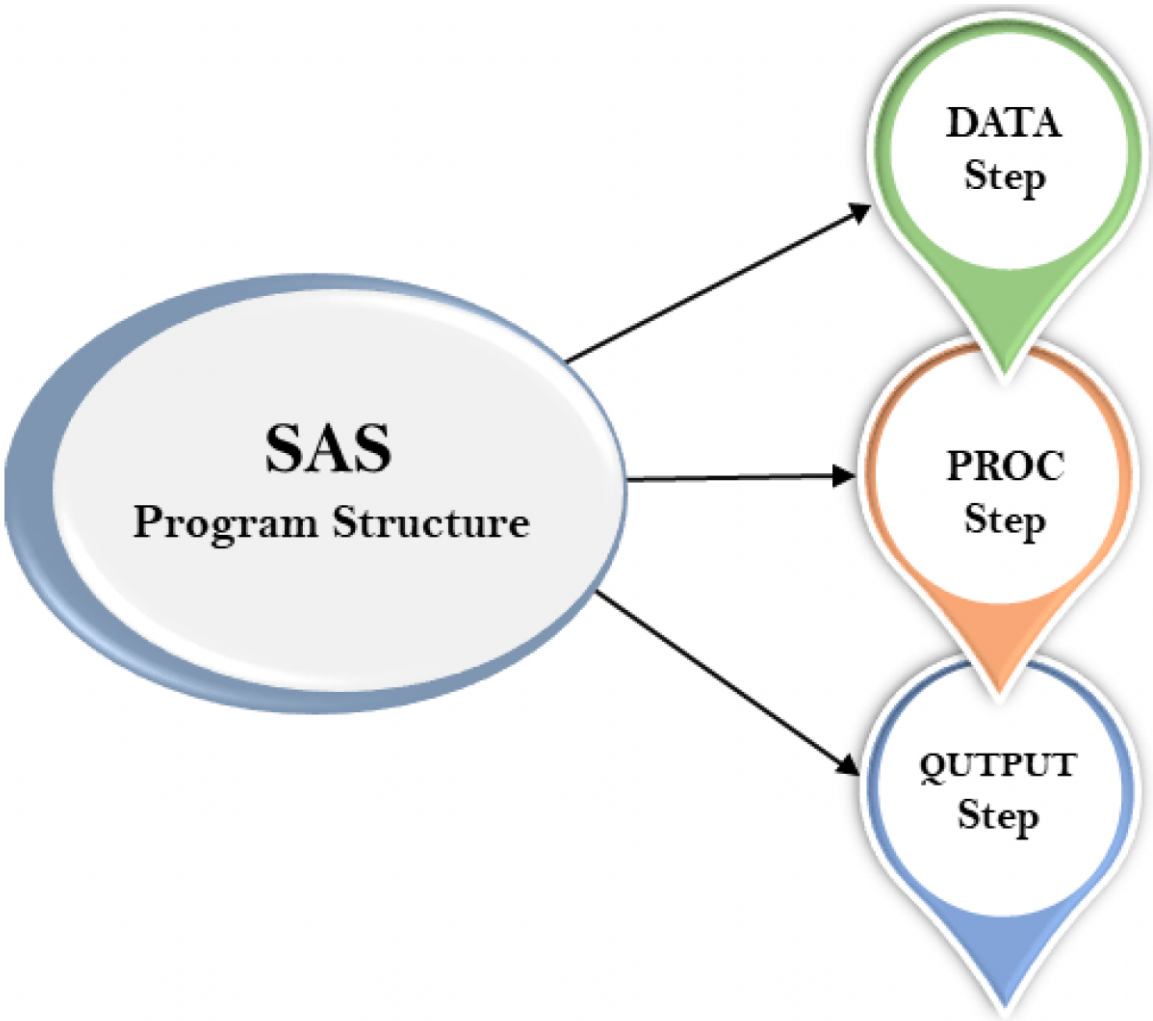
These windows can be moved or resized as desired. Only one SAS window is active at a time. The active window will have a shaded title bar at the top of the window, and a highlighted

windows bar at the bottom of the screen. In the above example, the Program Editor is the active window, with an "*Untitled*" program name. Note that the menu options for the SAS toolbar along the top of the screen depend on which window is currently active. (The active window can be changed by clicking on that window with the mouse, or by selecting the desired window from the Window menu.)

1.2 SAS Program

The programming structure of SAS consists of **three** significant steps:

- **DATA** step: create and modify a SAS data set for follow-up analysis
- **PROC** step: conduct data analysis
- **OUTPUT** step: show the analysis results




```
*Syntax of the SAS program:;
DATA dataset name; /* Name of the data set. */
INPUT var1,var2; /* Defines the variables in this data set. */
NEW_VAR; /* Creates a new variable. */
LABEL; /* Assign labels to variables. */
DATALINES; /* Enters the data. */
RUN;
```

1.3 SAS Dataset

SAS dataset is used to organize data values in a tabular form, i.e., in the form of rows for observations and columns for variables.

A SAS data set is a matrix whose each column is for each **variable** and whose each row for each **observation** (e.g., subject).

Data sets can be entered in the SAS programming code or can be read in from a variety of external sources, such as text files, csv files, and Microsoft Excel. In subsequent classes we will discuss reading in data sets from external files. Once a data set has been created, commands or procedures can operate on these data sets.

Other than these steps programming structure also includes data set, label, variables, values, and run.

1.4 SAS Examples

1.4.1 Creating a Dataset

Our first task in using SAS will be to create a small dataset and "print" that dataset to the output window. As we mentioned in previous paragraph SAS programs usually start with a DATA step where the dataset is created. Once the dataset is available, various procedures can be run on the dataset. The example below is written in the SAS Program window. The program creates a dataset called "People" with 3 variables (columns) which are 'gender', 'height', and 'weight' and 14 observations (rows). Note that the values of the variables on each line are separated by one or more blanks. A few other things that you should note:

- All SAS statements end with a semicolon (**;**)
- More than one SAS statement can be put on a line, or a SAS statement can continue across several lines, if every statement *ends with a semicolon.*
- Data listed as part of the program is also terminated with a semicolon. Data does not have to be entered in the program; it can also be read from files that are external to the SAS program (more on that next week)
- *gender* is a character variables as indicated by the $, and height and weight are numeric variables.
- Once the dataset is created, various SAS procedures (called **PROC**s) can be used to analyze the data and present results. We will start with a listing of the data created with a procedure called **PROC PRINT.**

```
title1 'STAT 8678 Example 1';
title2 'Your name';
```

```
DATA people;
INPUT gender $ height weight;

DATALINES;
m 63 125
m 76 195
f 62 109
m 75 186
f 67 115
f 60 120
m 75 205
m 71 185
m 63 140
f 59 135
f 65 125
m 68 167
m 72 220
f 66 155
;

PROC PRINT DATA=people;
RUN;
```

The LOG window gives information on the execution of the program. If your program did not execute properly you should examine the log for error messages that may explain the failure. The program would then be modified if necessary and rerun.

SAS creates a new window called the **Results Viewer** when the program is executed and produces output. This window is in HTML format and a new tab for the window is created below the left-hand windows.

1.4.2 Sorting Data

The data can be sorted (in our case by *gender*) using **PROC SORT** by adding the following lines to the program. We can then go ahead and print our new dataset sorted by gender with the proc print step.

> **Reminder**
>
> Any procedural step we do must begin with PROC and every line must end with a semicolon and the command run;

```
PROC SORT data = people;
BY gender;
RUN;

PROC PRINT data=people;
TITLE3 "Raw data sorted only by gender";
RUN;
```

> **Note**
>
> Any time we use PROC SORT our original dataset is sorted. SAS does not create a copy then sort!

1.5 Generating Summary Statistics

The last procedure to be executed in this exercise is **PROC UNIVARIATE**. This procedure will allow us to see summary statistics for any *quantitative variable*. The output from PROC UNIVARIATE will be important for the early part of this statistical methods class. It will provide measures of central tendency (mean, median, mode) and measures of dispersion (variance, standard deviation, range) as well as other basic statistics.

```
PROC UNIVARIATE DATA=people PLOT;
  BY gender;
  TITLE3 "Univariate procedure output done separately by gender";
  TITLE4 "The analysis was done for two quantitative variables";
  VAR HEIGHT WEIGHT;
RUN;
```

The code below applies the procedure only to the variable gender. It can be run on any quantitative variable and it could be run on several variables at the same time by listing several variables in the **VAR** statement, which would provide a separate analysis for each variable.

The results for PROC UNIVARIATE will be listed in the results viewer.

1.6 Other notes

Other Note

- SAS does not distinguish between upper-case letters or lower-case letters in the program, either can be used. However, it does distinguish between upper and lower case in datasets, so the character strings "Carol", "carol" and "CAROL" would be considered different values of the variable "name" in the program above.
- Comments: Additionally, you may add comments anywhere in your program either by beginning the statement with an asterisk (*) and ending it with a semicolon (;) or by beginning with /* and ending with */. These comments may be thought of as marginal notes, and will show in the program editor and log, but not in the output window.

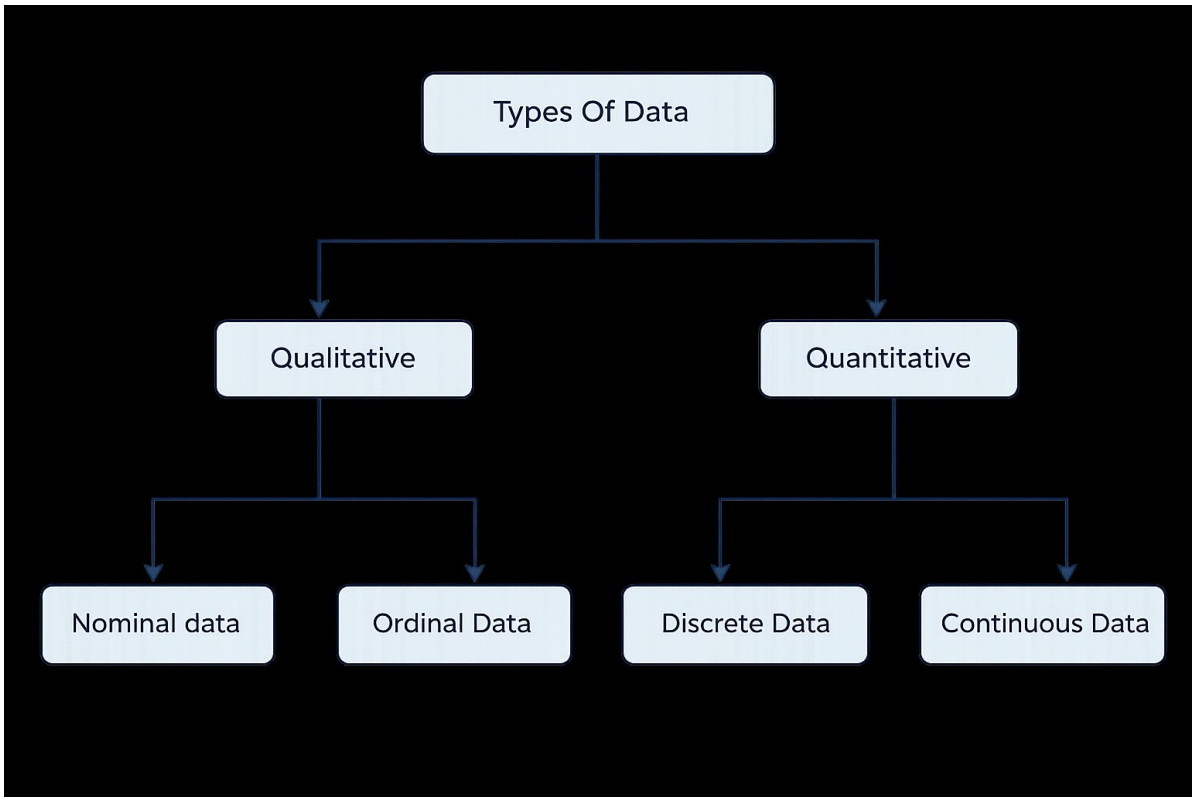
Shortcuts:

- F3 or the "running man": submits/run your program;
- F4: recalls text once the program editor;
- F5: directs user to the program editor;
- F6: directs user to the log;
- F7: directs user to the output;
- Ctrl+E clears content in the current window.
- Also, text can be copied with Ctrl+C, cut with Ctrl+X, or pasted with Ctrl+V.

1.7 Data Type

We can classify variables into *quantitative* variables and *qualitative* variables:

- Qualitative variables yield non-numerical information. Qualitative variables are often referred to as categorical variables, such as blood type. Qualitative variables can be further classified as
  - A nominal variable is a qualitative variable where no ordering is possible or implied in the levels, such as gender.
  - A ordinal variable is a qualitative variable with an order implied in the levels, such as health (poor, reasonable, good, or excellent)
- Quantitative variables yield numerical measurements. Quantitative variables can be further classified as discrete or continuous.
  - A discrete variable can assume only a countable number of values, such as headache severity scores.
  - A continuous variable is one that can take any one of an uncountable number of values in an interval, such as weight.



Question:

What is the type of each variable in the following dataset?

- AGE: The respondent's age in years
- GENDER: The respondent's sex coded 1 for male and 2 for female
- HAPPY: The respondent's general happiness, coded:1 for "Not too happy"2 for "Pretty happy"3 for "Very happy"
- TVHOURS: The average number of hours the respondent watched TV during a day

Table 1.1: Survey Respondent Summary

Respondent	AGE	Gen	HAPPY	TVHOURS
1	41	1	2	0
2	25	2	1	0
3	43	1	2	4
4	38	1	2	2
5	53	2	3	2
6	43	2	2	6

7	56	2	2	2

Answer:

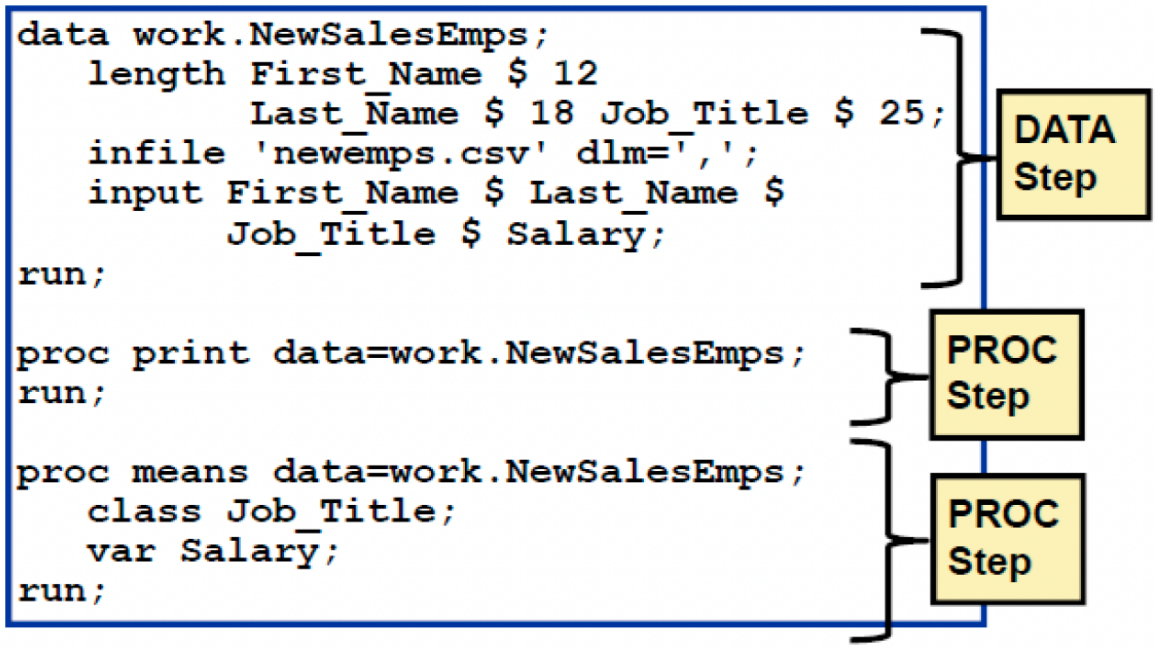
- Age: Continuous variable;
- SEX: qualitative;
- HAPPY: Discrete variable;
- TVHOURS: Continuous variable

2 Working with SAS Syntax

Learning objective:

1. Identify the characteristics of SAS statements.
2. Explain SAS syntax rules.
3. Insert SAS comments using two methods.
4. Identify SAS syntax errors.
5. Diagnose and correct a program with errors.
6. Save the corrected program

In general, writing a SAS program is to write a sequence of steps, and a step is a sequence of SAS statements:



How many statements are in the following step?

```
data work.NewSalesEmps;
   length First_Name $ 12
          Last_Name $ 18 Job_Title $ 25;
   infile 'newemps.csv' dlm=',';
   input First_Name $ Last_Name $
         Job_Title $ Salary;
run;
```

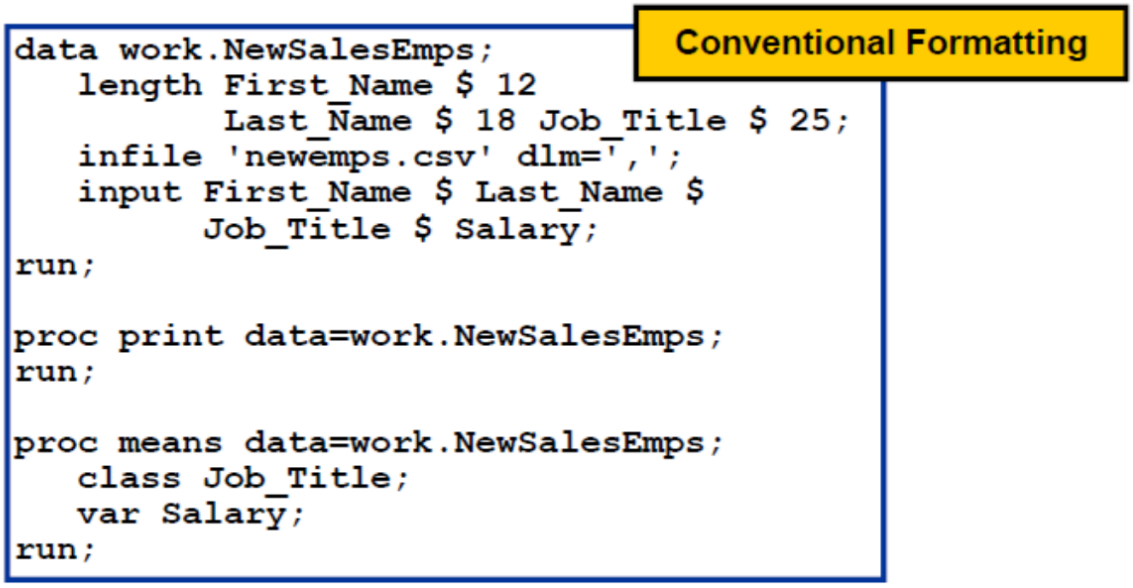
2.1 SAS Statments

SAS statements have these characteristics (Check the highlight context in the following examples):

- Usually begin with an identifying keyword.
- Always end with a semicolon.

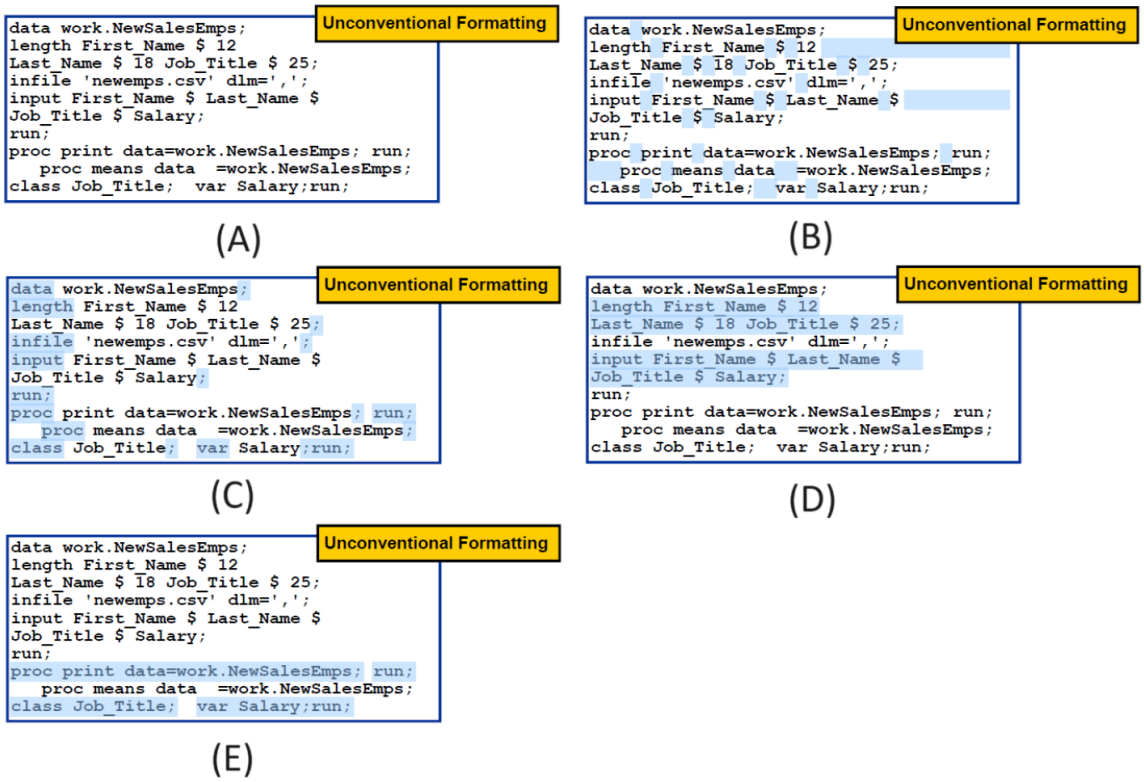
```
data work.NewSalesEmps;
   length First_Name $ 12
          Last_Name $ 18 Job_Title $ 25;
   infile 'newemps.csv' dlm=',';
   input First_Name $ Last_Name $
         Job_Title $ Salary;
run;

proc print data=work.NewSalesEmps;
run;

proc means data=work.NewSalesEmps;
   class Job_Title;
   var Salary;
run;
```

SAS programming statements are easier to read if you begin DATA, PROC, and RUN statements in column one and indent the other statements. This makes it more structured. Also, consistent spacing also makes a SAS program easier to read.



Overall, we can type SAS statements in the following ways:

- One or more blanks can be used to separate words.
- They can begin and end in any column.
- A single statement can span multiple lines.
- Several statements can be on the same line.



Sometimes we may want to make comments/notes to easy remember our SAS statements or to let other people easily understand what our code want to say. These comments are text that SAS ignores during processing. You can use comments anywhere in a SAS program to document the purpose of the program, explain segments of the program, or mark SAS code as non-executing text.

There are two ways to insert comments in a SAS program:

1. Using an asterisk (*) to begin the comment and a semicolon (;) to end the comment.

```
* This is a comment in SAS;
```

2. Using slash-asterisk (/*) to begin the comment and asterisk-slash (*/) to end the comment.

```
/* This is a comment in SAS */
```

Avoid placing the /* comment symbols in columns 1 and 2. On some operating environments, SAS might interpret these symbols as a request to end the SAS job or session. An example is given below:

```
*----------------------------------------*
|   This program creates and uses the    |
|   data set called work.NewSalesEmps.   |
*----------------------------------------*;
data work.NewSalesEmps;
   length First_Name $ 12 Last_Name $ 18
          Job_Title $ 25;
   infile 'newemps.csv' dlm=',';
   input First_Name $ Last_Name $
         Job_Title $ Salary /*numeric*/;
run;
/*
proc print data=work.NewSalesEmps;
run;
*/
proc means data=work.NewSalesEmps;
   *class Job_Title;
   var Salary;
run;
```

2.2 Diagnosing and Correcting Syntax Errors

Syntax errors occur when program statements do not conform to the rules of the SAS language.

Examples of syntax errors:

- misspelled keywords
- unmatched quotation marks
- missing semicolons
- invalid options

When SAS encounters a syntax error, SAS prints a warning or an error message to the log; for example,

```
ERROR 22-322: Syntax error, expecting one of the following:
              a name, a quoted string, (, /, ;, _DATA_, _LAST_,
              _NULL_.
```

When SAS encounters a syntax error, SAS underlines the error and the following information is written to the SAS log:

- the word **ERROR** or **WARNING**
- the location of the error + an explanation of the error

This program has three syntax errors. What are the errors?

```
daat work.NewSalesEmps;
   length First_Name $ 12
          Last_Name $ 18 Job_Title $ 25;
   infile 'newemps.csv' dlm=',';
   input First_Name $ Last_Name $
         Job_Title $ Salary;
run;

proc print data=work.NewSalesEmps
run;

proc means data=work.NewSalesEmps average max;
   class Job_Title;
   var Salary;
run;
```

2.3 Solution to the example

1. Exercise 1: 5
2. Exercise 2:

```
daat work.NewSalesEmps;
   length First_Name $ 12
          Last_Name $ 18 Job_Title $ 25;
   infile 'newemps.csv' dlm=',';
   input First_Name $ Last_Name $
         Job_Title $ Salary;
run;

proc print data=work.NewSalesEmps
run;

proc means data=work.NewSalesEmps average max;
   class Job_Title;
   var Salary;
run;
```

3 Import and Export Dataset

Learning objective:

1. Import a csv file into SAS
2. Export a csv file from SAS after data process

One of the strengths of SAS as a data analysis tool is its ability to read data from many sources, subset or combine data sets, and modify the datasets to accomplish various tasks. The most common types of external data sets used in SAS are EXCEL files (XLS extent), comma separated value files (CSV extent) and various space separate text files (PRN or TXT extent). A CSV file is actually a text file and can be read in any text reader (NOTEPAD or WORDPAD in Windows). In fact, the SAS files themselves, as well as the LOG and the LST files produced by a SAS by a batch submit, are also simple text files.

Format	Description	File Extension
WK1	Lotus 1 spreadsheet	.WK1
WK3	Lotus 3 spreadsheet	.WK3
WK4	Lotus 4 spreadsheet	.WK4
EXCEL	Excel Version 4 or 5 spreadsheet	.XLS
EXCEL4	Excel Version 4 spreadsheet	.XLS
EXCEL5	Excel Version 5 spreadsheet	.XLS
EXCEL97	Excel 97 spreadsheet	.XLS
DLM	delimited file (default delimiter is a blank)	.*
TAB	delimited file (tab-delimited values)	.TXT
CSV	delimited file (comma-separated values)	.CSV

3.1 Reading from External Files

The **PROC IMPORT** statement is the best way to enter external data sets. The CSV file we will be using is called "grades.csv". Download and save it in your favourite folder and mark the complete path to it. Then use the following code to import it, making sure you put the correct path on the **DATAFILE** argument.

```
PROC IMPORT OUT= GRADES_temp
  DATAFILE= "Put Your Path Here/grades_temp.csv" DBMS=CSV REPLACE;
  GETNAMES=YES;
  DATAROW=2;
RUN;
```

The **IMPORT** statement reads the dataset and stores it as the value designated by "OUT" in this case it will be saved as "Grades" in the library "Work".

The **DBMS** statement defines the type of input SAS should be reading. The following table gives you all the possible choices. The REPLACE argument forces SAS to overwrite any older datasets with the same name.

The **GETNAMES** = YES or NO statement for spreadsheets and delimited external files, determines whether to generate SAS variable names from the column names in the input file's first row of data. If you specify **GETNAMES** = NO or if the column names are not valid SAS names, PROC IMPORT uses the variable names VAR0, VAR1, VAR2, and so on. You may replace the equals sign with a blank.

The **DATAROW** argument tells SAS where to start reading for input data. In our case it is row 2 since row 1 is used for variable names.

We use the print procedure to see the dataset,

```
PROC PRINT DATA=GRADES_temp;
RUN
```

The first few rows are shown as follows

The SAS System

Obs	Student	Quiz1	Quiz2	Quiz3	Quiz4	Quiz5	Quiz6	Midterm	EC	Gender
1	.	7	7	7	7	7	7	25	2	M
2	1	4.55	6.8	6	5.1	5.75	6.7	13	1	M
3	2	5	6.1	6.7	5.4	7	3.8	21.5	1	M
4	3	3.75	6.5	6.9	2.1	6.6	5.5	21	2	M
5	4	5.05	6.2	7	5.9	6.75	6	24.5	1	M
6	5	5.35	6.8	6.6	5.1	6.6	6.5	24	1	M
7	6	7	7	6.7	6.1	7	7	25	1	F

3.2 Export a File from SAS

After reading a dataset into SAS, we may need to conduct some initial step/analysis to re-organize dataset for doing further data analysis. In the *grade* example, the first row shows the maximum points available for each quiz. We need to remove this row so that our analysis is correct. This can be done by the following SAS code

```
DATA GRADES;
  SET GRADES_temp NOBS=COUNT
    IF _n_         <= 1 THEN DELETE;
RUN;
```

Note: GRADES_temp is the name of the old dataset and *GRADES* is the name of the new dataset after the first row is deleted.

After re-organize the dataset, we may want to export the updated dataset for use in the future:

This can be done by

```
PROC EXPORT DATA= GRADES
  OUTFILE= "Put Your Path Here/grade_v2.csv" DBMS=CSV REPLACE;
  REPLACE;
RUN;
```

We will talk about more reorganizing skills in SAS in the next topic.

Which statement is true concerning the DATALINES statement based on reading the comment?

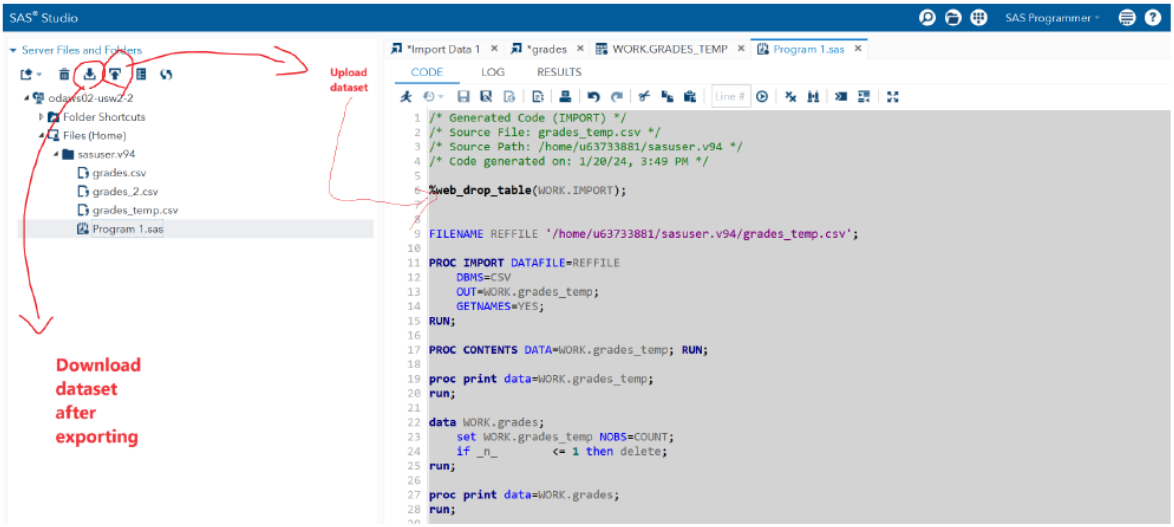
a. The DATALINES statement is used when reading data located in a raw data file.

b. The DATALINES statement is used when reading data located directly in the program.

3.3 Import & Export in SAS Virtual Studio

The key is

- Upload data before conducting "Import"
- Download data after conducting "Export"



A example given below will be demonstrated during the class.

```
/* Generated Code (IMPORT) */
/* Source File: grades_temp.csv */
/* Source Path: /home/u63733881/sasuser.v94 */
/* Code generated on: 1/20/24, 3:49 PM */

%web_drop_table(WORK.IMPORT);

FILENAME REFFILE '/home/u63733881/sasuser.v94/grades_temp.csv';

PROC IMPORT DATAFILE=REFFILE
    DBMS=CSV
    OUT=WORK.grades_temp;
    GETNAMES=YES;
RUN;

PROC CONTENTS DATA=WORK.grades_temp;
RUN;

PROC print data=WORK.grades_temp;
RUN;

DATA WORK.grades;
SET WORK.grades_temp NOBS=COUNT;
IF _n_ <= 1 THEN DELTE;
RUN;
```

```
PROC PRINT DATA=WORK.grades;
RUN;

PROC EXPORT data=WORK.grades
    OUTFILE = "/home/u63733881/sasuser.v94/grades_2.csv"
    DBMS = csv
    REPLACE;
RUN;
```

3.4 Solution to the example

Answer of the example: b

4 Random Variables

Learning Objectives

1. Distinguish between **discrete** and **continuous** random variables
2. Define random variables for real-world data problems
3. Identify appropriate probability distributions for common data types
4. Connect random variables to **data columns** used in SAS programs to represent a real world question

4.1 What Is a Random Variable?

A **random variable (RV)** is a numerical quantity whose value depends on the outcome of a random experiment.

We typically denote a random variable by an uppercase letter, such as X and its realized value by a lowercase letter, such as $X = x$. The random variable can be *continuous* or *discrete*.

In practice, we often observe multiple realizations, or running the random experiment multiple times, say n times or n realizations. We denote these realizations as:

$$X_1 = x_1,\ X_2 = x_2,\ \ldots,\ X_n = x_n.$$

The number n is called the **sample size**.

Define whether the following random variables are discrete or continuous, and the possible values that X takes.

- Number of emails received by a server in one hour ($X = 0, 1, 2, \ldots$: discrete)
- Time (in minutes) until a machine fails ($X = x \in [0, \infty)$: continuous)
- Total number of defects on a manufactured item ($X = 0, 1, 2, \ldots$: discrete)
- Daily maximum temperature in Atlanta (in degrees Fahrenheit) ($X = x \in (-\infty, \infty)$: continuous)
- Whether a randomly selected loan defaults within one year ($X = 1$ if default, 0 otherwise: discrete)

Define whether the following random variables are discrete or continuous, and the possible values that X takes.

1. Number of transactions made by a customer in a day
2. Response time (in seconds) of a web service request
3. Count of hospital admissions in a city per week discrete)
4. Proportion of time a system is idle during a day

4.2 Discrete vs Continuous Random Variables

4.2.1 Discrete Random Variables

A **discrete random variable** takes values in a **finite or countable set**.

Examples:

- Number of heads in three coin flips
- Number of students passing an exam
- Number of events occurring in a fixed time period

A discrete random variable is described by a **probability mass function (PMF)**:

Value of X	x_1	x_2	x_3	...	x_m
Probability	p_1	p_2	p_3	...	p_m

These probabilities satisfy:

- $0 \leq p_i \leq 1$,
- $\sum_{i=1}^{m} p_i = 1$.

We calculate the probability of events modelled by discrete random variables by summing up the probability p_i for the values x_i that make up the event.

Suppose the length X (in minutes) of an international phone call has distribution:

X	1	2	3	4
P(X)	0.2	0.5	0.2	0.1

Then, calculate the following probabilities

a. $P(X \leq 2)$
b. $P(X < 2)$
c. $P(X > 1)$

4.3 Common Discrete Distributions

4.3.1 Bernoulli Distribution

Used for **binary outcomes**:

- success / failure
- yes / no
- 1 / 0

Notation: $X \sim \text{Ber}(p)$, where p is the probability of success.

Example:

Whether a patient has diabetes (1 = yes, 0 = no).

How to specify the probability p will be discussed in the following lecture. This is related to the procedure of statistical inference.

4.3.2 Poisson Distribution

Used to model **counts of events over time or space**.

Notation: $X \sim \text{Poi}(\lambda)$

where λ is the **mean rate**.

Examples:

- Number of trades per day
- Number of system failures per week
- Number of arrivals to a service queue

Similar to the probability p from the Bernoulli distribution, how to specify the rate λ will be discussed in the following lecture.

4.4 Continuous Random Variables

A **continuous random variable** can take **any value** x in an interval.

Examples:

- Height of individuals
- Time until failure of a component
- Test scores treated as continuous

Probabilities are defined using **density functions**, not point probabilities. Some common continuous distributions are as follows.

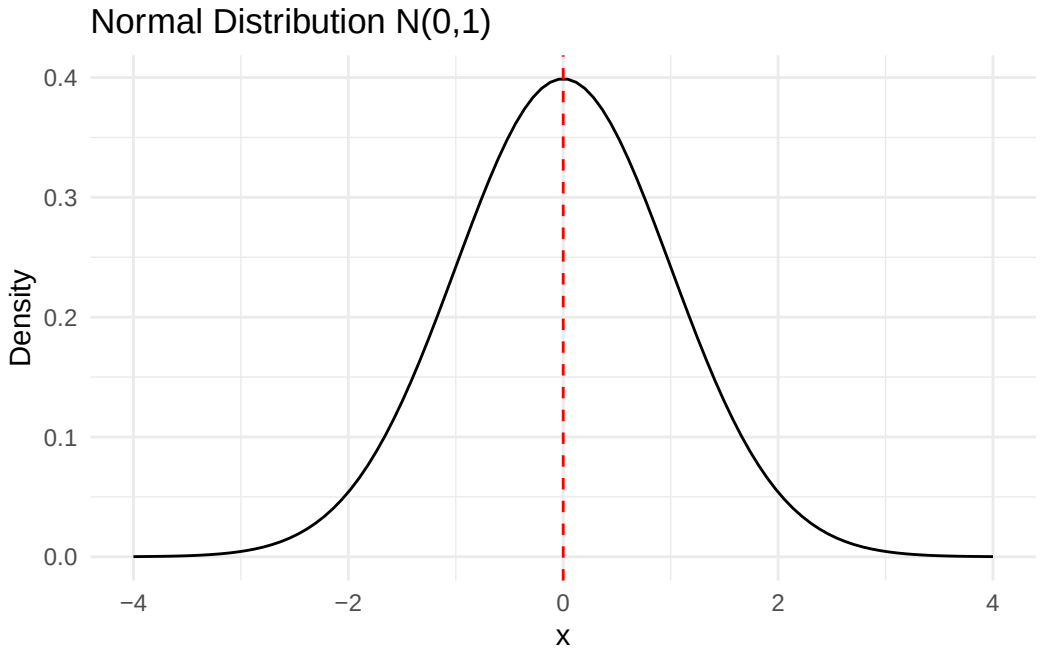
4.4.1 Normal Distribution

The normal distribution is also called the **Gaussian distribution**. It is characterized by two parameters: the mean μ and the standard deviation σ. It is unimodal and symmetric around the mean which is the centre of the mass. A continuous random variable X that has a normal distribution is said to be *normal* or *normally distributed*.

Notation: $X \sim N(\mu, \sigma^2)$. Sometimes the standard deviation may be used instead of the variance, which is the square of the variance.

Institution of the normal distribution:

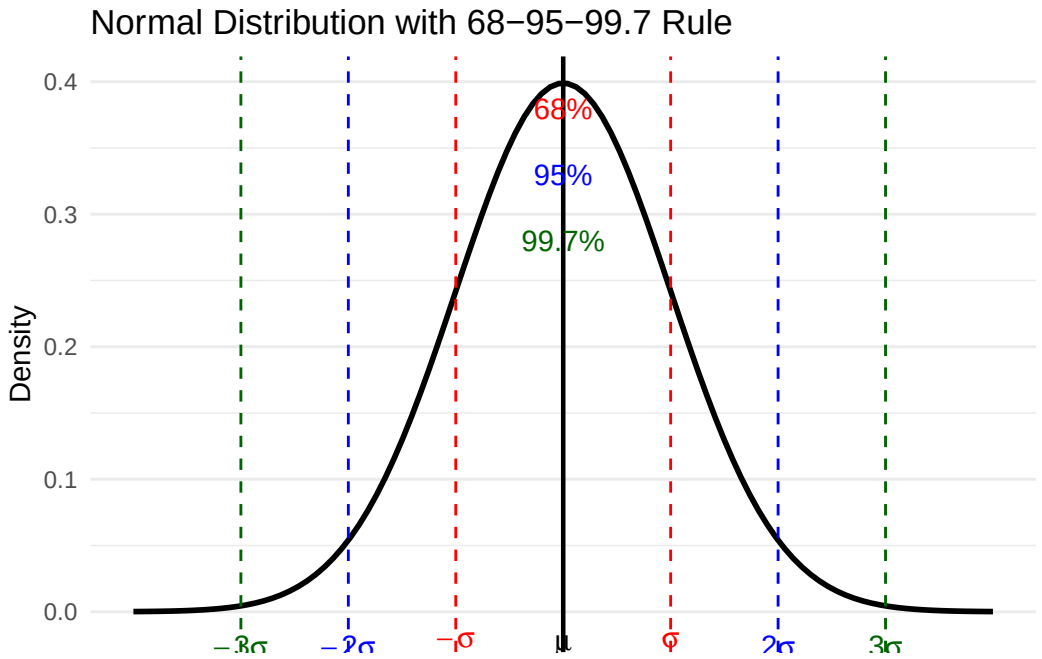
- mean: μ
- variance: σ^2



A distribution plot of the normal distribution with mean 0 and standard deviation 1 is shown above. It is often referred as the *standard normal distribution*. In practice, we would like to "standardized" the data to have mean 0 and standard deviation 1 for the subsequent analysis.

Normal distribution play an important role in statistical inference. One of the important property is the 68-95-99.7 rule: - About 68% of the data falls within one standard deviation of

the mean - About 95% of the data falls within two standard deviations of the mean - About 99.7% of the data falls within three standard deviations of the mean



4.4.2 Exponential Distribution

The exponential distribution is another common distribution where a few outcomes are most likely with a rapidly decreasing probability for larger values. It is often used to model waiting times or lifetimes of objects. It is similar to the *geometric distribution* in the discrete case.

Two ways to specify the parameter.

1. $X \sim \text{Exp}(\lambda)$, where λ is the *rate* parameter.
2. $X \sim \text{Exp}(\beta)$, where β is the *scale* parameter, and $\beta = 1/\lambda$.

The notation is either $X \sim \text{Exp}(\lambda)$ or $X \sim \text{Exp}(\beta)$. But to be sure which parameterization is used, we need to check the definition of the distribution.

Please define adequate random variables for the following data example, and thaink about what distribution can be used to model the data.

- Suppose we want to know whether the rate of diabetes of a certain area is too high. The researchers randomly discuss with 10 individuals in a certain area to know whether they have diabetes. The data are collected as "yes" or "no" answers as follows

Individual	1	2	3	4	5	6	7	8	9	10
Outcome	1	0	0	0	0	1	0	0	0	0

- A comapny that manufactures light blulb claims that a particular type of light bulb will last 850 hours on average. To justify the claim more scientifically, a researcher randomly selects 10 light bulbs of that type and measures the lifetimes (in hours) of the light bulbs as follows:

Light Bulb	1	2	3	4	5	6	7	8
	831	832	840	819	822	836	829	817

In the following classes we will assume the dataset we have can be well-modelled represented for the underline studies. The data collection, however, is beyond the scope of the class. For those interested in data collection, please refer to the *survey sampling* or *experimental design* area.

4.5 Solution to the exercise

Answer of the exercise

Exercise 1

1. Number of transactions made by a customer in a day ($X = 0, 1, 2, ...$: discrete)
2. Response time (in seconds) of a web service request ($X = x > 0$: continuous)
3. Count of hospital admissions in a city per week ($X = 0, 1, 2, ...$: discrete)
4. Proportion of time a system is idle during a day ($X = x \in [0, 1]$: continuous)

Exercise 2:

- $P(X \leq 2) = 0.7$
- $P(X < 2) = 0.2$
- $P(X > 1) = 0.8$

Part II

Statistical Analysis

5 Introduction to Statistical Inference I

Learning Objectives

1. Be familar with the difference about a probability problem and a statistical problem
2. Applied the one sample t-test to do inference for one sample mean problem

5.1 Probability versus Statistics

- In *probability*, we assume that random variables $X_1, \ldots, X_n$ follow a distribution with **known parameters**.
  Under this model, we can calculate probabilities of events of interest.
- In *statistics*, although we still use a distribution to model $X_1, \ldots, X_n$, the **parameters of the distribution are assumed to be unknown**.
  Our primary goal is to use observed data, the realization $X_1 = x_1, \ldots, X_n = x_n$—to make inference about these unknown parameters.

There are three common goals of statistical inference:

- **Point estimation**
- **Interval estimation**
- **Hypothesis testing**

> **Note**
>
> Note that there may be multiple valid methods for achieving each goal, even when working with the same dataset.

5.2 Example: One-Sample Mean Problem

The term *one sample* does **not** mean that there is only one observation. Instead, it means that there is **one population** under study.

In this problem, we are interested in making inference about the population mean, denoted by μ.

5.2.1 Problem Formulation

Suppose we want to conduct statistical inference on the mean length of a certain type of court case. Let

$$\mu = \text{the mean length of a certain type of court case.}$$

However, the true value of μ is unknown because we cannot observe all realizations of this process. As a result, statistical inference is required.

If μ were known, no inference would be necessary. What we can do in practice is to **collect data**. Suppose we randomly select 20 court cases of the same type from historical records and observe their case lengths (in days):

$$43,\ 90,\ 84,\ 87,\ 116,\ 95,\ 86,\ 99,\ 93,\ 92,$$
$$121,\ 71,\ 66,\ 98,\ 79,\ 102,\ 60,\ 112,\ 105,\ 98.$$

From a statistical perspective, these observed values are treated as **realizations of random variables**. Specifically, we denote

$$X_1 = 43,\ X_2 = 90,\ \dots,\ X_{20} = 98.$$

To model the data, we assume that the random variables

$$X_1, X_2, \dots, X_{20}$$

are **independent and identically distributed** according to a normal distribution with mean μ and variance σ^2, that is,

$$X_i \overset{\text{i.i.d.}}{\sim} N(\mu, \sigma^2), \qquad i = 1, \dots, 20.$$

Here, both μ and σ^2 are unknown parameters. In this example, our primary interest lies in estimating the population mean μ, while the variance σ^2 is treated as a nuisance parameter.

We can use the **maximum likelihood estimator** (MLE) or the **method of moments estimator** (MME) to construct a point estimator of μ. Under this model, however, they are coincide. The resulting estimator is the **sample mean**, denoted by

$$\hat{\mu} = \bar{X}_{20},$$

where

$$\bar{X}_{20} = \frac{1}{20}\sum_{i=1}^{20} X_i.$$

Under the assumed model, we may use either the **maximum likelihood estimator** or the **method of moments** to construct a point estimator for the population mean μ. In this model, both methods lead to the same estimator: the **sample mean**. We denote this estimator by

$$\hat{\mu} = \bar{X}_n.$$

Pay careful attention to the notation. We use a **capital letter** $\bar{X}_n$ to emphasize that the estimator itself is a **random variable**. Since the estimator is a function of the random variables $X_1, \ldots, X_n$, it is also random and therefore follows a probability distribution. The probability distribution of an estimator is called its **sampling distribution**. We will return to the concept of sampling distributions later in the course.

5.2.2 Formal Definitions

To be more precise, we introduce the following definitions.

- A **statistic** is any function of the random variables $X_1, \ldots, X_n$. Because it is a function of random variables, a statistic is itself a random variable and therefore has a probability distribution. If we use the parametric model to describe it, then there is a distribution that depends on the unknown parameters.
- A **(point) estimator** of a parameter θ, denoted by $\hat{\theta}$, is a statistic used to estimate θ. (Note, an estimator is a random variable because it is a statistics).
- A **(point) estimate** is the numerical value obtained by evaluating the estimator using the observed data $x_1, \ldots, x_n$. In particular, we use **lowercase notation** to indicate an estimate. For example, $\bar{x}_n$ denotes the observed value of the estimator $\bar{X}_n$.

So far, we have constructed a point estimator (and hence a point estimate). To construct a confidence interval or perform hypothesis testing, we need to know the **sampling distribution** of the estimator $\bar{X}_n$.

Under the normal model with unknown variance, the sampling distribution can be expressed as

$$\frac{\bar{X} - \mu}{S/\sqrt{n}} \sim t_{n-1},$$

where

$$S = \sqrt{\frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{n-1}},$$

and t_{n-1} denotes the **Student's** t distribution with $n-1$ degrees of freedom.

5.2.3 Confidence Interval for μ

Based on this sampling distribution, a $(1-\alpha) \times 100\%$ confidence interval for μ is given by

$$\left[\bar{X} - t_{n-1,\alpha/2}\frac{S}{\sqrt{n}},\ \bar{X} + t_{n-1,\alpha/2}\frac{S}{\sqrt{n}}\right].$$

5.2.4 Hypothesis Testing for μ

Using the same sampling distribution, we can conduct one of the following hypothesis tests:

- **Two-sided test**
$$H_0 : \mu = \mu_0 \quad \text{vs.} \quad H_1 : \mu \neq \mu_0$$
- **Left-tailed test**
$$H_0 : \mu = \mu_0 \quad \text{vs.} \quad H_1 : \mu < \mu_0$$
- **Right-tailed test**
$$H_0 : \mu = \mu_0 \quad \text{vs.} \quad H_1 : \mu > \mu_0$$

In some textbooks, equivalent hypotheses are written as

- $$H_0 : \mu = \mu_0 \quad \text{vs.} \quad H_1 : \mu \neq \mu_0$$
- $$H_0 : \mu \geq \mu_0 \quad \text{vs.} \quad H_1 : \mu < \mu_0$$
- $$H_0 : \mu \leq \mu_0 \quad \text{vs.} \quad H_1 : \mu > \mu_0$$

Fortunately, **SAS** can perform point estimation, confidence interval construction, and hypothesis testing simultaneously, provided we clearly specify:

- **The statistical question**
  (one-sample mean, one-sample variance, two-sample problem, etc.)
- **The inferential goal**
  (point estimation, confidence interval, hypothesis testing)
- **The statistical model and method**
  (for example, why we use the t distribution for the one-sample mean problem instead of the normal distribution or another alternative)

One of the main goals of this course is to help you build a mental "library" of statistical tools. Later, when you encounter a statistical question, you will be able to identify an appropriate method and then use SAS to obtain numerical results for interpretation.

5.3 Revisit the data example by using SAS

The court length data can be read by the following DATA step:

```
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;
```

The only variable in the DATA set, *time*, is assumed to be normally distributed. The trailing at signs (@@) indicate that there is more than one observation on a line. The following statements invoke PROC TTEST for a one-sample t test:

```
PROC TTEST H0=80 PLOTS(SHOWH0) SIDES=2 ALPHA=0.1;
  VAR TIME;
RUN;
```

- THE VAR statement indicates that the *time* variable is being studies
- the H0= option specifies that the mean of the time variable should be compared to the null value rather than the default value of 0
- the PLOTS(SHOWH0) option requests that this null value be displayed on all relevant graphs
- the SIDE=2 option specifies that a the focus of the research question, namely whether the mean count case length is not equal to 80days, rather than less or greater than 80 days (in which case you would use the SIDE=L or SIDE=U options, respectively)

- the ALPHA=0.1 option requests 90% confidence interval rather than the default 95% confidence interval

The output is presented in the table below.

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	82.4474	97.2526	19.1456	15.2002	26.2374

DF	t Value	Pr > |t|
19	2.30	0.0329

> **Note**
>
> Some SAS procedures produce graphs as automatically as they produce tables if the ODS GRAPHICS option is used. The graphs are integrated with tables in the ODS output.

```
ODS GRAPHICS ON;

PROC TTEST H0=80 PLOTS(SHOWH0) SIDES=U ALPHA=0.1;
  VAR TIME;
RUN;

ODS GRAPHICS OFF;
```

you will see the following results which include the same table you see in the last figure but with two more figures. We will talk those two Figures in the following classes, especially the QQ plot.

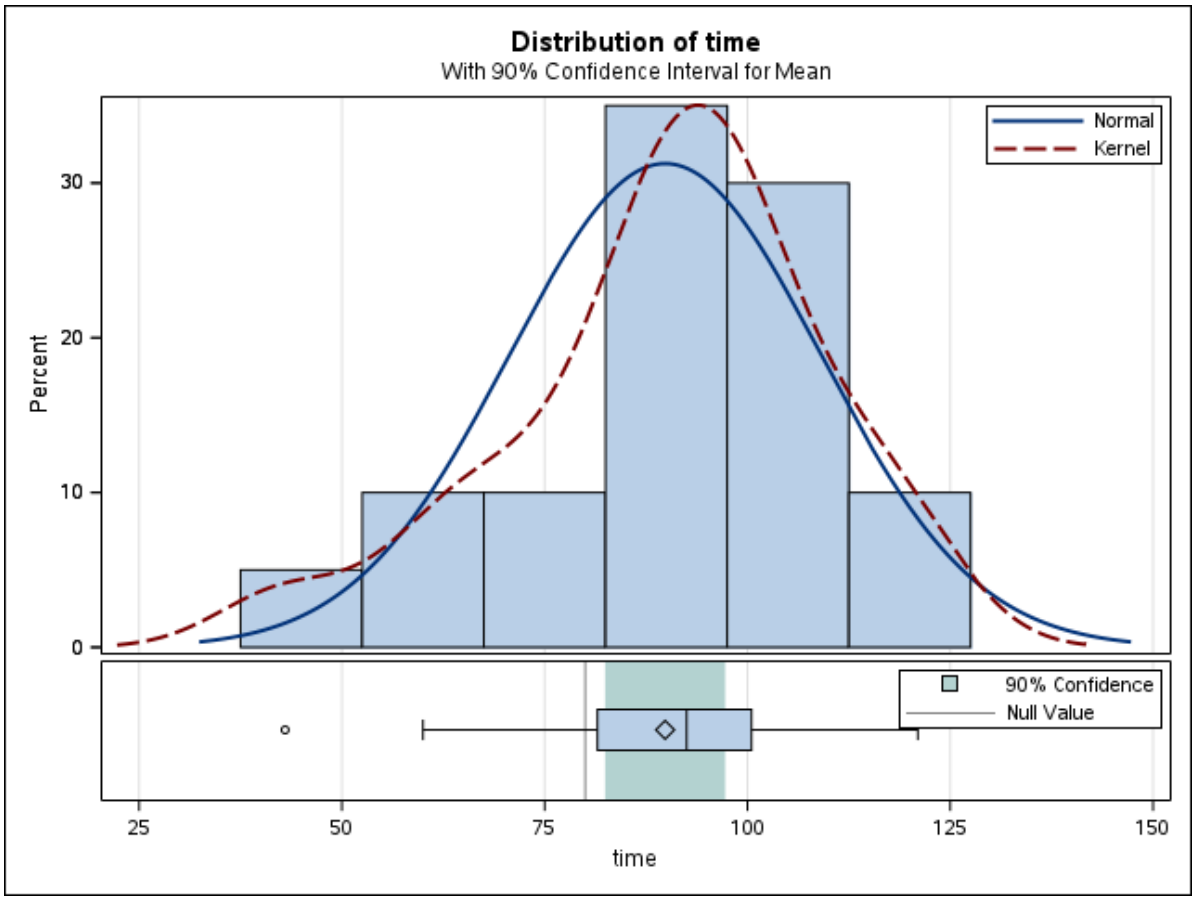
The TTEST Procedure

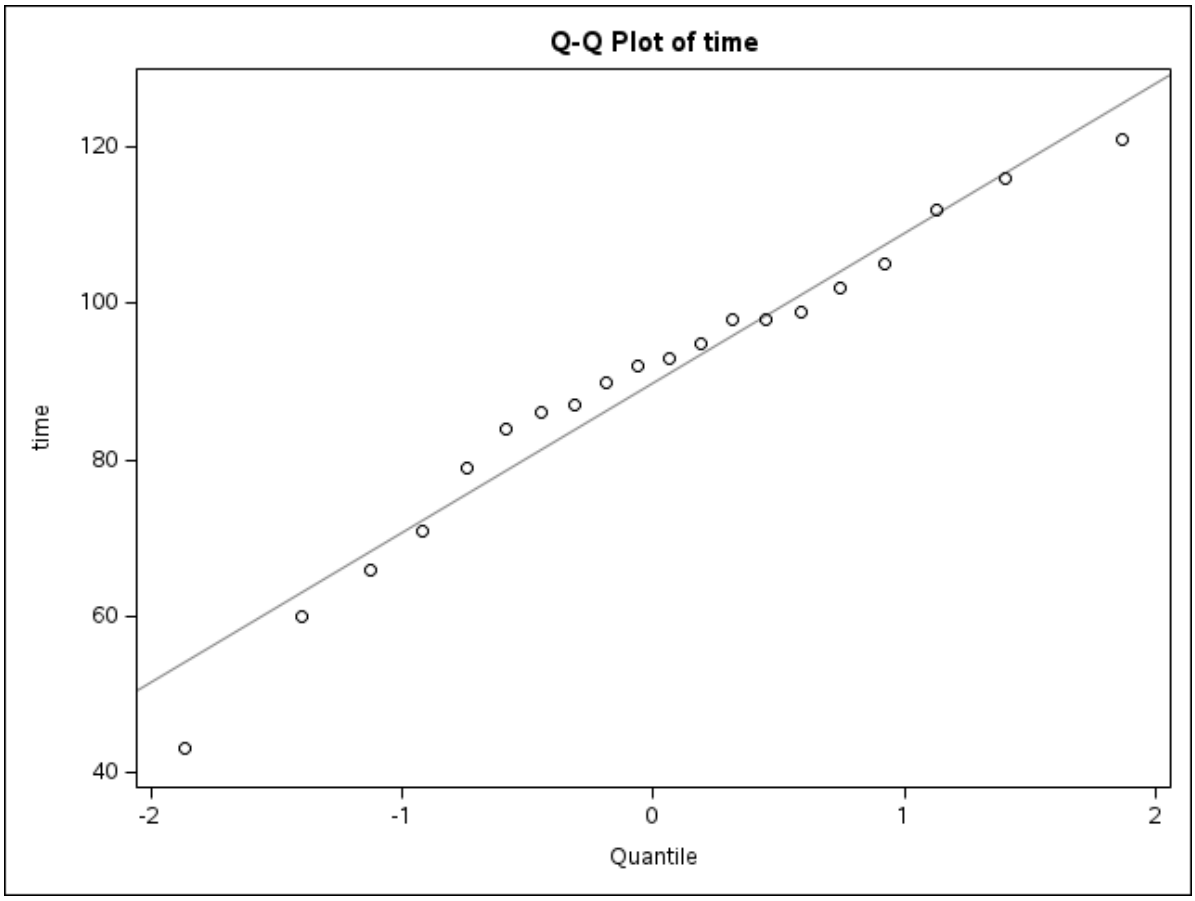
Variable: time

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	82.4474	97.2526	19.1456	15.2002	26.2374

DF	t Value	Pr > |t|
19	2.30	0.0329


Distribution of time
With 90% Confidence Interval for Mean
Normal
Kernel
Percent
30
20
10
0
90% Confidence
Null Value
25
50
75
100
125
150
time



5.4 More about hypothesis and confidence intervals

5.4.1 How to construct a confidence interval?

Consider a **95% confidence interval** for the population mean μ. Under the normal approximation (or by the Central Limit Theorem), we have

$$P(-1.96 < Z < 1.96) \approx 0.95,$$

where Z is a standard normal random variable.

Equivalently, this can be written as

$$P\left(-1.96 < \frac{\bar{X} - \mu}{\sigma/\sqrt{n}} < 1.96\right) \approx 0.95.$$

5.4.2 Rearranging the Inequality

Rearranging the terms inside the probability statement yields

$$P\left(\bar{X} - 1.96\frac{\sigma}{\sqrt{n}} < \mu < \bar{X} + 1.96\frac{\sigma}{\sqrt{n}}\right) \approx 0.95.$$

5.4.3 Large-Sample Confidence Interval

Thus, a **large-sample 95% confidence interval** for μ is given by

$$\left[\bar{X} - 1.96\frac{\sigma}{\sqrt{n}},\ \bar{X} + 1.96\frac{\sigma}{\sqrt{n}}\right].$$

More compactly, we often write this interval as

$$\bar{X} \pm 1.96\frac{\sigma}{\sqrt{n}}.$$

5.5 3.2 Hypothesis Testing and Confidence Intervals Always Agree

5.5.1 Interpretation of the p-value

The p-value is the probability of observing data **at least as favorable to the alternative hypothesis** H_A as the data actually observed, **assuming the null hypothesis** H_0 is true.

In this example, the observed sample mean is either greater than 3.56 or less than 3.18, and the null hypothesis assumes the true population mean is $\mu = 3.37$.

5.5.2 Computing the p-value

We compute the p-value as

$$P(\bar{X} > 3.56 \text{ or } \bar{X} < 3.18 \mid \mu = 3.37).$$

This can be written as the sum of two tail probabilities:

$$P(\bar{X} > 3.56 \mid \mu = 3.37) + P(\bar{X} < 3.18 \mid \mu = 3.37).$$

Standardizing using the normal distribution yields

$$P\left(Z > \frac{3.56 - 3.37}{0.31/\sqrt{147}}\right) + P\left(Z < \frac{3.18 - 3.37}{0.31/\sqrt{147}}\right).$$

Evaluating the standardized values gives

$$P(Z > 7.43) + P(Z < -7.43).$$

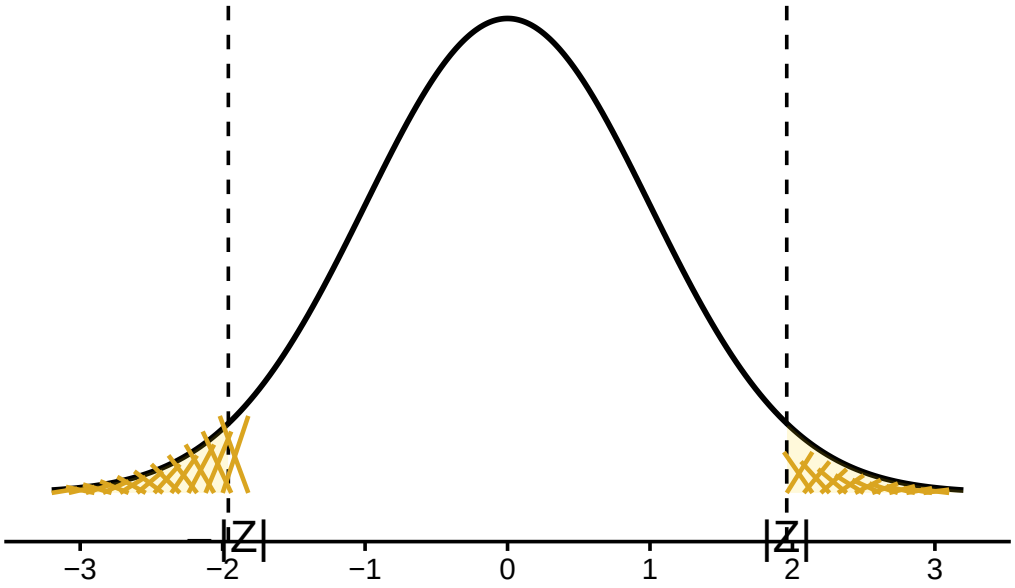
This probability is extremely small:

$$10^{-13} \approx 0.$$

5.5.3 Connection to Confidence Intervals

Because the p-value is essentially zero, we strongly reject H_0 at any reasonable significance level.

Equivalently, the hypothesized value $\mu = 3.37$ does **not** lie inside the corresponding confidence interval for μ.



This illustrates a key principle:

> **Hypothesis testing and confidence intervals always lead to the same conclusion when they are constructed at compatible significance levels.**

6 Introduction to Statistical Inference II

Learning Objectives

1. Distinguish between a *probability problem* and a *statistical (inference) problem*
2. Apply the *one-sample t* test to inference for a population mean
3. Conduct *model diagnostics* for the normality assumption
4. Correctly *interpret point estimates, confidence intervals, and hypothesis tests*

6.1 1. Recall: One-Sample Mean Problem

In the previous class, we studied inference for a *population mean* and show how to run SAS to obtain the statistical inference result from one sample t test. The example for illusrtratioe is to study the mean length of a certain type of the court case, i.e.,

$$\mu = \text{the mean length of a certain type of court case.}$$

The data for the study is

$$43,\ 90,\ 84,\ 87,\ 116,\ 95,\ 86,\ 99,\ 93,\ 92,$$
$$121,\ 71,\ 66,\ 98,\ 79,\ 102,\ 60,\ 112,\ 105,\ 98.$$

and we implement the statistical model:

$$X_i \overset{\text{i.i.d.}}{\sim} N(\mu, \sigma^2), \qquad i = 1, \dots, 20.$$

With this set up, we can use SAS, such as through the following code

```
ODS GRAPHICS ON;

PROC TTEST H0=80 PLOTS(SHOWH0) SIDES=U ALPHA=0.1;
  VAR TIME;
RUN;

ODS GRAPHICS OFF;
```

and obtain the statistical results. The results let us know

- A point estimate of μ
- An interval estimate of μ
- A hypothesis testing under $H_0 : \mu = 80$ versus H_1,

from t-distribution for the one sample mean problem.

From the example, we may think the beginning of conducting statistical analysis procedure as

i. Formulate the statistical problem

ii. Collection the adequate dataset and think the dataset as realization of some random variables

iii. Think some statistical models (i.e., assumptions) to be considered for the random variables. This step may include some [iv]

iv. preliminary data analysis

In this lecture, we will continue the remaining two steps:

v. Model diagnostics

vi. Interpret the results

6.2 Model Diagnostics

In this step, we pause and think carefully about the assumptions underlying our analysis.

Question:

What statistical models are we implementing?

In this course, the key assumptions for the one-sample mean problem are:

- **A normality assumption**
- **An independence assumption**

The independence assumption is primarily determined by *how the data are collected*. Once the data have been sampled, this assumption is generally *not testable* from the data alone.
We therefore assume that the sampling process was conducted correctly and focus our attention on checking the *normality assumption*.

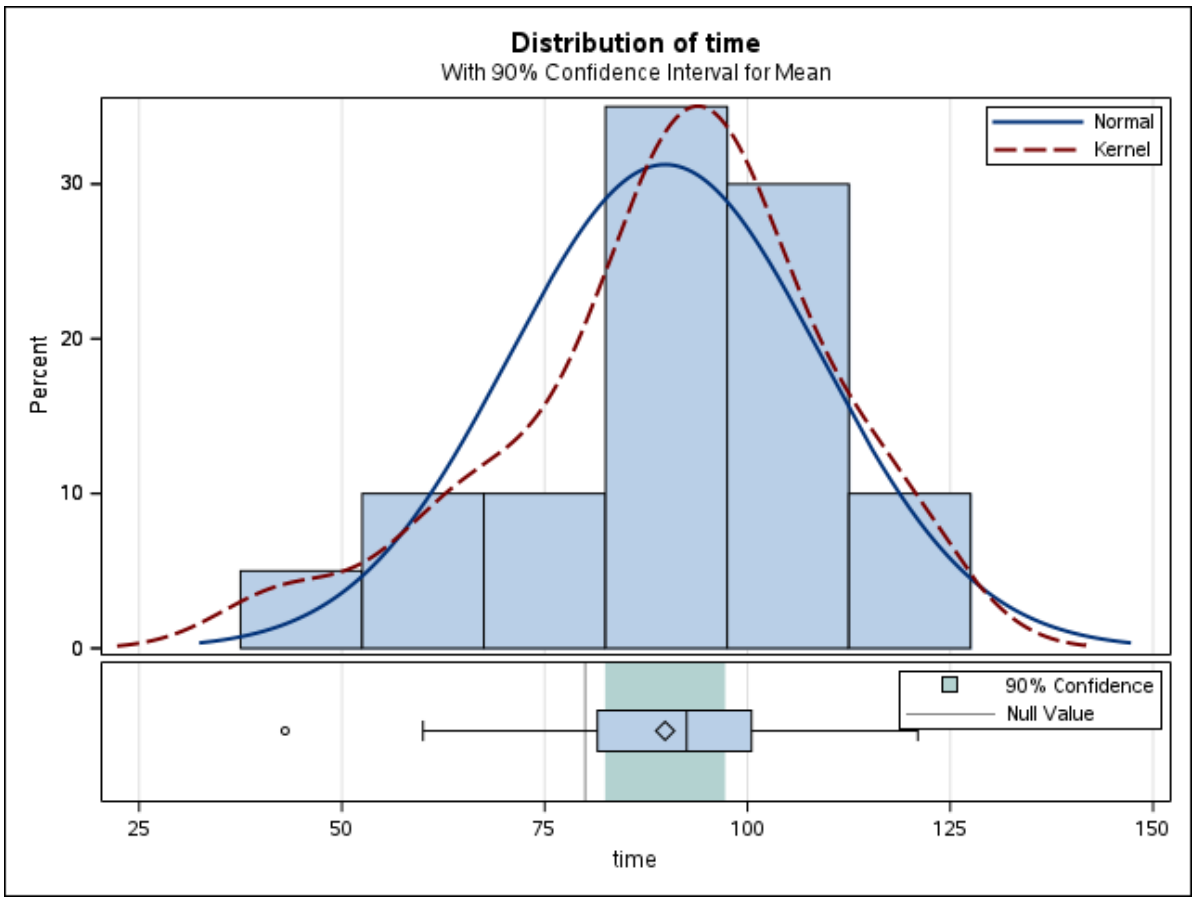
There are two broad classes of methods:

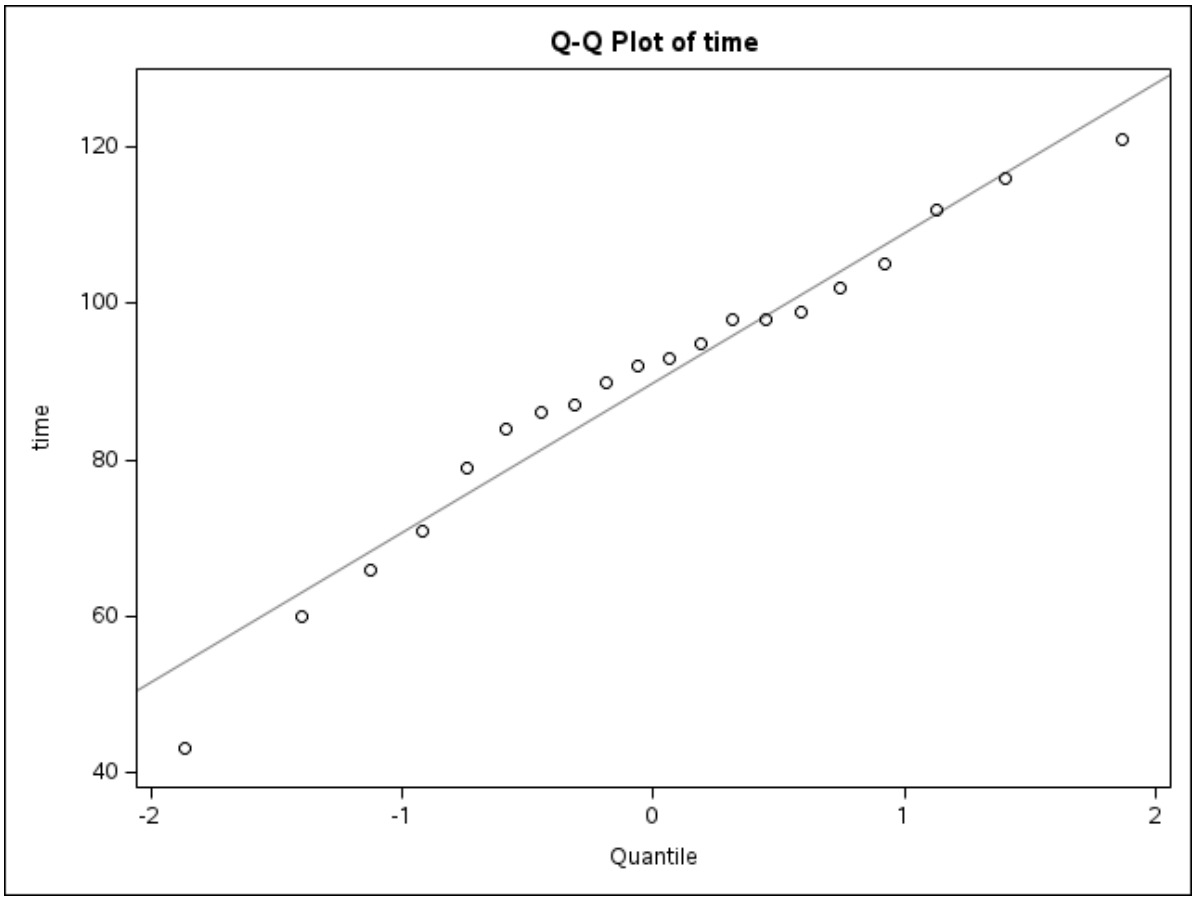
- *Graphical (visual) diagnostics*
- *Formal hypothesis testing methods*

6.2.1 Graphical Diagnostics for Normality

Common graphical tools include:

- **Histogram and Density Plots**:
  A kernel density estimate (KDE) provides a smooth estimate of the underlying probability density function, analogous to a histogram but without binning. KDEs represent the data using a continuous curve and are particularly useful for visualizing distributional shape.
- **Q–Q Plot (Quantile–Quantile Plot)**
  A Q–Q plot compares *empirical quantiles* of the observed data with *theoretical quantiles* from a normal distribution.
  If the normality assumption is reasonable, the points should fall approximately along a straight line.





6.2.2 Formal Hypothesis Testing for Normality

In addition to graphical methods, we can also assess the normality assumption using *formal numerical tests*.
One commonly used method is the **Kolmogorov–Smirnov (K–S) test**, which evaluates whether a sample plausibly comes from a specified distribution, such as the normal distribution.

The K–S test is widely used because many statistical procedures rely on the assumption that the data are normally distributed. When this assumption is violated, standard inference procedures may no longer be valid. The following step-by-step example demonstrates how to perform a Kolmogorov–Smirnov test for normality using **SAS**.

Step 1: Create the dataset

```
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;
```

Step 2: Perform the normality test

```
/* Perform Kolmogorov-Smirnov test */
PROC UNIVARIATE DATA=time;
    HISTOGRAM time / NORMAL(mu=est sigma=est);
RUN;
```

The UNIVARIATE Procedure
Fitted Normal Distribution for time

Parameters for Normal Distribution		
Parameter	**Symbol**	**Estimate**
Mean	Mu	89.85
Std Dev	Sigma	19.14563

Goodness-of-Fit Tests for Normal Distribution				
Test	**Statistic**		**p Value**	
Kolmogorov-Smirnov	**D**	0.12997262	**Pr > D**	>0.150
Cramer-von Mises	**W-Sq**	0.05410070	**Pr > W-Sq**	>0.250
Anderson-Darling	**A-Sq**	0.31058808	**Pr > A-Sq**	>0.250

Quantiles for Normal Distribution		
	Quantile	
Percent	**Observed**	**Estimated**
1.0	43.0000	45.3106
5.0	51.5000	58.3582
10.0	63.0000	65.3139
25.0	81.5000	76.9365
50.0	92.5000	89.8500
75.0	100.5000	102.7635
90.0	114.0000	114.3861
95.0	118.5000	121.3418
99.0	121.0000	134.3894

Remarks

Graphical diagnostics are informal but highly informative.
They allow us to detect skewness, heavy tails, and outliers that may invalidate normal-based inference.
Formal hypothesis tests for normality will be introduced next, but should always be interpreted in conjunction with these visual tools.

Important Note

For the best practice:
Never rely on a single normality test. Always combine numerical tests with visual inspection.

6.3 Interpretation of Results

6.3.1 Point Estimation

Point estimation is the process of using sample data to compute a single numerical value that estimates an unknown population parameter, such as the population mean.

When we report a point estimate, it is desirable to understand whether the estimator has good statistical properties. In particular, we often examine whether a point estimator is:

- **Consistent**: As the sample size increases, the estimator becomes closer to the true parameter value.
- **Unbiased**: The expected value of the estimator equals the true population parameter. For example, the sample mean is an unbiased estimator of the population mean.
- **Efficient (or best unbiased)**: Among all unbiased and consistent estimators, it has the smallest variance, meaning the estimator varies less from sample to sample.

How to rigorously verify these properties is a major topic in theoretical statistics courses. In practice, when these properties are unknown or difficult to assess, the safest interpretation of a point estimate is simply to report it with its associated unit. For example, using the court length data, we may state:

> *A reasonable estimate for the average court length is approximately 89.85 days.*

6.3.2 Confidence Intervals

A confidence interval (CI) provides a range of plausible values for an unknown population parameter. A common—but informal—interpretation of a 95% confidence interval is that we are "95% confident" the true parameter lies within the interval. While this interpretation is widely used, the **strictly correct interpretation** is based on repeated sampling:

> If the same study were repeated infinitely many times, and a 95% confidence interval were constructed each time, then approximately 95% of those intervals would contain the true parameter value.

As an example, suppose the 90% confidence interval for the population mean court length is $[15.2,\ 26.2]$. This means that, under the confidence interval procedure used, intervals constructed in this way would contain the true mean in 90% of repeated samples.

Factors Affecting Confidence Interval Width

The width of a confidence interval depends on several factors:

- **Sample size**
  Larger samples typically produce narrower (more precise) confidence intervals.
- **Variability of the outcome**
  For continuous outcomes, higher variability (larger standard deviation) leads to wider intervals.
- **Outcome type**
  - For binary outcomes, precision depends on the event probability.
  - For time-to-event outcomes, precision depends on the number of observed events.

All of these factors influence the standard error of the estimator, which directly determines the width of the confidence interval.

6.3.3 Hypothesis Testing

Hypothesis testing evaluates whether observed data are consistent with a specified statistical assumption, typically called the **null hypothesis**.

The result of a hypothesis test allows us to decide whether the assumption is supported by the data or whether there is sufficient evidence to reject it. The strength of this evidence is quantified by the *p-value*.

A **p-value** is defined as:

> The probability of observing data at least as extreme as the data actually observed, assuming the null hypothesis is true.

For example, if a hypothesis test yields a p-value of 0.01, this means that—if the null hypothesis were true—there would be only a 1% chance of observing data this extreme.

A small p-value provides evidence against the null hypothesis, while a large p-value indicates that the data are consistent with it.

If the observed data are very unlikely to occur under the conditions described by the null hypothesis, then the null hypothesis is unlikely to be true. In such cases, we reject the null hypothesis in favor of the alternative hypothesis, and the result is said to be **statistically significant**.

A commonly used decision rule is based on a significance level of 0.05. If the p-value is less than or equal to 0.05, this is typically taken as evidence against the null hypothesis, and we reject it in favor of the alternative hypothesis. However, a p-value cannot be used to prove that a hypothesis is true. Instead, it quantifies the strength of evidence against the null hypothesis.

Consider the court length data. Suppose we conduct a hypothesis test with

$$H_0 : \mu = 80 \quad \text{versus} \quad H_1 : \mu > 80,$$

and obtain a p-value of 0.0164.

If we choose a significance level of 0.05, then since $0.0164 < 0.05$,

we reject the null hypothesis. At the 5% significance level, there is strong statistical evidence that the population mean court length is **not equal to 80**.

When the p-value Is Large

If the p-value is greater than 0.05, we **do not reject** the null hypothesis. This does **not** mean that the null hypothesis is true. Rather, it means that the data do not provide strong enough evidence to conclude that the population mean court length differs from 80.

In hypothesis testing, failing to reject the null hypothesis should be interpreted as *insufficient evidence*, not as confirmation of the null hypothesis.

7 One Sample Nonparametric Test

Learning Objectives

1. Be familiar with the difference about parameter tests and nonparameteric tests
2. Applied the one sample non-parametricc test to do inference for one sample mean problem

7.1 Motivation

A **parametric test** specifies certain assumptions about the distribution of responses in the population from which the sample is drawn. The validity and interpretability of parametric test results depend critically on whether these assumptions are satisfied.

In contrast, a **nonparametric test** is based on a model that imposes only very general conditions and does **not** assume a specific parametric form for the population distribution. For this reason, nonparametric tests are often referred to as **distribution-free tests**.

Although nonparametric methods are not completely assumption-free, the assumptions they require are generally **fewer and weaker** than those of parametric tests.

7.1.1 Key Characteristics of Nonparametric Tests

Nonparametric test statistics typically rely on simple features of the data, such as:

- Signs of measurements
- Ranks or orderings
- Category or frequency counts

As a result:

- Linear transformations (stretching or compressing the scale) do **not** affect the test statistic.
- The null distribution of the test statistic can often be derived **without specifying the population distribution**.

Nonparametric tests therefore avoid assumptions such as:

- Normality
- Homogeneity of variance

Moreover, nonparametric methods usually compare **medians rather than means**, which makes them less sensitive to outliers.

Advantages of Nonparametric Tests

- Applicable to **all measurement scales**
- Particularly useful when the **sample size is very small**, unless the distribution is known
- Easier to learn and compute
- Require **fewer assumptions**
- More **robust** due to weaker modeling assumptions
- Do not require explicit population parameters
- In some cases, results can be **as exact** as parametric procedures

Disadvantages of Nonparametric Tests

- Often **less powerful** than parametric tests when parametric assumptions hold
- Provide less information about population parameters
- Interpretation is usually framed in terms of **location or rank**, not means
- Some procedures do not extend easily to complex models

Summary

- Parametric tests rely on strong distributional assumptions but can be powerful and informative.
- Nonparametric tests trade efficiency for **robustness and flexibility**.
- In practice, nonparametric methods serve as valuable alternatives when assumptions are questionable or sample sizes are limited.

In the next section, we will study **specific one-sample nonparametric procedures** and implement them in **SAS**.

7.2 One Sample Nonparametric Test in SAS

Base SAS provides two commonly used **one-sample nonparametric tests** through the `PROC UNIVARIATE` procedure:

- **Sign test**
- **Wilcoxon signed-rank test**

Both tests are designed for situations in which we want to make inference about the **location** of a population, typically interpreted as the **median** rather than the mean.

Suppose we are interested in testing whether the **median resting pulse rate of marathon runners** differs from a specified value. If the normality assumption required for a one-sample t-test is questionable, nonparametric alternatives provide a robust solution.

By default, both tests examine the hypothesis that the median of the population from which the sample is drawn is equal to a specific value,s which is zero by dafault. However, we note that

- **Wilcoxon signed-rank test**
  - Assumes the population distribution is **symmetric**
  - Generally more powerful when the symmetry assumption holds
- **Sign test**
  - Does **not** require symmetry
  - Uses only the **sign** of deviations from the hypothesized median
  - More robust but typically less powerful

Both tests can also be extended to **paired (related) samples**, which will be discussed later when we cover comparisons of two related samples.

Both the sign test and the Wilcoxon signed-rank test are **automatically available** in `PROC UNIVARIATE`.

```
/* Syntax: PROC UNIVARIATE */
PROC UNIVARIATE <options>;
    BY <variables>;
    CDFPLOT <variables> </ options>;
RUN;

CLASS variable-1 <(v-options)> <variable-2 <(v-options)>></ KEYLEVEL=value1 (value1 value2
)>;FREQ variable;HISTOGRAM <variables> </ options>;ID variables;INSET keyword-list </ options
<OUT=SAS-data-set> <keyword1=names ...keywordk=names> <percentile-options>;PPPLOT <variables>
</ options>;PROBPLOT <variables> </ options>;QQPLOT <variables> </ options>;VAR variables;WE
variable;
```

- The `PROC UNIVARIATE` statement invokes the **UNIVARIATE procedure**, which provides detailed descriptive statistics, distributional summaries, and diagnostic plots for numerical variables.
- The `VAR` Statement
  - Specifies the **numeric variables** to be analyzed.
  - **Required** if the `OUTPUT` statement is used.

  - If omitted, **all numeric variables** in the data set are analyzed.

- The PLOT statement (CDFPLOT, HISTOGRAM, PPPLOT, PROBPLOT, and QQPLOT) create graphical displays
- the INSET statement enhances these displays by adding a table of summary statistics directly on the graph.

You can specify one or more of each of the plot statements, the INSET statement, and the OUTPUT statement. If you use a VAR statement, the variables listed in a plot statement must be a subset of the variables listed in the VAR statement.

You can specify a `BY` statement to obtain separate analyses for each `BY` group. The `FREQ` statement specifies a variable whose values provide the frequency for each observation. The `ID` statement specifies one or more variables to identify the extreme observations. The `WEIGHT` statement specifies a variable whose values are used to weight certain statistics.

You can use a `CLASS` statement to specify one or two variables that group the data into classification levels. The analysis is carried out for each combination of levels in the input data set, or within each BY group if you also specify a BY statement. You can use the `CLASS` statement with plot statements to create comparative displays, in which each cell contains a plot for one combination of classification levels.

We revisit the **court length example** to demonstrate how one-sample **nonparametric tests** can be used as an alternative or complement to the one-sample t-test.

Suppose that, for some reason, we believe the t-test may not be an appropriate choice—perhaps due to concerns about normality—or we simply wish to **double-check our conclusions** using nonparametric methods. In this case, we can apply the nonparametric procedures available in `PROC UNIVARIATE`.

Step 1: Input the Data

We begin by entering the court length data into SAS.

```
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;
```

Step 2: Perform a One-Sample Nonparametric Test

To test whether the population median court length differs from 80 days, we use PROC UNIVARIATE with the MU0= option.

```
/* Perform one-sample nonparametric test */
PROC UNIVARIATE DATA=time MU0=80;
    VAR time;
RUN;
```

Step 3: Interpret the Output

From the output, SAS provides: + Test statistics + p-values for both nonparametric tests

These results allow us to assess whether there is statistical evidence that the population median court length differs from 80 days, without relying on the normality assumption required by the t-test.

7.3 Discussion: One-Sided vs Two-Sided Tests

Question:
Is this a *one-sided* or a *two-sided* test?

In this example, **SAS reports only two-sided p-values** by default for the nonparametric tests in `PROC UNIVARIATE`. If a **one-sided test** is desired, SAS does not directly provide the result.

However, a **simple (though not exact) workaround** can be used to approximate the one-sided p-value from the two-sided p-value.

7.3.1 Approximate One-Sided p-Value Calculation

1. Let

$$p^* = \frac{\text{two-sided p-value}}{2}.$$

2. Then proceed according to the alternative hypothesis:

7.3.1.1 Case 1: Right-sided test

Testing

$$H_1 : \mu > 80 \quad (\text{or } \mu > \mu_0)$$

- If the sample mean $\bar{x} > \mu_0$, then

$$\text{one-sided p-value} = p^*.$$

- If the sample mean $\bar{x} < \mu_0$, then

$$\text{one-sided p-value} = 1 - p^*.$$

7.3.1.2 Case 2: Left-sided test

Testing

$$H_1 : \mu < 80 \quad (\text{or } \mu < \mu_0)$$

- If the sample mean $\bar{x} < \mu_0$, then

$$\text{one-sided p-value} = p^*.$$

- If the sample mean $\bar{x} > \mu_0$, then

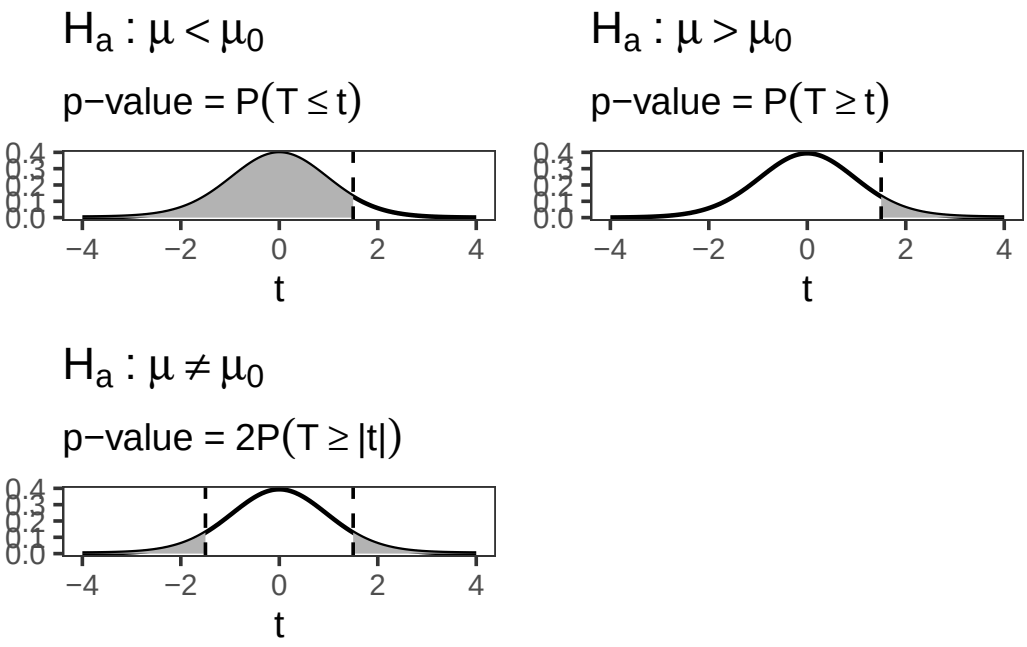
$$\text{one-sided p-value} = 1 - p^*.$$

7.3.2 Why Does This Work?

A two-sided p-value measures evidence in **both directions** away from the null hypothesis. Dividing it by two isolates the probability mass in **one tail** of the sampling distribution.

However, this adjustment is only valid when: - The test statistic is symmetric under the null hypothesis - The observed statistic is in the direction specified by the alternative

Because these conditions are not always guaranteed for nonparametric tests, this method should be viewed as an **approximation**, not an exact one-sided test.



Key takeaway:
SAS nonparametric procedures default to two-sided inference. While approximate one-sided p-values can be obtained manually, interpretation should be done with care.

8 One Sample Proportion Test

Learning Objectives

1. Apply the binomial test to conduct statistical inference for a population proportion.

8.1 Motivation

Suppose we observe a **categorical variable with two levels**, such as

- `SMOKES = 1`: smoker
- `SMOKES = 2`: non-smoker

In this setting, we are often interested in making inference about the **population proportion** of interest, such as the proportion of people who smoke.

We denote this population proportion by p.

Common inferential goals include:

- Estimating the population proportion p
- Constructing a confidence interval for p
- Performing a hypothesis test for p

Specifically, we may test

$$H_0 : p = p_0,$$

$$H_1 : \begin{cases} p \neq p_0, & \text{(two-sided)}, \\ p < p_0, & \text{(left-sided)}, \\ p > p_0, & \text{(right-sided)}. \end{cases}$$

8.1.1 Sampling Distribution of the Sample Proportion

Let $\hat{p}$ denote the **sample proportion**, computed as

$$\hat{p} = \frac{X}{n},$$

where

- X is the number of successes,
- n is the sample size.

When the sample size is sufficiently large (typically $n \geq 30$), the sampling distribution of $\hat{p}$ is approximately normal:

$$\hat{p} \sim N\left(p, \frac{p(1-p)}{n}\right).$$

This normal approximation forms the basis for both **confidence intervals** and **hypothesis testing** for a population proportion.

8.1.2 Confidence Interval for a Proportion

A $(1-\alpha) \times 100\%$ confidence interval for p is given by

$$\left[\hat{p} - Z_{\alpha/2}\sqrt{\frac{\hat{p}(1-\hat{p})}{n}}, \quad \hat{p} + Z_{\alpha/2}\sqrt{\frac{\hat{p}(1-\hat{p})}{n}}\right].$$

Here, $Z_{\alpha/2}$ is the standard normal quantile satisfying

$$P(Z \geq Z_{\alpha/2}) = \alpha/2, \quad Z \sim N(0,1).$$

Moreover, this normal approximation of $\hat{p}$ is also used to perform hypothesis testing for p.

8.1.3 Standard Error of the Sample Proportion

The **standard error** of $\hat{p}$ is

$$\text{SD}(\hat{p}) = \sqrt{\frac{\hat{p}(1-\hat{p})}{n}}.$$

This expression highlights an important principle:

> The uncertainty in an estimate decreases as the sample size increases.

8.1.4 Interpretation: Effect of Sample Size

Consider two examples where the estimated proportion is the same.

A small college has 8 physics majors, and 5 of them graduate within four years.

$$\hat{p} = \frac{5}{8} = 0.625$$

The standard error is

$$\text{SE}(\hat{p}) = \sqrt{\frac{0.6 \times 0.4}{8}} \approx 0.17.$$

An English department graduates 50 out of 80 students.

$$\hat{p} = \frac{50}{80} = 0.625$$

The standard error is

$$\text{SE}(\hat{p}) = \sqrt{\frac{0.6 \times 0.4}{80}} \approx 0.05.$$

Does this make sense?

> **Note**
>
> In practice, when the normal approximation may not be reliable (e.g., small n), we will instead rely on **exact binomial methods**, which do not require large-sample assumptions.

8.2 Computation

Suppose a college has six majors, labelled A, B, C, D, E, and F. For each major, we observe:

```
+ the number of students who graduated within four years (Grads), and
+ the total number of students in that major (Total).
```

```
DATA Grads;
  INPUT Major $ Grads Total @@;
  DATALINES;
A 10 22  B 10 32  C 17 25
D  4  7  E  8 14  F 16 28
;
RUN;
```

Obs	Major	Grads	Total
1	A	10	22
2	B	10	32
3	C	17	25
4	D	4	7
5	E	8	14
6	F	16	28

It is easy to write a short DATA step to compute the empirical proportions and a 95% confidence interval for each major.

```
DATA GradRate;
  SET Grads;

  /* Empirical proportion */
  p = Grads / Total;
```

```
  /* Standard error under normal approximation */
  StdErr = SQRT(p * (1 - p) / Total);

  /* 95% Wald confidence interval */
  z = QUANTILE("normal", 1 - 0.05/2);

  LCL = MAX(0, p - z * StdErr);   /* Lower bound */
  UCL = MIN(1, p + z * StdErr);   /* Upper bound */

  LABEL p   = "Proportion"
        LCL = "Lower 95% CI"
        UCL = "Upper 95% CI";
RUN;
```

Major	Grads	Total	Proportion	Lower 95% CL	Upper 95% CL
A	10	22	0.45455	0.24648	0.66261
B	10	32	0.31250	0.15190	0.47310
C	17	25	0.68000	0.49714	0.86286
D	4	7	0.57143	0.20483	0.93803
E	8	14	0.57143	0.31220	0.83065
F	16	28	0.57143	0.38813	0.75473

The output shows that although majors D, E, and F have the same four-year graduation rate (57%), the estimate for the D group, which has only seven students, has twice as much variability as the estimate for the F group, which has four times as many students.

8.3 Automating the Computations with PROC FREQ

In the previous section, we computed the Wald confidence interval for each major using a DATA step. That approach is straightforward, but it becomes inconvenient if we want other binomial confidence intervals (e.g., exact/Clopper–Pearson, Wilson, etc.). A convenient alternative is to use `PROC FREQ` with the `BINOMIAL` option. However, PROC FREQ expects the data in an

event / nonevent format (a binary outcome with a frequency count), rather than the events / trials format (Grads, Total) we currently have.

So we first convert each major into two rows: + Graduated="Yes" with Count = Grads + Graduated="No" with Count = Total - Grads

Step 1: Convert events/trials into event/nonevent format

```
/*-----------------------------------------------------------------
  Convert Event/Trials format (Grads, Total)
  into Event/Nonevent format (Graduated, Count)
-----------------------------------------------------------------*/
data GradFreq;
  set Grads;

  Graduated = "Yes";
  Count = Grads;
  output;

  Graduated = "No";
  Count = Total - Grads;
  output;
run;

proc print data=GradFreq;
run;
```

Obs	Major	Grads	Total	Graduated	Count
1	A	10	22	Yes	10
2	A	10	22	No	12
3	B	10	32	Yes	10
4	B	10	32	No	22
5	C	17	25	Yes	17
6	C	17	25	No	8
7	D	4	7	Yes	4
8	D	4	7	No	3
9	E	8	14	Yes	8
10	E	8	14	No	6
11	F	16	28	Yes	16
12	F	16	28	No	12

Step 2: Run PROC FREQ to compute binomial CIs by major

```
/*-------------------------------------------------------------
  Use PROC FREQ to analyze each major separately and compute
  binomial confidence intervals for Pr(Graduated="Yes")
-------------------------------------------------------------*/
proc freq data=GradFreq noprint;
  by notsorted Major;

  tables Graduated / binomial(level="Yes" CL=wald);
  weight Count;

  output out=FreqOut binomial;
run;
```

Step 3: Print a clean summary table

```
proc print data=FreqOut noobs label;
  var Major N _BIN_ L_BIN U_BIN;

  label _BIN_ = "Proportion"
        L_BIN = "Lower 95% CI"
        U_BIN = "Upper 95% CI";
run;
```

Major	Number of Subjects	Proportion	Lower 95% CI	Upper 95% CI
A	22	0.45455	0.24648	0.66261
B	32	0.31250	0.15190	0.47310
C	25	0.68000	0.49714	0.86286
D	7	0.57143	0.20483	0.93803
E	14	0.57143	0.31220	0.83065
F	28	0.57143	0.38813	0.75473

8.4 Visualization of Binomial Proportion

It is often helpful to visualize estimated proportions together with their confidence intervals. When plotting several proportions on the same graph, it is a good idea to **sort the groups** in a meaningful way—most commonly by the estimated proportion itself. If two groups have the same estimated proportion, the sample size can be used as a tie-breaker.

8.4.1 Why visualization matters

A single number rarely tells the whole story. Plotting proportions with confidence intervals allows us to:

- Compare graduation rates across majors at a glance
- Assess uncertainty in each estimate
- See how sample size affects precision

Groups with **smaller sample sizes** tend to have **wider confidence intervals**, reflecting greater uncertainty.

8.4.2 Adding a reference line

It is often informative to add a **reference line** representing the **overall proportion**, regardless of group membership.

For the graduation data in this example, the overall proportion of students who graduate in four years is

$$\hat{p}_{\text{overall}} = \frac{65}{128} \approx 0.5078.$$

This reference line helps contextualize each major's graduation rate relative to the overall average.

> **Tip:**
> You can obtain the overall proportion in SAS by repeating the `PROC FREQ` analysis **without** a `BY` statement.

8.4.3 Including sample size on the plot

Because uncertainty depends strongly on sample size, it is good practice to display the **number of observations per group** directly on the graph.

In SAS, this can be done using the `YAXISTABLE` statement, which adds a table aligned with the y-axis showing the sample size for each group.

8.4.4 Interpreting the plot

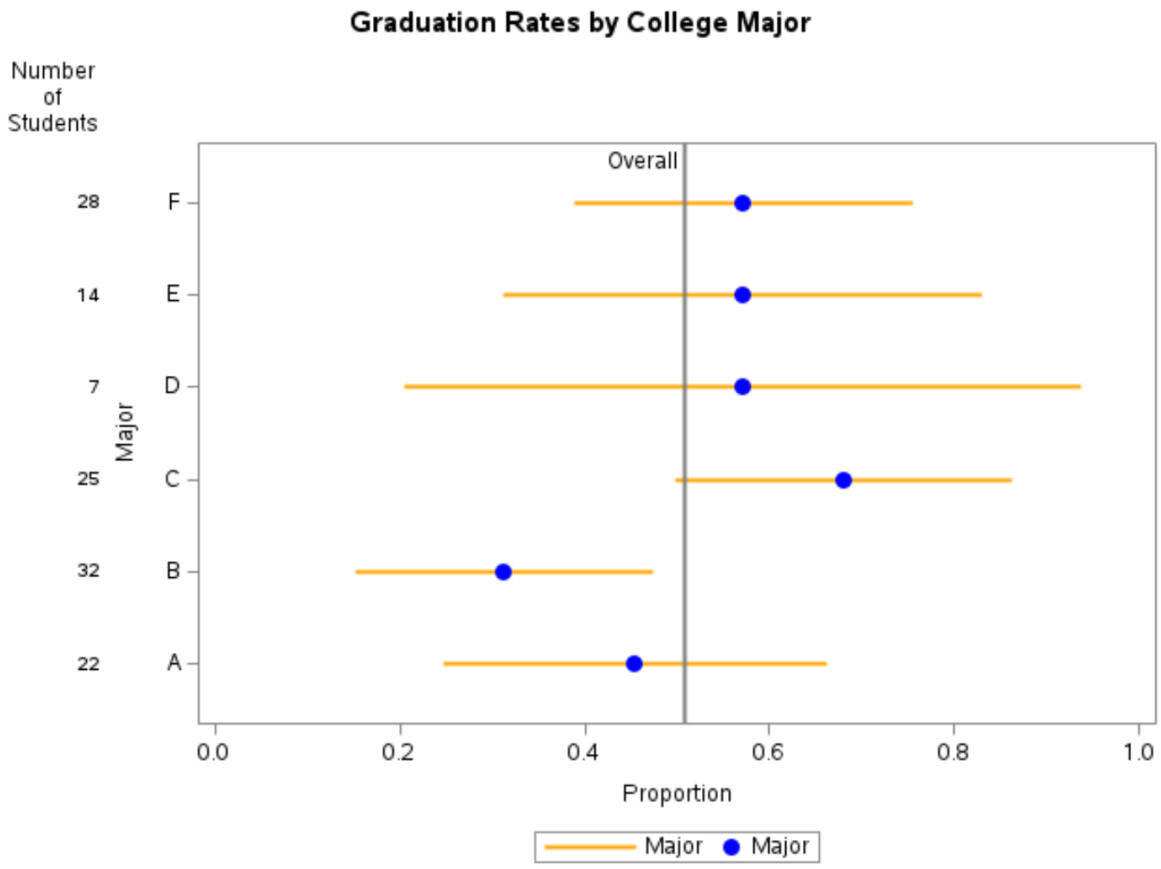
A typical plot of binomial proportions with confidence intervals shows:

- **Points**: estimated proportions
- **Error bars**: 95% confidence intervals
- **Vertical reference line**: overall proportion
- **Table entries**: number of students in each major

Such a graph clearly illustrates that majors with fewer students (for example, very small departments) have much wider confidence intervals than majors with larger enrollments.

8.4.5 Practical visualization advice

- If there are **10 or more categories**, consider using **alternating background color bands** to make it easier to associate confidence intervals with groups.
- Always label axes clearly (e.g., *Proportion* on the x-axis).
- Include the confidence level (e.g., "95% CI") in the caption or title.



8.4.6 Summary

This section demonstrates how visualization complements numerical output:

- `PROC FREQ` provides estimated proportions and confidence intervals.
- Graphing proportions with confidence intervals makes comparisons intuitive.
- Including sample sizes helps readers understand why some intervals are wider than others.

Well-designed graphics are one of the most effective ways to communicate results from binomial proportion analyses.

9 SAS Macro Program in One Sample Variance Problem

Learning Objectives

1. Apply hypothesis testing and confidence interval construction for **one population variance**.
2. Use this example to motivate **writing reusable SAS macro programs**.

9.1 Motivation

In many applications, we may care more about the **population variance** σ^2 than the population mean μ. For instance, in a manufacturing process, a **large variance** in product measurements indicates unstable quality, even if the mean is acceptable. This is not the only instance, as there are many interesting scientific questions that involve the population variance both in academia and in industry.

9.1.1 Examples

A SCUBA instructor records the dive depths of students during checkout. Although everyone should be at the same depth, the instructor is interested in how much the depths vary.

The instructor believes the standard deviation is **3 feet**, while the assistant thinks it is **less than 3 feet**.

The hypotheses are:

$$H_0 : \sigma^2 = 3^2 \qquad \text{vs} \qquad H_1 : \sigma^2 < 3^2$$

A company produces metal pipes of a standard length. Twenty years ago, the pipe lengths were normally distributed with standard deviation **1.1 cm**.

The company now tests a random sample of **30 pipes** and wants to construct a **95% confidence interval** for σ^2 to determine whether the production quality has changed.

For the examples above, the quantity of the interest is the population variance. But we need something to estiamte it. Let s^2 denote the sample variance. Then the pivotal quantity

$$\frac{(n-1)s^2}{\sigma^2},$$

which follows a chi-square distribution:

$$\frac{(n-1)s^2}{\sigma^2} \sim \chi^2_{n-1},$$

where

- n : sample size
- s^2 : sample variance
- σ^2 : population variance

You may think that s as the randomvariable in this test. The distribution is called chi-square distribution which depends on a parameter named the *degrees of freedom* denoted by $\text{df} = n-1$. A test of a single variance may be right-tailed, left-tailed or two-sided. This chi-square result is the foundation for both **confidence intervals** and **hypothesis tests** for σ^2.

> **Note**
>
> A $(1-\alpha) \times 100\%$ confidence interval for the population variance is
>
> $$\left(\frac{(n-1)s^2}{\chi^2_{1-\alpha/2,\, n-1}}, \frac{(n-1)s^2}{\chi^2_{\alpha/2,\, n-1}} \right),$$
>
> where $\chi^2_{p,\, n-1}$ denotes the p-th quantile of the χ^2_{n-1} distribution.
> I won't expect you to know the expression here for now.

Problem

Unfortunately, if you check the built-in SAS procedures, you will find that **there is no basic procedure** that directly handles the **one-sample variance inference problem**.

The good news is that SAS provides a powerful **MACRO programming language**, which allows us to **write our own SAS procedures**.

9.2 SAS Macro Program

When you write a program that will be run **repeatedly**, you may want to seriously consider using **macros**, because:

- **MARCROS allow centralized changes**:
  You can make a change in one location, and SAS will cascade that change throughout your program.
- **MACROS promote code reuse**:
  You can write a section of code once and reuse it many times, either within the same program or across different programs.
- **MARCROS enable data-driven programming**:
  SAS can decide what actions to take based on actual data values.

In general, macro code takes **longer to write and debug** than standard SAS code. Therefore, macros are usually **not recommended** for programs that will only be run a few times.

However, if you find yourself writing **similar code repeatedly**, macros can significantly improve efficiency and reduce errors.

Macros can help in several ways:

1. You can make a **single small change**, and SAS will propagate that change throughout your program.
2. You can **reuse code blocks**, avoiding duplication.
3. You can build **data-driven programs**, allowing SAS to adapt automatically to different datasets or inputs.

9.3 How Macros Work

When you submit a standard SAS program, SAS **compiles and executes it immediately**. When you write macro code, there is an **additional step**:

- SAS first sends MARCO statements to the **macro processor**.
- The macro processor **resolves the macro code**, generating standard SAS code.
- SAS then compiles and executes the generated code.

Because you are writing a program that **writes another program**, this process is sometimes referred to as **meta-programming**.

9.4 Designing Your Own Macros

To design effective SAS macros, we need to understand two fundamental concepts:

9.4.1 Macros vs. Macro Variables

As you construct macro programs, you will work with two basic building blocks:

- **MACROS** v.s.
- **MACRO variables**

To design your own SAS macros, we need to clearly distinguish between **macros** and **macro variables**. These are the two basic building blocks of SAS macro programming.

Conceptually, the workflow looks like this:

- You write **macro code**
- The **macro processor** resolves it
- The output is **ordinary SAS code**, which is then compiled and executed

Naming Conventions

You can tell macros and macro variables apart by their prefixes:

- **Macro variables** start with an ampersand: `&`
- **Macros** start with a percent sign: `%`

Examples:

- `&alpha`, `&n`
- `%myMacro`, `%DO`, `%IF`

Macro Variables

A **macro variable** is similar to a data variable, but with important differences:

- It does **not belong to a dataset**
- It has **only one value**
- That value is **always character**
- The value is **substituted directly into your program**

A macro variable's value can be:

- A variable name
- A number
- Text

- Any string you want substituted into your SAS code

Macro

A **macro** is a larger unit of code that can contain:

- DATA steps
- PROC steps
- Macro logic such as
  `%IF-%THEN-%ELSE`, `%DO-%END`

Macros often—but not always—use macro variables.

9.5 Think Globally and Locally: Scope of Macro Variables

Macro variables come in two scopes:

- **Local macro variables**
- **Global macro variables**

9.5.1 Local Macro Variables

- Defined **inside a macro**
- Exist **only within that macro**
- Cannot be referenced outside the macro

9.5.2 Global Macro Variables

- Defined in **open code** (outside any macro)
- Can be used **anywhere in the program**

9.5.3 Common Mistakes to Avoid

- Using a local macro variable outside its macro
- Accidentally creating **both local and global macro variables with the same name**

Keeping scope in mind will save you a lot of debugging time.

> **Note**
>
> The macro processor **does not resolve macro references inside single quotes**.
>
> - `'&var'` → macro not resolved
> - `"&var"` → macro resolved
>
> **Rule of thumb:**
>
> Use **double quotes** when quoted strings contain macro variables.

Substituting Text with %LET

The `%LET` statement assigns a value to a macro variable.

```
%LET alpha = 0.05;

%MACRO macro_name(param1, param2, ..., paramk);
   /* macro code */
%MEND macro_name;
```

9.6 Writing SAS Macros with Car dataset

Before writing our own macros, let us first understand how **macro variables** work through a simple example.

SAS provides a built-in **global macro variable** called `SYSDATE`, which represents the system date.

Suppose we want to print the system date automatically in the title of a SAS report every time the report is generated. The advantage is that we do **not** need to manually update the date in the code. We use the built-in SAS dataset `CARS` from the `SASHELP` library.

```
PROC PRINT DATA=SASHELP.CARS;
  WHERE MAKE = 'Audi' AND TYPE = 'Sports';
  TITLE "Sales as of &SYSDAY &SYSDATE";
RUN;
```

Sales as of Saturday 14FEB26

Obs	Make	Model	Type	Origin	DriveTrain	MSRP	Invoice	EngineSize	Cylinders	Horsepower	MPG_City	MPG_Highway	Weight	Wheelbase	Length
21	Audi	RS 6 4dr	Sports	Europe	Front	$84,600	$76,417	4.2	8	450	15	22	4024	109	191
22	Audi	TT 1.8 convertible 2dr (coupe)	Sports	Europe	Front	$35,940	$32,512	1.8	4	180	20	28	3131	95	159
23	Audi	TT 1.8 Quattro 2dr (convertible)	Sports	Europe	All	$37,390	$33,891	1.8	4	225	20	28	2921	96	159
24	Audi	TT 3.2 coupe 2dr (convertible)	Sports	Europe	All	$40,590	$36,739	3.2	6	250	21	29	3351	96	159

So far, we have used macro variables to make our SAS programs more flexible.
Now we take one more step forward and define a **macro program**, which is a reusable block of SAS code. Macro variables are referenced in SAS statements using the ampersand (&) character.

Now suppose we want our program to be more flexible. Instead of hard-coding the car MAKE and TYPE, we define macro variables so that we can easily change them without modifying multiple lines of code.

```
%LET MAKE_NAME = Audi;
%LET TYPE_NAME = Sports;

PROC PRINT DATA=SASHELP.CARS;
  WHERE MAKE = "&MAKE_NAME" AND TYPE = "&TYPE_NAME";
  TITLE "Sales as of &SYSDAY &SYSDATE";
RUN;
```

Sales as of Saturday 14FEB26

Obs	Make	Model	Type	Origin	DriveTrain	MSRP	Invoice	EngineSize	Cylinders	Horsepower	MPG_City	MPG_Highway	Weight	Wheelbase	Length
21	Audi	RS 6 4dr	Sports	Europe	Front	$84,600	$76,417	4.2	8	450	15	22	4024	109	191
22	Audi	TT 1.8 convertible 2dr (coupe)	Sports	Europe	Front	$35,940	$32,512	1.8	4	180	20	28	3131	95	159
23	Audi	TT 1.8 Quattro 2dr (convertible)	Sports	Europe	All	$37,390	$33,891	1.8	4	225	20	28	2921	96	159
24	Audi	TT 3.2 coupe 2dr (convertible)	Sports	Europe	All	$40,590	$36,739	3.2	6	250	21	29	3351	96	159

Reusing the Same Program with Different Values

Now, suppose we want to view Audi vehicles of type Wagon. We only need to change one line of code.

```
/* Change only the macro variable values */
%LET MAKE_NAME = 'Audi';
%LET TYPE_NAME = 'Wagon';

/* The SAS code below remains unchanged */
PROC PRINT DATA = SASHELP.CARS;
    WHERE MAKE = &MAKE_NAME AND TYPE = &TYPE_NAME;
    TITLE "Sales as of &SYSDAY &SYSDATE";
RUN;
```

Sales as of Saturday 14FEB26

Obs	Make	Model	Type	Origin	DriveTrain	MSRP	Invoice	EngineSize	Cylinders	Horsepower	MPG_City	MPG_Highway	Weight	Wheelbase	Length
25	Audi	A6 3.0 Avant Quattro	Wagon	Europe	All	$40,840	$37,060	3.0	6	220	18	25	4035	109	192
26	Audi	S4 Avant Quattro	Wagon	Europe	All	$49,090	$44,446	4.2	8	340	15	21	3936	104	179

More on Macro program

The following program defines a macro called `show_result`.
This macro takes two arguments:

- `make_`: the car manufacturer
- `type_`: the vehicle type

```
%MACRO show_result(make_, type_);

PROC PRINT DATA=SASHELP.CARS;
  WHERE MAKE = "&make_" AND TYPE = "&type_";
  TITLE "Sales as of &SYSDAY &SYSDATE";
RUN;

%MEND;
```

Calling a Macro Once a macro is defined, it can be called just like a function.

```
%show_result(BMW, SUV);
```

Sales as of Saturday 14FEB26

Obs	Make	Model	Type	Origin	DriveTrain	MSRP	Invoice	EngineSize	Cylinders	Horsepower	MPG_City	MPG_Highway	Weight	Wheelbase	Length
27	BMW	X3 3.0i	SUV	Europe	All	$37,000	$33,873	3.0	6	225	16	23	4023	110	180
28	BMW	X5 4.4i	SUV	Europe	All	$52,195	$47,720	4.4	8	325	16	22	4824	111	184

So after seeing the first example, we have a more clear sense about why Use Macro Programs?

Macro programs allow you to:

- Avoid rewriting the same code repeatedly
- Centralize logic in one place
- Make programs easier to maintain and extend
- Write data-driven and parameterized SAS programs

This idea becomes especially powerful when we need to repeat the same analysis for many datasets or parameter values.

9.7 Revisit the One-Sample Variance Test

As discussed earlier, SAS does NOT provide a built-in procedure for conducting one-sample variance inference (confidence intervals or hypothesis tests). However, SAS allows us to overcome this limitation by writing our own macro procedures. Fortunately, a one-sample variance inference macro is commonly used and can be written in a general form. In this course, we will use a macro named `vartest.sas`, which is shared in the iCollege and below.

```
%macro vartest(version,
                 data= _last_  ,
                 var=         ,
                 alpha=  0.05   ,
                 var0=,
                 sigma0=
              );

%if &version ne %then %put VARTEST macro Version 1.1;
%let opts = %sysfunc(getoption(notes))
            _last_=%sysfunc(getoption(_last_));
%if &data=_last_ %then %let data=&syslast;
options nonotes;

%if &sigma0=0 or &var0=0 %then %do;
  %put ERROR: The null hypothesis variance (VAR0=) or standard deviation;
  %put %str(      ) (SIGMA0=) must be greater than zero.;
  %goto exit;
%end;

proc summary data=&data;
  var &var;
  output out=_varn(keep=var n std) n=n var=var std=std;
run;

data _vartest;
  set _varn;
  df=n-1;
  %if &sigma0 ne %then %str(nullvar=&sigma0 * &sigma0;);
  %if &var0 ne %then %str(nullvar=&var0;);
  level=(1-&alpha)*100;
  call symput('level',trim(left(level)));
  %if &sigma0 ne or &var0 ne %then %do;
    chisq=((df)*(var))/nullvar;
    pvalue=1-probchi(chisq,df);
  %end;
  uclvar=df*var/cinv(&alpha/2,df); uclstd=sqrt(uclvar);
  lclvar=df*var/cinv(1-&alpha/2,df); lclstd=sqrt(lclvar);
  label var="Sample Variance" std="Sample Std Dev"
        chisq="Chi-Square" pvalue="Prob>Chi-Square"
        level="Confidence Level (%)"
        uclvar="UCL for Variance" lclvar="LCL for Variance"
```

```
        uclstd="UCL for Std Dev" lclstd="LCL for Std Dev";
  drop df  %if &sigma0 ne or &var0 ne %then nullvar;  ;
run;

data _ci;
  set _vartest;
  stat="Variance"; estimate=var; lcl=lclvar; ucl=uclvar; output;
  stat="Std Dev";  estimate=std; lcl=lclstd; ucl=uclstd; output;
  label stat="Statistic"
        estimate="Estimate"
        lcl="Lower Confidence Limit"
        ucl="Upper Confidence Limit";
  run;

proc print data=_ci label noobs;
  var stat estimate lcl ucl;
  title "&level.% Confidence Intervals for Variance and Std Dev";
  run;

%if &sigma0 ne or &var0 ne %then %do;
proc print data=_vartest label noobs;
  var chisq pvalue;
  format pvalue 6.4;
  title  "Hypothesis test for";
  title2;
 %if &var0 ne %then %do;
  title3 "H0: Variance(%upcase(&var)) = &var0";
  title4 "Ha: Variance(%upcase(&var)) > &var0";
 %end;
 %if &sigma0 ne %then %do;
  title3 "H0: StdDev(%upcase(&var)) = &sigma0";
  title4 "Ha: StdDev(%upcase(&var)) > &sigma0";
 %end;
run;
%end;

%exit:
title;
options &opts;
%mend;
```

With this Marcro, we can implement it to construct the hypothesis testing or confidence interval

for population variance σ^2.

```
/* Create a sample data set */
DATA DAT;
    INPUT X @@;
    DATALINES;
6.2 1.9 4.4 4.9 3.5
4.6 4.2 1.1 1.3 4.8
4.1 3.7 2.5 3.7 4.2
1.4 3.2 2.6 1.5 3.9
;
RUN;

/* Display the data */
PROC PRINT DATA = DAT;
RUN;
```

This dataset includes **20 measurements** of the length of a product from a manufacturing process. Again, our goal is to conduct **statistical inference on the population variance** of the product length. Specifically, we want to test whether the population variance is **greater than 2.25**.

```
DATA DAT;
    INPUT X @@;
    DATALINES;
6.2 1.9 4.4 4.9 3.5
4.6 4.2 1.1 1.3 4.8
4.1 3.7 2.5 3.7 4.2
1.4 3.2 2.6 1.5 3.9
;
RUN;

PROC PRINT DATA = DAT;
RUN;
```

9.7.1 Using the `VARTST` Macro in SAS

Once the dataset has been created in SAS and the macro has been saved in a file with a known path, we can use the following code to conduct the **one-sample variance test** and construct the corresponding **confidence intervals**.

Use the `%INCLUDE` statement to load the macro into your SAS session:

```
/* When using SAS Windows */
%INCLUDE
"C:/Users/cyeh/Desktop/Teaching/GSU/STAT8678-SAS/code/L10/vartest.sas";

/* in SAS studio */
%INCLUDE
"/home/u64378616/chikuang/Part2/L10_Marco/vartest.sas";

%VARTST(
    DATA  = DAT,
    VAR   = X,
    ALPHA = 0.05,
    VAR0  = 2.25
);
```

95% Confidence Intervals for Variance and Std Dev

Statistic	Estimate	Lower Confidence Limit	Upper Confidence Limit
Variance	1.98661	1.14894	4.23796
Std Dev	1.40947	1.07189	2.05863

Hypothesis test for

H0: Variance(X) = 2.25
Ha: Variance(X) > 2.25

Chi-Square	Prob>Chi-Square
16.7758	0.6051

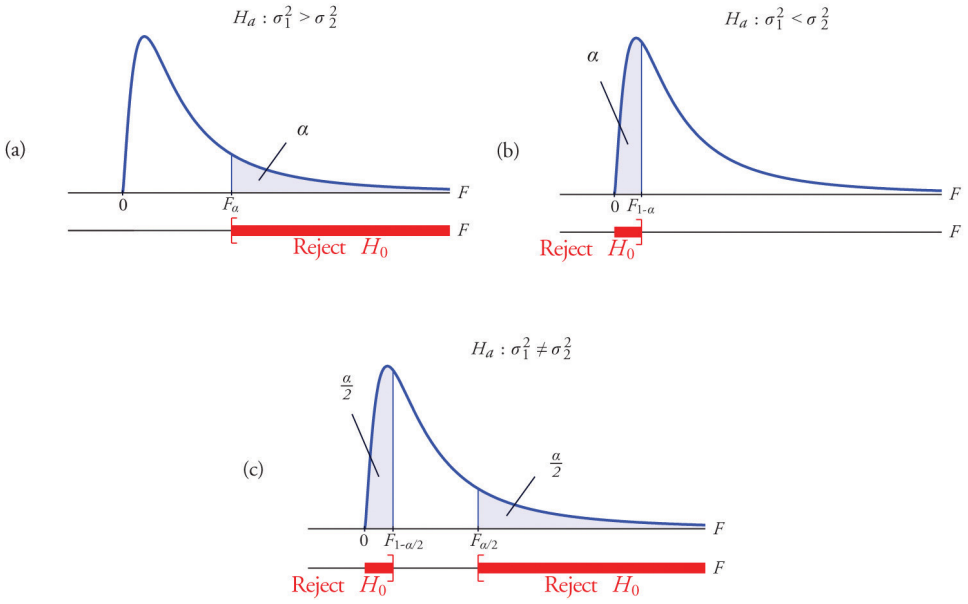
Question for you:

> How would you modify the macro call if you want to test whether the population standard deviation is greater than 1.5 instead of testing the population variance?

Answer with Explanation

To calculate the p-value for $H_1 : \sigma^2 < \sigma_0^2$ and $H_1 : \sigma^2 \neq \sigma_0^2$ from the p-value for $H_1 : \sigma^2 > \sigma_0^2$. We can do the following:
Let p be the p-value for the right-tailed test $H_1 : \sigma^2 > \sigma_0^2$.

- For the left-tailed test $H_1 : \sigma^2 < \sigma_0^2$, the p-value is $1 - p$.
- For the two-sided test $H_1 : \sigma^2 \neq \sigma_0^2$, the p-value is $\min\{p, 1-p\} \times 2$.



10 χ^2 Goodness of Fit Test

Learning Objectives

1. Explain what is a diagnostic test.
2. Explain what is a one-way contingency table.
3. Apply the diagnostic test to perform inference for analyzing a one-way contingency table.

10.1 Motivation: What Is a Goodness-of-Fit Test?

A **goodness-of-fit test**, in general, refers to measuring how well the observed data correspond to a fitted (assumed) model.

We will use this concept throughout the course as a way of checking model fit.
Similar to linear regression, in essence, a goodness-of-fit test compares:

- **Observed values**, and
- **Expected (fitted or predicted) values** under a specified model.

Hypotheses for a Goodness-of-Fit Test

A goodness-of-fit test examines the following hypotheses:

$$H_0 : \text{the model } M_0 \text{ fits}$$

versus

$$H_1 : \text{the model } M_0 \text{ does not fit.}$$

10.2 Example: Categorical Data (Dice Example)

Consider the simplest example of a goodness-of-fit test with **categorical data**.

Suppose we want to test whether a fair six-sided die has **equal probability** for each face. We compare the observed frequencies to those expected under the assumed model:

$$X \sim \text{Multinomial}\left(n = 30,\ \pi_0 = \left(\tfrac{1}{6}, \tfrac{1}{6}, \tfrac{1}{6}, \tfrac{1}{6}, \tfrac{1}{6}, \tfrac{1}{6}\right)\right).$$

In this setting, the goodness-of-fit test asks whether the probability in each cell is equal to the specified value π_0.

10.3 Hypotheses in Probability Vector Form

The hypotheses can be written as:

$$H_0 : \pi = \pi_0$$

versus

$$H_1 : \pi \neq \pi_0.$$

10.4 One-Way Contingency Table

In such problems, the data can be summarized using a **one-way contingency table**:

Categories	1	2	3	4	5	6
Number of Observations	O_1	O_2	O_3	O_4	O_5	O_6

Here,

- $O_1, O_2, \ldots, O_6$ are **observed counts** from your dataset.
- The expected counts are computed from the assumed probability model.

These observed and expected counts form the basis of the χ^2 goodness-of-fit test.

10.5 One-Way Contingency Table and Model Fit

Categories	1	2	3	4	5	6
Number of Observations	O_1	O_2	O_3	O_4	O_5	O_6
Expected Observations	$n\pi_1 = 5$	5	5	5	5	5

In the setting of one-way contingency tables, we measure how well an observed variable X corresponds to a model specified by a vector of cell probabilities.

In other words, under the null hypothesis, we assume the data come from a particular distribution, and we test whether this assumed model fits the data well when compared with the **saturated model** (the model that fits the data perfectly).

The rationale behind model fitting is the assumption that a complex data-generating mechanism can be reasonably approximated by a simpler model. The goodness-of-fit test is applied to assess whether this assumption is supported by the data.

10.6 Test Statistics

There are two commonly used test statistics for the χ^2 goodness-of-fit test.

Pearson Goodness-of-Fit Test Statistic

The Pearson goodness-of-fit statistic is defined as

$$X^2 = \sum_{j=1}^{k} \frac{(X_j - n\pi_{0j})^2}{n\pi_{0j}}.$$

An easy way to remember this formula is

$$X^2 = \sum_{j=1}^{k} \frac{(O_j - E_j)^2}{E_j},$$

where

- $O_j = X_j$ is the observed count in cell j, and
- $E_j = \mathbb{E}(X_j) = n\pi_{0j}$ is the expected count in cell j under the null hypothesis.

10.6.1 Deviance Test Statistic

The deviance statistic is defined as

$$G^2 = 2\sum_{j=1}^{k} X_j \log\left(\frac{X_j}{n\pi_{0j}}\right) = 2\sum_{j=1}^{k} O_j \log\left(\frac{O_j}{E_j}\right).$$

In some texts, G^2 is also referred to as the **likelihood ratio test (LRT) statistic**, which compares the log-likelihoods of two models:

- L_0: the reduced model under H_0, and
- L_1: the full (saturated) model under H_A.

$$G^2 = -2\log\left(\frac{\ell_0}{\ell_1}\right) = -2\left(L_0 - L_1\right)$$

Note that X^2 and G^2 are both functions of the observed data X and a vector of cell probabilities π_0. For this reason, we will sometimes write them as $X^2(x, \pi_0)$ and $G^2(x, \pi_0)$, respectively. When there is no ambiguity, we will simply use X^2 and G^2.

We will encounter these statistics repeatedly throughout the course, particularly in the analysis of two-way and k-way contingency tables, and when assessing the fit of log-linear and logistic regression models.

10.7 How Do They Work?

Both X^2 and G^2 measure how closely the assumed model—here, a multinomial model $\text{Mult}(n, \pi_0)$—fits the observed data.

When the null hypothesis H_0 is true, both statistics have an approximate chi-square distribution with $k - 1$ degrees of freedom. This allows us to use the chi-square distribution to obtain critical values and p-values for hypothesis testing.

- If the sample proportions $\hat{\pi}_j$ (i.e., the saturated model) are exactly equal to the model probabilities π_{0j} for all cells $j = 1, 2, \ldots, k$, then $O_j = E_j$ for all j, and both X^2 and G^2 are equal to zero. In this case, the model fits the data perfectly.
- If the sample proportions $\hat{\pi}_j$ deviate from the model probabilities π_{0j}, then both X^2 and G^2 are positive. Large values of X^2 and G^2 indicate that the data do not agree well with the assumed model M_0.

10.8 Dice Rolls Example

Suppose that we roll a die 30 times and observe the following table showing the number of times each face appears.

Categories	1	2	3	4	5	6
Number of Observations	$O_1 = 3$	$O_2 = 7$	$O_3 = 5$	$O_4 = 10$	$O_5 = 2$	$O_6 = 3$
Expected Observations						

We want to test the null hypothesis that the die is fair.

Under this hypothesis,

$$X \sim \text{Mult}(n = 30, \pi_0),$$

where

$$\pi_{0j} = \frac{1}{6}, \quad j = 1, \ldots, 6.$$

This is our assumed model. Under H_0, the expected counts are

$$E_j = \frac{30}{6} = 5 \quad \text{for each cell.}$$

We now have all the quantities needed to compute the goodness-of-fit statistics.

We now compute the two goodness-of-fit statistics explicitly.

10.8.1 Pearson chi-square statistic

$$\begin{aligned} X^2 &= \frac{(3-5)^2}{5} + \frac{(7-5)^2}{5} + \frac{(5-5)^2}{5} + \frac{(10-5)^2}{5} + \frac{(2-5)^2}{5} + \frac{(3-5)^2}{5} \\ &= 9.2 \end{aligned}$$

10.8.2 Deviance (likelihood-ratio) statistic

$$
\begin{aligned}
G^2 = 2\Bigg(& 3\log\frac{3}{5} + 7\log\frac{7}{5} + 5\log\frac{5}{5} \\
& + 10\log\frac{10}{5} + 2\log\frac{2}{5} + 3\log\frac{3}{5} \Bigg) \\
= 8.8 &
\end{aligned}
$$

Note that although X^2 and G^2 have the same approximate chi-square distribution, their realized numerical values need not be identical.

The corresponding p-values are

$$P(\chi^2_5 \geq 9.2) = 0.10, \qquad P(\chi^2_5 \geq 8.8) = 0.12.$$

Given a significance level of $\alpha = 0.05$, we **fail to reject** the null hypothesis.

Importantly, failing to reject H_0 does **not** mean that the null hypothesis is true. Rather, it means that we do not have sufficient evidence to conclude that it is false.

In this example, the fair-die model does not fit the data exactly, but the lack of fit is not large enough to conclude that the die is unfair at the 5% significance level.

The following SAS code implements the chi-square goodness-of-fit test for the dice example.

SAS code: Chi-square goodness-of-fit test

```
DATA DIE;
    INPUT FACE $ COUNT;
    DATALINES;
1 3
2 7
3 5
4 10
5 2
6 3
;
RUN;

PROC FREQ DATA=DIE;
    WEIGHT COUNT;
    TABLES FACE / CHISQ;
    /*
```

```
    TABLES FACE / NOCUM ALL
        CHISQ TESTP=(16.67 16.67 16.67 16.67 16.67 16.67);
    */
RUN;
```

The FREQ Procedure

FACE	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1	3	10.00	3	10.00
2	7	23.33	10	33.33
3	5	16.67	15	50.00
4	10	33.33	25	83.33
5	2	6.67	27	90.00
6	3	10.00	30	100.00

Chi-Square Test for Equal Proportions	
Chi-Square	9.2000
DF	5
Pr > ChiSq	0.1013

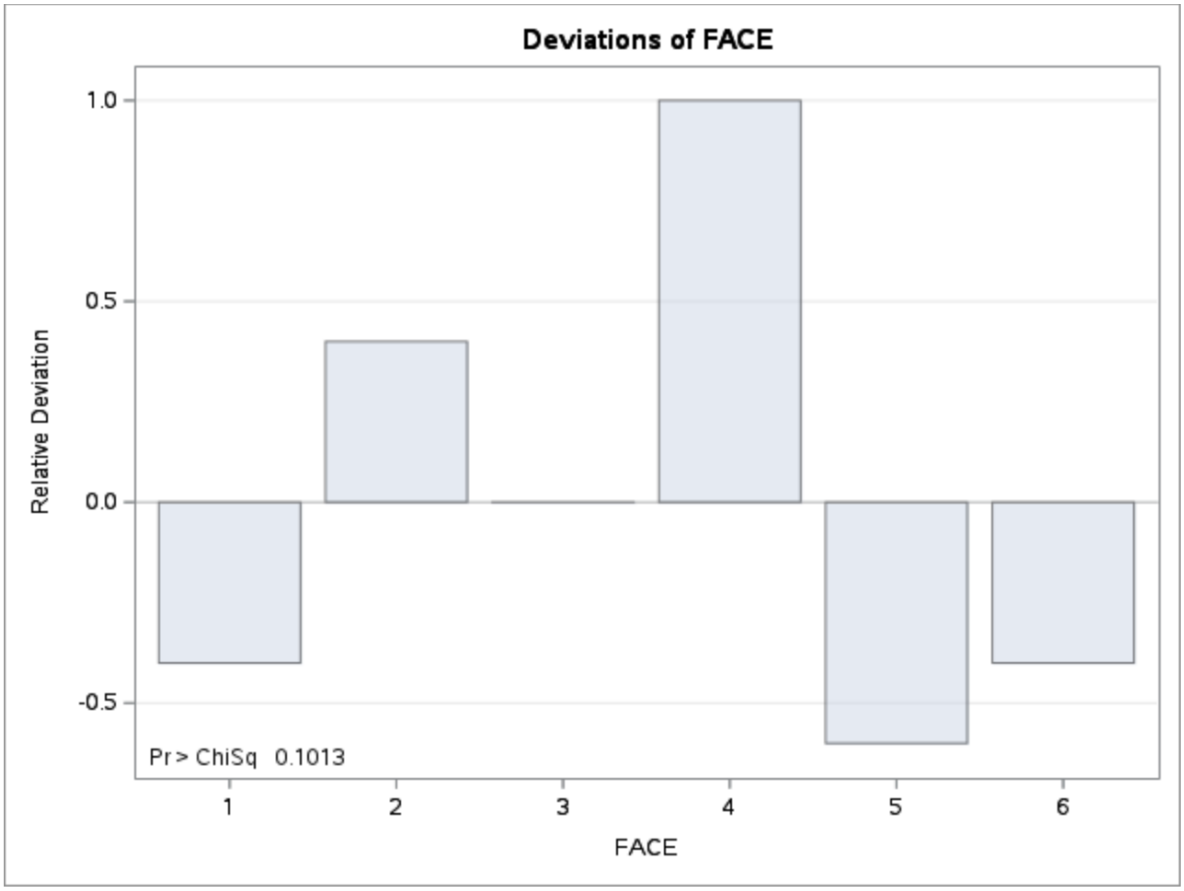
The `PROC FREQ` step computes the Pearson chi-square statistic for testing whether the die

is fair. The commented TESTP= option shows how expected probabilities can be specified explicitly.

Computing residuals and goodness-of-fit statistics manually

We now compute Pearson residuals, deviance residuals, and the test statistics X^2 and G^2 directly from the data.

```
DATA CAL;
    SET DIE;

    PI = 1/6;
    ECOUNT = 30 * PI;

    /* Pearson residual */
    RES = (COUNT - ECOUNT) / SQRT(ECOUNT);

    /* Deviance residual */
    DEVRES = SQRT(ABS(2 * COUNT * LOG(COUNT / ECOUNT)))
             * SIGN(COUNT - ECOUNT);
RUN;

PROC PRINT DATA=CAL;
RUN;
```



Sample Size = 30

Obs	FACE	COUNT	PI	ECOUNT	RES	DEVRES
1	1	3	0.16667	5	-0.89443	-1.75070
2	2	7	0.16667	5	0.89443	2.17039
3	3	5	0.16667	5	0.00000	0.00000
4	4	10	0.16667	5	2.23607	3.72330
5	5	2	0.16667	5	-1.34164	-1.91446
6	6	3	0.16667	5	-0.89443	-1.75070

Computing test statistics and p-values

```
PROC SQL;
    SELECT
        SUM((COUNT - ECOUNT)**2 / ECOUNT) AS X2,
        1 - PROBCHI(CALCULATED X2, 5)     AS PVALUE_X2,
        2 * SUM(COUNT * LOG(COUNT / ECOUNT)) AS G2,
        1 - PROBCHI(CALCULATED G2, 5)     AS PVALUE_G2
    FROM CAL;
QUIT;
```

X2	PVALUE_X2	G2	PVALUE_G2
9.2	0.101348	8.778485	0.118233

Note that PROC FREQ automatically reports the Pearson chi-square statistic, but does not directly report the deviance statistic. The deviance statistic G^2 must be computed manually as shown above.

10.9 Tomato Phenotypes Example

Tall cut-leaf tomatoes were crossed with dwarf potato-leaf tomatoes, and $n = 1611$ offspring were classified by their phenotypes as summarized in the table below.

Categories	tall cut-leaf	tall potato-leaf	dwarf cut-leaf	dwarf potato-leaf
Number of Observations	$O_1 = 926$	$O_2 = 288$	$O_3 = 293$	$O_4 = 104$
Expected Observation				

Genetic theory predicts that the four phenotypes should occur with relative frequencies

$$9 : 3 : 3 : 1,$$

which implies that the phenotypes are **not equally likely**. In particular, the dwarf potato-leaf phenotype is expected to be the least frequent.

We would like to test whether the observed data are consistent with this genetic theory.

10.9.1 Null hypothesis

Under the null hypothesis, the cell probabilities are

$$\pi_1 = \frac{9}{16}, \quad \pi_2 = \pi_3 = \frac{3}{16}, \quad \pi_4 = \frac{1}{16}.$$

10.9.2 Expected frequencies

The expected counts under H_0 are

$$\begin{aligned} E_1 &= 1611 \times \frac{9}{16} = 906.2, \\ E_2 = E_3 &= 1611 \times \frac{3}{16} = 302.1, \\ E_4 &= 1611 \times \frac{1}{16} = 100.7. \end{aligned}$$

10.9.3 Goodness-of-fit results

Using these expected counts, we compute the goodness-of-fit statistics and obtain

$$X^2 = 1.47, \qquad G^2 = 1.48.$$

The two statistics are nearly identical. Under the chi-square distribution with $k - 1 = 3$ degrees of freedom, the corresponding p-values are approximately

$$p \approx 0.69.$$

Since the p-value is much larger than the significance level $\alpha = 0.05$, we **fail to reject** the null hypothesis. The observed data are therefore **consistent with the genetic theory** predicting a $9 : 3 : 3 : 1$ ratio.

10.9.3.1 SAS Code: Goodness-of-Fit Test

Step 1: Enter the observed counts

```
DATA LEAF;
    INPUT TYPE $ COUNT;
    DATALINES;
tallc   926
tallp   288
dwarfc  293
dwarfp  104
;
RUN;
```

The FREQ Procedure

TYPE	Frequency	Percent	Test Percent	Cumulative Frequency	Cumulative Percent
tallc	926	57.48	56.25	926	57.48
tallp	288	17.88	18.75	1214	75.36
dwarfc	293	18.19	18.75	1507	93.54
dwarfp	104	6.46	6.25	1611	100.00

Step 2: Pearson chi-square goodness-of-fit test

```
PROC FREQ DATA=LEAF ORDER=DATA;
    WEIGHT COUNT;
    TABLES TYPE / ALL CHISQ
        TESTP=(56.25 18.75 18.75 6.25);
RUN;
```

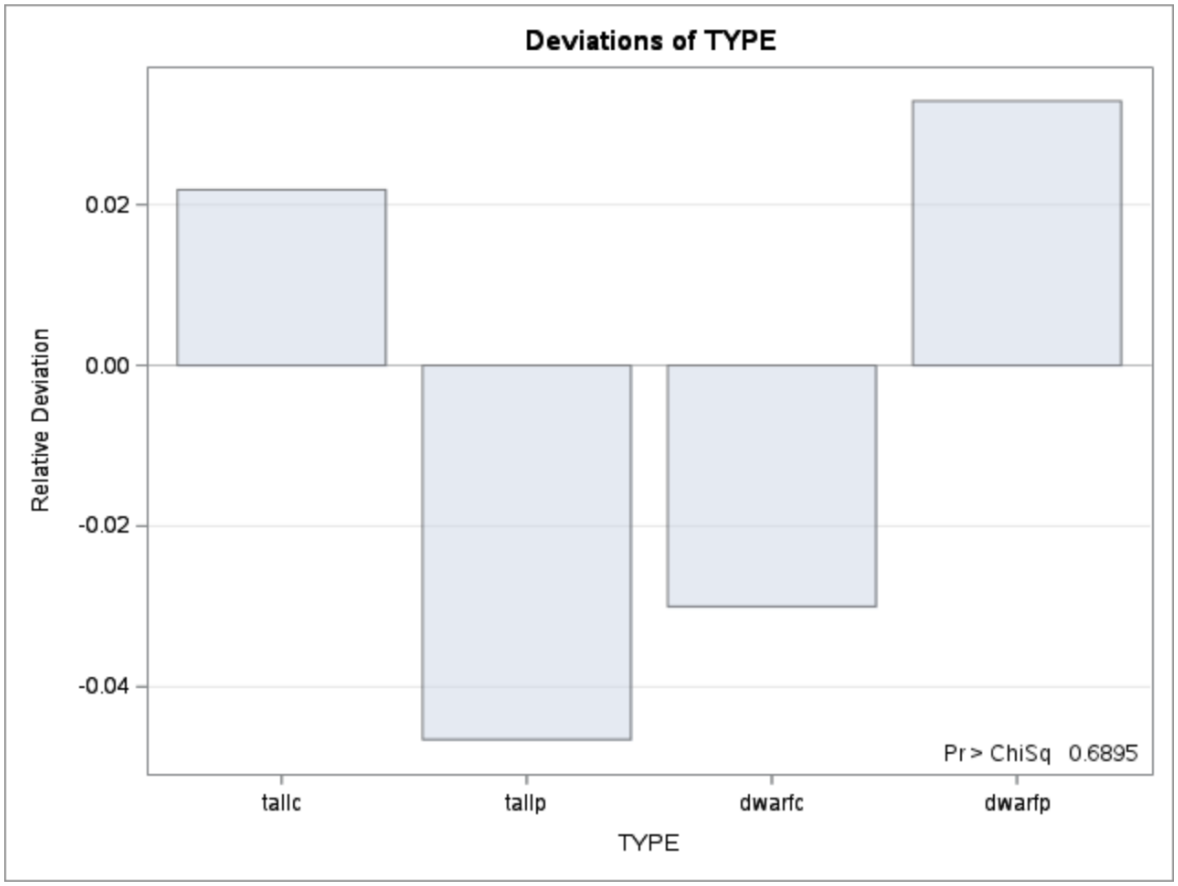
The proportions (56.25, 18.75, 18.75, 6.25) correspond to the theoretical probabilities $(9, 3, 3, 1)/16$.

Manual Computation of Residuals, X^2, and G^2

Chi-Square Test for Specified Proportions	
Chi-Square	1.4687
DF	3
Pr > ChiSq	0.6895

Step 3: Define theoretical probabilities

```
DATA PI;
    INPUT PI;
    PI = PI / 16;
CARDS;
9
3
3
1
;
RUN;
```



Sample Size = 1611

Step 4: Compute expected counts and residuals

```
DATA CAL;
    MERGE LEAF PI;
    ECOUNT = 1611 * PI;

    /* Pearson residual */
    RES = (COUNT - ECOUNT) / SQRT(ECOUNT);

    /* Deviance residual */
    DEVRES = SQRT(ABS(2 * COUNT * LOG(COUNT / ECOUNT)))
             * SIGN(COUNT - ECOUNT);
RUN;

PROC PRINT DATA=CAL;
RUN;
```

Obs	TYPE	COUNT	PI	ECOUNT	RES	DEVRES
1	tallc	926	0.5625	906.188	0.65816	6.32891
2	tallp	288	0.1875	302.063	-0.80912	-5.24022
3	dwarfc	293	0.1875	302.063	-0.52143	-4.22497
4	dwarfp	104	0.0625	100.688	0.33012	2.59476

Step 5: Compute X^2 and G^2 explicitly

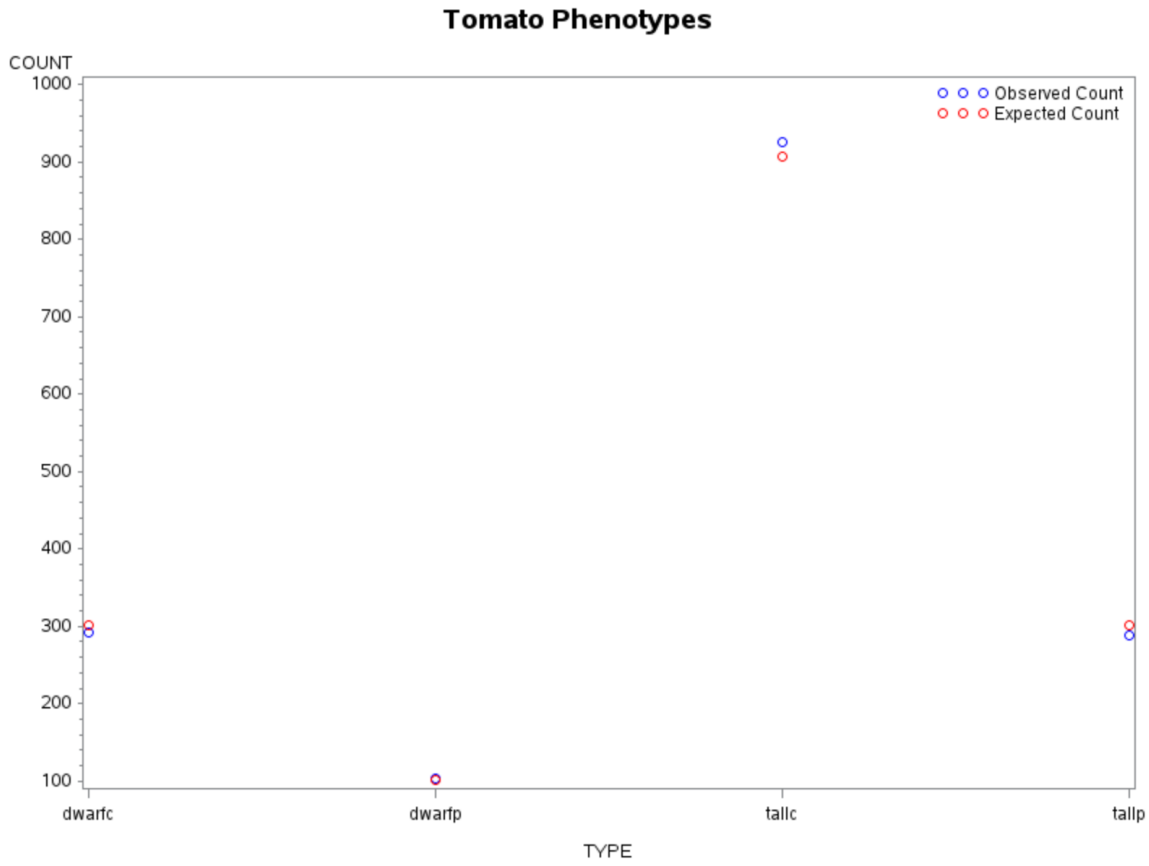
```
PROC SQL;
    SELECT
        SUM((COUNT - ECOUNT)**2 / ECOUNT) AS X2,
        1 - PROBCHI(CALCULATED X2, 3)     AS PVAL1,
        2 * SUM(COUNT * LOG(COUNT / ECOUNT)) AS G2,
        1 - PROBCHI(CALCULATED G2, 3)     AS PVAL2
    FROM CAL;
QUIT;
```

X2	PVAL1	G2	PVAL2
1.468722	0.689508	1.477587	0.687453

Visualization: Observed vs Expected Counts

```
GOPTIONS RESET=ALL;
SYMBOL1 V=CIRCLE C=BLUE I=NONE;
SYMBOL2 V=CIRCLE C=RED  I=NONE;

LEGEND1 LABEL=NONE
        VALUE=('Observed Count' 'Expected Count')
        ACROSS=1
        POSITION=(TOP RIGHT INSIDE)
        MODE=PROTECT;
```

```
TITLE "Tomato Phenotypes";

PROC GPLOT DATA=CAL;
    PLOT COUNT*TYPE ECOUNT*TYPE / OVERLAY LEGEND=LEGEND1;
RUN;
QUIT;

TITLE;
```



Interpretation

- Small values of X^2 and G^2 indicate good agreement with the model
- Large p-values confirm no strong evidence against the $9:3:3:1$ ratio
- Both Pearson and deviance statistics lead to the same conclusion

This example illustrates how goodness-of-fit testing connects theory, computation, and visualization in categorical data analysis.

11 Power Analysis with application in one sample t-test

Learning Objectives

1. Understand what power analysis is.
2. Explain why and when we need power analysis.

11.1 1 Introduction

A **power analysis** is a calculation that helps you determine the **minimum sample size** required for a study to reliably detect an effect of interest.

A power analysis is built around **four main components**.
If you know (or can reasonably estimate) **any three** of them, you can solve for the **fourth**.

11.1.1 Key Components of Power Analysis

- **Statistical power**
  The probability that a statistical test will correctly detect an effect of a certain size *when the effect truly exists*.
  Power is commonly set at **80% (0.8)** or higher.
- **Sample size**
  The minimum number of observations needed to detect an effect of a given size with a specified power level.
- **Significance level (α)**
  The maximum probability of rejecting a true null hypothesis (Type I error).
  Typically set at $\alpha = 0.05$.
- **Expected effect size**
  A standardized measure that quantifies the magnitude of the expected effect.
  This is often based on:
  - prior studies,

– pilot data, or
– subject-matter considerations.

11.2 Motivation Examples

A company that manufactures light bulbs claims that a particular type of light bulb lasts **850 hours on average**, with a **standard deviation of 50 hours**.

A consumer protection group believes that the manufacturer has **overestimated the average lifespan by about 40 hours**.

Question:
How many light bulbs does the consumer protection group need to test in order to demonstrate this discrepancy with reasonable confidence?

This is a classic **sample size determination problem**, where:

- the effect size is the difference between the claimed mean and the suspected true mean,
- the variability is known or estimated,
- and power and significance level must be specified.

It has been estimated that the average height of **American white male adults** is **70 inches**.

Suppose it is postulated that there is a **positive correlation between height and intelligence**.
If this is true, then the average height of **white male graduate students** on campus should be **greater than 70 inches**.

You plan to test this hypothesis by randomly sampling a group of white male graduate students.

Key question:
How small can the sample be (or how few individuals do you need to measure) and still have enough power to detect a meaningful difference?

This motivates power analysis for a **one-sided one-sample *t*-test**, where:

- the null hypothesis specifies a reference mean,
- the alternative hypothesis specifies a directional effect,
- and the goal is to balance feasibility (small sample size) with statistical reliability (adequate power).

In the remainder of this lecture, we will formalize these ideas and use the one-sample t-test to illustrate how power, sample size, effect size, and significance level are mathematically connected.

11.3 Before We Start the Analysis

For the power analysis in this lecture, we focus on the first example, which concerns testing the *average lifespan of a light bulb*. Our **first goal** is to determine the **number of light bulbs that must be tested**. That is, we want to find the **sample size** for a given significance level and power. Next, we will reverse the process and determine the **power**, given a sample size and significance level.

We know that the manufacturer claims the average lifespan of the light bulb is **850 hours**, with a **standard deviation of 50 hours**, while the consumer protection group believes that the manufacturer has overestimated the lifespan by about **40 hours**.

In terms of hypotheses:

- Null hypothesis: $H_0 : \mu = 850$
- Alternative hypothesis: $H_A : \mu = 810$

The **significance level** is the probability of a **Type I error**, that is, rejecting H_0 when it is actually true. We will set $\alpha = 0.05$.

The **power of the test** against H_A is the probability that the test **rejects H_0 when H_A is true**. We will set the desired power level to Power $= 0.90$.

Before proceeding with the power analysis, let us briefly discuss the role of the **standard deviation**. Intuitively, the number of light bulbs required for testing depends on the **variability** of their lifespans:

- If all light bulbs had exactly the same lifespan, then testing a single bulb would suffice.
- In reality, lifespans vary substantially. For example, some bulbs may last 1000 hours, while others may last 500 hours.

Therefore, the **standard deviation of the lifespan distribution** plays a critical role in determining the required sample size.

11.4 Power Analysis in SAS

In SAS, it is straightforward to perform a power analysis for **comparing means** using `PROC POWER`.

To compute the required sample size:

1. Specify the mean under the null hypothesis.
2. Specify the mean under the alternative hypothesis.
3. Specify the population standard deviation.
4. Set the significance level α (default is 0.05).

5. Specify the desired power level. Here we set Power = 0.90.
6. Indicate that the test is a **one-sample *t*-test**.

The following SAS code performs a power analysis for **Example 1**:

```
PROC POWER;
    ONESAMPLEMEANS TEST=t
        NULLMEAN = 850
        MEAN     = 810
        STDDEV   = 50
        POWER    = 0.9
        NTOTAL   = .;
RUN;
```

This code tells SAS to solve for the total sample size (ntotal) needed to achieve 90% power for a one-sample t-test at significance level ($\alpha = 0.05$).

The POWER Procedure
One-Sample t Test for Mean

Fixed Scenario Elements	
Distribution	Normal
Method	Exact
Null Mean	850
Mean	810
Standard Deviation	50
Nominal Power	0.9
Number of Sides	2
Alpha	0.05

Computed N Total	
Actual Power	**N Total**
0.909	19

The result tells us that we need a **sample size of at least 19 light bulbs** in order to reject H_0 under the alternative hypothesis H_A with a **power of 0.9**.

Next, suppose we have a sample of size $n = 10$. How much power do we have if we keep all other quantities the same? We can use the same SAS program to calculate the power.

```
PROC POWER;
    ONESAMPLEMEANS TEST=t
        NULLMEAN = 850
        MEAN     = 810
        STDDEV   = 50
        POWER    = .
        NTOTAL   = 10;
RUN;
```

The POWER Procedure
One-Sample t Test for Mean

Fixed Scenario Elements	
Distribution	Normal
Method	Exact
Null Mean	850
Mean	810
Standard Deviation	50
Total Sample Size	10
Number of Sides	2
Alpha	0.05

Computed Power
Power
0.616

You can see that the power is about 0.616 for a sample size of 10.

> This means that if we only test 10 light bulbs, we have about a 61.6% chance of correctly rejecting the null hypothesis when the true mean is 810 hours. This is below the commonly desired power level of 0.8, indicating that a sample size of 10 may not be sufficient to reliably detect the effect size of interest.

What if we have sample size of 15 or 20? We can use ta list of sample sizes as input to `PROC POWER`.

```
PROC POWER;
    ONESAMPLEMEANS TEST=t
        NULLMEAN = 850
        MEAN     = 810
        STDDEV   = 50
        POWER    = .
        NTOTAL   = 10 to 45 by 5;
RUN;
```

The POWER Procedure
One-Sample t Test for Mean

Fixed Scenario Elements	
Distribution	Normal
Method	Exact
Null Mean	850
Mean	810
Standard Deviation	50
Number of Sides	2
Alpha	0.05

Computed Power		
Index	**N Total**	**Power**
1	10	0.616
2	15	0.821
3	20	0.924
4	25	0.970
5	30	0.988
6	35	0.996
7	40	0.999
8	45	>.999

We can also expect that if we actually know the standard deviation is smaller, we would need fewer light bulbs to achieve the same power. We can vary the standard deviation in `PROC POWER` to see how it affects the required sample size.

```
PROC POWER;
    ONESAMPLEMEANS TEST=t
        NULLMEAN = 850
        MEAN     = 810
        STDDEV   = 30 to 100 by 10
        POWER    = 0.8
        NTOTAL   = . ;
RUN;
```

The POWER Procedure
One-Sample t Test for Mean

Fixed Scenario Elements	
Distribution	Normal
Method	Exact
Null Mean	850
Mean	810
Nominal Power	0.8
Number of Sides	2
Alpha	0.05

Computed N Total			
Index	**Std Dev**	**Actual Power**	**N Total**
1	30	0.834	7
2	40	0.803	10
3	50	0.821	15
4	60	0.807	20
5	70	0.815	27
6	80	0.808	34
7	90	0.803	42
8	100	0.808	52

Here, varying `STDDEV` to show how reduced variability lowers required sample size, while keeping the hypothesis, effect size, and power fixed.

11.5 Discussion

Normality assumption

One technical assumption underlying the power analysis: the **normality assumption**. If the variable of interest is not normally distributed, a small sample size will usually *not* achieve the power indicated by the theoretical results, because those results are derived under normality. In such cases, it may not even be appropriate to conduct a one-sample t-test with a very small sample size when the normality assumption is questionable.

Relative difference

There is another important technical point. What truly matters for power analysis is **not the individual values**, but the **difference between the two means**, relative to the variability of the data. In fact, what really determines the power is the ratio

$$\frac{\text{difference in means}}{\text{standard deviation}},$$

which is referred to as the **effect size**.

For example, we would obtain the same power if we subtracted 800 from both means, changing 850 to 50 and 810 to 10. The absolute scale of the data does not affect the power, only the standardized difference does.

```
PROC POWER;
    ONESAMPLEMEANS TEST=t
        NULLMEAN = 50
        MEAN     = 10
        STDDEV   = 50
        POWER    = 0.9
        NTOTAL   = . ;
RUN;
```

If we standardize our variable, we can calculate the means in terms of change in standard deviation.

```
PROC POWER;
    ONESAMPLEMEANS TEST=t
        NULLMEAN = 1
        MEAN     = 0.2
        STDDEV   = 1
        POWER    = 0.9
        NTOTAL   = . ;
RUN;
```

> **i** Key Takeaway
>
> It is usually not an easy task to determine the **"true" effect size**. In practice, we make our best guess based on existing literature or evidence from a pilot study. A good estimate of the effect size is **crucial** for a successful power analysis.

11.6 Math behind the power analysis

- We saw earlier that in tests of significance there are two types of errors. Each error has a given probability of occurring.
  - The **significance level** (Type I error) is

$$\alpha = P(\text{reject } H_0 \mid H_0 \text{ is true}).$$

  - The probability of a **Type II error** is

$$\beta = P(\text{fail to reject } H_0 \mid H_0 \text{ is false}).$$

- The **power of a test** is related to the probability of a Type II error. It is defined as

$$\begin{aligned} 1 - \beta &= 1 - P(\text{fail to reject } H_0 \mid H_0 \text{ is false}) \\ &= P(\text{reject } H_0 \mid H_0 \text{ is false}). \end{aligned}$$

- Therefore, to define power, we must be specific about what **"when H_0 is false"** means.

11.7 A Visualization Example

Bottles of a popular cola drink are supposed to contain 300 ml of cola.
There is some variation from bottle to bottle because the filling machine is not perfectly precise.

Suppose the distribution of the contents is normal with standard deviation

$$\sigma = 3 \text{ ml}.$$

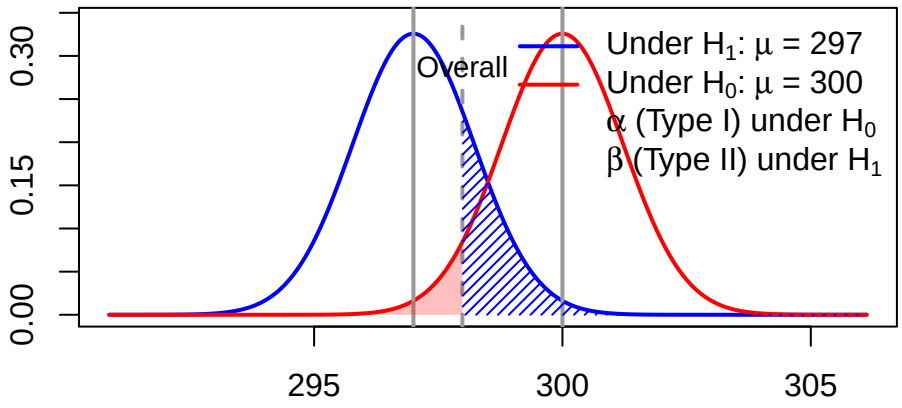
We want to carry out the following hypothesis test at significance level $\alpha = 0.05$,
based on a sample of six bottles of cola:

$$H_0 : \mu = 300 \quad \text{versus} \quad H_1 : \mu < 300.$$

If the power is calculated under the alternative hypothesis with

$$\mu = 297,$$

then we can visualize α and β as follows.



11.7.1 Interpretation of the Power Visualization

The figure above illustrates the concepts of **Type I error** (α), **Type II error** (β), and **power** ($1-\beta$) for a **left-tailed one-sample** t**-test**.

- The **red curve** represents the sampling distribution of the test statistic under the **null hypothesis**

$$H_0 : \mu = 300.$$

- The **blue curve** represents the sampling distribution under the **alternative hypothesis**

$$H_1 : \mu = 297.$$

- The **vertical cutoff line** corresponds to the critical value determined by the significance level $\alpha = 0.05$.

Type I Error (α)

- The **red shaded region** to the left of the cutoff is

$$\alpha = P(\text{reject } H_0 \mid H_0 \text{ is true}).$$

- This is the probability of incorrectly rejecting the null hypothesis.

Type II Error (β)

- The **blue shaded region** to the right of the cutoff is

$$\beta = P(\text{fail to reject } H_0 \mid H_0 \text{ is false}).$$

- This is the probability of failing to detect the true mean $\mu = 297$.

Power ($1-\beta$)

- Power increases when:
  - the true mean moves farther away from 300,
  - the sample size increases,
  - or the variability (σ) decreases.

This visualization highlights that **power is not a single number**, but depends on *which alternative value* of μ we consider when H_0 is false.

12 Two Sample t-test for Independent and Paired Data

Learning Objectives

1. Distinguish between **independent** and **paired** two-sample t-tests.
2. Identify the **appropriate test** based on study design.
3. State the **model assumptions** for each test.
4. Perform two-sample t-tests in **SAS** using `PROC TTEST`.
5. Interpret **test statistics, p-values, and confidence intervals**.

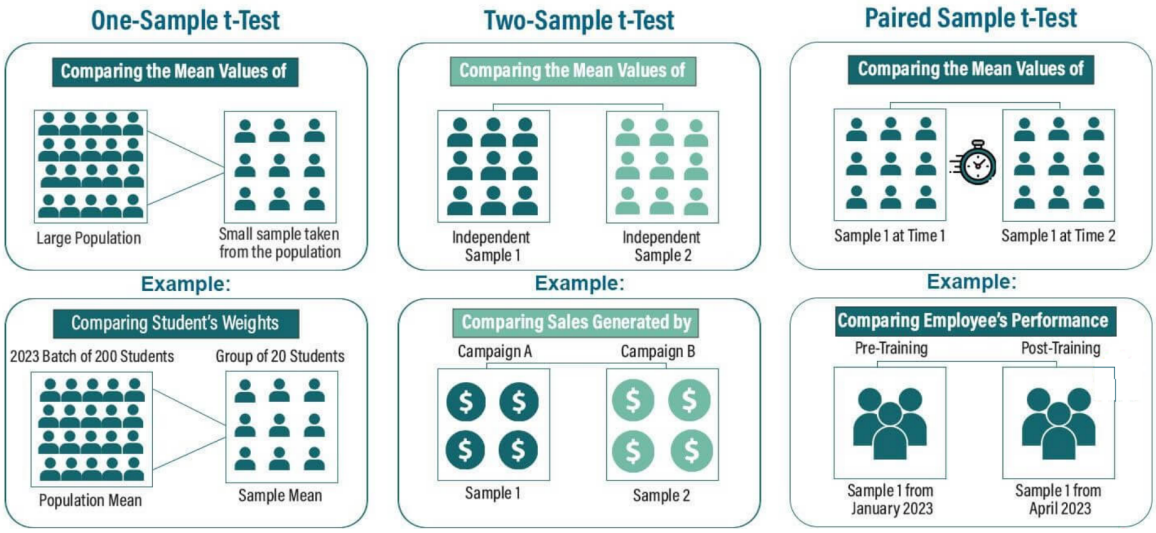
In many applications, we want to compare **two populations**, for example:

- Treatment vs control groups
- Male vs female outcomes
- Before vs after an intervention
- Method A vs Method B

The key question is:

Are the population means different?

This leads to **two-sample inference**.



Since we see that there are different scenarios for the t-test, we may have another question:

Which t-test should I use?

The answer depends on the **study design**. We will discuss two main types of two-sample t-tests:

1. **Independent two-sample t-test**: Used when the two samples are independent of each other (e.g., treatment vs control groups).

```
/*============================================================
  Create the dataset for two-sample t-test example
  Variables:
    Species : group indicator (1 or 2)
    Height  : numeric response
============================================================*/

DATA my_data;
    INPUT Species Height;
    DATALINES;
  1 13
  1 15
  1 15
  1 16
  1 16
  1 16
  1 17
```

```
  1 18
  1 18
  1 19
  1 20
  1 21
  2 15
  2 15
  2 16
  2 18
  2 19
  2 19
  2 19
  2 20
  2 21
  2 23
  2 23
  2 24
;
RUN;

/* Optional: check the data */
PROC PRINT DATA=my_data;
RUN;
```

Obs	Species	Height
1	1	13
2	1	15
3	1	15
4	1	16
5	1	16
6	1	16
7	1	17
8	1	18
9	1	18
10	1	19
11	1	20
12	1	21
13	2	15
14	2	15
15	2	16
16	2	18
17	2	19
18	2	19
19	2	19
20	2	20
21	2	21
22	2	23
23	2	23
24	2	24

2. **Paired two-sample t-test**: Used when the two samples are related or paired (e.g., before vs after measurements on the same subjects).

Student	Pre-Test Score	Post-Test Score
1	88	91
2	82	84
3	84	88
4	93	90
5	75	79
6	78	80
7	84	88
8	87	90
9	95	90
10	91	96
11	83	88
12	89	89
13	77	81
14	68	74
15	91	92

12.1 Inference Goals of the Two Sample *t*-Test

Suppose the two population means are denoted by:

$$\mu_1 \quad \text{and} \quad \mu_2.$$

A two-sample t-test always begins with the same **null hypothesis**:

$$H_0 : \mu_1 = \mu_2.$$

That is, the two population means are equal. The alternatively hypothesis can take different forms.

Alternative Hypotheses

Depending on the scientific question, the alternative hypothesis can take one of three forms.

1. Two-sided (two-tailed): $H_1 : \mu_1 \neq \mu_2$

   The two population means are different.

2. Left-tailed: $H_1 : \mu_1 < \mu_2$

   Population 1 has a *smaller* mean than population 2.

3. Right-tailed: $H_1 : \mu_1 > \mu_2$

   Population 1 has a *larger* mean than population 2.

Confidence Interval Interpretation

When constructing a confidence interval, inference is made on the **difference of means**: $\mu_1 - \mu_2$.

- If the confidence interval **contains 0**, we fail to reject H_0.
- If the confidence interval **does not contain 0**, we reject H_0.

12.2 Sample SAS code

12.2.1 Two Independent Sample t-test

```
/*=========================================================
  Example: Two-Sample t-Test (Independent Samples)
  STAT 8678 - Two-Sample t-Test
==========================================================*/

/*------------------------------
  Create the dataset
------------------------------*/
DATA my_data;
    INPUT Species $ Height;
    DATALINES;
1 13
1 15
1 15
1 16
1 16
1 16
1 17
1 18
1 18
1 19
1 20
```

```
1 21
2 15
2 15
2 16
2 18
2 19
2 19
2 19
2 20
2 21
2 23
2 23
2 24
;
RUN;

/*------------------------------
  Two-sample t-test
  H0:  1 -  2 = 0
  Two-sided test,   = 0.05
------------------------------*/
PROC TTEST DATA=my_data
           SIDES=2
           ALPHA=0.05
           H0=0;
    CLASS Species;
    VAR Height;
RUN;
```

The TTEST Procedure

Variable: Height

Species	Method	N	Mean	Std Dev	Std Err	Minimum	Maximum
1		12	17.0000	2.2962	0.6629	13.0000	21.0000
2		12	19.3333	3.0551	0.8819	15.0000	24.0000
Diff (1-2)	**Pooled**		-2.3333	2.7024	1.1033		
Diff (1-2)	**Satterthwaite**		-2.3333		1.1033		

Species	Method	Mean	95% CL Mean		Std Dev	95% CL Std Dev	
1		17.0000	15.5410	18.4590	2.2962	1.6266	3.8987
2		19.3333	17.3922	21.2744	3.0551	2.1642	5.1871
Diff (1-2)	**Pooled**	-2.3333	-4.6213	-0.0453	2.7024	2.0900	3.8249
Diff (1-2)	**Satterthwaite**	-2.3333	-4.6316	-0.0350			

Method	Variances	DF	t Value	Pr > |t|
Pooled	Equal	22	-2.11	0.0460
Satterthwaite	Unequal	20.422	-2.11	0.0469

Equality of Variances				
Method	**Num DF**	**Den DF**	**F Value**	**Pr > F**
Folded F	11	11	1.77	0.3577

Note on Two-Sample t-Test Methods in SAS

In `PROC TTEST`, SAS reports results from **two different methods** by default:

- **Method: Pooled**
  This corresponds to the **two-sample equal-variance t-test**, which assumes

$$\sigma_1^2 = \sigma_2^2.$$

The test statistic uses a **pooled variance estimator**.

- **Method: Satterthwaite**
  This corresponds to the **two-sample unequal-variance t-test**, also known as the **Welch t-test**.
  No assumption of equal variances is required, and the degrees of freedom are approximated using the **Satterthwaite formula**.

  Practical guidance:
  When the equal-variance assumption is questionable, the **Satterthwaite (Welch) test** is generally preferred and is the default recommendation in modern practice.

12.2.2 Paired Sample t-test

```
/* Create dataset */
DATA test_scores;
    INPUT PRE POST;
    DATALINES;
88 91
82 84
84 88
93 90
75 79
78 80
84 88
87 90
95 90
91 96
83 88
89 89
77 81
68 74
91 92
;
RUN;


/*view dataset*/
PROC PRINT DATA=test_scores;

/*perform paired samples t-test*/
PROC TTEST DATA=test_scores ALPHA=.05;
```

```
PAIRED PRE*POST;
RUN;
```

The TTEST Procedure

Difference: PRE - POST

N	Mean	Std Dev	Std Err	Minimum	Maximum
15	-2.3333	3.0394	0.7848	-6.0000	5.0000

Mean	95% CL Mean		Std Dev	95% CL Std Dev	
-2.3333	-4.0165	-0.6502	3.0394	2.2252	4.7935

DF	t Value	Pr > |t|
14	-2.97	0.0101

12.3 Statistical Assumptions of the Two Sample t-Test

The validity of a two-sample t-test relies on the following assumptions:

1. **Continuous data**
   The response variable is continuous (not discrete or categorical).
2. **Normality**
   The data in each population follow a normal distribution.
   - In practice, the two-sample t-test is reasonably robust to mild departures from normality, especially for moderate or large sample sizes.

3. **Equal variances (for the pooled t-test)**
   The population variances of the two groups are equal:

$$\sigma_1^2 = \sigma_2^2.$$

   - If this assumption is violated, the **Welch (Aspin–Welch / Satterthwaite) two-sample t-test** should be used instead.

4. **Independence of samples**
   The two samples are independent of each other.

   - There is no relationship between individuals in Sample 1 and Sample 2.
   - If observations are paired or matched, a **paired t-test** should be used instead.

5. **Random sampling**
   Both samples are simple random samples from their respective populations.
   Each individual in the population has an equal probability of being selected.

12.4 Other Two-Sample Tests

As discussed in the one-population setting, there are several hypothesis tests related to the two-sample problem beyond the two-sample t-test. The most common ones include:

12.4.1 Two-Proportion Z Test

This test is used to compare **two population proportions**.

Hypothesis Testing

Let p_1 and p_2 denote the two population proportions.

- **Null hypothesis**

$$H_0 : p_1 = p_2$$

  (the two population proportions are equal)

- **Alternative hypotheses**
  - Two-tailed:

$$H_1 : p_1 \neq p_2$$

    (the two population proportions are not equal)
  - Left-tailed:

$$H_1 : p_1 < p_2$$

    (population 1 proportion is less than population 2 proportion)

  - Right-tailed:
$$H_1 : p_1 > p_2$$
(population 1 proportion is greater than population 2 proportion)

- **Confidence Interval** Inference is made on the difference in proportions:
$$p_1 - p_2.$$

12.4.2 Two-Sample Variance *F* Test

This test is used to compare **two population variances**.

Hypothesis Testing

Let σ_1^2 and σ_2^2 denote the two population variances.

- **Null hypothesis**
$$H_0 : \sigma_1^2 = \sigma_2^2$$
(the two population variances are equal)

- **Alternative hypotheses**
  - Two-tailed:
$$H_1 : \sigma_1^2 \neq \sigma_2^2$$
(the two population variances are not equal)
  - Left-tailed:
$$H_1 : \sigma_1^2 < \sigma_2^2$$
(population 1 variance is less than population 2 variance)
  - Right-tailed:
$$H_1 : \sigma_1^2 > \sigma_2^2$$
(population 1 variance is greater than population 2 variance)

- **Confidence Interval** Inference is made on the ratio of variances:
$$\frac{\sigma_1^2}{\sigma_2^2}.$$

12.4.3 Summary

Test	Parameter of Interest	CI Targets
Two-sample t-test	$\mu_1 - \mu_2$	Difference in means
Two-proportion Z test	$p_1 - p_2$	Difference in proportions
Two-sample F test	σ_1^2/σ_2^2	Ratio of variances

Part III

Regression Analysis

13 Analysis of Variance

Learning Objectives

1. Understand what **ANOVA (Analysis of Variance)** is.
2. Explain **why and when** ANOVA is needed.
3. Recognize the connection between **two-sample t-tests** and **one-way ANOVA**.
4. Formulate ANOVA models and hypotheses.
5. Perform one-way ANOVA in **SAS** and interpret the output.

13.1 Introduction

In the previous lecture, we studied the **two-sample t-test**, where the goal was to compare the means of **two independent populations**.

Recall the equal-variance two-sample model:

$$\begin{aligned} X_{1i} &\overset{iid}{\sim} N(\mu_1, \sigma^2), \quad i = 1, \dots, n_1, \\ X_{2j} &\overset{iid}{\sim} N(\mu_2, \sigma^2), \quad j = 1, \dots, n_2, \end{aligned} \tag{13.1}$$

with inference focused on the mean difference

$$\mu_1 - \mu_2.$$

The corresponding hypothesis test was:

$$H_0 : \mu_1 = \mu_2.$$

13.1.1 From Two-Sample t-Test to ANOVA

A natural question arises:

What if we want to compare more than two population means?

For example:

- Comparing test scores across **three or more teaching methods**
- Comparing average yields across **multiple fertilizer types**
- Comparing mean response times across **several algorithms**

Running multiple pairwise t-tests is **not appropriate**, because:

- It inflates the **Type I error rate**
- It leads to inconsistent inference

This motivates the need for **ANOVA**.

13.2 ANOVA as a Generalization of the Two-Sample t-Test

To unify the two-sample setting, we can rewrite the data using an **indicator variable**.

Let the combined data be:

$$\{(I_{i,2}, X_i)\}_{i=1}^{n_1+n_2},$$

where

$$I_{i,2} = \begin{cases} 1, & \text{if observation } i \text{ comes from group 2,} \\ 0, & \text{otherwise.} \end{cases}$$

Then the two-sample model can be written as a **linear model**:

$$X_i = \alpha + \beta_1 I_{i,2} + \varepsilon_i, \quad \varepsilon_i \overset{iid}{\sim} N(0, \sigma^2). \tag{13.2}$$

Here, note that models Equation 13.1 and the linear model Equation 13.2 are **equivalent**. Specifically:

- $\alpha = \mu_1$, represents the mean of group 1. This is referred as the *baseline effect*
- $\beta_1 = \mu_2 - \mu_1$, this is the group effect, representing the difference between group 2 and group 1.
- Testing $\mu_1 = \mu_2$ is equivalent to testing:

$$H_0 : \beta_1 = 0$$

Note, we can also write the model in terms of group 2, and use it as the baseline group. In this case, β_1 is the difference effect form group 1, and the null hypothesis would be $H_0 : \beta_1 = 0$ as well.

13.3 ANOVA Analysis: Compare Multiple Mean Values

Now, we see how to extend Equation 13.2 to compare the population mean from multiple groups. Suppose now we have $K > 2$ groups.

The previous model Equation 13.2 can be extended to compare the population mean across **multiple groups (more than two)**.
Suppose we have **three groups** (1, 2, 3) and choose **group 1 as the baseline**.
An extension of the model is

$$X_i = \alpha + \beta_1 I_{i,2} + \beta_2 I_{i,3} + \epsilon_i$$

where

- $I_{i,2}$ indicates whether the observation belongs to **group 2**
- $I_{i,3}$ indicates whether the observation belongs to **group 3**

The parameters have the interpretation

$$\alpha = \mu_1$$

$$\beta_1 = \mu_2 - \mu_1$$

$$\beta_2 = \mu_3 - \mu_1$$

Thus, statistical inference on β_1 and β_2 allows us to determine whether the **group means differ**.
This type of analysis is known as **Analysis of Variance (ANOVA)**.

13.3.1 When Do We Use ANOVA?

In general, ANOVA is used under similar assumptions as the **two-sample t-test**.

- If the independent variable has **two levels**, both
  - a **two-sample t-test**, and
  - **ANOVA**
    can be used.

- If the independent variable has **three or more levels**,
  ANOVA is required.

Suppose three reading instruction methods (A, B, C) are given to *15 subjects*.

After instruction, a reading test is administered, and the number of words per minute is recorded.

The goal is to test whether the *instruction methods lead to different reading scores*.

13.3.2 Hypothesis Test

Global test

The ANOVA test examines whether **any group means differ**.

Our null hypothesis is:

$$H_0 : \mu_1 = \mu_2 = \mu_3.$$

This means, there is no differences among group means.

The alternative Hypothesis is:

$$H_1 : \text{At least one pair of means is different}$$

$$(\mu_1 \neq \mu_2 \text{ or } \mu_1 \neq \mu_3 \text{ or } \mu_2 \neq \mu_3)$$

After the Global Test

If the ANOVA test shows the null hypothesis is rejected, it tells us that **a difference exists**, but it does **not tell us where the difference occurs**.

Therefore, additional tests are performed to determine **which groups differ**.
These are called **multiple comparison procedures**.

Examples include:

- Tukey's HSD
- Bonferroni correction
- Scheffé test

13.4 ANOVA in SAS

```
DATA words;
INPUT words employee $;
DATALINES;
700  A
850  A
820  A
640  A
920  A
480  B
460  B
500  B
570  B
580  B
500  C
550  C
480  C
600  C
610  C
;
RUN;

PROC ANOVA DATA=words;
    TITLE Example of one-way ANOVA;
    CLASS employee;
    MODEL words = employee;
    MEANS employee / HOVTEST=WELCH;
RUN;

PROC MEANS DATA=words N MEAN STD;
    CLASS employee;
    VAR words;
RUN;
```

In the code chunk above:

- The `MEANS` statement generates the **group mean values** of the dependent variable (`words`).
- The option `HOVTEST` is used to **check the homogeneity of variance assumption** across groups.
- The option `WELCH` performs **Welch's ANOVA**, which is more appropriate when the **equal variance assumption is violated**.

- More details about **assumption checking** will be discussed below.

Example of one-way ANOVA

The ANOVA Procedure

Class Level Information		
Class	**Levels**	**Values**
employee	3	A B C

Number of Observations Read	15
Number of Observations Used	15

Example of one-way ANOVA

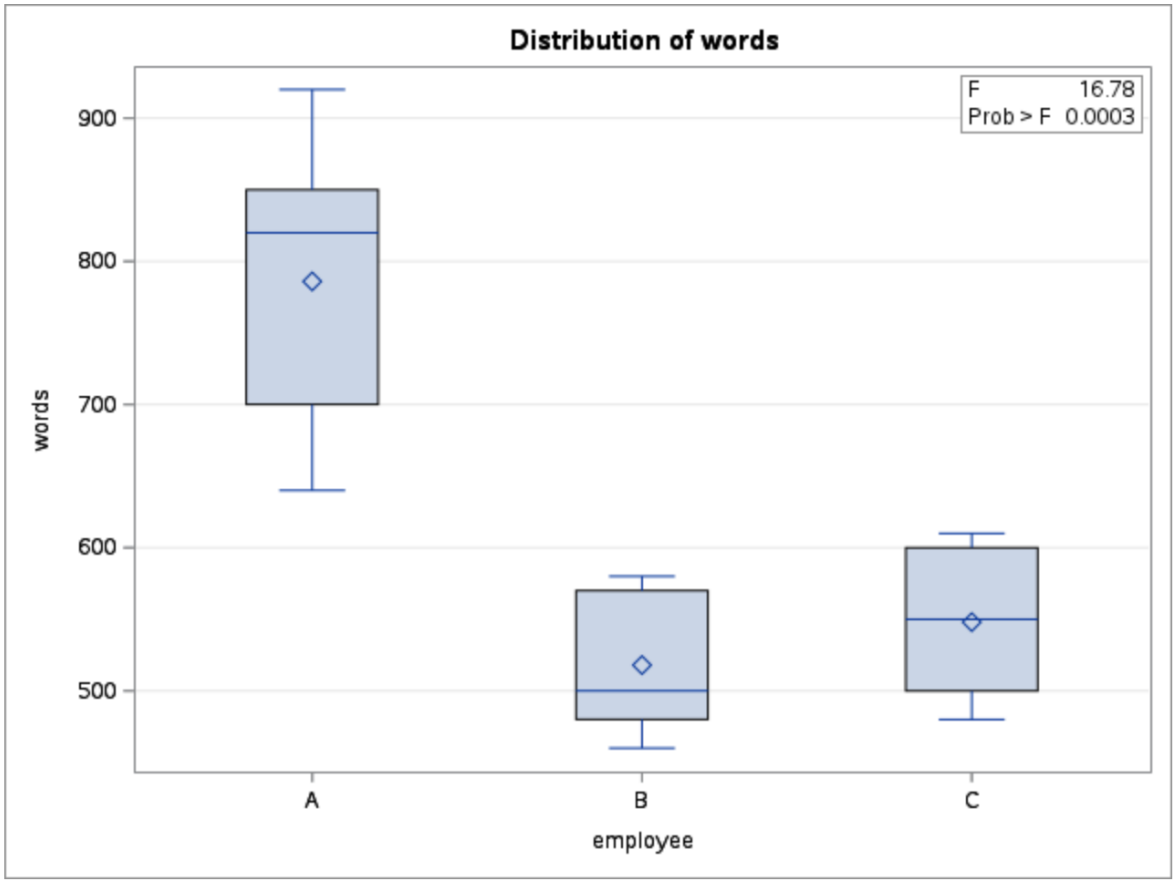
The ANOVA Procedure

Dependent Variable: words

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	2	215613.3333	107806.6667	16.78	0.0003
Error	12	77080.0000	6423.3333		
Corrected Total	14	292693.3333			

R-Square	Coeff Var	Root MSE	words Mean
0.736653	12.98256	80.14570	617.3333

Source	DF	Anova SS	Mean Square	F Value	Pr > F
employee	2	215613.3333	107806.6667	16.78	0.0003


Distribution of words
F 16.78
Prob > F 0.0003
900
800
700
600
500
words
A
B
C
employee

The MEANS Procedure

Analysis Variable : words				
employee	**N Obs**	**N**	**Mean**	**Std Dev**
A	5	5	786.0000000	113.9298029
B	5	5	518.0000000	54.0370243
C	5	5	548.0000000	58.0517011

13.4.1 Interpretation of the outputs

Based on the data, we conduct a hypothesis test (with a **0.05 significance level**) to determine whether the **three instruction methods have different effects on the reading result**.

The ANOVA output contains a table labeled **Source**, which shows the **sources of variation** in the data.
The term **Model** represents the effect of the independent variable (in this example, the **instruction method**).

One-way ANOVA is based on the **F-distribution**.
The computed **F-statistic is 16.78**, with a **p-value of 0.0003**.

Since the **p-value $<$ 0.05**, we **reject the null hypothesis**.

Therefore, we conclude that the **reading instruction methods do not all produce the same mean word counts**. In other words, **at least one instruction method differs from the others**.

Alternatively you can turn the `ODS GRAPHICS` on to get the diagnostic plots for ANOVA assumptions.

```
ODS GRAPHICS ON;
PROC ANOVA DATA=words;
    CLASS employee;
    MODEL words = employee;
```

```
    MEANS employee;
RUN;
ODS GRAPHICS OFF;
```

13.4.2 Assumption Validation for the ANOVA Test

The ANOVA test assumes that:

- The **dependent variable** (`word`) is **continuous**, and the **independent variable** (`method`) is **categorical**.
- The experiment follows a **random and independent design**.
  For example, the same person should **not be tested under all three instruction methods**, since that would create a **dependent (repeated-measures) design**.
- The samples (word counts from the three instruction methods) are **normally distributed** and have **equal variances**.

The **normality assumption** can be checked using the `PROC UNIVARIATE` procedure.

It is important to note that **ANOVA is relatively robust to mild departures from normality**, especially when sample sizes are similar.

The assumption of **equal variances (homogeneity of variances)** can be checked using the `HOVTEST` option in the `MEANS` statement:

```
MEANS method / HOVTEST=WELCH;
```

The result is what we have seen before,

Example of One-Way ANOVA

The ANOVA Procedure

Levene's Test for Homogeneity of words Variance ANOVA of Squared Deviations from Group Means					
Source	**DF**	**Sum of Squares**	**Mean Square**	**F Value**	**Pr > F**
employee	2	2.0668E8	1.0334E8	3.74	0.0545
Error	12	3.3121E8	27601053		

Welch's ANOVA for words			
Source	**DF**	**F Value**	**Pr > F**
employee	2.0000	10.52	0.0065
Error	7.5552		

As shown in the output, the p-value from **Levene's test** is **0.0545**, which is close to the significance level of **0.05**. Therefore, the result lies near the borderline.

Two interpretations are possible:

- If we use a significance level of **0.05**, there is **insufficient evidence to reject the null hypothesis of homogeneity of variances**.
- If a larger significance level is used (for example **0.10**), we would **reject the homogeneity assumption**. In that case, the **Welch ANOVA test** should be used instead.

Recall that:

- The p-value from the **regular ANOVA** (assuming equal variances, i.e., homogeneity) is **0.0003**.
- The p-value from **Welch's ANOVA** (not assuming equal variances, heterogeneity) is **0.0065**.

At the **0.05 significance level**, both tests lead to the **same conclusion**:
the **instruction methods have different effects on the reading results**.

13.5 Multiple Comparison

In general, methods used to identify **which group means differ after a significant global ANOVA test** are called **multiple comparison tests** or **post hoc tests**.

SAS provides several procedures to investigate differences between levels of the independent variable. Examples include:

- Duncan's multiple-range test (`DUNCAN`)
- Student–Newman–Keuls multiple-range test (`SNK`)
- Least significant difference test (`LSD`)
- Tukey's studentized range test (`TUKEY`)
- Scheffé's multiple-comparison procedure (`SCHEFFE`)

To request a multiple comparison test in SAS, place the desired test option **after a slash (`/`) in the `MEANS` statement**.

It is convenient to include the multiple comparison request **at the same time as the global ANOVA test**. However, the results of multiple comparisons should only be interpreted **after the global ANOVA test indicates a significant difference** among group means.

The following SAS code performs the SNK multiple comparison test at a significance level of 0.05:

```
PROC ANOVA DATA=words;
    TITLE "Multiple Comparison: SNK Test";
    CLASS EMPLOYEE;
    MODEL WORDS = EMPLOYEE;
    MEANS EMPLOYEE / SNK ALPHA=0.05;
RUN;
```

Multiple Comparison: SNK Test

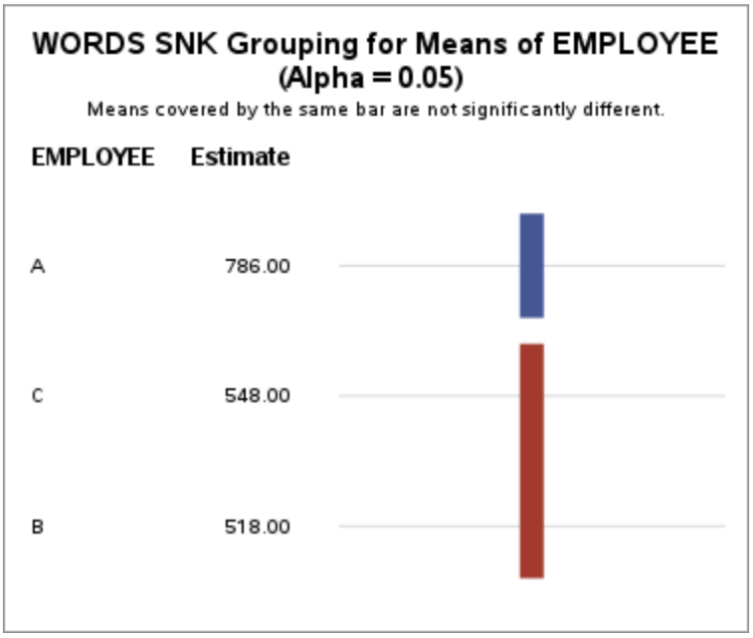
The ANOVA Procedure

Student-Newman-Keuls Test for WORDS

Note: This test controls the Type I experimentwise error rate under the complete null hypothesis but not under partial null hypotheses.

Alpha	0.05
Error Degrees of Freedom	12
Error Mean Square	6423.333

Number of Means	**2**	**3**
Critical Range	110.43659	135.22869



Under the **SNK grouping** column, groups that share the **same letter** are **not significantly different**.

For example, groups **C** and **B** both have the letter **"B"** in the grouping column, indicating that their means are **not significantly different**.

Group **A** has the letter **"A"**, meaning that its mean is **significantly different** (p-value $<$ 0.05) from the means of groups **B** and **C**.

Conclusion:
method A performs significantly better than methods B and C, while methods B and C are

not significantly different from each other.

Key Takeaways

- The **two-sample t-test is a special case of ANOVA** when there are only two groups.
- **ANOVA tests whether at least one group mean differs** among multiple groups using the **F-statistic**.
- Before interpreting ANOVA results, we should **check assumptions**, especially **homogeneity of variances** (e.g., using **Levene's test**).
- If the equal variance assumption is violated, **Welch's ANOVA** provides a more robust alternative.
- If the **global ANOVA test is significant**, **post-hoc multiple comparison tests** (e.g., SNK, Tukey, LSD) are used to determine **which groups differ**.

14 Regression Analysis

Learning Objectives

1. Understand what **regression analysis** is.
2. Explain the difference between **correlation** and **regression**.
3. Fit a **simple linear regression model**.
4. Interpret regression coefficients and statistical tests.
5. Perform regression analysis in **SAS** and interpret the output.
6. Check the **assumptions of linear regression** using diagnostic plots.

14.1 Introduction

In many studies, the goal is not only to determine whether variables are related, but also to *quantify how one variable influences another.*

For example:

- How does height affect weight?
- How does study time affect exam score?
- How does advertising expenditure affect sales?

These types of questions are addressed using *regression analysis.*

Regression analysis studies the relationship between a **dependent variable** and one or more **independent variables**, and can also be used for **prediction**.

14.2 Correlation vs Regression

Before fitting a regression model, it is often useful to examine the **correlation between variables**.

Correlation

- Measures the *strength and direction of linear association* between two variables.
- The correlation coefficient ranges from *-1 to 1.*

- Correlation is **symmetric**, meaning it does not distinguish between predictor and response variables.

However:

> **Correlation does not imply causation.**

A strong correlation between two variables does not necessarily mean that one variable **causes** the other.

Regression analysis goes further by specifying a **directional relationship** between variables.

14.3 Correlation between Two Variables

Suppose in a health screening, measurements are taken on several individuals:

- Gender
- Height
- Weight
- Age

An example dataset of such is given below.

```
DATA measurement;
  INPUT gender $ height weight age;
  DATALINES;
  M 68 155 23
  F 61 99 20
  F 63 115 21
  M 70 205 45
  M 69 170 35
  F 65 125 30
  M 72 220 48
;
RUN;
```

To examine the correlation between every paired variables, we can use the `PROC CORR` procedure in SAS.

```
PROC CORR DATA=measurement;
  TITLE "Correlation Analysis";
  VAR height weight age;
RUN;
```

Correlation Analysis

The CORR Procedure

3 Variables:	height weight age

Simple Statistics						
Variable	**N**	**Mean**	**Std Dev**	**Sum**	**Minimum**	**Maximum**
height	7	66.85714	3.97612	468.00000	61.00000	72.00000
weight	7	155.57143	45.79613	1089	99.00000	220.00000
age	7	31.71429	11.42679	222.00000	20.00000	48.00000

Pearson Correlation Coefficients, N = 7 Prob > |r| under H0: Rho=0			
	height	**weight**	**age**
height	1.00000	0.97165 0.0003	0.86467 0.0120
weight	0.97165 0.0003	1.00000	0.92621 0.0027
age	0.86467 0.0120	0.92621 0.0027	1.00000

The output will show the correlation coefficients between each pair of variables, along with the corresponding p-values to test for statistical significance.

The output from PROC CORR provides:

- Descriptive statistics for each variable
- A Pearson correlation matrix

Interpretation of the output

The top number in each cell is the Pearson correlation coefficient, which measures the strength of the linear association between two variables.

The number below it is the p-value for testing

$$H_0 : \rho = 0$$

against

$$H_1 : \rho \neq 0$$

For example:

- The correlation between height and weight is 0.9716.
- The corresponding p-value is 0.003.

Since the p-value is less than 0.05, we conclude that height and weight are significantly correlated.

The correlation matrix is symmetric along the diagonal line, and each variable has a correlation of 1 with itself.

Important Note

Even when two variables show a strong correlation, this does not necessarily imply a causal relationship.

A strong association may occur because:

- another hidden (confounding) variable influences both variables, or
- the relationship is coincidental.

14.4 Regression in SAS

Suppose our research question is:

> to investigate the *impact of one variable on another*, or to *predict the value of one variable based on another*.

If this is the case, the regression model can be used to achieve these goals.

In the example dataset, we are interested in studying the relationship between **height** and **weight**. For example:

- Does **height influence weight**?
- Can we **predict a person's weight given their height**?

To answer this question, we use a **simple linear regression model**.

The regression model can be written as

$$\text{Weight} \sim \text{Height}$$

which means we model **weight as a function of height**.

14.4.1 Regression Model

The mathematical form of the simple linear regression model is

$$Y = \beta_0 + \beta_1 X + \epsilon$$

where

- Y is the **dependent variable** (weight)
- X is the **independent variable** (height)
- β_0 is the **intercept**
- β_1 is the **slope**
- ϵ is the **random error term**

with

$$\epsilon \sim N(0, \sigma^2).$$

The slope β_1 represents the **change in the expected value of Y for a one-unit increase in X**.

SAS Implementation

In SAS, we fit the regression model using the `PROC REG` procedure.

```
PROC REG DATA=measurement;
  TITLE "Example of Linear Regression";
  MODEL weight = height;
RUN;
```

In this code:

- `PROC REG` fits a linear regression model.
- `MODEL weight = height` specifies that weight is the response variable and height is the predictor variable.

Example of Linear Regression

The REG Procedure
Model: MODEL1
Dependent Variable: weight

Number of Observations Read	7
Number of Observations Used	7

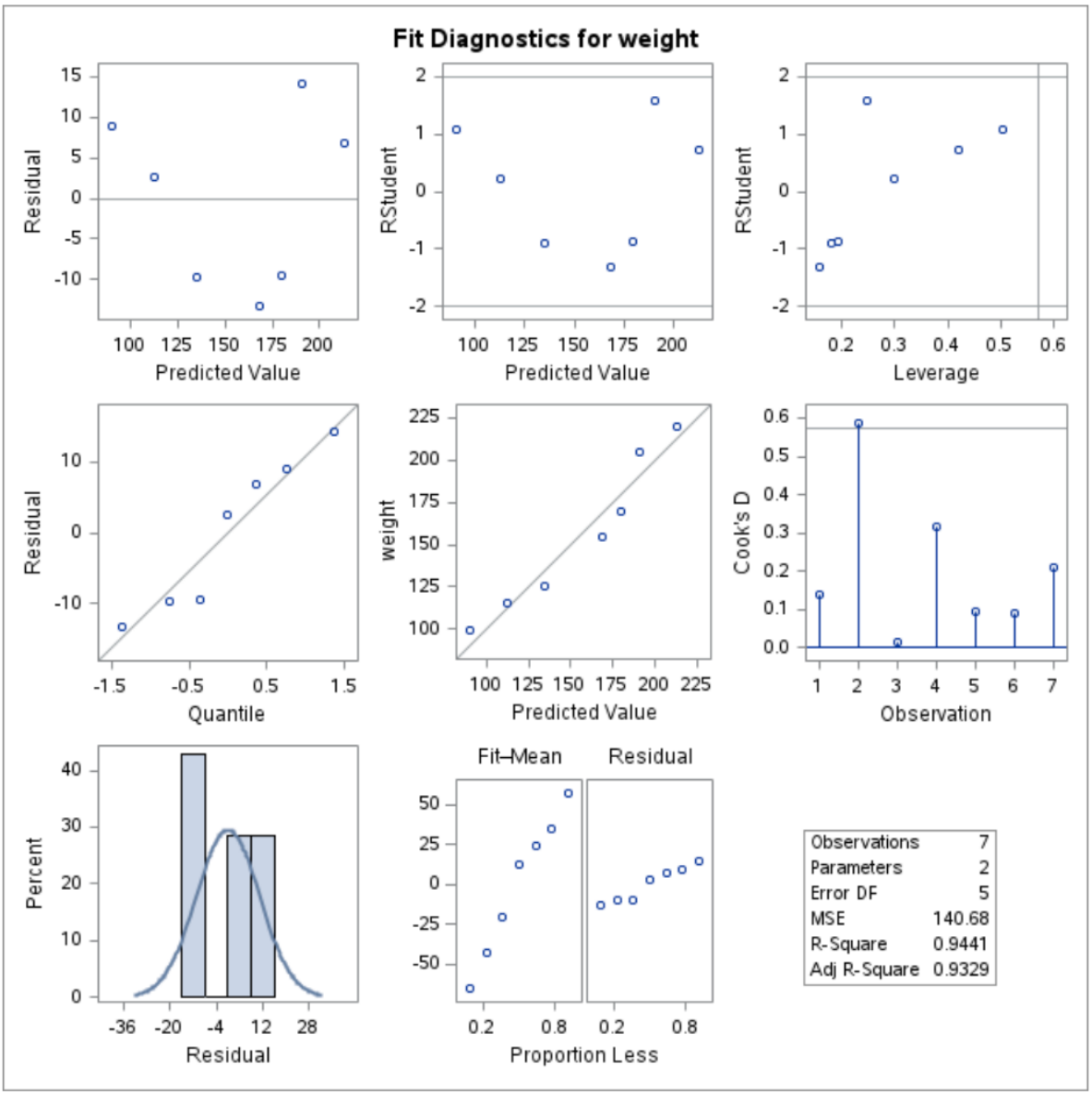
Analysis of Variance					
Source	**DF**	**Sum of Squares**	**Mean Square**	**F Value**	**Pr > F**
Model	1	11880	11880	84.45	0.0003
Error	5	703.38705	140.67741		
Corrected Total	6	12584			

Root MSE	11.86075	**R-Square**	0.9441
Dependent Mean	155.57143	**Adj R-Sq**	0.9329
Coeff Var	7.62399		

Parameter Estimates					
Variable	**DF**	**Parameter Estimate**	**Standard Error**	**t Value**	**Pr > |t|**
Intercept	1	-592.64458	81.54217	-7.27	0.0008
height	1	11.19127	1.21780	9.19	0.0003

Example of Linear Regression

The REG Procedure
Model: MODEL1
Dependent Variable: weight



14.5 Interpreting the Regression Output

Linear regression assumes that the **dependent variable** (e.g., Y) depends linearly on the **independent variable** (X).
The simple linear regression model is

$$Y = \beta_0 + \beta_1 X + \epsilon$$

where

- β_0 is the **intercept**
- β_1 is the **slope**
- ϵ is the **random error term**

with

$$\epsilon \sim N(0, \sigma^2)$$

The regression output provides estimates for β_0 and β_1.
In this example, the fitted regression equation is

$$\text{weight} = -592.64458 + 11.19127 \times \text{height}$$

This equation can be used to **predict weight from height**.

14.5.1 Example Prediction

For a person who is **70 inches tall**, the predicted weight is

$$\hat{weight} = -592.64458 + 11.19127 \times 70 = 190.66$$

Thus, the predicted weight for a **70-inch-tall person** is approximately **190.66 pounds**.

14.5.2 Testing the Slope

In the regression output, the **Parameter Estimates** table includes the following columns:

- **Standard Error**
- **t Value**
- **Pr** $>$ **|t|** (p-value)

These values are used to test the hypothesis

$$H_0 : \beta_1 = 0$$

which means that the predictor variable has **no linear effect** on the response variable.

If the p-value is **small (e.g., less than 0.05)**, we reject the null hypothesis.

In this example, the p-value for the slope is **0.0003**, which is much smaller than 0.05.

Therefore, we conclude that **height has a statistically significant linear effect on weight**.

14.6 Assumption Validation for Linear Regression

Linear regression assumes that the **relationship between two variables is linear**, and that the **residuals** are **normally distributed**.

Residuals are defined as

$$\text{Residual} = Y - \hat{Y}$$

where Y is the observed value and $\hat{Y}$ is the predicted value from the regression model.

These assumptions can be checked using **scatter plots** and **residual plots**.

The SAS regression procedure also produces several diagnostic figures that help evaluate the model assumptions.

The main diagnostic plots are shown below.

14.6.1 Residual Diagnostics

From the **residual plot**, we should check the following:

- **Does the residual plot show an evenly scattered pattern around 0?**

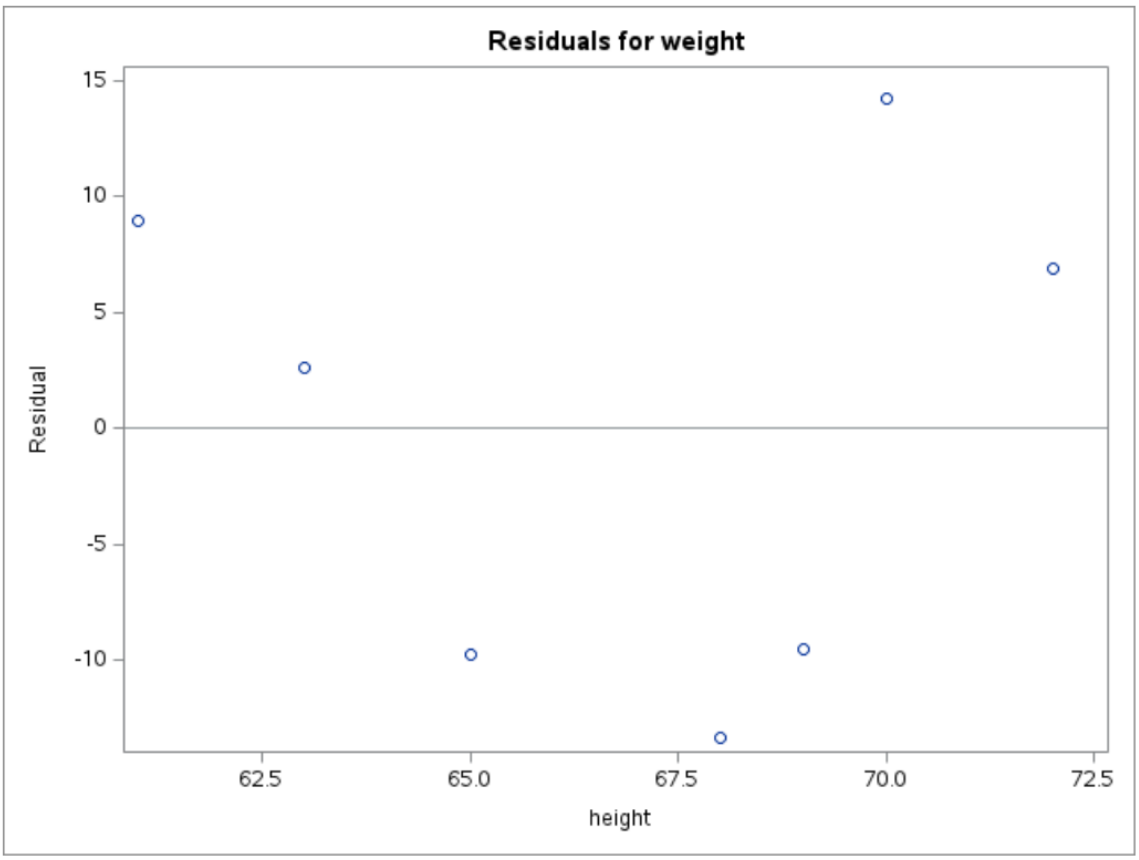
  A random "white-noise-like" scatter around zero suggests that the **linear model fits the data well**.

- **Do the residuals follow a normal distribution?**

  This can be examined using **normality diagnostics**, such as a **Q–Q plot** or formal statistical tests.

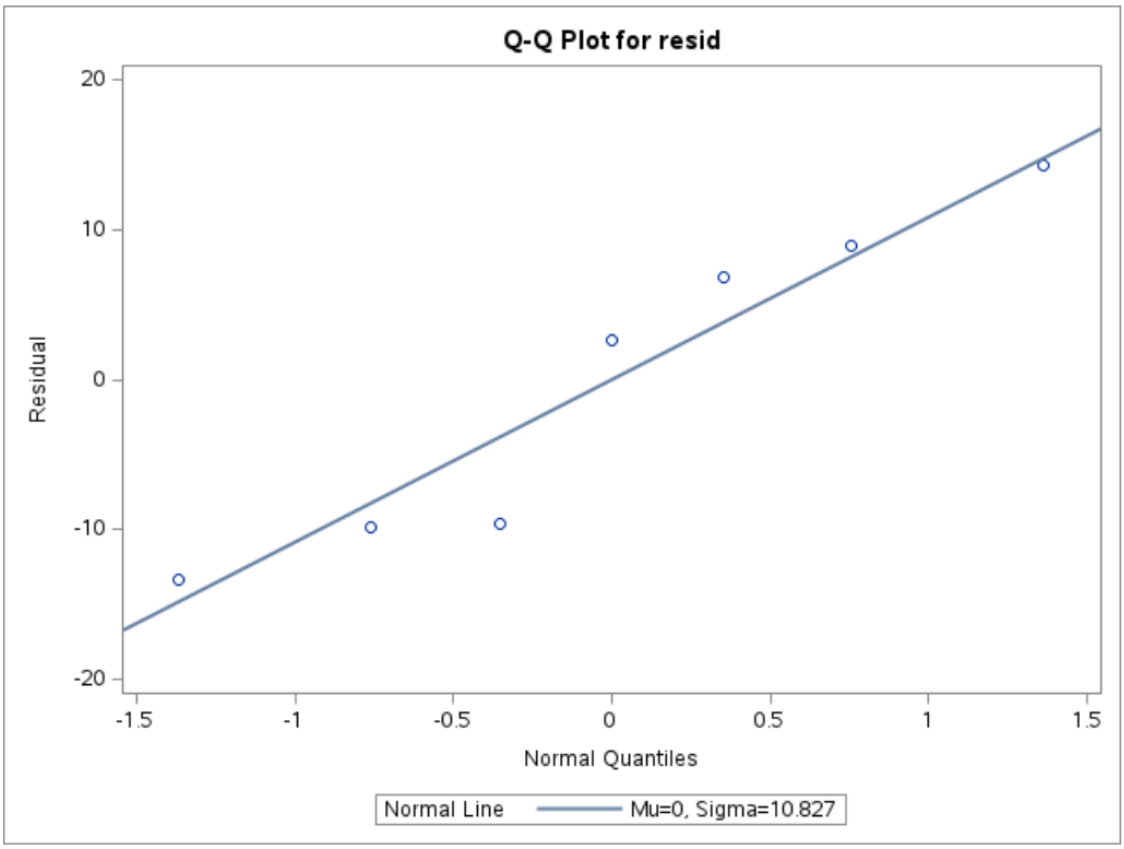
The normality of residuals can be checked using the following SAS code:

```
PROC REG DATA=measurement;
TITLE "Regression and residual plots";
MODEL weight = height;
OUTPUT OUT=myout R=resid;
RUN;

PROC UNIVARIATE DATA=myout NORMAL;
QQPLOT resid / NORMAL(mu=est sigma=est color=red l=1);
RUN;
```


Residuals for weight
15
10
5
0
-5
-10
Residual
62.5
65.0
67.5
70.0
72.5
height

Regression and residual plots

The UNIVARIATE Procedure



Regression and residual plots

The UNIVARIATE Procedure
Variable: resid (Residual)

Moments			
N	7	**Sum Weights**	7
Mean	0	**Sum Observations**	0
Std Deviation	10.8273346	**Variance**	117.231175
Skewness	-0.0425954	**Kurtosis**	-2.0229122
Uncorrected SS	703.387048	**Corrected SS**	703.387048
Coeff Variation	.	**Std Error Mean**	4.09234782

Basic Statistical Measures			
Location		**Variability**	
Mean	0.000000	**Std Deviation**	10.82733
Median	2.594880	**Variance**	117.23117
Mode	.	**Range**	27.61747
		Interquartile Range	18.76506

Tests for Location: Mu0=0				
Test	**Statistic**		**p Value**	
Student's t	**t**	0	**Pr > |t|**	1.0000
Sign	**M**	0.5	**Pr >= |M|**	1.0000
Signed Rank	**S**	-1	**Pr >= |S|**	0.9375

Tests for Normality				
Test	**Statistic**		**p Value**	
Shapiro-Wilk	**W**	0.903764	**Pr < W**	0.3544
Kolmogorov-Smirnov	**D**	0.239758	**Pr > D**	>0.1500
Cramer-von Mises	**W-Sq**	0.061099	**Pr > W-Sq**	>0.2500
Anderson-Darling	**A-Sq**	0.366219	**Pr > A-Sq**	>0.2500

Quantiles (Definition 5)	
Level	**Quantile**
100% Max	14.25602
99%	14.25602
95%	14.25602
90%	14.25602
75% Q3	8.97741
50% Median	2.59488
25% Q1	-9.78765
10%	-13.36145
5%	-13.36145
1%	-13.36145
0% Min	-13.36145

Extreme Observations			
Lowest		**Highest**	
Value	**Obs**	**Value**	**Obs**
-13.36145	1	-9.55271	5
-9.78765	6	2.59488	3
-9.55271	5	6.87349	7
2.59488	3	8.97741	2
6.87349	7	14.25602	4

As we can see that, the p-value for normality validation is 0.3544, which is greater than 0.05, so we fail to reject the null hypothesis of normality.

15 Regression with Interaction

Learning Objectives

1. Understand interaction effects in regression models
2. Distinguish between main effects and interaction effects
3. Interpret regression coefficients in an interaction model
4. Compute and interpret simple slopes

15.1 Why Do We Need Interaction?

In many real-world problems, the effect of one variable may depend on another variable.

For example:

- Does the effect of study time depend on gender?
- Does treatment effectiveness depend on age?
- Does price sensitivity depend on income?

If the relationship between predictors is not constant, a simple additive model is not sufficient.

This motivates the need for **interaction terms**.

Without interaction → parallel lines
With interaction → non-parallel lines

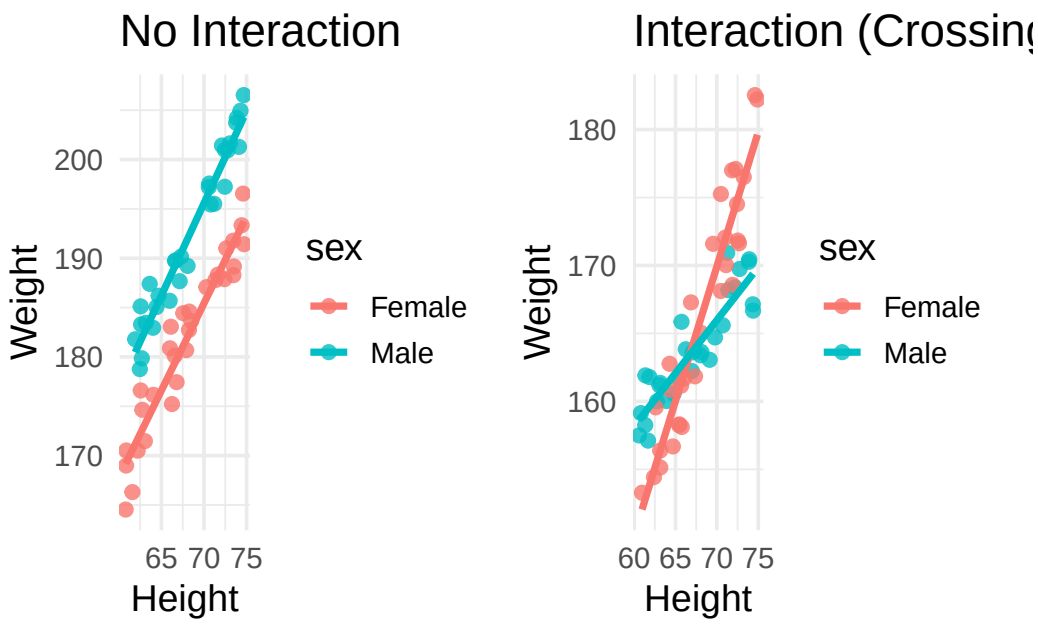
15.2 Visualizing Interaction Effects

The easiest way to understand interaction is through visualization.

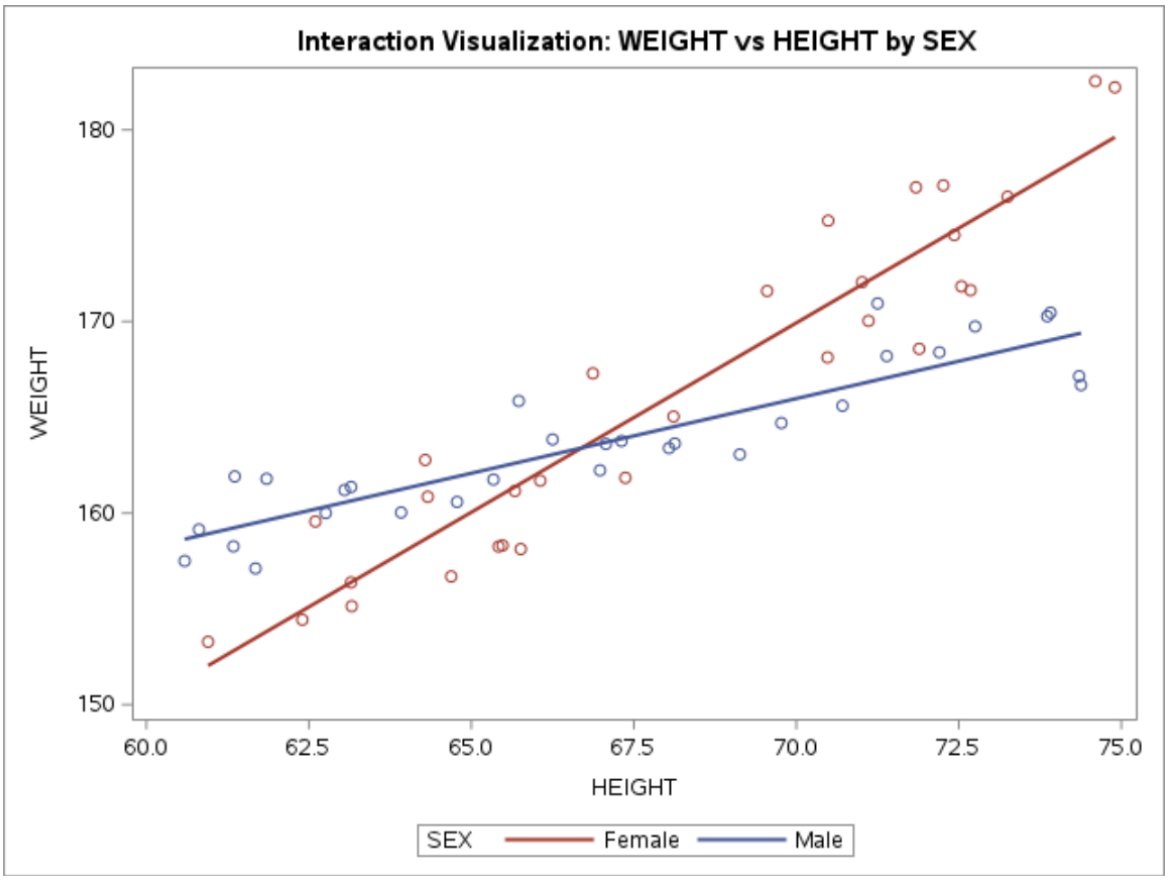
! How to visually detect interaction

For two groups:

- **Parallel regression lines** suggest **no interaction**
- **Non-parallel regression lines** suggest **interaction**
- **Crossing lines** indicate a strong interaction effect




```
PROC SGPLOT DATA=MEASUREMENT;
    TITLE "Interaction Visualization: WEIGHT vs HEIGHT by SEX";
    REG X=HEIGHT Y=WEIGHT / GROUP=SEX;
RUN;
TITLE;
```



15.3 Main-Effects Model vs Interaction Model

A **main-effects model** assumes each predictor has a constant effect across all levels of the other predictors.

For example, suppose we want to predict *weight* using a person's *sex* and *height*. A regression model with only main effects can be written as

$$\text{weight} = \beta_0 + \beta_s \text{SEX} + \beta_h \text{HEIGHT}$$

The coefficient β_h represents a weighted average of the height effects for males and females, had we modeled them separately.

Thus, the main-effects model assumes that the effect of height is the same for both males and females. However, if the effect of height differs by sex, then we need an **interaction term**.

The regression model with interaction becomes

$$\text{weight} = \beta_0 + \beta_s \text{SEX} + \beta_h \text{HEIGHT} + \beta_{sh}(\text{SEX} \times \text{HEIGHT})$$

> **Important**
>
> In a regression model that includes **interaction terms**, the coefficients of the individual variables are **no longer interpreted as overall main effects**.
> Instead, they represent the effect **when the interacting variable equals 0**.

Thus:

- β_h represents the effect of **HEIGHT when SEX = 0**
- β_s represents the effect of **SEX when HEIGHT = 0**

If we code

- SEX $= 0 \rightarrow$ male
- SEX $= 1 \rightarrow$ female

then we can construct the regression equations for **males and females** separately.

15.4 Regression Equations by Group

Substitute $SEX = 0$ into the interaction model:

$$\begin{aligned} \text{weight}_{male} &= \beta_0 + \beta_s(0) + \beta_h \text{HEIGHT} + \beta_{sh}(0)\text{HEIGHT} \\ &= \beta_0 + \beta_h \text{HEIGHT} \end{aligned}$$

Substitute $SEX = 1$:

$$\begin{aligned} \text{weight}_{female} &= \beta_0 + \beta_s(1) + \beta_h \text{HEIGHT} + \beta_{sh}(1)\text{HEIGHT} \\ &= \beta_0 + \beta_s + \beta_h \text{HEIGHT} + \beta_{sh}\text{HEIGHT} \\ &= \beta_0 + \beta_s + (\beta_h + \beta_{sh})\text{HEIGHT} \end{aligned}$$

Thus, **slope for males** is β_h and the **slope for females** is $\beta_h + \beta_{sh}$.

15.5 Interpreting the Interaction Coefficient

From the previous equations, we can see that the effect of **HEIGHT** differs for males and females.

- For males (SEX = 0), the slope of height is

$$\beta_h$$

- For females (SEX = 1), the slope of height is

$$\beta_h + \beta_{sh}$$

Thus, **HEIGHT has a different effect for males and females**.

The interaction coefficient

$$\beta_{sh}$$

represents the **difference between the height effects for males and females**.

In other words, a one-unit increase in **SEX** (changing from male to female) changes the slope of **HEIGHT** by β_{sh}.

It is important to note that this interpretation holds when the **main effects are included in the model together with the interaction term**.
If the interaction term were entered **without the corresponding main effects**, the interpretation of the coefficient would change.

Suppose the fitted model is

$$\widehat{\text{weight}} = 20 + 5\text{SEX} + 2\text{HEIGHT} + 1(\text{SEX} \times \text{HEIGHT})$$

Then:

- Male slope = 2
- Female slope = 2 + 1 = 3

Interpretation:

- For males, a one-unit increase in height increases expected weight by 2 units.
- For females, a one-unit increase in height increases expected weight by 3 units.

Thus, the effect of height is stronger for females.

15.6 SAS Implementation

15.6.1 Example dataset

```
DATA MEASUREMENT;
INPUT SEX $ HEIGHT WEIGHT;
DATALINES;
Male 67.07 163.61
Male 66.98 162.23
Male 69.13 163.06
Male 67.31 163.76
Male 66.25 163.84
Male 72.20 168.39
Male 71.39 168.19
Male 60.81 159.14
Male 64.78 160.58
Male 68.13 163.63
Male 63.15 161.36
Male 73.91 170.46
Male 74.38 166.68
Male 69.77 164.70
Male 73.86 170.27
Male 65.34 161.75
Male 72.75 169.74
Male 63.92 160.03
Male 61.85 161.79
Male 71.25 170.94
Male 61.68 157.11
Male 70.71 165.60
Male 65.73 165.85
Male 68.04 163.39
Male 62.76 160.00
Male 63.05 161.20
Male 61.34 158.26
Male 60.59 157.50
Male 74.35 167.14
Male 61.36 161.91
Female 65.42 158.25
Female 65.76 158.12
Female 74.60 182.54
Female 65.67 161.15
```

```
Female 66.06 161.69
Female 62.60 159.56
Female 72.26 177.10
Female 65.48 158.31
Female 66.87 167.29
Female 64.29 162.77
Female 71.11 170.03
Female 62.40 154.43
Female 60.95 153.28
Female 63.15 156.38
Female 72.54 171.83
Female 70.49 175.26
Female 72.68 171.63
Female 67.37 161.84
Female 68.11 165.04
Female 70.48 168.12
Female 64.33 160.85
Female 71.84 177.00
Female 69.55 171.59
Female 64.69 156.69
Female 74.90 182.22
Female 71.89 168.57
Female 73.25 176.51
Female 72.43 174.51
Female 71.01 172.05
Female 63.16 155.14
;
RUN;
```

15.6.2 Fit the interaction model in SAS

```
PROC GLM DATA=MEASUREMENT;
CLASS SEX;
MODEL WEIGHT = HEIGHT SEX HEIGHT*SEX;
STORE INTMODEL;
RUN;
```

15.6.3 Plot fitted interaction in SAS

```
PROC PLM RESTORE=INTMODEL;
EFFECTPLOT INTERACTION(X=HEIGHT SLICEBY=SEX);
RUN;
```

15.6.4 Quick visualization in SAS

```
PROC SGPLOT DATA=MEASUREMENT;
TITLE "Interaction Visualization: WEIGHT vs HEIGHT by SEX";
REG X=HEIGHT Y=WEIGHT / GROUP=SEX;
RUN;
TITLE;
```

15.7 Checking Regression Assumptions in SAS

The following SAS code fits a regression model and produces diagnostic plots for checking model assumptions.

```
PROC REG DATA=MEASUREMENT;
    TITLE "Regression and Residual Plots";
    MODEL WEIGHT = HEIGHT;
    OUTPUT OUT=MYOUT R=RESID;
RUN;

PROC UNIVARIATE DATA=MYOUT NORMAL;
    QQPLOT RESID / NORMAL(MU=EST SIGMA=EST COLOR=RED L=1);
RUN;
```

Regression and Residual Plots: WEIGHT on HEIGHT

The REG Procedure
Model: MODEL1
Dependent Variable: WEIGHT

Number of Observations Read	60
Number of Observations Used	60

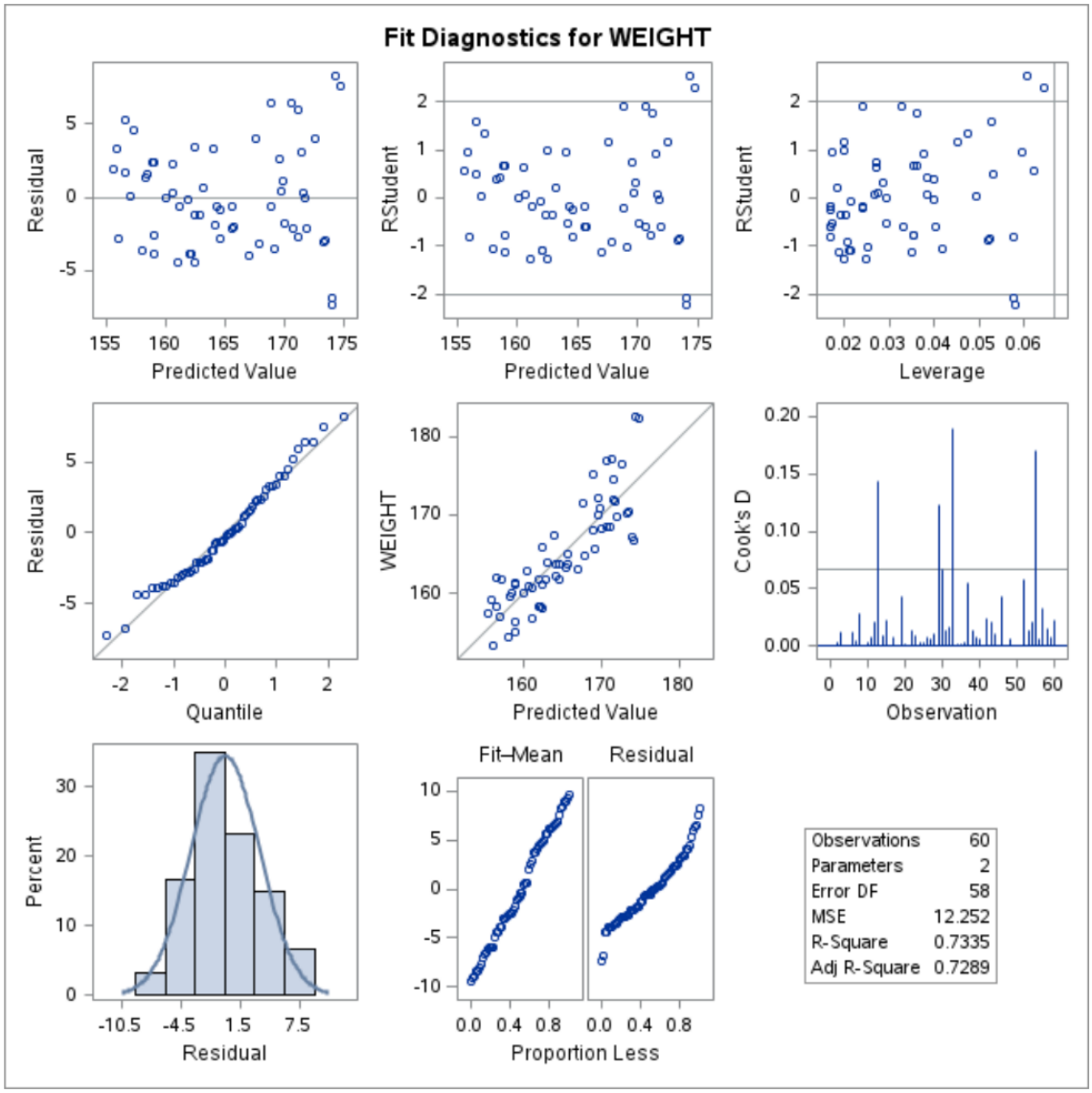
Analysis of Variance					
Source	**DF**	**Sum of Squares**	**Mean Square**	**F Value**	**Pr > F**
Model	1	1955.65136	1955.65136	159.62	<.0001
Error	58	710.61078	12.25191		
Corrected Total	59	2666.26214			

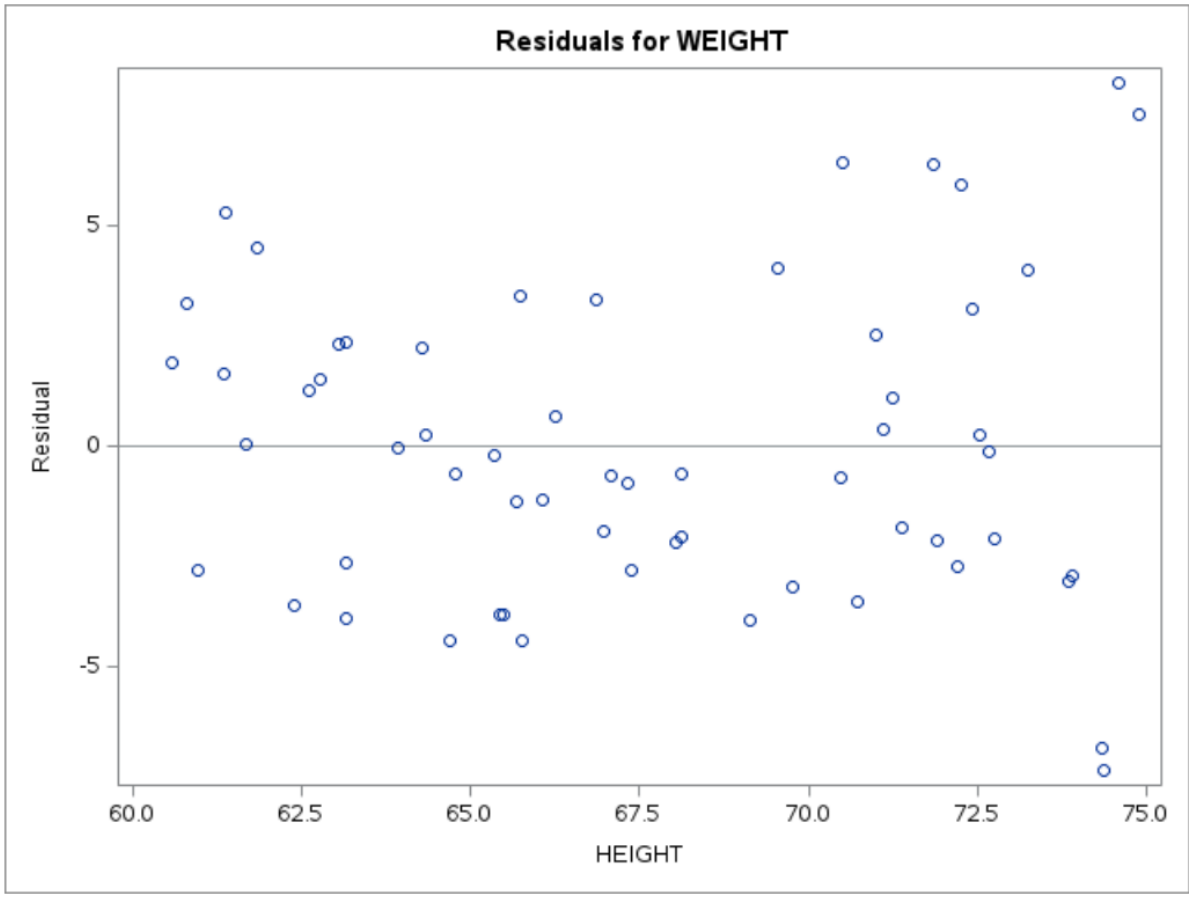
Root MSE	3.50027	**R-Square**	0.7335
Dependent Mean	165.03100	**Adj R-Sq**	0.7289
Coeff Var	2.12098		

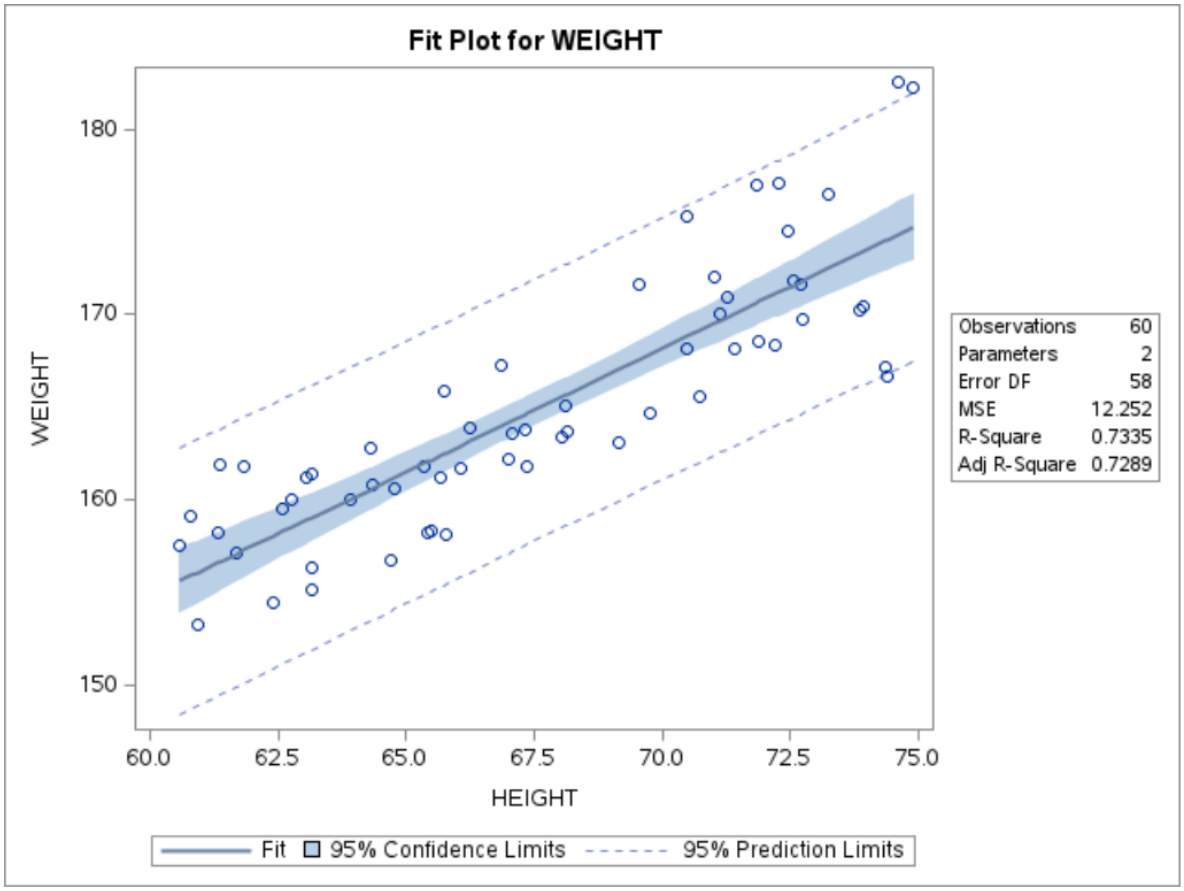
Parameter Estimates					
Variable	**DF**	**Parameter Estimate**	**Standard Error**	**t Value**	**Pr > |t|**
Intercept	1	74.69033	7.16481	10.42	<.0001
HEIGHT	1	1.33535	0.10569	12.63	<.0001

Example of Linear Regression

The REG Procedure
Model: MODEL1
Dependent Variable: WEIGHT




Residuals for WEIGHT
Residual
5
0
-5
60.0
62.5
65.0
67.5
70.0
72.5
75.0
HEIGHT


Fit Plot for WEIGHT
WEIGHT
180
170
160
150
60.0
62.5
65.0
67.5
70.0
72.5
75.0
HEIGHT
Observations 60
Parameters 2
Error DF 58
MSE 12.252
R-Square 0.7335
Adj R-Square 0.7289
Fit
95% Confidence Limits
95% Prediction Limits

Normality Check for Regression Residuals

The UNIVARIATE Procedure
Variable: HEIGHT

Moments			
N	60	**Sum Weights**	60
Mean	67.653	**Sum Observations**	4059.18
Std Deviation	4.31145117	**Variance**	18.5886112
Skewness	0.04225975	**Kurtosis**	-1.2804098
Uncorrected SS	275712.433	**Corrected SS**	1096.72806
Coeff Variation	6.37288985	**Std Error Mean**	0.55660595

Basic Statistical Measures			
Location		**Variability**	
Mean	67.65300	**Std Deviation**	4.31145
Median	67.19000	**Variance**	18.58861
Mode	63.15000	**Range**	14.31000
		Interquartile Range	7.51000

Tests for Location: Mu0=0				
Test	**Statistic**		**p Value**	
Student's t	**t**	121.5456	**Pr > |t|**	<.0001
Sign	**M**	30	**Pr >= |M|**	<.0001
Signed Rank	**S**	915	**Pr >= |S|**	<.0001

Tests for Normality				
Test	**Statistic**		**p Value**	
Shapiro-Wilk	**W**	0.944686	**Pr < W**	0.0088
Kolmogorov-Smirnov	**D**	0.110657	**Pr > D**	0.0676
Cramer-von Mises	**W-Sq**	0.149553	**Pr > W-Sq**	0.0236
Anderson-Darling	**A-Sq**	0.959271	**Pr > A-Sq**	0.0158

Quantiles (Definition 5)	
Level	**Quantile**
100% Max	74.900
99%	74.900
95%	74.365
90%	73.555
75% Q3	71.615
50% Median	67.190
25% Q1	64.105
10%	61.765
5%	61.145
1%	60.590
0% Min	60.590

Extreme Observations			
Lowest		**Highest**	
Value	**Obs**	**Value**	**Obs**
60.59	28	73.91	12
60.81	8	74.35	29
60.95	43	74.38	13
61.34	27	74.60	33
61.36	30	74.90	55

Normality Check for Regression Residuals

The UNIVARIATE Procedure
Variable: WEIGHT

Moments			
N	60	**Sum Weights**	60
Mean	165.031	**Sum Observations**	9901.86
Std Deviation	6.72241651	**Variance**	45.1908837
Skewness	0.64248991	**Kurtosis**	0.0107314
Uncorrected SS	1636780.12	**Corrected SS**	2666.26214
Coeff Variation	4.07342651	**Std Error Mean**	0.86786024

Basic Statistical Measures			
Location		**Variability**	
Mean	165.0310	**Std Deviation**	6.72242
Median	163.6200	**Variance**	45.19088
Mode	.	**Range**	29.26000
		Interquartile Range	9.58000

Tests for Location: Mu0=0				
Test	**Statistic**		**p Value**	
Student's t	**t**	190.1585	**Pr > |t|**	<.0001
Sign	**M**	30	**Pr >= |M|**	<.0001
Signed Rank	**S**	915	**Pr >= |S|**	<.0001

Tests for Normality				
Test	**Statistic**		**p Value**	
Shapiro-Wilk	**W**	0.962064	**Pr < W**	0.0594
Kolmogorov-Smirnov	**D**	0.120312	**Pr > D**	0.0296
Cramer-von Mises	**W-Sq**	0.11741	**Pr > W-Sq**	0.0673
Anderson-Darling	**A-Sq**	0.685605	**Pr > A-Sq**	0.0737

Description of the Code

- `PROC REG` fits the linear regression model relating weight to height.
- `OUTPUT OUT=MYOUT R=RESID` saves the residual values from the regression model into a new dataset called `MYOUT`.
- `PROC UNIVARIATE` is used to examine the distribution of the residuals.
- `QQPLOT RESID` produces a Q–Q plot to visually assess whether the residuals follow a normal distribution.

15.8 PROC PLM

This section relies heavily on **PROC PLM** to estimate, compare, and visualize the **conditional effects of interactions**. Before applying it to interaction models, we briefly introduce **PROC PLM** in general.

`PROC PLM` performs various analyses and plotting functions **after a regression model has been fitted**. It can be used for:

- testing **custom hypotheses** about model parameters
- computing **predicted values**
- estimating **simple effects and contrasts**
- creating **visualizations of predicted values**

Unlike most SAS procedures, **PROC PLM does not directly use a dataset as input**. Instead, it reads from an **item store**, which contains information about a previously fitted model.

An item store can be created by several SAS modeling procedures, including:

- `PROC GLM`
- `PROC GENMOD`
- `PROC LOGISTIC`
- `PROC PHREG`
- `PROC MIXED`
- `PROC GLIMMIX`

These procedures create the item store using a `STORE` statement.
Later, `PROC PLM` accesses this stored model using the `RESTORE=` option.

15.8.1 Example with PROC PLM

```
DATA EXERCISE;
    INPUT HOURS EFFORT $ LOSS;
    DATALINES;
1 Low 3.0
2 Low 4.0
3 Low 5.0
4 Low 6.0
5 Low 7.0
1 High 2.0
2 High 4.0
3 High 7.0
4 High 11.0
5 High 16.0
;
RUN;
```

```
PROC GLM DATA=EXERCISE;
    CLASS EFFORT;
    MODEL LOSS = HOURS|EFFORT;
    STORE CONTCONT;
RUN;

PROC PLM RESTORE=CONTCONT;
    EFFECTPLOT SLICEFIT(X=HOURS SLICEBY=EFFORT);
RUN;
```

Key Takeaways

- Interaction means the effect of one variable depends on another
- Main effects are no longer “global effects”
- Always interpret **simple slopes**
- Visualization is the easiest way to detect interaction

16 Linear Mixed Effect Models I

Learning Objectives

By the end of this lecture, you should be able to:

1. Understand why **ordinary linear regression may fail** for correlated data
2. Distinguish between **fixed effects** and **random effects**
3. Recognize **clustered** and **longitudinal** data structures
4. Write down the basic form of a **linear mixed effect model**
5. Fit a simple mixed model in **SAS using `PROC MIXED`**

16.1 Introduction

In previous lectures, we studied linear regression models of the form

$$Y_i = \beta_0 + \beta_1 X_i + \epsilon_i, \qquad \epsilon_i \sim N(0, \sigma^2).$$

A key assumption of this model is that the observations are **independent**.

However, in many real applications, this assumption is not reasonable. Observations often come in groups, or the same subject is measured repeatedly. In such settings, observations from the same group tend to be more similar to each other than to observations from other groups.

16.2 Two Common Settings Where Independence Fails

Clustered data arise when observations are grouped into natural clusters.

Examples include:

- students within schools
- patients within hospitals

- calves within farms

Observations within the same cluster tend to be similar. So for subjects j and k in the same cluster i,

$$\text{Corr}(Y_{ij}, Y_{ik}) \neq 0.$$

Longitudinal data arise when the same subject is measured repeatedly over time or across repeated trials.

Examples include:

- repeated test scores from the same student
- monthly measurements from the same patient
- repeated performance scores from the same worker

For subject i, repeated measurements such as

$$Y_{i1}, Y_{i2}, \ldots, Y_{iT}$$

are typically correlated.

16.3 Why Ordinary Regression Is Not Enough

If we ignore clustering or repeated measurements and fit an ordinary regression model, we often underestimate the variability in the data. As a result:

- p-values may be misleading
- confidence intervals may be too narrow
- conclusions may be overly optimistic

So the issue is not only the mean structure. We also need to model the **dependence structure** in the data.

Ordinary regression focuses on the **average relationship**.

Mixed models do two things at the same time:

- model the **average relationship**
- model the **within-group dependence**

16.4 Fixed Effects and Random Effects

To handle correlated data, we extend regression by adding **random effects**.

Fixed Effects

Fixed effects describe **population-level effects** that are of direct interest.

Examples:

- treatment group
- sex
- age
- trial number

If we write

$$\beta_1 = \text{effect of treatment},$$

then β_1 is a fixed effect parameter.

Random Effects

Random effects describe **group-specific deviations** from the overall population pattern.

For example, if subjects are measured repeatedly, each subject may have their own baseline level. We can write this as

$$u_i \sim N(0, \sigma_u^2),$$

where u_i is the random effect for subject i.

Intuition

- Fixed effects tell us about the **average relationship**
- Random effects allow each cluster or subject to have their **own deviation**

This leads to a model that combines both kinds of effects. That is why it is called a **mixed** model.

16.5 Linear Mixed Effected model

The simplest linear mixed effect model is the **random intercept model**, which is defined as

$$Y_{ij} = \beta_0 + \beta_1 X_{ij} + u_i + \epsilon_{ij},$$

where:

- β_0, β_1 are fixed effects
- u_i is a random effect for cluster or subject i
- ϵ_{ij} is the residual error

We usually assume

$$u_i \sim N(0, \sigma_u^2), \qquad \epsilon_{ij} \sim N(0, \sigma^2),$$

and that u_i and ϵ_{ij} are independent.

In this model:

- the fixed effects β_0, β_1 describe the overall trend
- the random effect u_i allows subject i to deviate from that overall trend
- the residual error ϵ_{ij} captures the remaining unexplained variation

16.5.1 Retionale behind of correlation in this model

Two observations from the same subject share the same random effect u_i. That is exactly what creates correlation.

For the same subject or cluster i,

$$Y_{ij} = \beta_0 + \beta_1 X_{ij} + u_i + \epsilon_{ij},$$

$$Y_{ik} = \beta_0 + \beta_1 X_{ik} + u_i + \epsilon_{ik}.$$

Because both contain the same u_i, they are correlated.

This is the main idea students should remember:

> the shared random effect makes observations from the same subject or cluster similar.

16.5.2 Variance Structure

For the random intercept model,

$$\text{Var}(Y_{ij}) = \sigma_u^2 + \sigma^2$$

and for two observations in the same cluster,

$$\text{Cov}(Y_{ij}, Y_{ik}) = \sigma_u^2.$$

Therefore,

$$\text{Corr}(Y_{ij}, Y_{ik}) = \frac{\sigma_u^2}{\sigma_u^2 + \sigma^2}.$$

> **Key Interpretation**
>
> - σ_u^2 measures **between-subject** or **between-cluster** variability
> - σ^2 measures **within-subject** variability
> - If σ_u^2 is large, observations within the same subject are strongly correlated

16.6 SAS Computing example Dataset

To illustrate the idea, consider the following repeated-measures dataset.

```
DATA performance;
    INPUT id $ age group $ trial score;
    DATALINES;
SY 34 A 1 14.3
SY 34 A 2 21.4
SY 34 A 3 27.6
SY 34 A 4 31.1
SY 34 A 5 33.2
WL 33 A 1 13.2
WL 33 A 2 21.4
WL 33 A 3 23.3
WL 33 A 4 30.0
WL 33 A 5 38.6
ZN 43 B 1 15.9
```

```
ZN 43 B 2 23.4
ZN 43 B 3 22.0
ZN 43 B 4 29.0
ZN 43 B 5 33.6
;
RUN;
```

In this dataset:

- `score` is the response variable
- `trial` is a fixed effect
- `group` is a fixed effect
- `id` identifies the subject

Because each subject appears multiple times, the observations are not independent.

16.6.1 Step 1: A Model That Ignores Correlation

If we fit ordinary regression,

```
PROC REG DATA=performance;
    MODEL score = trial;
RUN;
QUIT;
```

this assumes all observations are independent.

That is not appropriate here, because repeated scores from the same person are likely correlated.

The REG Procedure
Model: MODEL1
Dependent Variable: score

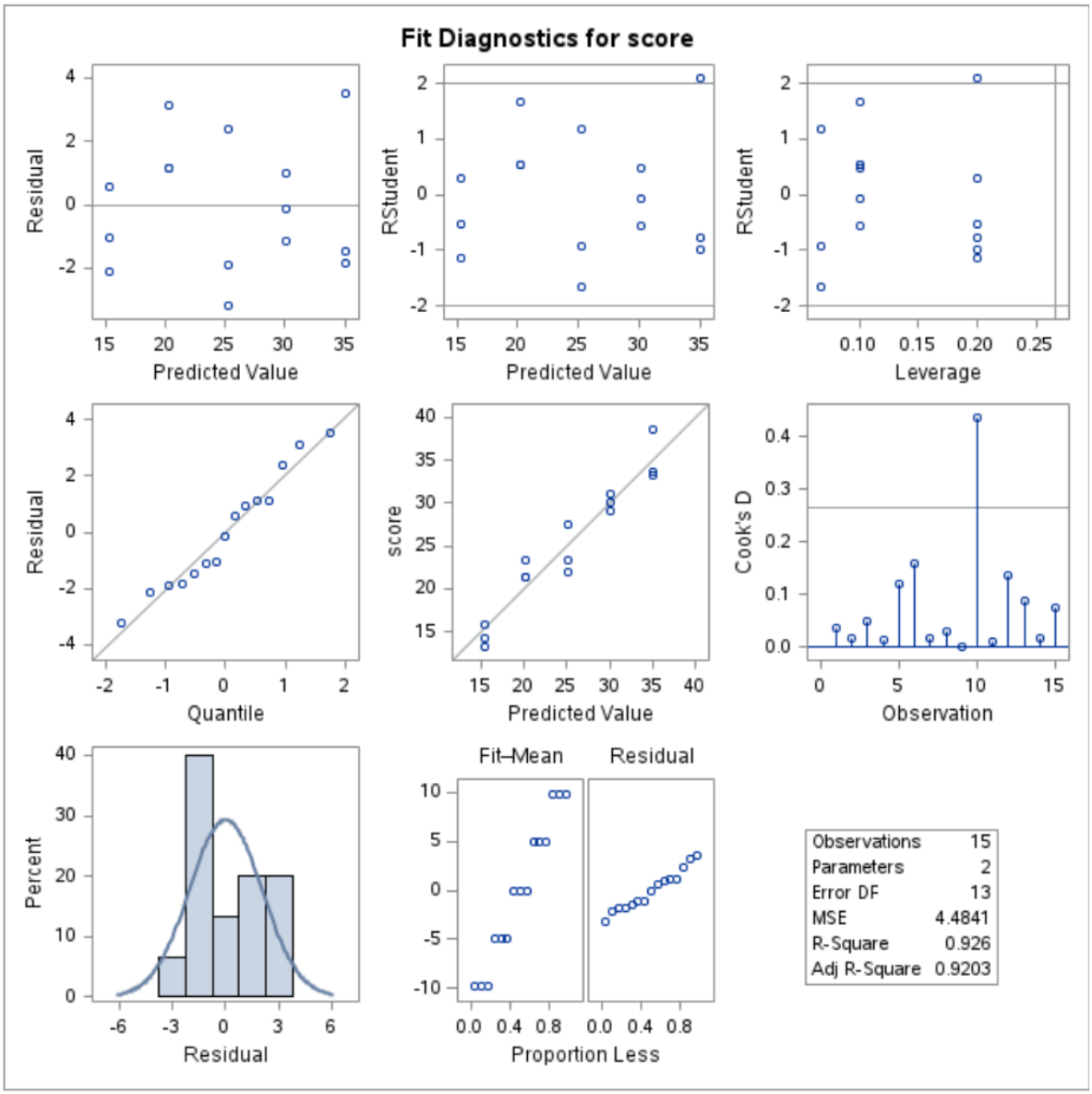
Number of Observations Read	15
Number of Observations Used	15

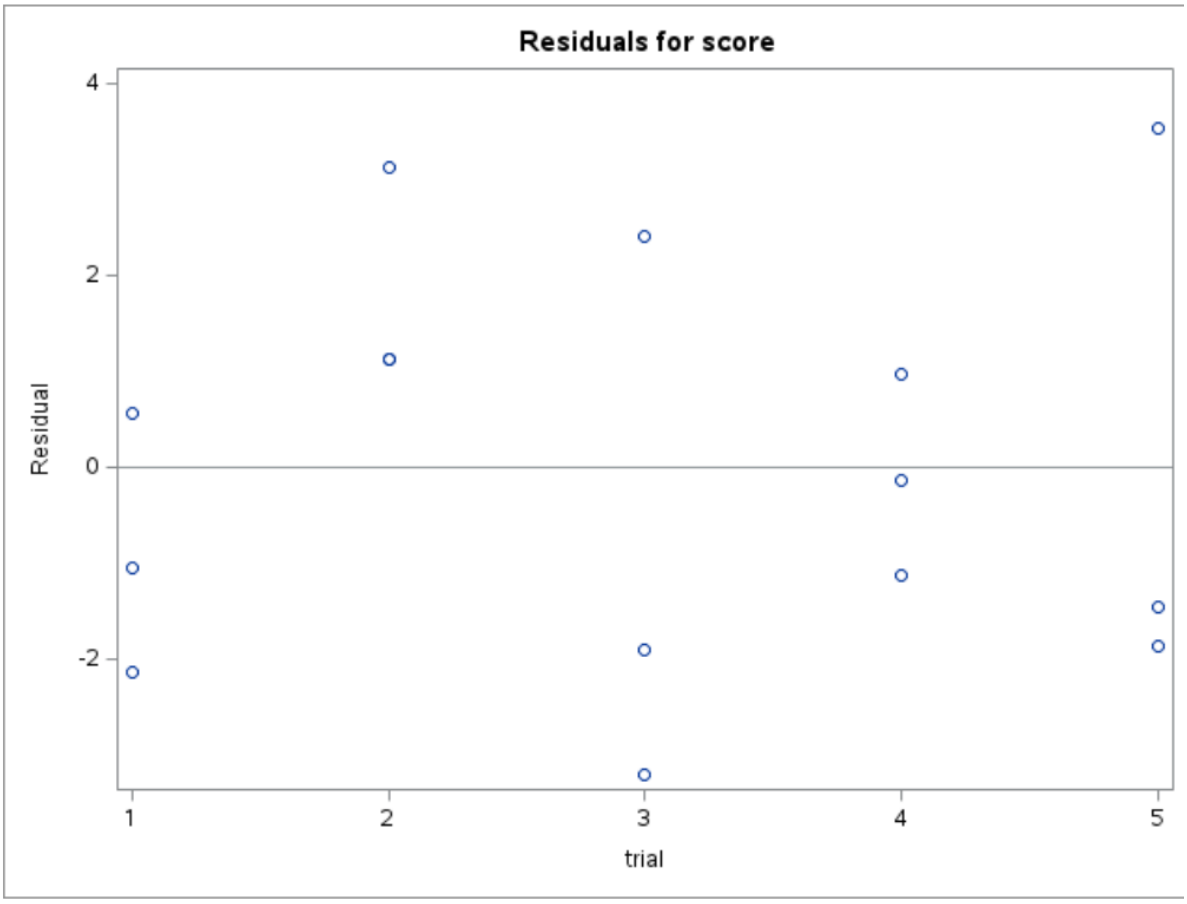
Analysis of Variance					
Source	**DF**	**Sum of Squares**	**Mean Square**	**F Value**	**Pr > F**
Model	1	729.14700	729.14700	162.61	<.0001
Error	13	58.29300	4.48408		
Corrected Total	14	787.44000			

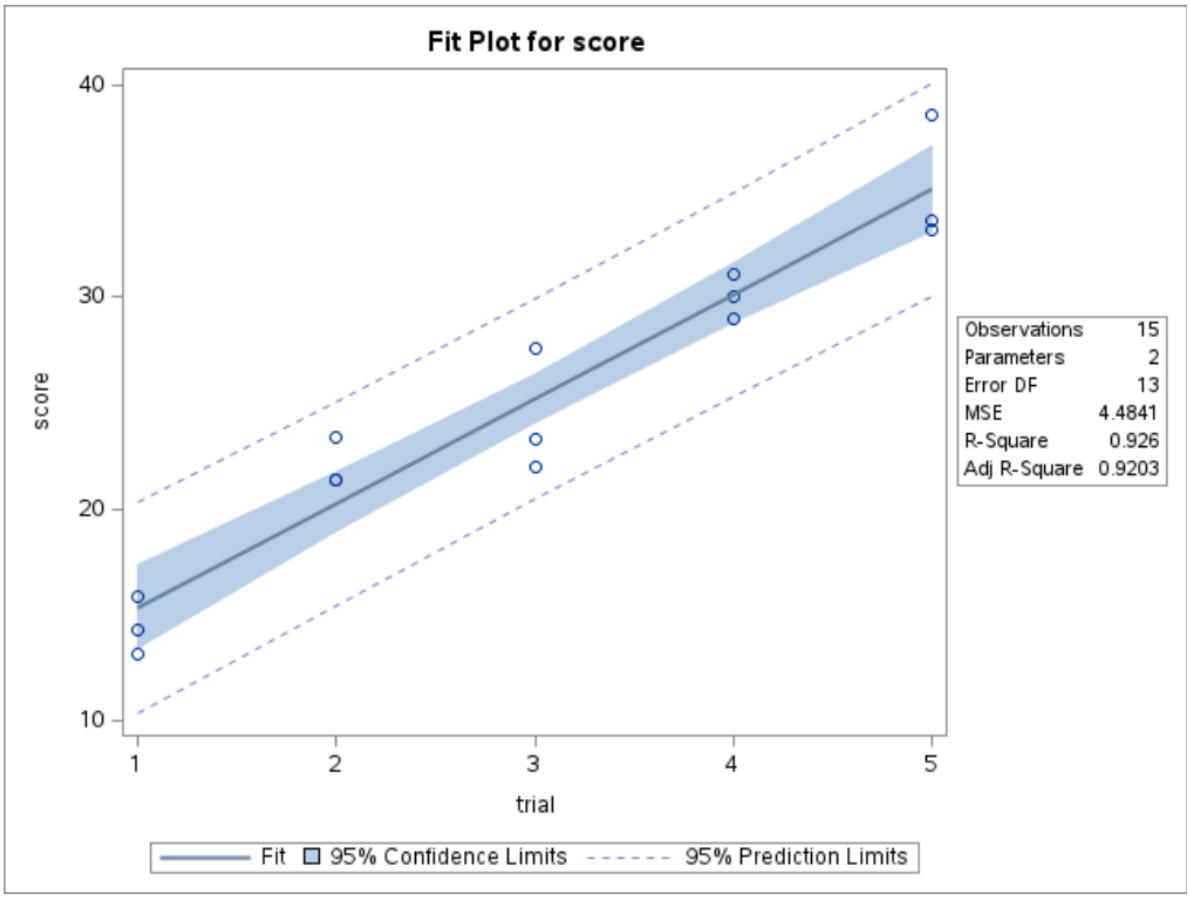
Root MSE	2.11756	**R-Square**	0.9260
Dependent Mean	25.20000	**Adj R-Sq**	0.9203
Coeff Var	8.40303		

Parameter Estimates					
Variable	**DF**	**Parameter Estimate**	**Standard Error**	**t Value**	**Pr > |t|**
Intercept	1	10.41000	1.28225	8.12	<.0001
trial	1	4.93000	0.38661	12.75	<.0001

The REG Procedure
Model: MODEL1
Dependent Variable: score




Residuals for score
Residual
4
2
0
-2
1
2
3
4
5
trial



> **Warning**
>
> `PROC REG` is not designed for repeated-measures data with within-subject dependence. It may still produce estimates, but the inference can be misleading because the independence assumption is violated.

16.6.2 Step 2: Random Intercept Model

A natural first mixed model is the **random intercept model**.

```
PROC MIXED DATA=performance;
    CLASS id;
    MODEL score = trial;
    RANDOM INTERCEPT / SUBJECT=id;
RUN;
```

The Mixed Procedure

Model Information	
Data Set	WORK.PERFORMANCE
Dependent Variable	score
Covariance Structure	Variance Components
Subject Effect	id
Estimation Method	REML
Residual Variance Method	Profile
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Containment

Class Level Information		
Class	**Levels**	**Values**
id	3	SY WL ZN

Dimensions	
Covariance Parameters	2
Columns in X	2
Columns in Z per Subject	1
Subjects	3
Max Obs per Subject	5

Number of Observations	
Number of Observations Read	15
Number of Observations Used	15
Number of Observations Not Used	0

Iteration History		
Iteration	**Evaluations**	**-2 Res Log**
0	1	62.508
1	1	62.508

Convergence criteria r

Estimated G matrix is not po

Covariance Parameter Es		
Cov Parm	**Subject**	
Intercept	id	
Residual		

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

Type 3 Tests of Fixed E			
Effect	**Num DF**	**Den DF**	**F**
trial	1	11	

This corresponds to the model

$$Y_{ij} = \beta_0 + \beta_1 \, trial_{ij} + u_i + \epsilon_{ij}.$$

Interpretation

- β_0 is the overall intercept
- β_1 is the average effect of trial
- u_i allows each subject to have their own baseline level

So this model says:

> everyone follows the same overall slope, but each subject may start from a different intercept.

In the code above:

- `CLASS id;` tells SAS that `id` is a grouping variable
- `MODEL score = trial;` specifies the fixed-effects part of the model
- `RANDOM INTERCEPT / SUBJECT=id;` tells SAS to fit a subject-specific random intercept

This is usually the first mixed model to try when each subject is observed multiple times.

16.6.3 Step 3: Add Another Fixed Effect

We can also include `group` as a fixed effect.

```
PROC MIXED DATA=performance;
    CLASS id group;
    MODEL score = trial group;
    RANDOM INTERCEPT / SUBJECT=id;
RUN;
```

This corresponds to

$$Y_{ij} = \beta_0 + \beta_1 \, trial_{ij} + \beta_2 \, group_{ij} + u_i + \epsilon_{ij}.$$

Now the model includes:

- an overall effect of trial
- an overall difference between groups
- a subject-specific random intercept

The Mixed Procedure

Model Information	
Data Set	WORK.PERFORMANCE
Dependent Variable	score
Covariance Structure	Variance Components
Subject Effect	id
Estimation Method	REML
Residual Variance Method	Profile
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Containment

Class Level Information		
Class	**Levels**	**Values**
id	3	SY WL ZN
group	2	A B

Dimensions	
Covariance Parameters	2
Columns in X	4
Columns in Z per Subject	1
Subjects	3
Max Obs per Subject	5

Number of Observations	
Number of Observations Read	15
Number of Observations Used	15
Number of Observations Not Used	0

Iteration Histo		
Iteration	**Evaluations**	**-2 Res L**
0	1	60.0
1	1	60.0

Convergence criteri

Estimated G matrix is not

Covariance Parameter	
Cov Parm	**Subject**
Intercept	id
Residual	

Fit Statistics
-2 Res Log Likelihood
AIC (Smaller is Better
AICC (Smaller is Bett
BIC (Smaller is Better

Type 3 Tests of Fixed		
Effect	**Num DF**	**Den DF**
trial	1	11
group	1	11

Intuition

- `trial` asks whether scores tend to change over trials
- `group` asks whether group A and group B differ on average
- `id` as a random effect accounts for repeated measurements on the same subject

16.6.4 Step 4: Random Slope Model

A more flexible model allows the effect of trial to vary by subject.

```
PROC MIXED DATA=performance;
    CLASS id;
    MODEL score = trial;
    RANDOM INTERCEPT trial / SUBJECT=id;
RUN;
```

The Mixed Procedure

Model Information	
Data Set	WORK.PERFORMANCE
Dependent Variable	score
Covariance Structure	Variance Components
Subject Effect	id
Estimation Method	REML
Residual Variance Method	Profile
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Containment

Class Level Information		
Class	**Levels**	**Values**
id	3	SY WL ZN

Dimensions	
Covariance Parameters	3
Columns in X	2
Columns in Z per Subject	2
Subjects	3
Max Obs per Subject	5

Number of Observations	
Number of Observations Read	15
Number of Observations Used	15
Number of Observations Not Used	0

Iteration History		
Iteration	**Evaluations**	**-2 Res Log**
0	1	62.508
1	4	62.508

Convergence criteria

Estimated G matrix is not p

Covariance Parameter E		
Cov Parm	**Subject**	
Intercept	id	
trial	id	
Residual		

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

Type 3 Tests of Fixed E			
Effect	**Num DF**	**Den DF**	**F**
trial	1	2	

This corresponds to

$$Y_{ij} = \beta_0 + \beta_1 trial_{ij} + u_{0i} + u_{1i} trial_{ij} + \epsilon_{ij}.$$

Interpretation

- u_{0i} allows subject i to have their own intercept
- u_{1i} allows subject i to have their own slope

This means different subjects can improve at different rates across trials.

- A **random intercept model** says subjects start at different levels
- A **random slope model** says subjects may also change at different rates

16.7 Conceptual Visualization

A helpful way to think about the difference between regression and mixed models is:

- ordinary regression: one fitted line for everyone
- random intercept model: many subject-specific lines with different intercepts
- random slope model: many subject-specific lines with different intercepts and different slopes

This is one reason mixed models are so useful for repeated-measures data.

16.8 In-Class Questions

Which variable is most appropriate as a random effect?

- (A) `trial`
- (B) `score`
- (C) `id`
- (D) `group`

Why include a random effect?

- (A) To increase sample size
- (B) To model dependence
- (C) To remove variables
- (D) To normalize the data

Ignoring clustering most directly leads to:

- (A) Correct inference
- (B) Smaller residuals
- (C) Incorrect standard errors
- (D) Better prediction

16.9 Summary of this class

This first lecture on linear mixed models is mainly about **intuition**.

The big picture is:

1. Ordinary regression assumes independence
2. Clustered and longitudinal data violate this assumption
3. Random effects are used to model within-group dependence
4. Mixed models combine:
    - fixed effects for the mean structure
    - random effects for the dependence structure

In the next lecture, we will go further into:

- more detailed interpretation of `PROC MIXED` output
- covariance structures
- repeated-measures formulations
- model comparison and diagnostics

> **Key Takeaways**
>
> - Mixed models are used when observations are **correlated**
> - Two common settings are:
>   - **clustered data**
>   - **longitudinal data**
> - Fixed effects describe **population-level effects**
> - Random effects describe **subject- or cluster-specific deviations**
> - A random effect creates correlation among observations from the same subject or cluster
> - In SAS, the most basic mixed model can be fitted using `PROC MIXED`

17 Linear Mixed Effect Models II

Learning Objectives

By the end of this lecture, you should be able to:

1. Implement **linear mixed models in SAS using `PROC MIXED`**
2. Understand the roles of the **CLASS**, **MODEL**, and **RANDOM** statements
3. Interpret key output tables, including:
   - covariance parameter estimates
   - tests of fixed effects
   - fit statistics
4. Compare **fixed-effects models** and **mixed-effects models**
5. Explain why modeling correlation can change standard errors, p-values, and conclusions

17.1 Introduction

In the previous lecture, we introduced the conceptual framework of linear mixed models:

$$Y = X\beta + Z\gamma + \epsilon,$$

where:

- $X\beta$ represents the **fixed-effects part** of the model
- $Z\gamma$ represents the **random-effects part** of the model
- ϵ is the residual error

In this lecture, we focus on **how to fit and interpret these models in SAS** using `PROC MIXED`.

Main Goal of This Lecture

Last time, the main question was:

> **Why do we need mixed models?**

This time, the main question is:

> **How do we fit, read, and interpret mixed models in SAS?**

17.2 Why `PROC MIXED`?

Recall the roles of some common SAS procedures:

- `PROC REG` fits ordinary regression models and assumes **independent observations**
- `PROC GLM` handles fixed-effects models, including ANOVA-style models with categorical predictors
- `PROC MIXED` extends these ideas to data with **dependence, clustering, or repeated measurements**

Typical use cases include:

- clustered data, such as students within schools or patients within hospitals
- repeated-measures data, such as multiple observations on the same subject

A General Template for `PROC MIXED`

A basic mixed model in SAS has the form

```
PROC MIXED DATA=dataset;
    CLASS categorical_variables;
    MODEL response = fixed_effects;
    RANDOM random_effects / SUBJECT=cluster;
RUN;
```

How to read this syntax?

- `CLASS` tells SAS which variables are categorical
- `MODEL` specifies the **fixed-effects part**
- `RANDOM` specifies the **random-effects part**
- `SUBJECT=` identifies the clustering variable

A mixed model has **two parts**:

1. A **fixed-effects part**, which describes the average relationship

2. A **random-effects part**, which accounts for dependence among observations from the same subject or cluster

17.3 Example Dataset: Family Heights

To keep the ideas concrete, consider the following family-height data.

```
DATA heights;
    INPUT Family Gender $ Height @@;
    DATALINES;
1 F 67 1 F 66 1 F 64 1 M 71 1 M 72
2 F 63 2 F 67 2 M 69 2 M 68 2 M 70
3 F 63 3 M 64
4 F 67 4 F 66 4 M 67 4 M 67 4 M 69
;
RUN;
```

In this dataset:

- `Height` is the response variable
- `Gender` is a categorical explanatory variable
- `Family` identifies the cluster

So the natural grouping structure is:

> observations within the same family may be correlated

This means a standard fixed-effects model that assumes independence may not be appropriate.

17.3.1 Step 1: A Fixed-Effects Model

We begin with a model that treats family as a fixed effect.

```
PROC MIXED DATA=heights;
    CLASS Family Gender;
    MODEL Height = Gender Family Family*Gender;
RUN;
```

What this model does

This model is similar to a fixed-effects ANOVA model:

- `Gender` is treated as a fixed effect
- `Family` is also treated as a fixed effect
- `Family*Gender` allows the gender difference to vary across families

Interpretation

This approach is useful if:

- the specific families in the dataset are the only ones of interest
- you do not want to generalize beyond these observed families

However, this model does **not** use the idea that families may be viewed as a random sample from a larger population of families.

> **Important**
>
> If family is really a random grouping factor, treating it only as fixed may not be the best way to represent the data structure.

17.3.2 Step 2: A Mixed-Effects Model

Now we fit a mixed model that treats family-related effects as random.

```
PROC MIXED DATA=heights;
    CLASS Family Gender;
    MODEL Height = Gender;
    RANDOM Family Family*Gender;
RUN;
```

This model can be written conceptually as

$$Y_{ij} = \beta_0 + \beta_1 \,\text{Gender}_{ij} + u_{\text{Family}} + u_{\text{Family}\times\text{Gender}} + \epsilon_{ij}.$$

Interpretation

- β_0 and β_1 are **fixed effects**
- u_{Family} allows different families to have different baseline levels
- $u_{\text{Family}\times\text{Gender}}$ allows the gender effect to vary by family
- ϵ_{ij} is the residual error

This model explicitly acknowledges that observations from the same family may be correlated.

Why the Mixed Model Is Different

The mixed model is different from the fixed-effects model because it recognizes that:

- families are a source of variability
- part of the total variation in height comes from **between-family differences**
- part of the total variation comes from **within-family differences**

That is why the mixed model gives more realistic standard errors and can lead to different conclusions.

The Most Important Output Tables

When we first use `PROC MIXED`, the output can feel overwhelming. The goal is **not** to read every table in detail. Instead, focus on a few key tables.

17.4 1. Covariance Parameter Estimates

This table reports estimated variance components for the random part of the model.

Typical rows include:

- `Family`
- `Family*Gender`
- `Residual`

17.4.1 How to interpret this table

- A larger variance component means that source contributes more variability
- If the `Family` variance is large, families differ substantially in baseline height
- If the `Family*Gender` variance is large, the gender difference varies across families
- The residual variance reflects remaining unexplained within-family variation

> **Key Interpretation**
>
> Covariance parameter estimates tell us **where the variability comes from**.

17.5 2. Tests of Fixed Effects

This table tests the fixed-effects part of the model.

In the example above, the most important fixed effect is usually:

- `Gender`

This table answers questions such as:

- Is there evidence of an overall gender difference in height?
- After accounting for family-level variation, does gender still matter?

17.5.1 Why this matters

The p-values here can differ from those in a fixed-effects model because the mixed model adjusts for the dependence structure in the data.

17.6 3. Fit Statistics

This table often includes quantities such as:

- -2 log likelihood
- AIC
- BIC

These are useful when comparing competing models.

17.6.1 General interpretation

- Smaller values often indicate better fit, especially when comparing models fit to the same dataset
- AIC and BIC are particularly useful when deciding whether adding random effects improves the model

If two models are fitted to the same data, and one has clearly smaller AIC and BIC, that model is usually preferred from a model-fit perspective.

17.7 Why Modeling Correlation Changes Inference

One of the biggest lessons from mixed models is this:

> if you ignore correlation, your inference may change.

In practice, adding random effects can change:

- standard errors
- test statistics
- p-values
- confidence intervals

This does **not** necessarily mean the fixed-effect estimate changes dramatically. Sometimes the estimate is similar, but the uncertainty around it changes.

That is one reason mixed models are so important.

17.8 Comparing a Fixed-Effects Model and a Mixed Model

A useful way to learn mixed models is to compare the two approaches.

Fixed-effects model

```
PROC MIXED DATA=heights;
    CLASS Family Gender;
    MODEL Height = Gender Family Family*Gender;
RUN;
```

Mixed-effects model

```
PROC MIXED DATA=heights;
    CLASS Family Gender;
    MODEL Height = Gender;
    RANDOM Family Family*Gender;
RUN;
```

Conceptual comparison

- The fixed-effects model treats families as specific categories of direct interest
- The mixed-effects model treats family as a source of random variability
- The mixed-effects model is more natural when families are thought of as sampled from a larger population

17.9 Another Example: Repeated Measures

Now return to the repeated-measures dataset from the previous lecture.

```
DATA performance;
    INPUT id $ age group $ trial score;
    DATALINES;
SY 34 A 1 14.3
SY 34 A 2 21.4
SY 34 A 3 27.6
SY 34 A 4 31.1
SY 34 A 5 33.2
WL 33 A 1 13.2
WL 33 A 2 21.4
WL 33 A 3 23.3
WL 33 A 4 30.0
WL 33 A 5 38.6
ZN 43 B 1 15.9
ZN 43 B 2 23.4
ZN 43 B 3 22.0
ZN 43 B 4 29.0
ZN 43 B 5 33.6
;
RUN;
```

A basic random intercept model is

```
PROC MIXED DATA=performance;
    CLASS id group;
    MODEL score = trial group;
    RANDOM INTERCEPT / SUBJECT=id;
RUN;
```

Intrepreation of the model

- `trial` measures the average trend across repeated trials
- `group` compares group A and group B
- `RANDOM INTERCEPT / SUBJECT=id;` allows each subject to have their own baseline

17.9.1 What output should students focus on?

Again, focus on three things:

1. covariance parameter estimates
2. tests of fixed effects
3. fit statistics

17.9.2 How to Explain Results in Words?

Students often know how to read p-values but struggle to explain mixed-model results in plain language. Here are some useful sentence patterns.

For a fixed effect

- "After accounting for family-level variability, there is evidence of a gender difference in height."
- "After accounting for repeated measurements within subjects, score tends to increase with trial."

For a random effect

- "There is substantial between-family variability in baseline height."
- "There is meaningful variation across subjects in their baseline scores."

17.9.3 For model comparison

- "The mixed model fits better than the fixed-effects model based on smaller fit statistics."

17.10 In-Class Questions

In the code below, which part specifies the fixed effects?

```
PROC MIXED DATA=performance;
    CLASS id group;
    MODEL score = trial group;
    RANDOM INTERCEPT / SUBJECT=id;
RUN;
```

- (A) `CLASS id group;`

- (B) `MODEL score = trial group;`
- (C) `RANDOM INTERCEPT / SUBJECT=id;`
- (D) `PROC MIXED DATA=performance;`

What does the `RANDOM` statement do?

- (A) It creates a new variable
- (B) It defines the variance structure due to clustering
- (C) It removes dependence from the data
- (D) It converts a continuous variable into a categorical variable

If a mixed model and a fixed-effects model give different p-values, which is the most likely explanation?

- (A) The raw data changed
- (B) The mixed model accounts for correlation
- (C) The response variable changed units
- (D) SAS made an error

17.11 What to Remember from This Lecture

The main lesson of this second lecture is:

> `PROC MIXED` allows us to estimate both the fixed-effects part and the random-effects part of a model.

When you read the output, focus on:

1. the fixed effects
2. the random-effect variance components
3. the fit statistics

These three pieces usually tell the main story.

Key Takeaways

- `PROC MIXED` is used to fit linear mixed models in SAS
- `CLASS` identifies categorical variables
- `MODEL` specifies the fixed-effects part
- `RANDOM` specifies the random-effects part
- Covariance parameter estimates show how variability is split across sources

- Tests of fixed effects tell us whether predictors are associated with the response
- Modeling correlation can change standard errors, p-values, and conclusions

18 Model Selection in SAS

Learning Objectives

By the end of this lecture, you should be able to:

1. Understand the main goal of **model selection**
2. Explain the **bias-variance trade-off**
3. Understand why **training error is not enough**
4. Use **cross-validation** to assess prediction performance
5. Interpret **AIC** and **BIC** as model comparison tools
6. Implement model selection in SAS using `PROC GLMSELECT`

18.1 Introduction

In many modern data problems, we often have several possible predictors.

A natural practical question is:

> We have many variables, but which ones should we use in the model?

More generally, model selection includes questions such as:

- which predictors to include
- whether to use a simpler or more complex model
- whether interaction terms are needed
- how to choose tuning parameters or model forms

The goal of model selection is **not** simply to fit the largest model possible. Instead, we want a model that is:

- reasonably simple and interpretable
- accurate for prediction on new data

18.2 Why Model Selection Matters

A model that is too simple may miss important patterns in the data.

A model that is too complex may fit the training data very well, but perform poorly on new data.

So model selection is really about finding a model that **generalizes well**.

> **i** Main Idea
>
> A good model balances:
>
> - **fit**
> - **simplicity**
> - **prediction performance**

18.3 Bias-Variance Trade-Off

A central idea in model selection is the **bias-variance trade-off**.

Suppose

$$Y = f(X) + \epsilon, \qquad \epsilon \sim N(0, \sigma_\epsilon^2).$$

Then the expected prediction error at a point (x) can be decomposed as

$$E[(Y - \hat{f}(x))^2] = \text{Bias}^2 + \text{Variance} + \text{Irreducible Error}.$$

18.3.1 Intuition

- A **simple model** often has **high bias** but **low variance**
- A **complex model** often has **low bias** but **high variance**

As model complexity increases:

- bias usually decreases
- variance usually increases

So the total prediction error is often minimized at some intermediate level of complexity.

- If a model is **too simple**, it may underfit the data

- If a model is **too flexible**, it may overfit the data
- The best model is often somewhere in between

18.4 Why Training Error Is Not Enough

Suppose we fit a model using the training data.

Then the **training MSE** often underestimates the true prediction error, because the model has already seen those data.

So if our goal is prediction, we need a better way to estimate how well the model performs on new observations.

This motivates **cross-validation**.

18.5 Cross-Validation

The goal of cross-validation is to estimate how well a model predicts new data.

18.5.1 Basic idea

1. Split the data into two parts:
    - a pseudo-training set
    - a pseudo-test set
2. Fit the model on the pseudo-training set
3. Evaluate prediction error on the pseudo-test set

This gives an estimate of test MSE.

18.6 K-Fold Cross-Validation

In **K-fold cross-validation**:

1. Randomly divide the data into (K) folds
2. For each fold (k):
    - use fold (k) as the test set
    - use the remaining (K-1) folds as the training set
3. Compute the prediction error for each fold

4. Average the errors

The cross-validation estimate is

$$\widehat{\mathrm{MSE}}_{CV} = \frac{1}{K}\sum_{k=1}^{K}\widehat{\mathrm{MSE}}_k.$$

If (K=n), this becomes **leave-one-out cross-validation (LOOCV)**.

A single train/test split can be unstable.

K-fold cross-validation improves stability by averaging prediction error across multiple splits.

18.7 Information Criteria: AIC and BIC

Besides prediction-based approaches, model selection can also be based on **information criteria**.

Two common criteria are:

- Akaike's Information Criterion (AIC)
- Bayesian Information Criterion (BIC)

18.7.1 Main idea

Both AIC and BIC try to balance:

- goodness of fit
- model complexity

A model with more parameters may fit the data better, but it is also penalized for being more complex.

18.7.2 Interpretation

- Smaller **AIC** is better
- Smaller **BIC** is better

In general:

- AIC tends to be a bit more flexible
- BIC tends to penalize complexity more strongly

> **Practical Interpretation**
>
> If two models are fitted to the same data, the one with the smaller AIC or BIC is usually preferred.

18.8 `PROC GLMSELECT`

In SAS, one important procedure for model selection is `PROC GLMSELECT`.

It is designed for model selection in general linear models, especially when there are many candidate effects.

18.8.1 Why `PROC GLMSELECT` is useful

It provides flexibility for:

- large numbers of candidate predictors
- interaction terms
- classification effects
- training / validation / testing partitions
- cross-validation
- different selection criteria and stopping rules

18.8.2 Common selection methods

Some common methods include:

- `FORWARD`
- `BACKWARD`
- `STEPWISE`
- `LASSO`

18.9 Example: Model Selection with a Toy Dataset

To make the ideas concrete, consider a small dataset with a practical meaning.

Suppose we want to predict a student's **final exam score** using:

- `hours` = weekly study hours
- `attendance` = class attendance percentage

- `homework` = homework average
- `sleep` = average hours of sleep per night
- `parttime` = part-time work hours per week

Our goal is to decide which predictors should be included in the model.

18.10 Step 1: Create the Dataset

```
DATA studentperf;
    INPUT hours attendance homework sleep parttime final;
    DATALINES;
5  92 88 7.5  5 84
8  95 91 7.0  2 90
3  85 80 6.5 12 72
10 98 94 7.2  0 95
6  90 86 8.0  8 83
4  88 78 6.8 15 74
9  96 93 7.1  4 92
7  91 89 7.4  6 87
2  80 75 6.0 18 68
11 99 96 7.3  0 97
1  78 70 5.8 20 63
6  89 84 7.6 10 81
8  94 90 7.0  3 89
5  87 82 6.9 14 77
9  97 95 7.8  1 94
3  83 76 6.4 16 70
7  92 88 7.1  7 86
4  86 79 6.7 13 75
10 98 97 7.5  0 96
2  81 73 6.2 17 67
;
RUN;
```

18.10.1 Variable meanings

- `final` = final exam score
- `hours` = study hours per week
- `attendance` = attendance percentage
- `homework` = homework average

- `sleep` = average sleep per night
- `parttime` = work hours per week

18.11 Why This Is a Model Selection Problem

We have several candidate predictors, but not all of them may be equally useful.

Questions include:

- Do we need all predictors?
- Which predictors improve prediction?
- Can a simpler model perform as well as a larger model?

18.12 Step 2: Fit a Full Model

```
PROC REG DATA=studentperf;
    MODEL final = hours attendance homework sleep parttime;
RUN;
QUIT;
```

18.12.1 Discussion

This model includes **all candidate predictors**.

Questions to ask:

- Are all predictors significant?
- Are some predictors redundant?
- Can a smaller model perform similarly?

> **Important**
>
> A full model is not always the best model.
> Some predictors may add complexity without adding much predictive value.

18.13 Step 3: Use PROC GLMSELECT with Cross-Validation

```
PROC GLMSELECT DATA=studentperf PLOTS=ALL;
    MODEL final = hours attendance homework sleep parttime
          / SELECTION=STEPWISE
            CHOOSE=CV
            CVMETHOD=RANDOM(5);
RUN;
```

18.13.1 What this does

- `SELECTION=STEPWISE` asks SAS to perform stepwise model selection
- `CHOOSE=CV` tells SAS to choose the model using cross-validation
- `CVMETHOD=RANDOM(5)` uses 5-fold cross-validation

This helps us select a model that balances:

- prediction accuracy
- model simplicity

18.14 Step 4: Alternative Selection with AIC

You can also use AIC as the selection criterion.

```
PROC GLMSELECT DATA=studentperf PLOTS=ALL;
    MODEL final = hours attendance homework sleep parttime
          / SELECTION=FORWARD
            CHOOSE=AIC;
RUN;
```

18.14.1 Interpretation

This version uses:

- `SELECTION=FORWARD` for forward selection
- `CHOOSE=AIC` to pick the model with the best AIC

So this is still model selection, but based on a different criterion.

18.15 How to Read the ASE Plot

When you use `PLOTS=ALL`, one useful graph is the ASE plot.

The ASE plot helps compare:

- training error
- validation or cross-validation error

18.15.1 Interpretation

- Training error usually decreases as the model becomes more complex
- Validation or test error may decrease at first, then increase
- If validation error starts increasing, the model may be overfitting
- The best model is often near the minimum of the validation or test ASE curve

> **Important**
>
> A smaller training error does **not** always mean a better model.
> For prediction, validation or test error is more informative.

18.16 In-Class Questions

Why is training MSE usually too optimistic as an estimate of prediction error?

What is the main purpose of K-fold cross-validation?

If a more complex model has lower training error but higher test error, what problem is occurring?

What does a smaller AIC or BIC usually indicate?

Why is `PROC GLMSELECT` useful when there are many candidate predictors and interactions?

Which variables seem most useful for predicting final exam score in this example?

18.17 Teaching Interpretation of the Toy Example

This example has a natural real-world meaning:

- More study hours may improve exam performance
- Better attendance may help
- Strong homework performance may indicate understanding
- Too many part-time work hours may reduce study time
- Sleep may matter, but perhaps less strongly than the others

This makes it easier to think about:

- variable importance
- model simplicity
- prediction versus interpretation

18.18 What to Remember from This Lecture

The central question in model selection is:

> Which model gives the best balance between fit, prediction accuracy, and simplicity?

There are two major perspectives:

1. **Prediction-based selection**
   - training/test split
   - validation data
   - cross-validation
2. **Criterion-based selection**
   - AIC
   - BIC

In SAS, `PROC GLMSELECT` provides a practical way to carry out model selection with both classical and modern options.

Key Takeaways

- Model selection is about choosing a model that is useful, not just complicated
- Prediction error is often more important than training error
- Cross-validation helps estimate out-of-sample performance
- AIC and BIC balance fit and complexity
- `PROC GLMSELECT` is a powerful SAS tool for model selection

- Overfitting occurs when a model fits training data well but performs poorly on new data

19 Nested Models

Learning Objectives

By the end of this lecture, you should be able to:

1. Distinguish between **nested models** (full vs reduced models) and **nested designs** (hierarchical data structures).
2. Perform and interpret a **partial F-test** and a **likelihood ratio test (LRT)** for model comparison.
3. Explain the difference between **crossed** and **nested** factors.
4. Implement nested model testing in SAS using `PROC REG` and the `TEST` statement.
5. Specify hierarchical nested effects in `PROC GLM` and `PROC MIXED` using `B(A)` syntax.

19.1 Introduction

In the previous chapter, we studied model selection methods such as AIC, BIC, and cross-validation for comparing competing models. In this chapter, we focus on a more structured type of comparison: **nested structures**.

The word **nested** appears in two related but distinct contexts in statistics:

1. **Nested models**: a simpler model is obtained from a more complex model by imposing parameter restrictions.
2. **Nested designs**: one factor exists only within the levels of another factor, creating a hierarchical data structure.

This chapter connects regression, ANOVA, and mixed models by treating both ideas carefully.

A model M_0 is said to be **nested** within a model M_1 if M_0 can be obtained from M_1 by imposing constraints on the parameters, usually by setting some coefficients equal to zero.

19.1.1 Full and Reduced Models

Consider the following regression models.

Full model M_1:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \varepsilon$$

Reduced model M_0:

$$Y = \beta_0 + \beta_1 X_1 + \varepsilon$$

The reduced model is nested within the full model because it is obtained by imposing the restrictions

$$\beta_2 = \beta_3 = 0.$$

In other words, the reduced model is a special case of the full model.

Nested models allow us to answer a very specific question:

> Do the additional terms in the full model provide enough improvement to justify their inclusion?

This leads naturally to formal hypothesis testing.

19.2 Partial F-test for Nested Linear Models

For linear regression models fit by least squares, the standard test for comparing nested models is the **partial F-test**.

We test

$$H_0 : \beta_2 = \beta_3 = 0$$

against

$$H_a : \text{at least one of } \beta_2, \beta_3 \text{ is nonzero.}$$

Let:

- SSE_R: error sum of squares for the reduced model
- SSE_F: error sum of squares for the full model
- df_R: error degrees of freedom for the reduced model
- df_F: error degrees of freedom for the full model

Then the test statistic is

$$F = \frac{(SSE_R - SSE_F)/(df_R - df_F)}{SSE_F/df_F}.$$

Interpretation

- If the full model does **not** improve fit much, then $SSE_R - SSE_F$ will be small, and the F-statistic will be small.
- If the full model improves fit substantially, then the F-statistic will be large.

A small p-value provides evidence against H_0, suggesting that the additional predictors should be retained.

19.3 Likelihood Ratio Test (LRT)

For likelihood-based models, such as generalized linear models and many mixed models, we often compare nested models using the **likelihood ratio test**.

Let ℓ_{Full} and ℓ_{Reduced} be the maximized log-likelihoods under the full and reduced models, respectively. The likelihood ratio test statistic is

$$LRT = 2\left(\ell_{\text{Full}} - \ell_{\text{Reduced}}\right).$$

Under regularity conditions and under H_0,

$$LRT \sim \chi^2_{df_R - df_F},$$

where $df_R - df_F$ is the difference in the number of free parameters between the two models.

When to use which?

- **Partial F-test**: typically used for nested linear models fit by least squares.
- **LRT**: commonly used for generalized linear models, mixed models, and other likelihood-based settings.

19.4 SAS Implementation

In SAS, we can test nested regression models directly using `PROC REG` and the `TEST` statement.

```
PROC REG DATA=my_data;
    MODEL Y = X1 X2 X3;

    /* Test whether the reduced model is sufficient */
    TEST X2 = 0, X3 = 0;
RUN;
QUIT;
```

19.4.1 What Does This Do?

This code fits the full model

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \varepsilon$$

and tests whether the restrictions

$$\beta_2 = \beta_3 = 0$$

are reasonable.

If the test is significant, then the reduced model is too simple and the additional terms should remain in the model.

19.5 Concept 2: Nested Designs

In ANOVA and mixed models, the word **nested** refers to a hierarchical factor structure.

A factor B is nested within a factor A if each level of B appears in only one level of A.

Suppose:

- A = Region
- B = City

If each city belongs to exactly one region, then **City is nested within Region**.

We write this as:

$$\text{City(Region)}.$$

19.6 Crossed vs Nested Factors

It is important to distinguish **crossed** factors from **nested** factors.

Crossed Factors

Two factors are **crossed** if every level of one factor appears with every level of the other factor.

- Treatment and gender
- Every treatment level can be observed for every gender

In this case, an interaction term such as $A \times B$ is meaningful.

Nested Factors

A factor is **nested** when its levels exist only within the levels of another factor.

Example:

- Students within classrooms
- Classrooms within schools
- Cities within regions

You cannot meaningfully combine a nested level with all levels of the parent factor. For example, New York City is not a city within the Midwest region.

Summary Table

Structure	Description
Crossed	Every level of A appears with every level of B
Nested	Levels of B exist only within levels of A

19.7 Statistical Formulation for a Nested Design

A standard nested ANOVA model is

$$y_{ijk} = \mu + \alpha_i + \beta_{j(i)} + \varepsilon_{ijk},$$

where

- μ is the overall mean,
- α_i is the effect of factor A (for example, Region),
- $\beta_{j(i)}$ is the effect of factor B nested within A (for example, City within Region),
- ε_{ijk} is the random error term.

This is a **hierarchical** structure rather than a factorial one.

19.8 SAS Implementation for Nested Designs

19.8.1 Example Data

```
DATA nested_data;
    INPUT Region $ City $ Y;
    DATALINES;
NE NY 30
NE NY 35
NE Pitt 18
NE Pitt 20
MW Chicago 10
MW Chicago 9
;
RUN;
```

19.8.2 `PROC GLM`: Fixed-Effects Perspective

```
PROC GLM DATA=nested_data;
    CLASS Region City;
    MODEL Y = Region City(Region);
RUN;
QUIT;
```

This specification treats the nested structure explicitly using `City(Region)`.

19.8.3 `PROC MIXED`: Hierarchical or Random-Effects Perspective

```
PROC MIXED DATA=nested_data;
    CLASS Region City;
    MODEL Y = Region;
    RANDOM City(Region);
RUN;
QUIT;
```

This version is often more appropriate when the nested factor is viewed as a **random effect**.

19.9 Fixed vs Random Effects

A key modelling decision is whether factors are treated as **fixed** or **random**.

Component	Fixed Effect Interpretation	Random Effect Interpretation
Region	Compare these specific regions	Regions are sampled from a larger population
City(Region)	Compare these specific cities within each region	City-to-city variation within region

> ⚠ Important Note
>
> `PROC GLM` and `PROC MIXED` are not simply interchangeable. The interpretation changes depending on whether the factor is treated as fixed or random.
>
> - Use `PROC GLM` when you want to compare specific factor levels directly.
> - Use `PROC MIXED` when you want to model hierarchical variation and random effects.

Interpreting the Output

In a nested design, the total variation is decomposed into:

1. variation **between** regions,
2. variation **among cities within regions**,
3. residual variation **within cities**.

This is called **hierarchical variance decomposition**.

So the interpretation is different from a crossed two-way ANOVA. We are not asking whether Region and City interact; instead, we are asking how variability is distributed across levels of a hierarchy.

19.10 Connection to Model Selection

This chapter connects naturally to the previous lecture on model selection.

- General Model Selection Asks

  Which model fits the data best overall?

- Nested Model Comparison Asks

Does the added complexity of a larger model significantly improve fit over a simpler baseline?

This is why nested models are so important:

- they provide the formal basis for **partial F-tests**,
- they motivate **likelihood ratio tests**,
- and they help us think carefully about when extra structure is statistically justified.

At the same time, nested designs remind us that data structure also matters. A model can be more complex not only because it has more parameters, but also because it reflects a deeper hierarchical organization of the data.

19.11 Summary

Nested structures connect three core ideas in statistical modelling:

1. **Model building**: specifying appropriate structure for the data,
2. **estimation**: fitting full and reduced models or decomposing variance,
3. **inference**: deciding whether added complexity is justified.

19.11.1 Key Takeaways

- **Nested models** concern **parameter restrictions**.
- **Nested designs** concern **hierarchical factor structures**.
- The **partial F-test** is used for nested linear models.
- The **likelihood ratio test** is used for many likelihood-based nested comparisons.
- In SAS, nested structures are expressed using `TEST`, `B(A)`, and `RANDOM`.

19.12 Practice Problems

1. Show mathematically how a reduced regression model is nested within a full model through parameter constraints.
2. Suppose students are nested within classrooms, and classrooms are nested within schools. Write an appropriate SAS model specification.
3. Fit a full regression model using `PROC REG`, carry out a `TEST` statement, and verify the partial F-statistic manually using the reduced and full model SSE values.
4. Explain the difference between `PROC GLM` and `PROC MIXED` for a nested design.
5. Give one example of crossed factors and one example of nested factors, and explain why they differ.

Part IV

Advanced Topics

20 Predictive Modelling and Model Evaluation in SAS

Learning Objectives

By the end of this lecture, you should be able to:

1. Explain the difference between **statistical inference** and **prediction**.
2. Understand why we split data into **training** and **test** sets.
3. Evaluate predictive performance using measures such as **RMSE**, **MAE**, and **classification accuracy**.
4. Fit predictive models in SAS using regression-based procedures.
5. Generate predictions and assess out-of-sample model performance in SAS.

20.1 Introduction

So far in this course, we have focused mainly on statistical modelling for explanation and inference. We built regression models, interpreted coefficients, tested hypotheses, and compared models using methods such as AIC and BIC.

In many real-world problems, however, the main goal is not necessarily to **explain the relationship between variables**. Instead, the goal is to make **accurate predictions** for future observations.

Examples include:

- predicting house prices,
- forecasting sales,
- predicting whether a customer will respond to a marketing campaign,
- estimating the probability that a patient has a disease.

This shift in focus leads us to **predictive modelling**.

20.2 Inference vs Prediction

Although the same model can often be used for both purposes, inference and prediction are not the same.

Inference

Inference focuses on questions such as:

- Is a predictor statistically significant?
- What is the effect of a one-unit increase in X?
- Is there evidence that the mean response differs across groups?

The emphasis is on:

- parameter estimation,
- confidence intervals,
- hypothesis testing,
- interpretability.

Prediction

Prediction focuses on questions such as:

- How well can the model predict outcomes for new observations?
- Which model gives the smallest prediction error?
- How accurate are predicted probabilities or classifications?

The emphasis is on:

- predictive accuracy,
- generalization to new data,
- out-of-sample performance.

Key Idea

A model can be useful for inference but not especially good for prediction, and vice versa.

A model with many terms may fit the current data very well, but may perform poorly on future data. This is one reason why predictive modelling requires different evaluation tools.

20.3 Training Error vs Test Error

Suppose we fit a model using a given dataset.

- The **training error** measures how well the model fits the data used to estimate the model.
- The **test error** measures how well the model predicts new observations that were not used for fitting.

20.3.1 Why does this matter?

If a model is too flexible, it may fit noise in the training sample rather than the true signal. This is called **overfitting**.

As a result:

- training error may be very small,
- test error may still be large.

20.3.2 Main Principle

> A good predictive model should perform well on new data, not only on the training data.

20.4 Data Splitting

A standard approach in predictive modelling is to divide the data into separate parts.

Common split

- **Training set**: used to fit the model
- **Validation set**: sometimes used to tune or compare models
- **Test set**: used for final evaluation

In this lecture, we focus on the simpler and very common setup:

- **training set**
- **test set**

Question: Why not use all observations for fitting?

Because if we evaluate the model on the same data used to fit it, we usually obtain an overly optimistic assessment of performance.

Data Leakage

A major concern in predictive modelling is **data leakage**, which occurs when information from the test data influences model building. This makes performance look better than it really is.

- choosing variables after looking at the test set,
- using test-set summaries in data preprocessing,
- evaluating many models on the same test set and reporting only the best one.

20.5 Predictive Models for Continuous and Binary Outcomes

20.5.1 Continuous Response

When the response variable is continuous, common predictive models include:

- linear regression,
- polynomial regression,
- more flexible machine learning models.

Examples:

- house price prediction,
- salary prediction,
- demand forecasting.

20.5.2 Binary Response

When the response variable is binary, common predictive models include:

- logistic regression,
- classification trees,
- other classifiers.

Examples:

- disease vs no disease,
- default vs no default,
- spam vs not spam.

In this lecture, we will use familiar regression-based models to build a bridge from classical statistics to predictive modelling.

20.6 Performance Metrics for Regression

Suppose we observe actual values $y_1, \ldots, y_n$ and predicted values $\hat{y}_1, \ldots, \hat{y}_n$.

1. Mean Squared Error (MSE)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

2. Root Mean Squared Error (RMSE)

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

- RMSE is popular because it is measured on the *same scale* as the response.

3. Mean Absolute Error (MAE)

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

- Smaller RMSE or MAE indicates better predictive performance.
- RMSE penalizes large errors more heavily than MAE.
- MAE is often easier to interpret directly.

20.7 Performance Metrics for Classification

For binary outcomes, the model often produces either:

- a predicted class, or
- a predicted probability.

1. Confusion Matrix

   A confusion matrix summarizes:

- true positives,
- true negatives,
- false positives,

- false negatives.

2. Classification Accuracy

$$\text{Accuracy} = \frac{\text{Number of correct classifications}}{\text{Total number of observations}}$$

3. Other Useful Metrics

- sensitivity,
- specificity,
- precision,
- ROC curve,
- AUC.

In this lecture, we focus on basic classification accuracy and predicted probabilities, while later topics in ML can expand these ideas.

> **Predictive Modelling Workflow**
>
> A common workflow for predictive modelling is:
>
> 1. split the data into training and test sets,
> 2. fit a model using the training set,
> 3. evaluate the model on the test set,
> 4. compare predicted and observed outcomes,
> 5. compute predictive performance measures.

In this example, we use a regression model to predict a continuous response.

Step 1: Split the data

```
PROC SURVEYSELECT DATA=mydata
    OUT=split_data
    SEED=12345
    SAMPRATE=0.70
    OUTALL;
RUN;
```

The chunk above creates an output dataset with an indicator variable named `Selected`:

- `Selected=1`: observation goes to the training set
- `Selected=0`: observation goes to the test set

Step 2: Create training and test datasets

```
DATA train test;
    SET split_data;
    IF Selected = 1 THEN OUTPUT train;
    ELSE OUTPUT test;
RUN;
```

Step 3: Fit a regression model on the training set

```
PROC REG DATA=train;
    MODEL Y = X1 X2 X3;
    SCORE DATA=test OUT=test_pred PREDICT;
RUN;
QUIT;
```

Step 4: Compute prediction errors

```
DATA test_pred;
    SET test_pred;
    error  = Y - MODEL1;
    abserr = ABS(error);
    sqerr  = error**2;
RUN;

PROC MEANS DATA=test_pred MEAN;
    VAR abserr sqerr;
RUN;
```

The mean of `abserr` is the MAE. The mean of `sqerr` is the MSE. Taking the square root of the MSE gives the RMSE.

Now suppose the response is binary, such as whether a customer responds to an offer.

Step 1: Fit logistic regression on the training data

```
PROC LOGISTIC DATA=train;
    MODEL response(event='1') = X1 X2 X3;
    SCORE DATA=test OUT=logit_scored;
RUN;
```

Step 2: Convert probabilities into predicted classes

A common rule is:

- predict 1 if predicted probability ≥ 0.5,
- predict 0 otherwise.

```
DATA logit_scored;
    SET logit_scored;
    pred_class = (P_1 >= 0.5);
RUN;
```

Step 3: Compare predicted and observed classes

```
PROC FREQ DATA=logit_scored;
    TABLES response*pred_class / NOCOL NOROW NOPERCENT;
RUN;
```

From this table, we can calculate classification accuracy.

20.8 Why Predictive Modelling Changes the Way We Think

In classical regression analysis, we often ask:

- Which variables are significant?
- What is the interpretation of each coefficient?

In predictive modelling, we instead ask:

- Does this model predict well on new data?
- Which model gives the best out-of-sample performance?
- Is a simpler model good enough?

> ⚠ Important Shift
>
> A variable can be **statistically significant** but **add little predictive value.**
> Conversely, a model chosen for prediction may prioritize accuracy over easy interpretation.

20.9 Relationship to Model Selection

This lecture connects directly to the previous lecture on model selection.

In model selection for inference, we often ask:

- Is the coefficient significant?
- Which model has the smallest AIC or BIC?

In predictive modelling, we often ask:

- Which model has the smallest test RMSE?
- Which classifier performs best on unseen data?

These are related questions, but they are not identical.

Key Message

> A good explanatory model is not always the best predictive model.

This distinction is central to modern data analysis and machine learning.

20.10 Practical Notes for SAS Users

When building predictive models in SAS, keep the following points in mind:

1. Always separate training and test data when evaluating prediction performance.
2. Record the random seed so your results are reproducible.
3. Evaluate performance on the test set, not only the training set.
4. Use appropriate metrics for the outcome type:
    - RMSE or MAE for continuous response
    - accuracy or related measures for binary response
5. Keep the full modelling pipeline clear and reproducible.

This is especially important in real data analysis projects.

20.11 Things to think about

This lecture provides the bridge from classical statistics to machine learning.

Once we adopt the predictive modelling viewpoint, it becomes natural to ask:

- Can we use more flexible models?
- Can we improve prediction beyond linear and logistic regression?
- How do tree-based methods work?
- What happens when we care more about prediction than parameter interpretation?

These questions motivate the next lecture on **machine learning in SAS**.

20.12 Summary

In this lecture, we introduced predictive modelling as a different perspective on statistical modelling.

Main Ideas

- Inference and prediction are related but distinct goals.
- Training performance and test performance are not the same.
- Data splitting helps us assess out-of-sample performance.
- Regression and logistic regression can both be used as predictive models.
- Predictive performance should be evaluated using suitable metrics.

SAS Commands

- splitting data using `PROC SURVEYSELECT`,
- fitting predictive models with `PROC REG` and `PROC LOGISTIC`,
- scoring test data,
- evaluating performance using prediction errors and classification tables.

20.13 Practice Problems to yourself

1. Explain the difference between inference and prediction in your own words.
2. Why is test error usually more informative than training error in predictive modelling?
3. Use `PROC SURVEYSELECT` to create training and test sets from a dataset of your choice.
4. Fit a linear regression model on the training set and compute the test-set RMSE.
5. Fit a logistic regression model on the training set and compute classification accuracy on the test set.

6. Give an example of a model that may be easy to interpret but not necessarily best for prediction.

21 Introduction to Machine Learning in SAS

Learning Objectives

By the end of this lecture, you should be able to:

1. Explain what **machine learning** means in a predictive modelling context.
2. Distinguish between **supervised** and **unsupervised** learning.
3. Understand the basic ideas behind **decision trees**, **random forests**, and **regularization**.
4. Compare classical statistical models with more flexible machine learning methods.
5. Implement basic machine learning workflows in SAS for prediction and model comparison.

21.1 Introduction

In the previous lecture, we introduced predictive modelling and emphasized the importance of evaluating models using out-of-sample performance. That lecture served as a bridge from classical statistical modelling to modern prediction.

In this lecture, we take one more step and ask:

- What do people mean by **machine learning** (ML)?
- How is it related to regression and classification?
- What kinds of predictive methods are available in SAS?
- When might a more flexible model be useful?

The goal of this lecture is not to turn this course into a full machine learning course. Instead, the goal is to introduce several important ideas in a way that is connected to the statistical models you already know.

21.2 What Is ML?

There is no single perfect definition of machine learning, but in many practical settings it refers to:

> using data-driven algorithms to learn patterns from data for prediction, classification, or discovery.

Compared with traditional statistical modelling, machine learning often places stronger emphasis on:

- prediction,
- flexibility,
- automatic pattern detection,
- performance on unseen data.

21.2.1 Important Note

Machine learning is **not separate from statistics**. Many machine learning methods are extensions of ideas you already know:

- regression,
- classification,
- optimization,
- variable selection,
- bias-variance trade-off.

In many real applications, the boundary between statistics and machine learning is not sharp.

21.3 Supervised vs Unsupervised Learning

A useful first distinction is between **supervised learning** and **unsupervised learning**.

Supervised Learning

In supervised learning, we observe:

- predictors X,
- a response Y.

The goal is to learn a rule that predicts Y from X.

- predicting house price from location and size,
- predicting whether a patient has a disease,

- predicting whether an email is spam.

Typical supervised learning methods include:

- linear regression,
- logistic regression,
- decision trees,
- random forests,
- support vector machines,
- neural networks.

Unsupervised Learning

In unsupervised learning, we observe predictors X but no response variable.

The goal is to discover structure in the data.

- grouping customers into segments,
- detecting clusters,
- dimension reduction.

Typical unsupervised learning methods include:

- clustering,
- principal component analysis.

21.3.1 Focus of This Lecture

We focus mainly on **supervised learning**, because it connects directly to predictive modelling and to many SAS procedures.

21.4 Classical Models vs ML Models

Classical statistical models often emphasize:

- interpretability,
- effect estimation,
- statistical significance,
- model assumptions.

ML methods often emphasize:

- predictive accuracy,
- flexibility,

- automatic adaptation to nonlinear patterns,
- performance on new data.

Comparison

Perspective	Classical Statistical Modelling	Machine Learning
Main goal	explanation and inference	prediction
Focus	parameter interpretation	predictive performance
Model form	often specified in advance	often more flexible
Evaluation	significance, AIC, BIC	test error, accuracy, AUC

21.4.1 Important Message

This is a difference in emphasis, not a strict separation. For example:

- linear regression can be used as a predictive model,
- machine learning models can sometimes still be interpreted.

21.5 Why Flexible Models Can Help

A linear model assumes a relatively simple relationship between the response and predictors.

But in real data, relationships may be:

- nonlinear,
- involve interactions,
- vary across different parts of the predictor space,
- too complex to describe with a simple formula.

More flexible models can adapt to such patterns.

However, flexibility comes with a cost:

- a flexible model may overfit,
- interpretation may become harder,
- tuning becomes more important.

This is why we must always evaluate models on new data.

21.6 Decision Trees

One of the most intuitive machine learning methods is the **decision tree**.

21.6.1 Basic Idea

A decision tree splits the predictor space into regions by asking a sequence of questions such as:

- Is age < 40?
- Is income $> 60{,}000$?
- Is blood pressure > 130?

For regression, each terminal node gives a predicted numeric value.
For classification, each terminal node gives a predicted class or class probability.

21.6.2 Why Students Usually Like Trees

Decision trees are appealing because they are:

- easy to visualize,
- easy to explain,
- capable of capturing nonlinear relationships,
- capable of handling interactions automatically.

A decision tree for loan default might work like this:

1. Split first on income.
2. Among lower-income individuals, split on credit score.
3. Among higher-income individuals, split on debt level.

The final prediction depends on the path through the tree.

Strengths and Weaknesses of Trees

- Strengths
  - easy to interpret,
  - naturally capture interactions,
  - do not require linearity,
  - useful for both regression and classification.
- Weaknesses
  - can be unstable,

  - small changes in data may lead to a different tree,
  - a single tree may not predict as accurately as ensemble methods.

This motivates methods such as **bagging** and **random forests**.

21.7 Random Forests and Ensemble Thinking

A random forest is an **ensemble method** built from many decision trees.

21.7.1 Main Idea

Instead of relying on one tree, we build many trees using random variation in the data and predictors, then combine their predictions.

This often improves predictive performance because:

- a single tree may be noisy or unstable,
- averaging many trees reduces variability,
- the combined model is often more robust.

21.7.2 Why Random Forests Work Well

Random forests can:

- model nonlinear relationships,
- handle many predictors,
- capture interactions,
- provide strong predictive accuracy in many settings.

21.7.3 Trade-off

Compared with a single tree, a random forest is usually:

- more accurate,
- less interpretable.

This is a common pattern in ML: better predictive performance may come at the cost of simpler interpretation.

21.8 Regularization and High-Dimensional Prediction

Another important machine learning idea is **regularization**.

21.8.1 Motivation

Suppose we have many predictors.

Problems may include:

- overfitting,
- unstable coefficient estimates,
- multicollinearity,
- poor test performance.

Regularization adds a penalty that discourages overly complex models.

21.8.2 Example: LASSO

LASSO regression estimates coefficients by balancing:

- goodness of fit,
- model complexity.

As a result, some coefficients may be shrunk toward zero, and some may become exactly zero.

This makes LASSO useful for:

- variable selection,
- prediction in larger models,
- controlling overfitting.

21.8.3 Big Picture

Regularization helps us move from classical regression toward modern predictive modelling in a principled way.

21.9 A Simple Machine Learning Workflow in SAS

A practical ML workflow in SAS looks like this:

1. split the data into training and test sets,
2. fit multiple candidate models,
3. score the models on the test set,
4. compare predictive performance,
5. choose a model based on out-of-sample results.

This is similar to the workflow from the previous lecture, but now we may compare more flexible models.

21.10 SAS Example: Decision Tree

One useful SAS procedure for tree-based methods is `PROC HPSPLIT`.

21.10.1 Example: Classification Tree

```
PROC HPSPLIT DATA=train;
    CLASS response;
    MODEL response = X1 X2 X3 X4 X5;
    GROW GINI;
    PRUNE COSTCOMPLEXITY;
    CODE FILE='tree_score.sas';
RUN;
```

21.10.2 Explanation

- `GROW GINI` is a common splitting rule for classification.
- `PRUNE COSTCOMPLEXITY` reduces overfitting by simplifying the tree.
- `CODE FILE=...` creates scoring code that can be applied to new data.

21.10.3 Scoring the Test Set

```
DATA tree_scored;
    SET test;
    %INCLUDE 'tree_score.sas';
RUN;
```

After scoring, we can compare predicted classes to observed classes.

21.11 SAS Example: Random Forest

A useful SAS procedure for random forests is `PROC HPFOREST`.

```
PROC HPFOREST DATA=train;
    TARGET response / LEVEL=nominal;
    INPUT X1 X2 X3 X4 X5 / LEVEL=interval;
    SCORE OUT=forest_train;
RUN;
```

Depending on your SAS environment, you may also use procedure-specific scoring output for validation or test datasets.

21.11.1 Interpretation

The random forest builds many trees and aggregates their predictions. In practice, this often gives better predictive accuracy than a single tree.

21.12 SAS Example: Regularization with GLMSELECT

SAS also provides tools for regression-based model selection and shrinkage.

A useful procedure is `PROC GLMSELECT`.

```
PROC GLMSELECT DATA=train;
    MODEL Y = X1 X2 X3 X4 X5 / SELECTION=LASSO;
    PARTITION FRACTION(TEST=0.3);
RUN;
```

21.12.1 Why This Is Useful

This procedure illustrates an important idea:

- classical regression and machine learning are connected,
- variable selection can be handled in a prediction-oriented framework,
- model complexity can be controlled automatically.

21.13 Comparing Models

One of the most important lessons in ML is that we should compare models using the same evaluation framework.

For example, we might compare:

- logistic regression,
- a classification tree,
- a random forest.

21.13.1 Questions to Ask

- Which method has the highest test accuracy?
- Which method gives the best balance between performance and interpretability?
- Is the improvement in prediction large enough to justify a more complex model?

21.13.2 Example Discussion

A logistic regression model may be:

- easier to interpret,
- easier to communicate,
- better when the relationship is approximately linear.

A tree-based model may be:

- better at capturing nonlinear relationships,
- better at capturing interactions,
- harder to summarize with a few coefficients.

A random forest may be:

- even more accurate,
- but much less transparent.

21.14 Bias-Variance Trade-off

A central concept in machine learning is the **bias-variance trade-off**.

21.14.1 Informal Idea

- a very simple model may be too rigid and miss important patterns,
- a very flexible model may follow noise too closely.

So good prediction often requires balancing:

- **bias**: error from being too simple,
- **variance**: error from being too sensitive to the sample.

21.14.2 Connection to Earlier Topics

This idea is closely related to:

- overfitting,
- model selection,
- regularization,
- test-set evaluation.

In other words, machine learning builds on ideas that are already present in statistics.

21.15 Why This Matters for SAS Users

Students sometimes think machine learning is something completely separate that requires a different software ecosystem.

That is not true.

SAS supports important ML workflows, including:

- data splitting,
- regression-based prediction,
- tree-based methods,
- forests and ensemble methods,
- regularization and model comparison.

This means the tools you have learned in SAS can be extended naturally into modern predictive analytics.

21.16 Practical Advice

When using machine learning methods, remember:

1. Do not evaluate performance only on the training set.
2. A more complex model is not automatically better.
3. Interpretability still matters in many applications.
4. Always compare models using a clear test-set or validation framework.
5. Good predictive modelling requires both statistical thinking and computational implementation.

21.17 Looking Ahead

This lecture is only an introduction, but it shows how SAS can be used for modern predictive analysis.

After this course, students interested in advanced topics could continue with:

- cross-validation,
- boosting,
- support vector machines,
- neural networks,
- deep learning,
- high-dimensional data analysis.

The key point is that the ideas from this course already provide a strong foundation for these topics.

21.18 Chapter Summary

In this lecture, we introduced machine learning as an extension of predictive modelling.

21.18.1 Main Ideas

- Machine learning emphasizes prediction, flexibility, and out-of-sample performance.
- Supervised learning uses predictors and a response; unsupervised learning looks for structure without a response.
- Decision trees provide intuitive nonlinear prediction rules.
- Random forests improve prediction by aggregating many trees.
- Regularization helps control model complexity and improve prediction.

21.18.2 SAS Skills Introduced

- tree modelling with `PROC HPSPLIT`,
- random forests with `PROC HPFOREST`,
- regularized regression with `PROC GLMSELECT`,
- model comparison using predictive performance.

21.18.3 Final Message

> Machine learning does not replace statistics.
> It extends statistical modelling when prediction and flexibility become central goals.

21.19 Practice Problems

1. Explain the difference between supervised and unsupervised learning.
2. Why might a decision tree outperform a linear model in some datasets?
3. What is the main advantage of a random forest over a single tree?
4. Why does regularization help when many predictors are available?
5. Compare logistic regression and a tree-based classifier in terms of interpretability and flexibility.
6. In SAS, which procedures introduced in this lecture would you consider for:
    - a classification tree,
    - a random forest,
    - LASSO regression?

Part V

Final Project

Data Analysis Using SAS

Weight: 20% of final grade
Due Date: May 1, 2026
Last modified: Apr 17, 2026

Project Overview

The goal of the final project is to apply **statistical methods and SAS programming skills** learned throughout the course to conduct a **complete data analysis pipeline**.

Students will:

- Work with a **real-world open dataset**
- Perform **data cleaning, visualization, and modelling**
- Use **SAS procedures and macros**
- Communicate results clearly through a **written report**

Group Information

Group	Member	Topic
1	Zong Yang	Financial Metric Transmission Mechanism within Coca-Cola's Income Statement
2	Mangsa Limbu Pheudin, Shaswat Bhushan Jangam, Allswell Akomanyi-Addo	Analysis of blood lead levels in the treatment of lead-exposed children
3	Wenpu Ma	Cleveland Heart Disease Data
4	Vu, Ha	Drug Safety Classification and Risk Analysis of FDA Drug Adverse Reports
5	Umma Hafsah Himu	Clinical and Demographic Predictors of Heart Disease
6	Kalen Jinnah, Ivanna Poliashenko	Baseball bat speed Analysis
7	Zhe Zhong	Bank Marketing Dataset

Group	Member	Topic
8	Andrew Krause	Heart Disease Dataset
9	Taiwo Ayeni	Cardiovascular Risk Factors in the Framingham Heart Study
10	Maya Creary	Is there a statistically significant difference in the "Inattentive" vs. "Hyperactive" symptom scores between male and female subjects?
11	Seoyoung Kim	CDC Disability data
12	Sifan You, Boshu Zhang	Relationship between car's weight and its fuel efficiency
13	Mostafa Farhadian	Flood Insurance Claims Dataset
14	Michael Dixon	African American high school students' Math scores
15	Kamana Joshi, Sushil Khadka	Spotify Audio Features
16	Seth Kwarteng	UCI Parkinsons dataset
17	Sierra Doherty	Braves ABS challenges
18	Daniel Geiger	African Mosquito Dataset

Data Requirement

You must use a **public/open dataset**.

Recommended sources:

- UCI Machine Learning Repository
- Kaggle (public datasets only)
- Data.gov
- CDC / WHO datasets
- Built-in SAS datasets (e.g., `sashelp.cars`, `sashelp.heart`, etc.)

Requirements:

- At least **100 observations**
- At least **3–5 variables**

- Must include:
  - At least **one categorical variable**
  - At least **one numerical variable**

Group:

You can form group of **2–3 students**. If you prefer to work alone, that is also acceptable. PhD students MUST work individually.

Project Components

Your project must include the following components:

Research Question (10%)

Clearly state:

- What question are you trying to answer?
- Why is it interesting or important?

Examples:

- Does treatment A outperform treatment B?
- What factors affect income?
- Is there an association between two categorical variables?

Data Cleaning & Preparation (15%)

- Import dataset into SAS
- Handle:
  - Missing values
  - Data types
  - Outliers (if necessary)

Suggested tools:

- `DATA` step
- `PROC IMPORT`
- `PROC FORMAT`

Exploratory Data Analysis (EDA) (15%)

Use SAS to explore the data:

- Summary statistics:
  - `PROC MEANS`
  - `PROC FREQ`
- Visualizations:
  - `PROC SGPLOT`

Examples:

- Histograms
- Boxplots
- Scatterplots

Statistical Analysis (30%)

You must include **at least TWO of the following methods**:

- Chi-square test (`PROC FREQ`)
- Two-sample t-test (`PROC TTEST`)
- ANOVA (`PROC ANOVA` or `PROC GLM`)
- Regression (`PROC REG`)
- Correlation (`PROC CORR`)

You should:

- State hypotheses
- Report test statistics and p-values
- Interpret results in context

SAS Macro (15%)

You must create **at least one SAS macro**.

Examples:

- A macro for:
  - Summary statistics
  - Running multiple regressions
  - Automating hypothesis tests

Your macro must:

- Take at least **2 input arguments**
- Be reusable
- Be clearly documented

Example structure:

```
%MACRO analysis(data, var);
    PROC MEANS DATA=&data;
        VAR &var;
    RUN;
%MEND;
```

Interpretation & Conclusion (15%)

You must:

- Explain results in plain language
- Connect findings to your research question
- Discuss:
- Limitations
- Possible improvements

Deliverables

1. Written Report (PDF)

Length: 8–10 pages (excluding references and appendices). You may put additional Figures and Tables in the appendix.

Include:

- Introduction
- Data description
- Methods
- Results
- Conclusion

2. SAS Code File (.sas)

Must include:

- Clean and well-commented code
- Macro implementation

3. (Optional Bonus +5%)

- Visualization dashboard
- Additional modeling (e.g., interaction, model comparison)
- Advanced macro usage

Grading Breakdown

Component	Points
Research Question	10
Data Preparation	15
EDA	15
Statistical Analysis	30
SAS Macro	15
Interpretation	15
Total	**100 (20% course weight)**

Important Notes

- You must write your own SAS code
- Collaboration is allowed for discussion, but not for code sharing between the groups
- Plagiarism will result in zero credit

Suggested Workflow

1. Pick dataset early
2. Perform EDA first
3. Decide appropriate statistical methods
4. Write SAS code
5. Wrap repeated tasks into macros
6. Write report last

Key Learning Outcome

This project integrates:

- Statistical thinking
- SAS programming
- Reproducible analysis
- Communication skills

This is designed to simulate a real-world data analysis task.

Part VI

Hands on Class

Writing Macro in SAS

Learning Objectives

In this hands-on session, we will practice writing **SAS macros** through a concrete example.
The goal is to help you develop a **systematic workflow** for turning regular SAS code into reusable and flexible macro functions.

By the end of this activity, you should be able to:

1. Understand the logic behind SAS macro programming.
2. Identify macro inputs and outputs.
3. Write simple SAS macros for statistical analysis.
4. Apply macros to automate repetitive tasks.

Guideline: How to Write SAS Macros

Based on experience, a reliable way to write SAS macros is to follow **three steps**:

Step 1: Write Regular SAS Code First

- Write SAS code that produces the desired result **without using macros**.
- Clearly identify:
  - What the code does
  - Which values may change (dataset name, variable name, parameters, etc.)

Step 2: Introduce Macro Variables Using `%LET`

- Replace hard-coded values with **macro variables** using `%LET`.
- This makes the code flexible and easier to generalize.

Step 3: Wrap the Code into a Macro

- Convert the `%LET` variables into **macro arguments**.
- Place the code inside `%MACRO ... %MEND`.
- Remove the `%LET` statements.
- Your macro now behaves like a function.

This three-step strategy is also useful when writing functions in **R** or **Python**.

Practice Examples in this activity

To practice these steps, in this activity, we will:

1. Write the first macro to compute the **sample mean**, **standard deviation** as well as the p-value
2. Write the second macro to conduct a **one-sample Z-test**

Eventually, you will practice the three-step workflow **twice** through those two macros

Dataset for this exercise: Court Length Data

We reuse the court length example from previous lecture.

```
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;
```

First SAS Macro for Reporting Mean and Standard Deviation

Step 1: Write Regular SAS Code (No Macros)

First, write SAS code that computes the `sample mean`, `sample size`, and `standard deviation` from a dataset.

Key: Replace `?` and `??` with the dataset name and variable name.

```
PROC MEANS DATA=?;
    VAR ??;
    OUTPUT OUT=zdata
           MEAN=Mean
           N=n
           STDDEV=Std_Dev;
RUN;
```

At this stage:

- Do not worry about macros.
- Focus only on producing the correct output.

Step 2: Introduce Flexibility Using %LET

Next, make the code flexible by replacing hard-coded values with macro variables.

```
%LET D = time;    /* dataset name */
%LET V = time;    /* variable name */

PROC MEANS DATA=&D;
    VAR &V;
    OUTPUT OUT=zdata
           MEAN=Mean
           N=n
           STDDEV=Std_Dev;
RUN;
```

Key idea:

- %LET creates macro variables D and V
- &D controls which dataset is used
- &V controls which variable is analyzed

This allows the same code to work for any dataset and variable.

Step 3: Change everything to a MACRO syntax

Finally, wrap the code into a macro function, that is, replace %LET variables with macro arguments and place the code inside %MACRO ... %MEND.

```
%MACRO myFunction(input1, input2);
...
%MEND myFunction;

%myFunction(input1, input2);
```

Second SAS Macro for creating confidence interval and p-value from a Z-test

In this section, we apply the **three-step macro writing strategy** again to build a SAS macro that computes:

- a $(1-\alpha)100\%$ confidence interval for the population mean, and
- the **Z-test statistic and p-value** for testing

$$H_0 : \mu = \mu_0.$$

You should **think carefully through each step** before combining everything into a macro.

You may need to utilize the following SAS functions:

Code chunk 1: Dataset

```
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;
```

Code chunk 2: Mean, standard deviation, and sample size calculation

```
PROC MEANS DATA=time;
    VAR time;
    OUTPUT OUT=zdata
           MEAN=Mean
           N=n
           STDDEV=Std_Dev;
RUN;
```

For this code chunk, you may verify that:

- Mean, Std_Dev, and n are correctly stored in the dataset zdata.

Code chunk 3: CI and Z-test Quantities

```
DATA ztest;
    SET zdata;

    /* Confidence level */
    CL = (1 - &alpha) * 100;

    /* Confidence interval */
    ZLower = Mean - probit(1 - &alpha/2) * Std_Dev / sqrt(n);
    ZUpper = Mean + probit(1 - &alpha/2) * Std_Dev / sqrt(n);

    /* Z-test statistic */
    zts = (Mean - &muzero) / (Std_Dev / sqrt(n));
RUN;
```

This code chunk creates:

- confidence level,
- confidence interval bounds,
- Z-test statistic.

Notes:

- `&alpha` is the significance level
- `&muzero` is the hypothesized mean under H_0
- `probit()` returns the standard normal quantile

Step 3: Introduce Macro Variables for Flexibility

Now, think about what should be user-controlled inputs:

- dataset name
- variable name
- significance level α
- null mean μ_0

Example macro variables:

```
%let D = time;
%let V = time;
%let alpha = 0.05;
%let muzero = 90;
```

Update your code by replacing fixed values with macro variables.

Convert Everything into a Macro

Finally, combine everything we learned from this activity into a reusable SAS macro.

Step 1

```
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 MEANS DATA=TIME;
    VAR TIME;
    OUTPUT OUT=ZDATA MEAN=MEAN N=N STDDEV=STD_DEV;
RUN;

DATA ZTEST;
    SET ZDATA;
    CL = (1 - &ALPHA) * 100;
    ZLOWER = MEAN - PROBIT(1 - &ALPHA / 2) * STD_DEV / SQRT(N);
    ZUPPER = MEAN + PROBIT(1 - &ALPHA / 2) * STD_DEV / SQRT(N);
    ZTS = (MEAN - &MUZERO) / (STD_DEV / SQRT(N));
    IF &SIDE = 'L' THEN ZPVALUE = PROBNORM(ZTS);
    IF &SIDE = 'U' THEN ZPVALUE = 1 - PROBNORM(ZTS);
    IF &SIDE = 'B' THEN ZPVALUE = 2 * MIN(1 - PROBNORM(ZTS), PROBNORM(ZTS));
RUN;

PROC PRINT DATA=ZTEST NOOBS;
    VAR N MEAN STD_DEV ZLOWER ZUPPER ZPVALUE;
    TITLE 'Z-INTERVALS AND Z-TEST';
RUN;
```

Step 2

```
%LET ALPHA = 0.05;   *VALUE OF ALPHA;
%LET MUZERO = 80;
%LET SIDE = 'B';

PROC MEANS DATA=TIME;
```

```
    VAR TIME;
    OUTPUT OUT=ZDATA MEAN=MEAN N=N STDDEV=STD_DEV;
RUN;

DATA ZTEST;
    SET ZDATA;
    CL = (1 - &ALPHA) * 100;
    ZLOWER = MEAN - PROBIT(1 - &ALPHA / 2) * STD_DEV / SQRT(N);
    ZUPPER = MEAN + PROBIT(1 - &ALPHA / 2) * STD_DEV / SQRT(N);
    ZTS = (MEAN - &MUZERO) / (STD_DEV / SQRT(N));
    IF &SIDE = 'L' THEN ZPVALUE = PROBNORM(ZTS);
    IF &SIDE = 'U' THEN ZPVALUE = 1 - PROBNORM(ZTS);
    IF &SIDE = 'B' THEN ZPVALUE = 2 * MIN(1 - PROBNORM(ZTS), PROBNORM(ZTS));
RUN;

PROC PRINT DATA=ZTEST NOOBS;
    VAR N MEAN STD_DEV ZLOWER ZUPPER ZPVALUE;
    TITLE 'Z-INTERVALS';
RUN;
```

Step 3

```
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;

%MACRO ZTEST_SELF(D, V, ALPHA, MUZERO, SIDE);

PROC MEANS DATA=&D;
    VAR &V;
    OUTPUT OUT=ZDATA MEAN=MEAN N=N STDDEV=STD_DEV;
RUN;

DATA ZTEST;
    SET ZDATA;
    CL = (1 - &ALPHA) * 100;
    ZLOWER = MEAN - PROBIT(1 - &ALPHA / 2) * STD_DEV / SQRT(N);
    ZUPPER = MEAN + PROBIT(1 - &ALPHA / 2) * STD_DEV / SQRT(N);
```

```
    ZTS = (MEAN - &MUZERO) / (STD_DEV / SQRT(N));
    IF &SIDE = 'L' THEN ZPVALUE = PROBNORM(ZTS);
    IF &SIDE = 'U' THEN ZPVALUE = 1 - PROBNORM(ZTS);
    IF &SIDE = 'B' THEN ZPVALUE = 2 * MIN(1 - PROBNORM(ZTS), PROBNORM(ZTS));
RUN;

PROC PRINT DATA=ZTEST NOOBS;
    VAR N MEAN STD_DEV CL ZLOWER ZUPPER ZPVALUE;
    TITLE 'Z-INTERVALS & TESTS';
RUN;

%MEND ZTEST_SELF;

%ZTEST_SELF(TIME, TIME, 0.05, 80, 'B');
```

Example Macro Call

```
%ZTEST_SELF(TIME, TIME, 0.05, 80, 'B');
```

- TIME (first): input dataset
- TIME (second): variable of interest
- 0.05: significance level
- 80: hypothesized mean
- 'B': two-sided Z-test
- 'L' = left-tailed
- 'U' = right-tailed
- 'B' = two-tailed

Summary

By following the three-step macro workflow, you have:

1. Written correct and verifiable SAS code,
2. Introduced flexibility using macro parameters,
3. Built a reusable SAS macro for Z-test inference.

This same workflow applies directly to:

- t-tests,
- confidence intervals,
- ANOVA summaries,
- regression diagnostics.

Once you master this pattern, writing SAS macros becomes systematic, reusable, and scalable, just like functions in R or Python.

Two-Sample t-Test and ANOVA in SAS

Learning Objectives

In this hands-on session, we will practice applying **two-sample t-tests** and **one-way ANOVA** in **SAS** using a built-in dataset from **SASHELP**. The goal is to help you develop a **systematic workflow** for comparing group means and interpreting the results.

By the end of this activity, you should be able to:

1. Understand when to use a **two-sample t-test**.
2. Understand when to use **one-way ANOVA**.
3. Identify the response variable and grouping variable in SAS code.
4. Write SAS code for two-sample t-tests and ANOVA.
5. Interpret p-values, assumptions, and post-hoc comparisons.

Guideline: How to Analyze Group Mean Comparisons in SAS

Based on experience, a reliable way to solve mean-comparison problems is to follow **three steps**:

Step 1: Understand the data and the question

- Identify the **response variable**
- Identify the **grouping variable**
- Determine how many groups are being compared

Step 2: Choose the correct inferential method

- If there are **two groups**, use a **two-sample t-test**
- If there are **three or more groups**, use **one-way ANOVA**

Step 3: Write SAS code and interpret the output

- Run the statistical procedure
- Check assumptions when appropriate
- Interpret the p-value in context
- For ANOVA, if the global test is significant, run a **multiple comparison procedure**

This same workflow is also useful when analyzing grouped data in **R** or **Python**.

Practice Examples in this Activity

To practice these steps, in this activity, we will:

1. Use a **two-sample t-test** to compare city fuel efficiency between cars from **Asia** and the **USA**
2. Use **one-way ANOVA** to compare **horsepower** across different vehicle types

Eventually, you will see that the **two-sample t-test is a special case of ANOVA**.

Dataset for this exercise: `SASHELP.CARS`

In this activity, we use the built-in SAS dataset `SASHELP.CARS`.

This dataset contains information on many car models and includes variables such as:

- `Origin`: region where the car was produced
- `Type`: vehicle type
- `MPG_City`: city miles per gallon
- `MPG_Highway`: highway miles per gallon
- `Horsepower`: engine horsepower
- `Weight`: vehicle weight

Since `SASHELP.CARS` is built into SAS OnDemand / SAS Studio, **no data import is required**.

First Practice: Two-Sample t-Test

In this first example, we compare the **city fuel efficiency** (`MPG_City`) of cars from **Asia** and the **USA**.

Step 1: Explore the Dataset

First, look at the structure of the dataset and preview a few rows.

```
PROC CONTENTS DATA=SASHELP.CARS;
RUN;

PROC PRINT DATA=SASHELP.CARS (OBS=10);
RUN;
```

At this stage:

- Focus on identifying useful variables
- Decide which variable is the **response**
- Decide which variable defines the **groups**

Step 2: Create a Dataset with Two Groups

Since we only want to compare **Asia** and **USA**, we subset the built-in dataset.

```
DATA CARS_SUB;
  SET SASHELP.CARS;

  IF ORIGIN IN ("Asia", "USA");
RUN;
```

Key idea:

- `MPG_City` is the response variable
- `Origin` is the grouping variable
- There are **two groups**, so the correct method is a **two-sample t-test**

Step 3: Run the Two-Sample t-Test

The hypotheses are

$$H_0 : \mu_{\text{Asia}} = \mu_{\text{USA}}$$

and

$$H_1 : \mu_{\text{Asia}} \neq \mu_{\text{USA}}.$$

Use the following code:

```
PROC TTEST DATA=CARS_SUB;
CLASS ORIGIN;
VAR MPG_CITY;
RUN;
```

Code Description

- `CLASS ORIGIN` defines the grouping variable
- `VAR MPG_CITY` defines the response variable
- `PROC TTEST` performs the two-sample t-test

Step 4: Visualize the Two Groups

It is often helpful to examine the two groups visually.

```
PROC SGPLOT DATA=CARS_SUB;
TITLE "City MPG by Car Origin";
VBOX MPG_CITY / CATEGORY=ORIGIN;
RUN;
```

In-Class Questions

1. Which group appears to have the larger mean city MPG?
2. What is the null hypothesis being tested?
3. If the p-value is smaller than 0.05, what conclusion should we make?
4. Does SAS report only one t-test result, or more than one method?

Second Practice: One-Way ANOVA

In this second example, we compare **horsepower** across different **vehicle types**.

Step 1: Understand the Problem

Now the response variable is

- `Horsepower`

and the grouping variable is

- `Type`

Since `Type` has **more than two levels**, we use **one-way ANOVA**.

Step 2: Compute Summary Statistics

Before fitting the ANOVA model, compute summary statistics by group.

```
PROC MEANS DATA=SASHELP.CARS N MEAN STD;
CLASS TYPE;
VAR HORSEPOWER;
RUN;
```

This code helps you inspect:

- sample size in each group
- mean horsepower in each group
- standard deviation in each group

Step 3: Visualize the Groups

```
PROC SGPLOT DATA=SASHELP.CARS;
TITLE "Horsepower by Vehicle Type";
VBOX HORSEPOWER / CATEGORY=TYPE;
RUN;
```

Step 4: Write the Global Hypotheses

The one-way ANOVA hypotheses are

$$H_0 : \mu_1 = \mu_2 = \cdots = \mu_k$$

versus

$$H_1 : \text{At least one group mean differs.}$$

Step 5: Fit the ANOVA Model

```
PROC ANOVA DATA=SASHELP.CARS;
CLASS TYPE;
MODEL HORSEPOWER = TYPE;
RUN;
```

Code Description

- `CLASS TYPE` defines the grouping variable
- `MODEL HORSEPOWER = TYPE` tests whether mean horsepower differs across vehicle types
- The global ANOVA test is based on the **F-statistic**

In-Class Questions

1. What does the global ANOVA test tell us?
2. If the ANOVA p-value is smaller than 0.05, what can we conclude?
3. If the ANOVA test is significant, do we immediately know which groups differ?

Assumption Checking for ANOVA

One important ANOVA assumption is **homogeneity of variances**.

We can check this using **Levene's test**.

```
PROC ANOVA DATA=SASHELP.CARS;
CLASS TYPE;
MODEL HORSEPOWER = TYPE;
MEANS TYPE / HOVTEST=LEVENE WELCH;
RUN;
```

Code Description

- `HOVTEST=LEVENE` requests **Levene's test** for equal variances
- `WELCH` requests **Welch's ANOVA**, which is useful when the equal variance assumption is questionable

In-Class Questions

1. What does Levene's test check?
2. If Levene's test has a p-value below 0.05, what does that suggest?
3. Why might Welch's ANOVA be preferable in that case?

Multiple Comparison After ANOVA

If the global ANOVA test is significant, we often want to know:

> Which groups are different?

We can do this with a **multiple comparison procedure**.

Example: Tukey Test

```
PROC ANOVA DATA=SASHELP.CARS;
CLASS TYPE;
MODEL HORSEPOWER = TYPE;
MEANS TYPE / TUKEY;
RUN;
```

Code Description

- `TUKEY` requests Tukey's multiple comparison procedure
- This compares all pairs of group means while controlling the overall error rate

In-Class Questions

1. Why should we only interpret Tukey results after the global ANOVA test is significant?
2. What is the purpose of a post-hoc test?
3. How is this different from simply running many separate t-tests?

Compare the Two Methods

At this point, compare the two procedures used in this activity.

Two-Sample t-Test

Used when:

- there are **exactly two groups**
- the goal is to compare **two group means**

Example in this activity:

- `MPG_City` for **Asia vs USA**

One-Way ANOVA

Used when:

- there are **three or more groups**
- the goal is to determine whether **at least one mean differs**

Example in this activity:

- `Horsepower` across **vehicle types**

Important Connection

When there are only **two groups**, the **two-sample t-test** and **one-way ANOVA** test the **same null hypothesis**.

So:

> The **two-sample t-test is a special case of ANOVA**.

Suggested Practice Tasks

Work through the following during class.

Task 1

Compare **MPG_Highway** between **Asia** and **Europe** cars.

Suggested steps:

1. Create a subset dataset
2. Identify the response and grouping variables
3. Run a two-sample t-test
4. Draw a boxplot
5. Interpret the p-value

Task 2

Test whether **vehicle weight** differs by **Origin**.

Suggested steps:

1. Compute summary statistics
2. Draw a boxplot
3. Fit the ANOVA model
4. Check the p-value
5. If needed, perform a multiple comparison test

Summary

By following the group-comparison workflow, you have practiced how to:

1. Identify whether a problem involves **two groups** or **multiple groups**
2. Choose between a **two-sample t-test** and **one-way ANOVA**
3. Write SAS code for both procedures
4. Interpret p-values and global test results
5. Check assumptions and perform post-hoc comparisons

This same workflow applies directly to:

- regression with categorical predictors,
- factorial ANOVA,
- linear models,
- and many later topics in data analysis.

Once you master this pattern, mean-comparison problems become much more systematic and easier to solve.

Linear Regression and Interaction

Learning Objectives

By the end of this activity, you should be able to:

1. Fit and interpret a simple linear regression model in SAS
2. Check regression assumptions
3. Understand when interaction is needed
4. Fit and interpret regression with interaction
5. Visualize and interpret interaction effects

Structure of This Activity

This activity follows a three-stage workflow:

1. Build a baseline regression model
2. Diagnose model assumptions
3. Extend to an interaction model

Dataset for This Exercise

We will use a dataset relating:

- height
- sex
- weight

```
DATA MEASUREMENT;
    INPUT SEX $ HEIGHT WEIGHT;
    DATALINES;
Male 67.07 163.61
Male 66.98 162.23
Male 69.13 163.06
Male 67.31 163.76
```

```
Male 66.25 163.84
Male 72.20 168.39
Male 71.39 168.19
Male 60.81 159.14
Male 64.78 160.58
Male 68.13 163.63
Male 63.15 161.36
Male 73.91 170.46
Male 74.38 166.68
Male 69.77 164.70
Male 73.86 170.27
Male 65.34 161.75
Male 72.75 169.74
Male 63.92 160.03
Male 61.85 161.79
Male 71.25 170.94
Male 61.68 157.11
Male 70.71 165.60
Male 65.73 165.85
Male 68.04 163.39
Male 62.76 160.00
Male 63.05 161.20
Male 61.34 158.26
Male 60.59 157.50
Male 74.35 167.14
Male 61.36 161.91
Female 65.42 158.25
Female 65.76 158.12
Female 74.60 182.54
Female 65.67 161.15
Female 66.06 161.69
Female 62.60 159.56
Female 72.26 177.10
Female 65.48 158.31
Female 66.87 167.29
Female 64.29 162.77
Female 71.11 170.03
Female 62.40 154.43
Female 60.95 153.28
Female 63.15 156.38
Female 72.54 171.83
Female 70.49 175.26
```

```
Female 72.68 171.63
Female 67.37 161.84
Female 68.11 165.04
Female 70.48 168.12
Female 64.33 160.85
Female 71.84 177.00
Female 69.55 171.59
Female 64.69 156.69
Female 74.90 182.22
Female 71.89 168.57
Female 73.25 176.51
Female 72.43 174.51
Female 71.01 172.05
Female 63.16 155.14
;
RUN;
```

Part 1: Simple Linear Regression

Fit a simple regression model:

```
PROC REG DATA=MEASUREMENT;
    MODEL WEIGHT = HEIGHT;
RUN;
QUIT;
```

Questions

1. What is the estimated slope?
2. Interpret the slope in context.
3. Is HEIGHT a significant predictor?
4. What does the intercept represent here?

Part 2: Diagnostic Checking

```
PROC REG DATA=MEASUREMENT;
    MODEL WEIGHT = HEIGHT;
    OUTPUT OUT=MYOUT R=RESID;
RUN;
QUIT;

PROC UNIVARIATE DATA=MYOUT NORMAL;
    QQPLOT RESID / NORMAL(MU=EST SIGMA=EST);
RUN;
```

Questions

1. Are the residuals approximately normal?
2. Do you see any obvious violations of model assumptions?
3. What would you check next if the model looked problematic?

Part 3: Add a Categorical Variable

Now include SEX:

```
PROC GLM DATA=MEASUREMENT;
    CLASS SEX;
    MODEL WEIGHT = HEIGHT SEX;
RUN;
QUIT;
```

Questions

1. What does the coefficient or effect of SEX represent?
2. Does this model assume the same slope for males and females?
3. Is SEX associated with average differences in weight after accounting for HEIGHT?

Key Concept

This model assumes:

> The effect of HEIGHT is the same for both groups.

Part 4: Visual Check for Interaction

```
PROC SGPLOT DATA=MEASUREMENT;
    REG X=HEIGHT Y=WEIGHT / GROUP=SEX;
RUN;
```

Questions

1. Are the two fitted lines roughly parallel?
2. Do you suspect interaction?
3. Which group appears to have the steeper slope?

Part 5: Fit the Interaction Model

```
PROC GLM DATA=MEASUREMENT;
    CLASS SEX;
    MODEL WEIGHT = HEIGHT SEX HEIGHT*SEX;
RUN;
QUIT;
```

Questions

1. Is the interaction term significant?
2. How does this model differ from the previous one?
3. Should we keep the interaction term?

Part 6: Interpret the Model

The interaction model is

$$\text{weight} = \beta_0 + \beta_s \text{SEX} + \beta_h \text{HEIGHT} + \beta_{sh}(\text{SEX} \times \text{HEIGHT}).$$

Assume coding:

- SEX = 0 for males
- SEX = 1 for females

Task

Derive the fitted equations for each group.

For males

$$\text{weight}_{male} = \beta_0 + \beta_h \text{HEIGHT}$$

For females

$$\text{weight}_{female} = \beta_0 + \beta_s + (\beta_h + \beta_{sh})\text{HEIGHT}$$

Questions

1. What is the slope for males?
2. What is the slope for females?
3. What does β_{sh} represent?

Key Insight

Interaction = difference in slopes.

Part 7: Visualization Using PROC PLM

```
PROC GLM DATA=MEASUREMENT;
    CLASS SEX;
    MODEL WEIGHT = HEIGHT SEX HEIGHT*SEX;
    STORE INTMODEL;
RUN;
QUIT;

PROC PLM RESTORE=INTMODEL;
    EFFECTPLOT INTERACTION(X=HEIGHT SLICEBY=SEX);
RUN;
```

Questions

1. Does the plot confirm interaction?
2. Which group changes faster as HEIGHT increases?
3. Is the interaction easier to understand using the plot?

Part 8: Numerical Interpretation Practice

Suppose the fitted interaction model is

$$\widehat{\text{weight}} = 20 + 5\text{SEX} + 2\text{HEIGHT} + 1(\text{SEX} \times \text{HEIGHT}).$$

Questions

1. What is the slope for males?
2. What is the slope for females?
3. For a person of height 70, what is the predicted weight for a male?
4. For a person of height 70, what is the predicted weight for a female?

Final Reflection

Discuss the following:

1. When should we include an interaction term?
2. Why is interpretation harder once interaction is added?
3. What is the danger of ignoring interaction when it is truly present?

Summary

- A simple regression model assumes a constant effect.
- Adding a group variable still assumes equal slopes unless interaction is included.
- Interaction allows the effect of one variable to depend on another.
- Once interaction is present, interpretation becomes conditional.
- Visualization is often the clearest way to understand interaction.

Instructor Timing Guide (75 minutes)

Section	Time
Part 1–2	20 min
Part 3–4	15 min
Part 5–6	20 min
Part 7–8	15 min
Final reflection	5 min

Linear Mixed Effects Model

Learning Objectives

By the end of this activity, you should be able to:

1. Identify clustered and hierarchical data structures
2. Distinguish between fixed effects and random effects
3. Specify linear mixed models
4. Implement PROC MIXED in SAS
5. Interpret fixed and random effects
6. Compare models with and without random effects

Structure of This Activity (75 Minutes)

- Part 1 (15 min): Understanding the data
- Part 2 (15 min): Model formulation
- Part 3 (25 min): SAS implementation
- Part 4 (20 min): Interpretation and discussion

Dataset: Multi-Location Crop Yield Study

We study crop yields across:

- Multiple locations (farms)

- Machine types (draper vs stripper)
- Crop varieties (v1, v2)

Each location is **randomly sampled**, so observations within the same location are correlated.

SAS Dataset

```
DATA crop;
INPUT location $ machine $ variety $ yield;
DATALINES;
  A draper v1 35.2
  A draper v2 34.8
  A stripper v1 38.5
  A stripper v2 39.1
  B draper v1 30.5
  B draper v2 31.2
  B stripper v1 34.0
  B stripper v2 35.5
  C draper v1 28.9
  C draper v2 29.5
  C stripper v1 32.1
  C stripper v2 33.0
  D draper v1 36.0
  D draper v2 35.7
  D stripper v1 40.2
  D stripper v2 41.0
  E draper v1 33.3
  E draper v2 34.1
  E stripper v1 37.5
  E stripper v2 38.0
;
RUN;
```

Part 1: Understanding the Data (15 min)

If we would like to understand the data, first of all, we can ask ourselves some questions about the data structure and the variables involved.

Questions

1. What is the response variable in this study?
Answer: The response variable is `yield`.

2. What are the fixed effects in this study?
Answer: The fixed effects are `machine`, `variety`, and possibly the interaction `machine*variety`.

3. What is the random effect in this study?
Answer: The random effect is `location`.

4. Why is `location` treated as a random effect rather than a fixed effect?
Answer: Because the locations are viewed as a random sample from a larger population of farms or locations. We are not only interested in these five specific locations, but in the variability across locations more generally. Treating `location` as a random effect allows us to account for location-to-location variability and to make inference beyond the observed sample.

5. Why are observations from the same location likely to be correlated?
Answer: Because observations from the same location share common environmental and management conditions, such as soil, weather, and farm-specific characteristics. This creates within-location similarity and therefore correlation.

6. If we ignore the location effect, what problem might occur?
Answer: Ignoring the location effect can lead to incorrect standard errors, misleading p-values, and overly optimistic conclusions, because the dependence among observations within the same location is not being modeled.

Part 2: Model Formulation (15 min)

7. Write a linear model for this study if we ignore the random location effect.

One possible fixed-effects model is

$$Y = \beta_0 + \beta_1 \text{machine} + \beta_2 \text{variety} + \beta_3 (\text{machine} \times \text{variety}) + \epsilon.$$

Here:

- β_0 is the overall intercept
- β_1 is the machine effect

- β_2 is the variety effect
- β_3 is the interaction effect between machine and variety
- ϵ is the residual error

8. Write a mixed model for this study by adding a random effect for location.

A mixed model is

$$Y = \beta_0 + \beta_1 \text{machine} + \beta_2 \text{variety} + \beta_3 (\text{machine} \times \text{variety}) + u_{\text{location}} + \epsilon.$$

9. What does u_{location} represent?
Answer:
u_{location} represents the random deviation associated with each location. It captures the idea that some locations may have systematically higher or lower yields than others because of unobserved location-specific conditions.

10. Why does u_{location} induce correlation?
Answer:
Because all observations from the same location share the same random effect u_{location}. This shared term makes observations within the same location more similar to each other than to observations from different locations, which induces within-location correlation.

11. In words, what is the difference between the fixed-effects model and the mixed-effects model?
Answer:
The fixed-effects model only describes the average effects of machine type, variety, and their interaction.
The mixed-effects model does the same, but also accounts for extra variability across locations by including a random location effect.

12. Why is the mixed model more appropriate here?
Answer:
Because the data are grouped by location, and observations from the same location are likely correlated. The mixed model explicitly accounts for that clustering structure.

Part 3: SAS Implementation (25 min)

In this part, we compare a model **without random effects** and a model **with a random location effect**.

The main idea is:

- the fixed-effects model treats all observations as independent after accounting for the explanatory variables
- the mixed-effects model recognizes that observations from the same location may still be correlated

Step 1: Fixed-effects model

```
PROC GLM DATA=crop;
    CLASS machine variety location;
    MODEL yield = machine variety machine*variety location;
RUN;
QUIT;
```

Explanation

This model treats:

- `machine` as a fixed effect
- `variety` as a fixed effect
- `machine*variety` as an interaction effect
- `location` as a fixed effect

So this model asks:

- Is `yield` associated with `machine` type?
- Is `yield` associated with crop `variety`?
- Does the `machine` effect depend on `variety`?
- Do the five observed `locations` differ from one another?

However, this model treats the five locations as the only locations of interest. It does not explicitly model location-to-location variability as a random source of variation.

What assumption does this model make about independence?

Step 2: Mixed-effects model

```
PROC MIXED DATA=crop;
    CLASS location machine variety;
    MODEL yield = machine variety machine*variety;
    RANDOM location;
RUN;
```

Explanation

This model treats:

- `machine` as a fixed effect
- `variety` as a fixed effect
- `machine*variety` as a fixed interaction effect
- `location` as a random effect

This means that the model now assumes the observed locations are a random sample from a larger population of locations.

The line `RANDOM location;` adds a random effect for location, so the model can capture location-to-location variability.

Questions

15. What does RANDOM location; do?
Answer: It adds a random effect for location, allowing each location to have its own deviation from the overall mean. This captures the variability across locations.

16. Why is this useful here?
Answer: Because observations from the same location are likely correlated. The random effect accounts for that within-location dependence.

17. What is the main conceptual difference between this model and the previous one?
Answer: The previous model treats location as fixed, while this model treats location as random and explicitly models variability across locations.

Step 3: Alternative interaction notation

```
PROC MIXED DATA=crop;
    CLASS location machine variety;
    MODEL yield = machine|variety;
    RANDOM location;
RUN;
```

Explanation

In SAS, the notation `machine|variety` is shorthand for `machine variety machine*variety`.

So this model is mathematically the same as the previous mixed model.

It is often more convenient because it automatically includes:

- the main effect of machine
- the main effect of variety
- the interaction machine*variety

Questions

18. What does machine|variety mean in the MODEL statement?
Answer: It tells SAS to include machine, variety, and their interaction machine*variety.

19. Is this model different from the previous mixed model?
Answer: No. It is just a shorter way to write the same fixed-effects structure.

Part 4: Interpretation (20 min)

Question 20

Suppose the fitted mixed model gives:

- machine effect = 3.2
- interaction effect = 1.1

How would you interpret these?

Answer:

- Machine effect = 3.2: On average, changing from draper to stripper is associated with an increase of 3.2 units in yield, for the reference variety.
- Interaction effect = 1.1: The effect of machine type depends on crop variety. In particular, the difference between stripper and draper changes by 1.1 units when moving from the reference variety to the other variety.

Question 21

Why should we be careful when interpreting the machine effect if the interaction is present?

Answer:
Because when an interaction is included, the main effect of machine is interpreted conditionally. It usually represents the machine effect only for the reference level of variety, not an overall effect across all varieties.

Question 22: Model Comparison

Suppose we compare the following two models:

Task 8: Model Comparison

Model	AIC
No random effect	210
With random effect	165

Which model is better? Why?

Answer:
The model with random effect is better because it has the smaller AIC value. In model comparison, a smaller AIC indicates a better balance between model fit and model complexity.

Question 23

What does this comparison suggest about the role of location?

Answer: It suggests that location-to-location variability is important and should be included in the model. The data are better explained when location is treated as a random effect.

Question 24

What might happen if we ignore the random location effect?

Answer:
Ignoring the random effect can lead to incorrect standard errors, misleading p-values, and conclusions that are too optimistic, because the within-location dependence is not being modeled.

What happens if we ignore the random effect?

(A) Correct inference

(B) Smaller variance

(C) Incorrect standard errors

(D) No difference

Key Takeaways

- `PROC GLM` can fit a **fixed-effects model** with location treated as a **fixed factor**
- `PROC MIXED` allows us to treat location as a **random effect**
- `RANDOM location;` captures variability across locations
- `machine|variety` is shorthand for main effects plus interaction
- When interaction is present, main effects must be interpreted carefully

- AIC can help compare competing models

References