CARDAMOM PLANTERS' ASSOCIATION COLLEGE
(Re-Accredited With 'A' Grade By NAAC)
Pankajam Nagar, Bodinayakanur - 625 582.

Department of CS & IT

Statistical data analysis

Unit – 5

PROC UNIVARIATE, PROC MEANS, PROC CORR, PROC PLOT, PROC FREQ, PROC TTEST, PROC NPAR, PROC ANOVA, PROC REG, PROC ARIMA.

5.1. PROC UNIVARIATE in SAS

- **PROC UNIVARIATE** is one of the most powerful procedures in SAS used for **descriptive statistics and distribution analysis of a single variable**.
- It helps analysts **understand the shape, spread, and summary of data** before performing advanced statistical modeling.

5.1.1. Purpose of PROC UNIVARIATE

1. Summary Statistics

PROC UNIVARIATE provides important statistical measures such as:

- **Mean** – Average value
- **Median** – Middle value
- **Mode** – Most frequent value
- **Variance** – Measure of data variability
- **Standard Deviation** – Spread of data
- **Skewness** – Symmetry of distribution
- **Kurtosis** – Peakedness of distribution

2. Distribution Analysis

It performs detailed **distribution analysis**, including:

- **Percentiles**
- **Quartiles**
- **Normality tests**
- **Extreme values**

This helps determine whether data follows a **normal distribution**.

3. Graphical Output

PROC UNIVARIATE can generate several plots to visualize data:

- **Histogram**
- **Box Plot**
- **Stem-and-Leaf Plot**
- **Normal Probability Plot**

These graphs help understand **data patterns and distribution shape**.

4. Outlier Detection

PROC UNIVARIATE helps identify:

- **Extreme values**
- **Unusual observations**
- **Possible outliers**

This is important before performing **statistical modeling or machine learning analysis**.

Basic Syntax

```
proc univariate data=dataset_name;
  var variable_name;
run;
```

- data= → dataset name
- var → variable to be analyzed

Example 1: Simple PROC UNIVARIATE

```
proc univariate data=students;
  var marks;
run;
```

Explanation

This program analyzes the **marks variable** in the students dataset.

It produces statistics such as:

- Mean
- Median
- Mode
- Standard Deviation
- Minimum and Maximum
- Skewness
- Kurtosis

5.1.2. Commonly Used Statements & Options in PROC UNIVARIATE (SAS)

PROC UNIVARIATE has several statements and options that help analyze data and create visualizations.

1. CLASS Statement

The **CLASS** statement calculates statistics **for each group level** of a categorical variable.

Example

1. Create Sample Dataset

```
data students;
  input name $ department $ marks;
  datalines;
```

Arun CSE 85

Ravi ECE 90

Meena CSE 78

Kiran ECE 88

Priya IT 92

Rahul IT 80

;

run;

Example Dataset

Name	Department	Marks
Arun	CSE	85
Ravi	ECE	90
Meena	CSE	78
Kiran	ECE	88
Priya	IT	92
Rahul	IT	80

2. PROC UNIVARIATE with Class

```
proc univariate data=students;
  class department;
  var marks;
run;
```

✓ This computes statistics for **each department separately**.

Output Produced

SAS will produce **separate statistical results for each department**.

Example Output

Department: CSE

Statistic	Value
Mean	81.5
Min	78
Max	85

Department: ECE

Statistic	Value
Mean	89
Min	88
Max	90

Department: IT

Statistic	Value

Mean	86
Min	80
Max	92

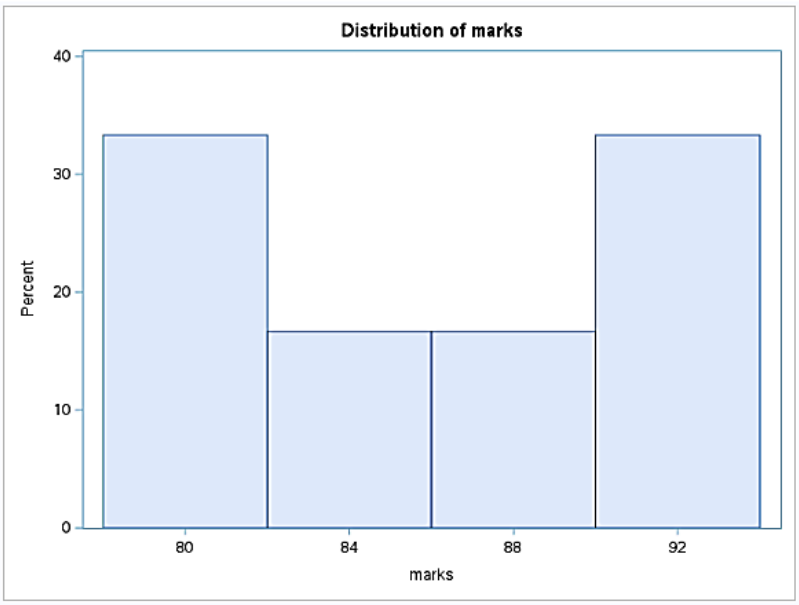
2. HISTOGRAM Statement

The **HISTOGRAM** statement creates a **histogram graph** to visualize the data distribution.

Example

```
data students;
  input name $ department $ marks;
  datalines;
Arun CSE 85
Ravi ECE 90
Meena CSE 78
Kiran ECE 88
Priya IT 92
Rahul IT 80
;
run;
proc univariate data=students;
  var marks;
  histogram;
run;
```

✓ Displays a **histogram showing the distribution of marks**.



3. NORMAL Option

The **NORMAL** option performs **normality tests** to check whether the data follows a **normal distribution**.

Common test:

- **Shapiro–Wilk Test**

  The **Shapiro–Wilk Test** is a **statistical test used to check whether a dataset follows a normal distribution**.
  It is commonly used in **PROC UNIVARIATE** when the **NORMAL option** is specified.

  Purpose

  The test helps determine whether data is **normally distributed**, which is important before applying many statistical methods.

  Hypotheses

  - **Null Hypothesis (H_0):** Data follows a **normal distribution**
  - **Alternative Hypothesis (H_1):** Data **does not follow a normal distribution**

  Decision Rule

P-value	Interpretation
P > 0.05	Data is **normally distributed**
P ≤ 0.05	Data is **not normally distributed**

  Example

  1. Create a Sample Dataset

```
data students;
  input name $ marks;
  datalines;
Arun 85
Ravi 90
Meena 78
Kiran 88
Priya 92
Rahul 80
;
run;
```

  ✓ This creates a dataset called **students** with student marks.

  2. Perform Shapiro–Wilk Normality Test

```
proc univariate data=students normal;
  var marks;
run;
```

  Explanation

  - **PROC UNIVARIATE** → Performs descriptive statistical analysis
  - **NORMAL option** → Requests normality tests
  - **VAR marks** → Tests whether the variable *marks* follows a normal distribution

Output

The SAS output will show a **Tests for Normality table** containing:

Test	Statistic	p-value
Shapiro–Wilk	W value	p-value

Example:

Test	Statistic	p-value
Shapiro–Wilk	0.95	0.95

Interpretation

- p-value > 0.05 → Data is normally distributed
- p-value ≤ 0.05 → Data is not normally distributed

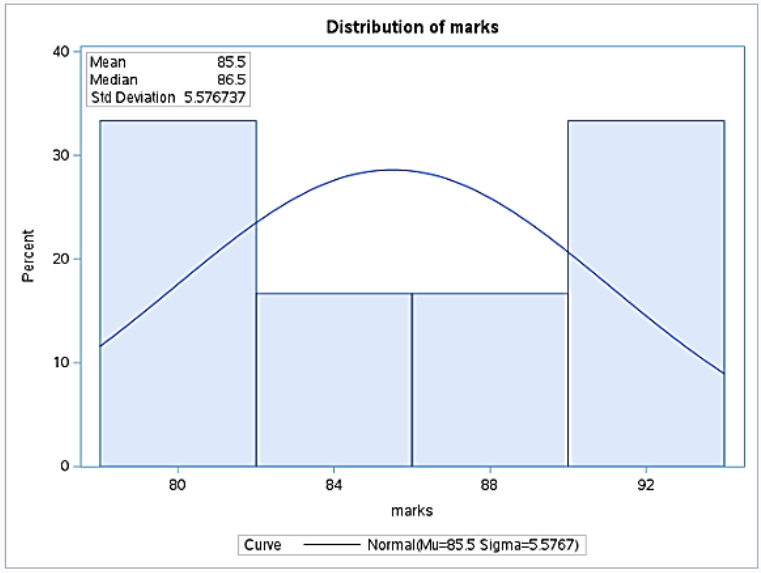
4. INSET Statement

The **INSET** statement adds **summary statistics directly on the plot**.

Example

```
data students;
  input name $ department $ marks;
  datalines;
Arun CSE 85
Ravi ECE 90
Meena CSE 78
Kiran ECE 88
Priya IT 92
Rahul IT 80
;
run;
proc univariate data=students;
  var marks;
  histogram / normal;
  inset mean median std;
run;
```

✓ Displays statistics like **mean, median, and standard deviation on the histogram**.



5. OUTPUT OUT= Option

The **OUTPUT OUT=** option creates a **new dataset containing calculated statistics**.

Example

```
proc univariate data=students;
  var marks;
  output out=stats
  mean=avg
  median=med
  std=sd;
run;
```

✓ Creates a dataset **stats** containing:

- average
- median
- standard deviation

Summary Table

Statement / Option	Purpose
CLASS	Calculates statistics by group
HISTOGRAM	Creates histogram plot
NORMAL	Performs normality tests
INSET	Displays statistics on graph
OUTPUT OUT=	Stores results in a new dataset

5.2. PROC MEANS

PROC MEANS is a SAS procedure used to calculate **summary statistics for numeric variables** in a dataset.
It is commonly used to compute measures such as **mean, sum, minimum, maximum, and standard deviation**.

PROC MEANS helps summarize large datasets quickly and is widely used in **data analysis and reporting**.

5.2.2. Purpose

PROC MEANS is used to:

- Calculate **average (mean) values**
- Find **sum of values**
- Determine **minimum and maximum**
- Compute **standard deviation and variance**
- Summarize numeric data efficiently

5.2.3. Syntax

```
proc means data=dataset_name options;
  class variable;
  var numeric_variable;
run;
```

5.2.4. Common Statistics in PROC MEANS

Statistic	Description
MEAN	Average value
SUM	Total value
MIN	Smallest value
MAX	Largest value
STD	Standard deviation
VAR	Variance
N	Number of observations

Sample Program

Step 1: Create Dataset

```
data students;
  input name $ department $ marks;
  datalines;
Arun CSE 85
Ravi ECE 90
Meena CSE 78
Kiran ECE 88
Priya IT 92
Rahul IT 80
;
run;
```

Step 2: Use PROC MEANS

```
proc means data=students;
  var marks;
run;
```

Output

SAS calculates statistics such as:

- Mean
- Minimum
- Maximum
- Standard Deviation

for the variable **marks**.

6. PROC MEANS with CLASS Statement

```
proc means data=students;
  class department;
  var marks;
run;
```

✓ This calculates statistics **for each department separately**.

Example groups:

- CSE
- ECE
- IT

7. Specifying Required Statistics

```
proc means data=students mean sum min max;
  var marks;
run;
```

✓ Displays only selected statistics.

When to Use

- **Quick summaries** of numeric variables.
- **Grouped statistics** (e.g., average sales by region).
- **Data validation** (checking ranges, missing values).
- **Preliminary analysis** before deeper distribution checks.

5.3. PROC CORR in SAS

PROC CORR is a SAS procedure used to **measure the relationship between two or more numeric variables**.
It calculates the **correlation coefficient**, which shows how strongly variables are related.

The most common correlation method used is **Pearson correlation**.

5.3.2. Purpose

PROC CORR is used to:

- Determine the **relationship between variables**
- Measure **strength and direction of correlation**
- Identify **positive or negative relationships**
- Analyze **linear association between variables**

5.3.3. Types of Correlation

Type	Description
Positive Correlation	Both variables increase together
Negative Correlation	One variable increases while the other decreases
No Correlation	No relationship between variables

Syntax

```
proc corr data=dataset_name options;
  var variable1 variable2;
run;
```

Sample Program

Step 1: Create Dataset

```
data students;
  input id marks attendance;
  datalines;
1 85 90
2 78 80
3 92 95
4 88 85
5 75 70
;
run;
```

Step 2: Apply PROC CORR

```
proc corr data=students;
  var marks attendance;
run;
```

Output

PROC CORR produces a **correlation matrix**.

Example:

Variable	Marks	Attendance
Marks	1.00	0.89
Attendance	0.89	1.00

✓ **0.89 indicates strong positive correlation**.

Correlation Value Interpretation

Correlation Value	Interpretation
+1	Perfect positive correlation
0	No correlation
-1	Perfect negative correlation

5.3.4. Additional Options

Option	Purpose
PEARSON	Pearson correlation
SPEARMAN	Rank correlation
PLOTS=	Creates scatter plots

Example:

```
proc corr data=students plots=matrix;
  var marks attendance;
run;
```

5.3.5. Advantages

- Identifies relationships between variables
- Helps in **predictive analysis**
- Useful in **statistical modeling**

5.4. PROC PLOT in SAS

PROC PLOT is a SAS procedure used to create **simple graphical plots of data**.
It is mainly used to produce **scatter plots and line plots** to visualize the relationship between variables.

PROC PLOT is one of the **basic plotting procedures in SAS**.

Purpose

PROC PLOT is used to:

- Visualize relationships between variables
- Create **scatter plots**
- Understand **data patterns and trends**
- Perform **basic graphical analysis**

Syntax

```
proc plot data=dataset_name;
  plot y_variable * x_variable;
run;
```

✓ y_variable → Variable on Y-axis
✓ x_variable → Variable on X-axis

Key Features

Feature	Example	Output
Simple scatter plot	PLOT height*weight;	Height vs. weight
Multiple plots	PLOT score*age score*hours;	Several plots in one run
Grouping	BY gender;	Separate plots by group
Overlaying	PLOT y1*x y2*x;	Multiple Y variables vs. same X

Sample Program

Step 1: Create Dataset

```
data students;
  input id marks attendance;
  datalines;
1 85 90
2 78 80
3 92 95
4 88 85
5 75 70
;
run;
```

Step 2: Create Plot

```
proc plot data=students;
  plot marks*attendance;
run;
```

Explanation

- marks → Y-axis
- attendance → X-axis

This program produces a **scatter plot showing the relationship between marks and attendance**.

Multiple Plots

You can plot multiple variables in the same graph.

Example:

```
proc plot data=students;
  plot marks*attendance attendance*marks;
run;
```

✓ Creates two plots.

Advantages

- Simple and easy to use
- Helps visualize data quickly
- Useful for identifying **patterns and relationships**

Limitation

- PROC PLOT produces **text-based graphs**, so it is less advanced compared to modern procedures like **PROC SGPLOT**.
- **PROC PLOT** in SAS is used to create **basic scatter plots and line graphs** for visualizing relationships between variables.

5.5. PROC FREQ in SAS

PROC FREQ is a SAS procedure used to **analyze categorical data by calculating frequency counts and percentages**.
It shows how many times each value of a variable occurs in a dataset.

5.5.1. Purpose

PROC FREQ is used to:

- Count **number of occurrences of values**
- Generate **frequency tables**
- Calculate **percentages and cumulative percentages**
- Analyze **relationships between categorical variables**

Syntax

```
proc freq data=dataset_name;
  tables variable;
run;
```

✓ tables statement specifies the variable for which the frequency distribution is calculated.

5.5.2. Key Features

Feature	Example	Output
One-way frequency	TABLES gender;	Count & % by gender
Two-way table	TABLES gender*grade;	Cross-tabulation
Chi-square test	TABLES gender*grade / CHISQ;	Chi-square results
Measures of association	TABLES var1*var2 / MEASURES;	Phi, Cramer's V
Expected values	TABLES var1*var2 / EXPECTED;	Expected cell counts

Sample Program

Step 1: Create Dataset

```
data students;
  input name $ department $ gender $;
  datalines;
Arun CSE M
Ravi ECE M
Meena CSE F
Kiran ECE M
Priya IT F
Rahul IT M
;
run;
```

Step 2: Frequency Distribution

```
proc freq data=students;
  tables department;
run;
```

Output Example

Department	Frequency	Percent
CSE	2	33.3
ECE	2	33.3
IT	2	33.3

✓ Shows **how many students belong to each department**.

5.5.2.1. Two-Way Frequency Table

PROC FREQ can analyze the relationship between two categorical variables.

```
proc freq data=students;
  tables department*gender;
run;
```

✓ Produces a **cross-tabulation table**.

Example:

Department	Male	Female
CSE	1	1
ECE	2	0
IT	1	1

5.5.3. Options to Customize Output in PROC FREQ

PROC FREQ provides several **options to control how the output table is displayed**.

1. NOPERCENT

The **NOPERCENT option** suppresses the **percentage column** in the frequency table.

Example

```
proc freq data=students;
  tables department / nopercent;
run;
```

Output

Department	Frequency
CSE	2
ECE	2
IT	2

✓ The **percent column is removed**.

2. NOCUM

The **NOCUM option** suppresses **cumulative frequency and cumulative percentage**.

Example

```
proc freq data=students;
  tables department / nocum;
run;
```

✓ Removes:

- Cumulative Frequency
- Cumulative Percent

3. ORDER=FREQ

The **ORDER=FREQ option** sorts the categories **from highest frequency to lowest frequency**.

Example

```
proc freq data=students order=freq;
  tables department;
run;
```

Output Example

Department	Frequency
CSE	3
IT	2
ECE	1

✓ Categories appear **in descending order of frequency**.

4. CHISQ

The **CHISQ option** performs a **Chi-Square test** for two-way tables to check if two categorical variables are related.

Example

```
proc freq data=students;
  tables department*gender / chisq;
run;
```

✓ Produces:

- Contingency table
- Chi-square statistic
- P-value

This helps determine **whether two variables are statistically associated**.

Summary Table

Option	Function
NOPERCENT	Removes percentage column

NOCUM	Removes cumulative statistics
ORDER=FREQ	Sorts categories by frequency
CHISQ	Performs chi-square test

Advantages

- Easy analysis of categorical data
- Generates frequency and percentage tables
- Useful for **survey and demographic analysis**

It is used to **analyze categorical variables by generating frequency tables and percentages**, helping users understand the distribution of data.

5.6. PROC TTEST in SAS

PROC TTEST is a SAS procedure used to perform a **t-test**, which compares the **means of two groups** to determine whether they are significantly different.

It is commonly used in statistics to analyze **small sample data**.

5.6.1. Purpose

PROC TTEST is used to:

- Compare **means of two groups**
- Test **hypotheses about population means**
- Determine whether the difference between groups is **statistically significant**
- Analyze **experimental and survey data**

5.6.2. Types of T-Tests

Type	Description	Sample Code	Explanation
One-Sample T-Test	Compares sample mean with a known value	PROC TTEST DATA=students H0=50; VAR score; RUN;	H0=50 → tests if mean score differs from 50.
Two-Sample T-Test	Compares means of two independent groups	PROC TTEST DATA=students; CLASS gender; VAR score; RUN;	CLASS gender → compares male vs female scores.
Paired T-Test	Compares means of paired observations	PROC TTEST DATA=experiment; PAIRED before*after; RUN;	PAIRED before*after → compares matched measurements.

Syntax

```
proc ttest data=dataset_name;
  class group_variable;
  var numeric_variable;
run;
```

✓ class → Defines the groups to compare
✓ var → Specifies the numeric variable

Sample Program

Step 1: Create Dataset

```
data students;
  input name $ gender $ marks;
  datalines;
Arun M 85
Ravi M 78
Kiran M 88
Meena F 92
Priya F 80
Anita F 87
;
run;
```

Step 2: Apply PROC TTEST

```
proc ttest data=students;
  class gender;
  var marks;
run;
```

Explanation

- class gender → Compares **male vs female students**
- var marks → Tests whether **average marks differ between genders**

Output

PROC TTEST produces:

- Group statistics
- Mean difference
- t-value
- Degrees of freedom
- **p-value**

Interpretation

p-value	**Conclusion**
p ≤ 0.05	Significant difference
p > 0.05	No significant difference

Example Result

Gender	**Mean Marks**
Male	83.7
Female	86.3

If **p = 0.04**, the difference is **statistically significant**.

Advantages

- Simple method for comparing group means
- Useful in **research and experiments**
- Provides **statistical significance testing**

It is used to **compare the means of two groups and determine whether the difference between them is statistically significant** using the t-test method.

5.7. PROC NPAR1WAY in SAS

PROC NPAR1WAY is a SAS procedure used to perform **nonparametric statistical tests** for comparing groups.
It is used when the data **does not follow a normal distribution** or when the assumptions of parametric tests (like t-test or ANOVA) are not satisfied.

5.7.1. Purpose

PROC NPAR1WAY is used to:

- Compare **two or more groups**
- Perform **nonparametric tests**
- Analyze **ordinal or non-normally distributed data**
- Test **differences between groups without assuming normal distribution**

5.7.2. Common Nonparametric Tests in PROC NPAR1WAY

Test	Purpose
Wilcoxon Test	Alternative to two-sample t-test
Kruskal–Wallis Test	Alternative to one-way ANOVA
Median Test	Compares medians of groups
Van der Waerden Test	Normal scores test

Syntax

```
proc npar1way data=dataset_name options;
  class group_variable;
  var numeric_variable;
run;
```

✓ class → Defines groups to compare
✓ var → Specifies the variable being analyzed

Sample Program

Step 1: Create Dataset

```
data students;
  input name $ department $ marks;
  datalines;
Arun CSE 85
Ravi ECE 78
Meena CSE 92
Kiran ECE 88
Priya IT 80
```

```
Rahul IT 75
;
run;
```

Step 2: Apply PROC NPAR1WAY

```
proc npar1way data=students wilcoxon;
  class department;
  var marks;
run;
```

Explanation

- wilcoxon → Requests **Wilcoxon rank-sum test**
- class department → Groups data by department
- var marks → Variable being tested

Output

PROC NPAR1WAY produces:

- Rank sums
- Test statistics
- **p-value**
- Median values for groups

Interpretation

p-value	Conclusion
p ≤ 0.05	Significant difference between groups
p > 0.05	No significant difference

Advantages

- Works with **non-normal data**
- Suitable for **small sample sizes**
- Does not require strict statistical assumptions

PROC NPAR1WAY in SAS is used to perform **nonparametric statistical tests to compare groups when data does not meet normal distribution assumptions**.

5.8. PROC ANOVA in SAS

PROC ANOVA (Analysis of Variance) is a SAS procedure used to **compare the means of three or more groups** to determine whether there is a **significant difference among them**.

It is an extension of the **t-test** for multiple groups.

Purpose

PROC ANOVA is used to:

- Compare **means of multiple groups**
- Test **statistical significance of differences**
- Analyze **variation within and between groups**

- Perform **experimental data analysis**

Basic Concept

ANOVA works by dividing total variation into:

- **Between-group variation**
- **Within-group variation**

Then it calculates the **F-statistic** to test significance.

Syntax

```
PROC ANOVA DATA=dataset;
  CLASS categorical_variable;
  MODEL numeric_variable = categorical_variable;
  MEANS categorical_variable / TUKEY CLDIFF;
RUN;
```

- **CLASS** → defines the categorical grouping variable(s).
- **MODEL** → specifies the dependent variable and factors.
- **MEANS** → requests post-hoc tests (e.g., Tukey) for pairwise comparisons.

Sample Program

Step 1: Create Dataset

```
data students;
  input name $ department $ marks;
  datalines;
Arun CSE 85
Ravi ECE 78
Meena CSE 92
Kiran ECE 88
Priya IT 80
Rahul IT 75
;
run;
```

Step 2: Apply PROC ANOVA

```
proc anova data=students;
  class department;
  model marks = department;
run;
```

Explanation

- class department → Groups data by department
- model marks = department → Compares **marks across departments**

Output

PROC ANOVA produces:

- ANOVA table

- Sum of Squares (SS)
- Mean Squares (MS)
- **F-value**
- **p-value**

Interpretation

p-value	Conclusion
p ≤ 0.05	Significant difference between groups
p > 0.05	No significant difference

Example Result

If output shows:

- **F = 4.25**
- **p = 0.03**

✓ Conclusion: There is a **significant difference in marks among departments**.

Key Features

1. One-way ANOVA

Description:
Tests mean differences across **one categorical variable**.

✓ Used when:

- One independent variable (factor)
- One dependent variable

Example

```
proc anova data=students;
  class department;
  model marks = department;
run;
```

✓ Compares **marks across departments**

2. Two-way ANOVA

Description:
Tests mean differences across **two categorical variables** and also checks their **interaction effect**.

✓ Used when:

- Two independent variables
- One dependent variable

Example

```
proc anova data=students;
  class department gender;
```

```
  model marks = department gender department*gender;
run;
```

✓ Analyzes:

- Effect of department
- Effect of gender
- Interaction effect (department × gender)

3. Post-hoc Tests

Description:
Used after ANOVA to identify **which specific groups differ**.

Common tests:

- **Tukey**
- **Bonferroni**
- **Scheffé**

Example

```
proc anova data=students;
  class department;
  model marks = department;
  means department / tukey;
run;
```

✓ Shows **pairwise comparisons between groups**

4. Homogeneity Tests

Description:
Checks whether **group variances are equal** (important ANOVA assumption).

- Common test: **Levene's Test**

✓ If variances are not equal → ANOVA assumptions may be violated.

5. Multivariate Option (MANOVA)

Description:
Used when there are **multiple dependent variables**.

Example

```
proc anova data=students;
  class department;
  model marks attendance = department;
  manova h=department;
run;
```

✓ Analyzes multiple outputs:

- Marks
- Attendance

Advantages

- Compares multiple groups simultaneously
- Reduces error compared to multiple t-tests
- Widely used in **research and experiments**

PROC ANOVA in SAS is used to **analyze differences among group means** and determine whether those differences are statistically significant using the F-test.

5.9. PROC REG

PROC REG is a SAS procedure used to perform **linear regression analysis**.
It helps in modeling the **relationship between a dependent variable and one or more independent variables**.

Purpose

PROC REG is used to:

- Analyze **relationships between variables**
- Perform **simple and multiple linear regression**
- Predict **dependent variable values**
- Evaluate model performance using **R^2, p-values, and coefficients**

Types of Regression

Type	Description
Simple Linear Regression	One independent variable
Multiple Linear Regression	More than one independent variable

Syntax

```
proc reg data=dataset_name;
  model dependent_variable = independent_variables;
run;
quit;
```

✓ model statement defines the regression equation

Sample Program

Step 1: Create Dataset

```
data students;
  input marks attendance study_hours;
  datalines;
85 90 4
78 80 3
92 95 5
88 85 4
75 70 2
;
run;
```

Step 2: Apply PROC REG

```
proc reg data=students;
  model marks = attendance study_hours;
```

```
run;
quit;
```

Explanation

- marks → Dependent variable
- attendance, study_hours → Independent variables

✓ This model predicts **marks based on attendance and study hours**.

Output

PROC REG produces:

- Regression equation
- Coefficients (β values)
- **R^2 (coefficient of determination)**
- **p-values**
- ANOVA table

Example Regression Equation

Marks = 10 + 0.5(Attendance) + 2(Study Hours)

✓ Interpretation:

- Increase in attendance → increases marks
- Increase in study hours → increases marks

Interpretation

Value	Meaning
R^2	Model accuracy
p-value	Significance of variables
Coefficient	Impact of independent variable

Advantages

- Predicts future values
- Identifies important factors
- Useful in **data analysis and forecasting**

PROC REG in SAS is used to perform **linear regression analysis**, helping to understand relationships between variables and make predictions.

5.10. PROC ARIMA

PROC ARIMA is a SAS procedure used for **time series analysis and forecasting**.
It applies the **ARIMA model (Auto Regressive Integrated Moving Average)** to analyze past data and predict future values.

Purpose

PROC ARIMA is used to:

- Analyze **time-based data**

- Identify **patterns and trends**
- Build **forecasting models**
- Predict **future values**
- Handle **seasonal and non-stationary data**

Components of ARIMA Model

ARIMA model has three parts:

Component	Meaning
AR (p)	AutoRegressive part (past values)
I (d)	Integrated part (differencing)
MA (q)	Moving Average part (past errors)

Syntax

```
proc arima data=dataset_name;
  identify var=variable;
  estimate p= ar_value q= ma_value;
  forecast lead=number_of_periods out=output_dataset;
run;
quit;
```

Steps in PROC ARIMA

1. Identify

- Checks whether data is **stationary**
- Helps determine model parameters (p, d, q)

2. Estimate

- Estimates parameters of ARIMA model

3. Forecast

- Predicts future values

Sample Program

Step 1: Create Dataset

```
data sales;
  input month sales;
  datalines;
1 100
2 120
3 130
4 150
5 170
6 180
;
run;
```

Step 2: Apply PROC ARIMA

```
proc arima data=sales;
  identify var=sales;
  estimate p=1 q=1;
  forecast lead=3 out=forecast_data;
run;
quit;
```

Explanation

- identify var=sales; → Analyzes time series data
- estimate p=1 q=1; → Fits ARIMA(1,1) model
- forecast lead=3; → Predicts next **3 periods**
- out=forecast_data; → Stores results in a new dataset

Output

PROC ARIMA produces:

- Model parameters
- Diagnostic statistics
- Forecast values
- Confidence intervals

Advantages

- Handles **time series data effectively**
- Useful for **forecasting future trends**
- Works with **seasonal and non-stationary data**

PROC ARIMA in SAS is used for **time series modeling and forecasting**, allowing users to analyze trends and predict future values using ARIMA models.